Decentralised Key Management for Delay Tolerant Networks

by

Christopher I. Djamaludin

Bachelor of Engineering (Electrical)(Hons I)
(University of Queensland) – 2007

Thesis submitted in accordance with the regulations for the Degree of Doctor of Philosophy

Information Security Discipline
Science and Engineering Faculty
Queensland University of Technology

2016

Keywords

Delay Tolerant Network, Decentralised, Public Key Infrastructure, Public Key Authentication, Key Management, Key Distribution, Key Revocation, Autonomous, Trust, Reputation.

Abstract

Advances in ubiquitous mobile computing has given rise to the pervasive deployment of physical devices embedded with sensors, software, and wireless communications that collect and exchange data. These devices are adept for deployment in environments with minimal existing infrastructure, as the devices themselves become the components of a functioning network. With the addition of mobility, these devices operate where disruption between entities is high, resulting in dynamic, fragmented, and ephemeral networks. Such networks are considered Delay Tolerant Networks (DTNs). The pervasiveness of DTN nodes, and their varied deployment environments leads to two important motivations for securing DTNs. First, the data collected, stored, and transferred between nodes can be of high value due to commercial, safety, or national security reasons. Second, persistent threats from adversaries are common. Consequently, the development of techniques to secure communications from persistent threats from within a DTN is an important area of research.

One particular challenge that is foundational for securing DTN is public key management, in particular the provision of public key authentication. This is the ability for a node to verify the identity-public key binding of another node. The lack of public key authentication provides an adversary the capability of modifying the identity-public key bindings. This allows them to eavesdrop and modify contents of communications, as well as assume identities of others in the network for authentication purposes. As a result, the ability to verify the identity-public key binding is foundational to providing confidentiality, integrity, and message authentication that ensures security in a DTN. Public key authentication is achieved using Public Key Infrastructure (PKI). Centralised hierarchical PKI implementations rely on pre-established trust in a Certificate Authority (CA), while decentralised implementations such as Pretty Good Privacy (PGP) rely on humans for trust establishment. These two methods of public key authentication

are unsuitable in an autonomous DTN, where there is no pre-established trust, and human involvement. Therefore, distributed, self-organised, and autonomous approaches for providing public key authentication and key management are needed.

The work presented in this thesis addresses the above challenges and makes several related contributions. The *first* contribution of this thesis is the development and evaluation of a proof-of-concept public key authentication scheme for DTNs. The proposed scheme called the Leverage of Common Friends (LCF) trust system, utilised a trust system to provide confidence in the identity-key binding of an autonomous DTN node. The scheme was evaluated by introducing an adversarial agent performing a key spoof attack, and was effective in mitigating the distribution of adversarial keys by 40%.

The *second* contribution is the evaluation of the LCF trust system in a realistic large scale geographic environment with a large quantity of autonomous nodes. The proposed scheme was subjected to mobility movement data of taxi cabs in downtown San Francisco, along with the introduction of varying amounts and varieties of adversarial agents. The trust system was successful in mitigating the distribution of adversarial keys in some experiments by 70%.

The *third* contribution is the development and evaluation of a public key authentication scheme that expanded the LCF trust system to include co-localisation information. The proposed scheme called Location based Leverage of Common Friends (LLCF) was also evaluated to the realistic mobility movement data and varying quantities and varieties of adversarial agents. The addition of co-localisation data was found to improve security against stationary adversaries by an additional 50% in comparison to the LCF trust system, at a small cost of 13% to key distribution performance.

The *fourth* contribution of the thesis is the development and evaluation of an unplanned public key revocation and replacement scheme for autonomous DTNs. The proposed scheme was developed and evaluated for an environment with no Trusted Third Party (TTP) or human involvement, and was simulated on a large geographic scale. The public key revocation and replacement scheme was found to be successful in the timely removal of the old public key, and the efficient distribution of the new public key, even whilst under internal attack from adversaries. A typical experiment resulted in an additional 35% more new public keys being distributed, while reducing the adversary keys by 40%.

Dedication

To my Dad.

Contents

List of Figures

List of Tables

List of Acronyms

ADOPT Ad-hoc Distributed OCSP for Trust

CA Certificate Authority

CBD Central Business District

CRL Certificate Revocation List

CSR Certificate Signing Request

DDoS Distributed Denial of Service

DoS Denial of Service

DS Distributed Signing

DSRC Dedicated Short Range Communications

DTN Delay Tolerant Network

ECC Elliptic Curve Cryptography

EU European Union

FRESH Fresher Encounter SearcH

GPG GNU Privacy Guard

GPS Global Positioning System

HMAC Hash-Based Message Authentication Code

IBC Identity Based Cryptography

IoT Internet of Things

IP Internet Protocol

IPv6 Internet Protocol version 6

ITS Intelligent Transportation System

KDC Key Distribution Centre

KGC Key Generation Centre

LCF Leverage of Common Friends

LLCF Location based Leverage of Common Friends

MANET Mobile Ad-Hoc Network

MITM Man In The Middle

OCSP Online Certificate Status Protocol

P2P Peer to Peer

PGP Pretty Good Privacy

PKI Public Key Infrastructure

PKG Private Key Generator

PROPHET PRObabilistic Protocol using History of Encounters and Transitivity

QoS Quality of Service

RA Registration Authority

RO Remove Only

RR Remove and Replace

RSU Road Side Unit

TCP/IP Transmission Control Protocol/Internet Protocol

TLS Transport Layer Security

TTP Trusted Third Party

US United States

V2I Vehicle to Infrastructure

V2V Vehicle to Vehicle

VANET Vehicle Ad-Hoc Network

VARS Vehicular Ad-Hoc Network Reputation System

WSN Wireless Sensor Network

Declaration

The work contained in this thesis has not been previously submitted to meet requirements for an award at this or any other higher education institution. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made.

Signed: **Date:** 2016.04.11

Previously Published Material

The following articles have been published, and contain material based on the content of this thesis.

- **Journal Papers**
  - C. I. Djamaludin, E. Foo, and P. Corke, "Establishing Initial Trust in Autonomous Delay Tolerant Networks without Centralised PKI," *Computers & Security*, vol. 39, Part B, pp. 299 - 314, 2013.
  - C. I. Djamaludin, E. Foo, S. Camtepe, and P. Corke, "A Self-Organising Initial Trust Establishment Scheme for Autonomous VANETs," *Journal of Networks and Computer Applications*, (In Revision).
  - C. I. Djamaludin, E. Foo, S. Camtepe, and P. Corke, "Revocation and Update of Trust in Autonomous Delay Tolerant Networks," *Computers & Security*, 2016. (Article in press).
- **Conference Papers**
  - E. Foo, C. I. Djamaludin, and A. Rakotonirainy, "Security Issues for Future Intelligent Transport Systems (ITS)," in *Proceedings of the 2015 Australasian Road Safety Conference*, 2015.

Acknowledgements

Over the duration it has taken for me to complete this work, many have contributed and encouraged me throughout this process. My appreciation and gratitude is heartfelt to the following.

First and foremost, my sincere and heartfelt thanks to Dr. Ernest Foo. Who has been my principle supervisor, advisor, mentor, and a good friend. This work would not have been possible without his guidance, wisdom, and technical direction. For his perseverance and endless patience with me, particularly during times when I was not. This thesis is as much his achievement as it is mine.

Dr. Seyit Camtepe, my associate supervisor and good friend, who through his mentor-ship and friendship has provided great technical direction, as well as encouragement at times when it was much needed. His meticulous approach to proof-reading and generous feedback has greatly improved the quality of research in this thesis.

Professor Peter Corke, my associate supervisor. For the wealth of experience he brought to this research, and who guided me through the beginnings of my research and this work.

Professor Gordon Wyeth, and Associate Professor Geoffrey Walker, who were both instrumental in motivating me to undertake this research.

To the numerous reviewers who have provided helpful critical feedback on this work. Their suggestions and improvements have helped focus this research. In particular, the panel members, Professor Josef Pieprzyk and Dr. Dhammika Jayalath for their feedback and encouragement in this work.

Past and present academics of the past Information Security Institute, now Information Security Research Discipline. They include, Professor Colin Boyd, Professor Ed Dawson, Professor Colin Fidge, Associate Professor Xavier Boyen, Dr. Leonie Simpson, Dr. Douglas Stebila, and Dr. Desmond Schmidt. For their encouragement and wealth of broad knowledge collectively.

My fellow colleagues and friends, Dr. Sajal Bhatia, Dr. Farzad Salim, Dr. Andrew White, Dr. Christophe Hauser, Dr. Chai Wen Chuah, Dr. Sui Guan Teo, Dr. Kaleb Lee, Dr. Janaka Alawatugoda, Raphael Amoah, James Akande, Ben Dowling, Qinyi Li, Udyani Herath, David Myers, and Nishchal Kush for their friendship and support. Special thanks to Thomas Haines who always paid for coffee.

Nicholas Rodofile, for his friendship, generosity, and frequent outings to lunch.

Dr. Mark Branagan (Special Agent) from [redacted], who has provided [redacted], and coffee.

The Queensland University of Technology, for the free lunch.

My nieces and nephew, Abigail, Benjamin, and Lilla, for providing a much welcomed distraction in the form of Star Wars and Lego.

My brother Daniel and his wife Rachel, and extended family Uncle Ricky, Aunty Viviana, Mama, for encouraging me.

My Oma, who has always encouraged me, even from the very beginning.

My parents, who has always encouraged and never doubted me throughout this whole process. They have tirelessly worked to provide an education for me, and imparted its importance to me. My Dad, to whom this thesis is dedicated to, for his curiosity to learn, and whom I follow. My Mum, who knows what it's like to research.

My amazing wife Rachael, whom I am blessed to have. Who has endured so much over the years, supported me in everything, and is instrumental in my motivation for this thesis. My heartfelt appreciation and gratitude for all that you have endured and done, and all that you have endured and done that I didn't know. These words are as much yours, as they are mine.

And finally, to God. For without Him, I am nothing.

Chapter 1

Introduction

Advances in a combination of fields ranging from semi-conductor design, power and battery technology, and wireless communications have given rise to ubiquitous mobile computing. Physical objects embedded with sensors, software, and wireless communications that collect and exchange data. These devices are embedded everywhere, and their pervasiveness has now encompassed every part of modern society. Having now been heavily reliant on these devices for communications (mobile phones and tablets), commuting (Intelligent Transportation Systems (ITSs)), manufacturing (control, monitoring and instrumentation systems), health (health monitoring devices), and national security (battlefield networks and autonomous drones). Typically, these devices are connected to the Internet, to make the Internet of Things (IoT). However, due to the mobility of devices, and deployment environments, there is an increase in network applications where disruption between network entities is high. These networks are considered *Delay Tolerant Networks (DTNs)* or *Disruption Tolerant Networks*.

Securing such networks has become a deliberate necessity for two reasons. First, the data collected, stored and transferred can be of high value due to commercial, safety, personal, or national security reasons. Second, the sheer pervasiveness of these devices means they are varied, numerous, and potentially deployed in hostile environments. These two reasons results in a high probability for adversaries to operate in the network, and the motivation for compromising the network is attractive.

In comparison to conventional networks, DTNs are highly dynamic and dis-

tributed networks that create opportunistic ephemeral connections between nodes that can span large geographic areas. There is no guarantee that a reliable source to destination path can be maintained for long periods of time. This means information transfer between mobile nodes is in the form of data bundles, as a store-carry-and-forward scheme relayed through multiple intermediary autonomous nodes or through multiple paths. Since multiple intermediary nodes will store, carry, and forward a data bundle, it is important to provide some general security properties in the network. Confidentiality is required when information in the data bundle is intended only for the source and destination nodes. Intermediate carrier nodes passing on data bundles may eavesdrop on information unintended to be disclosed to them. Integrity of data is also required, as there is the potential for the modification of information as it is passed between intermediate carrier nodes. In some applications, it may be required for both the data source and contents of the data be verifiable, thereby providing message authentication. Typically these general security properties can be achieved using combinations of cryptographic technologies such as cryptographic hash functions, and private and public (asymmetric) keys. One aspect of providing security to protect and mitigate from such attacks is *Key Management* and *Public Key Authentication*, which is the subject of this thesis.

The remainder of this chapter is organised as follows. Section 1.1 provides a brief overview of security in DTNs, public key authentication, its importance, and application to autonomous DTNs. It also explains the research motivation of this thesis. Section 1.2 states the overarching research objective and identifies the related research questions associated. Section 1.3 summarises the contributions of this thesis. Finally, Section 1.4 outlines the overall structure of the thesis.

1.1 Background and Motivation

The implementation of a public key cryptographic system to achieve these security properties in a DTN raises the important issue of *Public Key Authentication*: The ability for an entity to verify the identity-public key binding of another entity. Without public key authentication, any adversary would be capable of modifying the identity-public key bindings to eavesdrop on passing communications, modify the contents of data bundles, and change the origin of the data. Because of this, the concept of public key authentication is the foundation from

which all other security properties are achieved. It is therefore a significant aspect of security. An example is in the critical application of battlefield networks. Soldiers are often deployed in environments that are remote, isolated, and void of any infrastructure. Their mobility and lack of infrastructure forms a DTN. To operate safely and effectively, soldiers require many security properties such as confidentiality, data integrity, and message authentication. Communications between members of the group will need to be confidential, as to prevent the adversary from eavesdropping on battle plans, movements and orders. Data integrity of the messages sent is required to prevent intentional tampering and modification. Without data integrity, orders such as "Enemy location 27-28-0 S, 153-2-0-E" can be modified to "Enemy location 28-27-0 S, 163-2-0-E" by adversarial agents. In addition, message authentication, the requirement to authenticate the source of a message is important. Soldiers will want to ensure an order was sent by their group commander, not an adversary. An adversary is capable of performing these attacks if there is no public key authentication. Key spoof attacks can lead to Man In The Middle (MITM) and Sybil attacks, thereby undermining the entire security architecture of the network. As a result, public key authentication is a critical aspect of security in any network, and its provision is still an open research problem.

The provision of public key authentication is through Public Key Infrastructure (PKI), a set of policies dependent on the concept of trust. Conventional networks with dependable infrastructure components such as the Internet rely on a centralised Trusted Third Party (TTP) where trust is binary; either the party is trustworthy or untrustworthy. While more distributed models such as Pretty Good Privacy (PGP) utilise discrete states; where human reasoning is required to determine the criteria for each state. While such PKI implementations are suitable to conventional networks, the characteristics of DTNs make them unsuitable for public key authentication. The problem is further compounded with the lack of human intervention to guide trust in an autonomous DTN. The use of a trust or reputation system in autonomous networks is already feasible to provide assurance and security. However, its application to assist the provision of public key authentication is still an active area for research and development.

Over the years, ubiquitous mobile computing has encompassed every part of modern society. With autonomous DTNs now being deployed in greater numbers and in more hostile environments, the information collected, stored and trans-

ferred has forced the necessity to secure it. As a result, providing public key authentication in autonomous DTNs is critical to securing communications and data in such networks.

1.2 Research Aims and Questions

The overarching aim of this thesis is *to investigate, develop, and evaluate techniques to provide public key authentication during all phases of public key management for a DTN comprising of autonomous nodes.* From this overall aim, the following research questions have been identified from the various phases of key management:

Research Question 1: *Can a trust or reputation system be utilised to assist in DTN Key Distribution such that Public Key Authentication can be achieved without a trusted third party, but by automatically including mobility parameters, behaviour, and levels of collaboration into trust?* This question explores the utilisation of a trust or reputation system to address public key authentication. The background and related work presented in Chapter 2 indicates that trust and reputation systems are used to provide adversary detection or optimal message routing. The literature also explores that public key authentication during key distribution is traditionally achieved by a TTP such as a Certificate Authority (CA) or signing keys through a Web of Trust. Combining these two concepts directly addresses the issue of public key authentication between autonomous nodes during the key distribution phase.

Research Question 2: *Is it possible to apply a trust or reputation system for DTN Key Distribution for a large scale realistic DTN application?* This question further investigates the suitability and effectiveness of a public key authentication scheme for DTNs. Previous proposals for providing public key authentication in DTNs evaluate the schemes in small, controlled, and closed environments. The potential pervasive deployment of DTN nodes creates questions on how security is designed and implemented for larger scale realistic DTNs. In particular, where persistent adversarial agents exist in the network.

Research Question 3: *Is it possible to leverage location data to assist a trust or reputation system for DTN Key Distribution?* DTN applications such

as Vehicle Ad-Hoc Networks (VANETs) are large scale realistic environments. VANET nodes (cars, taxis, buses) also provide additional resources in the form of Global Positioning System (GPS) driven location data. This additional information may assist autonomous nodes in public key authentication by feeding location data into a trust or reputation system.

Research Question 4: *Is it possible to utilise a trust or reputation system to assist in DTN Key Revocation such that Public Key Authentication can be achieved without a trusted third party?* Key revocation is an integral phase of the key management lifecycle. Effective key management schemes have provisions for the efficient removal of public keys from operation. Traditional key revocation schemes, are heavily reliant on a TTP or human intervention. This is to provide the necessary public key authentication to prevent false and erroneous key revocation by an adversarial agent. Extending public key authentication for the key revocation phase when the entities are autonomous, with no human intervention gives rise to this research question.

Research Question 5: *Is it possible to provide trust transferral of an old compromised public key to a newly generated public key without a trusted third party during an unplanned key revocation event?* This final research question addresses a specific un-addressed but critical revocation event. Revocation of public keys may be *planned*, due to a finite lifetime for keys, or organisational and environmental policy. Trust transferral between the old public key, and the new public key can be transferred through various planned and organised methods (see Section 2.3 for details). However, the *unplanned* scenario where public keys need to be revoked is usually due to private key compromise by an adversary. In this scenario, trust transferral between the old public key and the new public key is either non-existent, or difficult, and requires human intervention. The motivation to provide trust transferral between the old public key and the new public key for autonomous nodes gives rise to this final research question.

1.3 Research Contributions

This thesis has provided various research contributions to DTNs, by providing secure key management for DTNs using trust and reputation systems. More

specifically, this research has resulted in the following contributions and outcomes:

Contribution 1: Chapter 3 investigated, developed, and evaluated a proof-of-concept public key authentication scheme. The proposed scheme called the Leverage of Common Friends (LCF) trust system, extended previous research in this field by utilising a trust system to provide public key authentication in an autonomous DTN, thereby addressing Research Question 1. The proposed scheme implemented a common and effective linear trust computation engine that was evaluated in extensive simulations with the introduction of an adversarial agent. The results from the evaluation, showed the system significantly mitigated the impact an adversary would have on the network. To the best of our knowledge, no other key distribution scheme for autonomous DTNs has utilised a trust or reputation system to assist with key management. This research outcome led to the following publication:

> C. I. Djamaludin, E. Foo, and P. Corke, "Establishing Initial Trust in Autonomous Delay Tolerant Networks without Centralised PKI," *Computers & Security*, vol. 39, Part B, pp. 299 - 314, 2013.

Contribution 2: Chapter 4 evaluated the proof-of-concept public key authentication scheme proposed in Chapter 3 by deploying and simulating it in a realistic large scale geographic environment with a large quantity of autonomous nodes, over a longer duration, thereby addressing Research Question 2. The proposed scheme was subjected to mobility movement data from taxi cabs in downtown San Francisco, along with the introduction of varying amounts and varieties of adversarial agents. The public key authentication scheme proposed in Chapter 3 was evaluated and found to be successful in providing public key authentication, as public keys of legitimate nodes were distributed, whilst mitigating the impact of multiple adversaries.

Chapter 4 also investigated, developed, and evaluated an extension of the public key authentication scheme proposed in Chapter 3 with the addition of co-localisation information to the trust system. The proposed scheme called Location based Leverage of Common Friends (LLCF) was also evaluated to the realistic mobility movement model and varying quantities and varieties of adversarial agents. The addition of co-localisation data was found to improve security

against stationary adversaries at a small cost to key distribution performance. This research contribution addresses Research Question 3. Addressing Research Questions 2 and 3 has led to the following publications:

> E. Foo, C. I. Djamaludin, and A. Rakotonirainy, "Security Issues for Future Intelligent Transport Systems (ITS)," in *Proceedings of the 2015 Australasian Road Safety Conference*, 2015.
>
> C. I. Djamaludin, E. Foo, S. Camtepe, and P. Corke, "A Self-Organising Initial Trust Establishment Scheme for Autonomous VANETs," *Journal of Networks and Computer Applications*, (In Revision).

Contribution 3: Chapter 5 presents an unplanned public key revocation and replacement scheme for autonomous DTNs. The proposed scheme was developed and evaluated for an environment with no TTP or human intervention, and was simulated on a similar scale as the evaluation of Chapter 4. This addressed Research Question 4. The public key revocation and replacement scheme was found to be successful in the timely removal of the old public key, and the efficient distribution of the new public key, even whilst under internal attack from adversaries. Chapter 5 also investigates trust transferral between the old public key and new public key during an unplanned revocation event, through the use of the trust system employed during the key distribution phase. This addresses Research Question 5. To the best of our knowledge, no other key revocation and replacement scheme for autonomous DTNs has addressed public key revocation and replacement with the provision of trust transferral. Addressing Research Questions 4 and 5 has led to the following publication:

> C. I. Djamaludin, E. Foo, S. Camtepe, and P. Corke, "Revocation and Update of Trust in Autonomous Delay Tolerant Networks," *Computers & Security*, 2016. (Article in press).

1.4 Organisation of Thesis

The following five chapters of this thesis are organised as follows:

Chapter 2 introduces the concepts of DTNs, in particular the characteristics, routing, applications, and challenges that differentiate a DTN to traditional networks such as the Internet. Two concepts important to this research are also covered. The first is trust and reputation systems. General concepts of trust and reputation systems and how they are important for the operation of an autonomous network are outlined. It also covers commonly implemented trust and reputation system architectures and computation engines. The second concept is PKI and Key Management. This includes an overview of the general concepts of PKI, in particular the two traditional methodologies of providing public key authentication, CAs and OpenPGP Web of Trust. Specific trust and reputation system proposals for DTNs and VANETs are also discussed, in particular how they are applied for adversary detection and optimal message routing. More specific key management proposals for DTNs and VANETs are also discussed, focussing on the two key management phases of interest; distribution, and revocation.

Chapter 3 proposes a public key distribution scheme that integrates a trust system to provide public key authentication. The chapter provides a proof-of-concept implementation called LCF using a common and effective linear computation engine that leverages social contacts between autonomous nodes. A selection of trust variables and weightings are analysed. The implementation is simulated in a controlled closed environment with an adversarial agent to evaluate various security and performance metrics. The results indicated an effective mitigation of the adversary from distributing spoofed keys by exploiting the lack of a TTP to provide public key authentication.

Chapter 4 built on the LCF proof-of-concept public key distribution scheme developed in Chapter 3 by deploying it in a realistic VANET environment. It was also extended with the addition of co-localisation data to form a location based trust system for key distribution called LLCF. Both LCF and LLCF trust systems were evaluated using a realistic mobility movement model of taxi cabs in downtown San Francisco. The evaluation tested the scalability of both trust systems for deployment in a large scale geographic environment, with a large quantity of nodes. Varying quantities and varieties of adversarial agents were also introduced to evaluate the effectiveness of both trust systems in mitigating a key spoof attack. Both stationary and moving adversaries were introduced to help

determine the impact of including co-localisation data in the LLCF trust system. The results from the experiments showed significant mitigation of spoofed public keys in the network for both the LCF and LLCF trust systems. The addition of co-localisation data in the LLCF trust system provided additional security against a stationary adversary at minimal cost to performance.

Chapter 5 proposes a decentralised public key revocation and replacement scheme that provides trust transferral between the old public key and new public key during an unplanned revocation. The scheme was compared and evaluated to similar LCF based revocation schemes using a simulation environment similar to the experiments of Chapter 4. Varying quantities and varieties of adversarial agents performing a Sybil attack were also introduced to evaluate the effectiveness of old key removal, new key distribution, and trust transferral. The results from the experiments showed significant mitigation of spoofed public keys in the network from adversaries, as well as the complete prevention of an adversary triggering a false revocation event.

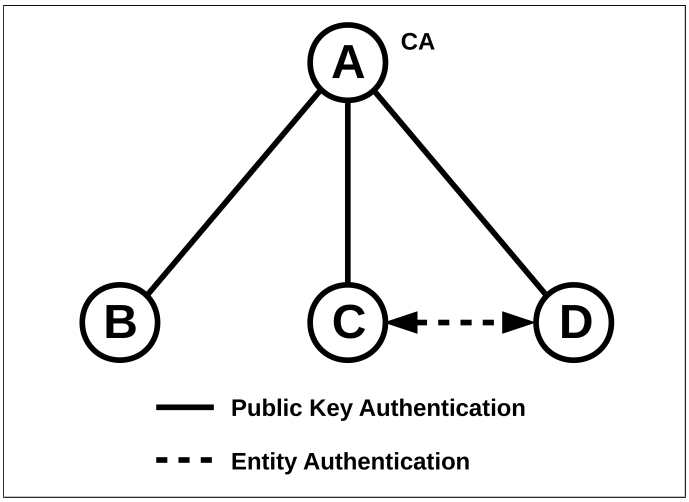
Chapter 6 concludes and summarises the research outcomes and results presented in this thesis. Future work derived from this research is also discussed.

Chapter 2

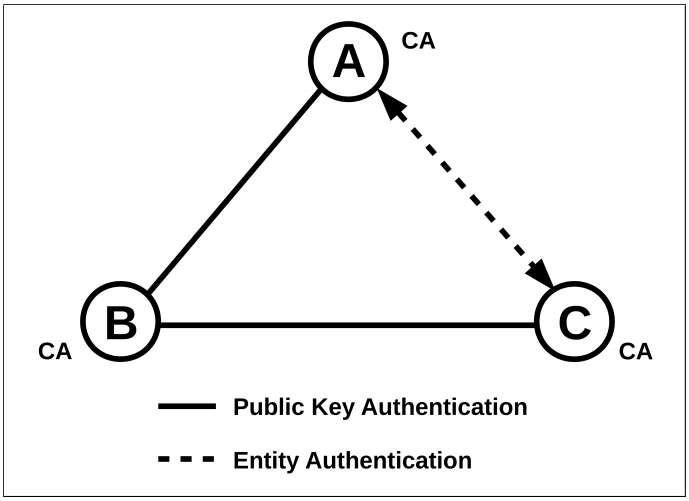
Background and Related Works

The primary research aim of this thesis, as described in Chapter 1, is to provide a complete decentralised key management scheme for autonomous Delay Tolerant Networks (DTNs). In particular, the provision of *Public Key Authentication* during the various key management phases. Public key authentication is the ability for an entity to verify the identity-public key binding of another entity. This concept provides the foundation for security in a DTN. Figure 2.1a depicts a traditional hierarchical Certificate Authority (CA) trust model for public key authentication. *Pre-established* trust in the CA (A) by all nodes (B, C, D) provides public key authentication, and by extension entity authentication between two nodes (C and D). In contrast, Figure 2.1b depicts a distributed Pretty Good Privacy (PGP) trust model for public key authentication. Initial trust is established between network entities by *humans*, and everyone is their own CA. Alice (A) and Bob (B) provide public key authentication to each other. Bob (B) and Carol (C) also provide public key authentication to each other. Entity authentication is achieved through Bob acting as the CA between Alice and Carol. The CA model in Figure 2.1a requires *pre-established* trust in a central CA, while the PGP model in Figure 2.1b requires *humans* to provide the initial trust establishment. However, there exists a case scenario for *autonomous* DTNs where neither *pre-established* trust or *humans* are available. Thereby creating the problem of public key authentication in an autonomous DTN as depicted in Figure 2.1c.

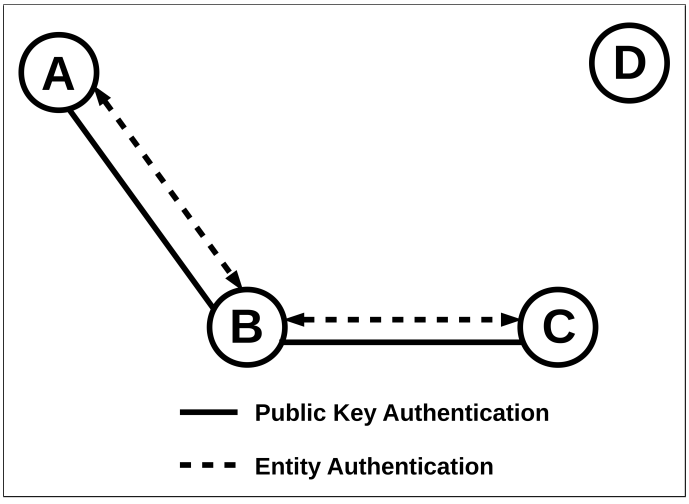
To provide public key authentication for the various key management phases



(a) CA - With Pre-established Trust



(b) PGP - With Human Trust Establishment



(c) DTN - With Autonomous Trust Establishment

Figure 2.1: Public Key Authentication

in an autonomous DTN, several important concepts are covered. First, essential background on the DTN environment, applications and routing are covered, as this provides the motivation of securing communications in a DTN. Autonomous DTN applications dictate how trust and reputation systems are utilised to provide adversary detection and general management of the network with no human involvement. An outline of existing Public Key Infrastructure (PKI) and how they provide public key authentication through different methodologies are also covered. Together, trust and key management are two important concepts in providing public key authentication. Finally, more specific DTN applications of trust and reputation systems as well as PKI schemes are reviewed to identify existing issues and research gaps for the contribution of this thesis.

This chapter outlines the background and related works, with additional specific literature expanded in Chapters 3, 4, and 5. Each of these chapters will continue the gap analysis specific to the contributions of each chapter. This chapter is organised as follows. Section 2.1 introduces the concept and characteristics of DTNs and how they differ from traditional networks. Issues such as network traffic routing, applications, and current and future challenges are also considered. Section 2.2 provides a background on trust and reputation systems and an overview on how they can be applied to autonomous DTNs. Different architecture and trust computation engines are also discussed. Section 2.3 covers background on two different types of existing PKI and how they provide public key authentication. Related work on more specific applications of DTN trust and reputation system and PKI schemes are discussed in Sections 2.4 and 2.5 respectively. From these schemes, Section 2.6 then presents the research challenges associated with providing the complete decentralised key management solution for autonomous DTNs. Finally the chapter is summarised in Section 2.7.

2.1 Delay Tolerant Networks

Securing DTNs is necessary due to their characteristics and potential applications in hostile environments. They consist of highly mobile nodes that are intermittently connected, often forming opportunistic ephemeral connections [8, 132]. These networks typically have very long delay network paths and lack a complete source-to-destination path [72]. In existing Transmission Control Protocol/Internet Protocol (TCP/IP) based networks such as the Internet, a number of

assumptions on the operation of the network are made. The first is that a persistent end-to-end path exists between the source and destination. The second is that the maximum round trip between nodes of the network is not excessive in duration, and the third is that the probability of end-to-end packet loss is small. One of the foundation papers in DTN research by Fall [40], categorises DTNs as networks that violate one or more of these assumptions.

Fall [40] compares DTNs to the Internet due to its scale and influence. DTNs have a high latency and low data rate in comparison. The data flow of the network is also asymmetric, with one direction of data exceeding the opposite data flow. In some instances, there may not be any return channel as evident in submarines, battlefield, or covert communications. Fall also discusses the state of disconnect between the nodes, in particular how end-to-end disconnection is more common than connection. This is evident in applications where nodes are highly mobile, and disconnection due to nodal movement can either be predictable or opportunistic. DTN applications where nodes are mobile and have a predictable disconnection include satellite passes with orbital movement, buses, trains and transportation with timetable scheduling. Applications where mobile nodes are opportunistic in their disconnection include aspects of randomness, such as nodes on a random path movement model, or taxis in a city. Fall also outlines that nodes can be disconnected due to low-duty-cycle operation in nodes which have limited power resources. Since DTN nodes spend more time disconnected than connected to other nodes, they also have long queuing times. Messages from the source node may take several magnitudes of time to reach the destination node. Due to this delay, it is also resource expensive to re-initiate transmission of the message from the source.

DTN applications are varied and widespread. With the rise of the Internet of Things (IoT), DTNs fall into a specific category of IoT devices and applications. The first are terrestrial mobile networks, where parts of the network are partitioned either by unintentional reasons such as interference, or from intentional and periodic reasons such as buses that act as data mules for remote villages. Such a network may exhibit typical characteristics of DTNs, such as frequent disconnection and high latency. Interplanetary space networks are a form of DTN, in particular the long distances of deep space networks cause long delay times, and potential interference from massive objects. Defence networks for battlefield deployment are a common form of DTNs, where soldiers are de-

ployed in an area with little or no infrastructure. Ad-hoc networks are created between the soldiers to provide communications and data between each combatant. Such a network may also face intentional signal jamming or Denial of Service (DoS) attacks from hostile forces. Covert operations or submarines may operate on an asymmetric data communications link [40] where messages are received. However, to comply with radio silence, no return message is sent. An important application of DTNs is of Mobile Ad-Hoc Networks (MANETs) and Vehicle Ad-Hoc Networks (VANETs), where people, mobile devices, and vehicles become nodes in the network.

The characteristics of a DTN are a contrast to the model of the Internet today. A network that mainly relies on persistent wired or wireless connections to provide a complete and continuous source to destination path. It has relatively fast round trip times, and a small probability of packet loss. Many papers such as Zhu et al. [136], Bhutta et al. [8], and Wood et al. [132] re-iterate the characteristics of DTN systems and emphasise that they are connection opportunistic in model. These characteristics in contrast to the Internet, result in a different methodology for securing and routing data communications in a DTN.

2.1.1 Routing

Secure routing in a DTN is important because the data flow and transfer model is different to a more conventional network such as the Internet. Due to this, different security considerations are required. The characteristics of DTNs as outlined in [40] is that a complete and reliable source to destination path between nodes is not always present. This can be due to node mobility, limited communications range, or physical obstructions [53]. Guo et al. [53] outlines that the common data delivery mechanism in any DTN takes the form of a *store-carry-and-forward* scheme, thereby exploiting the mobility of nodes. This method of routing is important in a DTN, as it forms the primary method of transfer for all data, ranging from messages, to public keys and Certificate Revocation Lists (CRLs).

Routing of data in DTNs can be categorised into two approaches [13]; proactive and reactive. Proactive routing protocols are when nodes store routing information to every other node in the DTN, while reactive protocols only initiate route discovery between two nodes when communication is needed. A more recent classification to DTN routing categorises them into the amount of knowledge used for routing [120].

Flooding based routing has little to no knowledge of the network, resulting in all nodes becoming relay intermediaries. Vahdat and Becker [127] proposed an early message routing scheme using an epidemic flooding model. This scheme results in high congestion in the network, but provides a quick source to destination route. Improvements to the flooding based routing were proposed by limiting the number of message instances in the network [125], and limiting the lifetime of message instances [62, 58].

History based routing relied on past interactions between nodes to provide additional information on routing. A node that meets the destination node of a message regularly and often is likely to meet again. Proposals such as PRObabilistic Protocol using History of Encounters and Transitivity (PROPHET) [88] utilised past encounters to determine the predictability of message delivery to each node, while the Fresher Encounter SearcH (FRESH) algorithm [34] forwarded messages to nodes that had encountered the destination node. Contextual information including time delay between meeting destination nodes [52], and mobility patterns of nodes [82] are also utilised to assist in DTN routing.

The addition of social relationships between nodes can help establish message routing [45]. Hui et al. [69] proposed leveraging social networks to reduce the source to destination hops for message delivery, by using the bubble algorithm. This establishes a route between the most popular nodes of both the source and destination social networks.

The final approach to data routing in DTNs is special devices. This approach includes the use of dedicated messenger nodes [133]. Nodes wanting to send a message push data to these messenger nodes, who are then responsible for delivering the message. Messenger nodes that specifically pull data are considered as data mules, such as trains and buses that pass access points and pull awaiting messages. Proposals that utilise messenger nodes may share messenger nodes between multiple DTN nodes [23, 128], or provide dedicated messenger nodes for each DTN node [53].

The various approaches to message routing in a DTN can be utilised for different messages. A flooding based approach would be best utilised for the dissemination of a blacklist of malicious nodes, or a CRL. The combination of history based routing and social relationships assists in trust and reputation, while the deployment of special devices in the form of static infrastructure Road Side Units (RSUs) for VANETs would assist in message routing. However, ir-

respective of the type data that is being routed, it is passed through multiple intermediary nodes as no persistent end-to-end connection is available, making DTNs highly susceptible to Man In The Middle (MITM) attacks. It therefore becomes important to secure communications and routing in DTNs.

Securing a DTN requires varying and different security considerations than securing traditional networks. The wide variety of DTN applications results in scalability issues. DTN schemes are required to be flexible and *scalable* to provide security from small unplanned and unmanaged deployments, to large scale vehicular deployments. Common between all DTN deployments is the general method of routing. Characteristics such as mobility and frequent disconnection result in a store-carry-and-forward routing scheme, which rely on the nodes to provide the network and communications infrastructure. This makes a DTN more susceptible to MITM attacks, as data bundles are passed between intermediary nodes. Key management is the common requirement for the security mechanisms (confidentiality, integrity, and message authentication) to remedy these attacks. Key management comes with two inherent fundamental issues:

1. Distribution, operation and revocation of public and private keys using PKI.

2. Public key authentication - verifying the identity-public key binding of another entity.

As a result, background on both *Trust* and *PKI* are covered in the next two sections.

2.2 Background on Trust and Reputation Systems

There is an increasing number of DTN deployments where the nodes operate autonomously, or with little to no human intervention. This is particularly pertinent in deployments with large quantities of nodes over large geographic areas, where the autonomous nature contributes to the ease of network operation and maintenance. In such applications, trust and reputation systems become an important security mechanism to allow autonomous operation of DTNs, particularly in routing, adversary detection, and potentially key management. Trust is a concept that is evident in daily human interaction. However, it is difficult

to define and describe as reflected by the varying publications that attempt to define the notion of trust and reputation [74].

Josang et al. [74] segregates the notion of trust, and the notion of reputation into individual components. This thesis will use these definitions of trust and reputation. Reliability trust as defined by Gambetta [43] is the subjective probability of individual A expecting another individual B of performing an action which its welfare depends on. However, Falcone and Castelfranchi [39] note that a high reliability in an individual may not lead to an action of dependence of that individual as risks are involved. To include risk based on the stake of the action, and the probability of failure, McKnight and Chervany [97] define trust as the following.

Trust *is the extent to which one party depends on another with relative security, even with the possibility of a negative outcome* [97].

In general, trust is the relationship that one party has of the other for certain actions involving risk. It is established through the dealings of the two parties. Trust systems produce a score or result that subjectively reflects the relationship between two parties. However, the notion of reputation builds on the notion of trust, and is defined as the following.

Reputation *can be considered a public aggregation and collation of trust of a party, from members of a community* [74].

Josang et al. establishes the difference between trust and reputation. A party may trust another party because of their good reputation. However, a party can still trust another party despite a bad reputation. Josang et al. distinguishes that trust is a direct, personal, and relational phenomenon and carries more weight over reputation. They also state that in the absence of a direct relationship between two parties, trust is formed through reputation, second hand relationships, and referrals. Reputation systems produce a public score or result of a party that is a culmination of the entire community. The decentralised nature of DTNs makes reputation systems difficult to implement. The lack of a persistent source to destination channel severely hinders the aggregation of trust scores from a variety of nodes to form a reputation. This operation is also communications intensive, resulting in high communications overheads just to aggregate trust scores, and subsequently distribute reputation scores. These reasons, make

the application of a trust system independently managed by nodes more suitable than an aggregated reputation system.

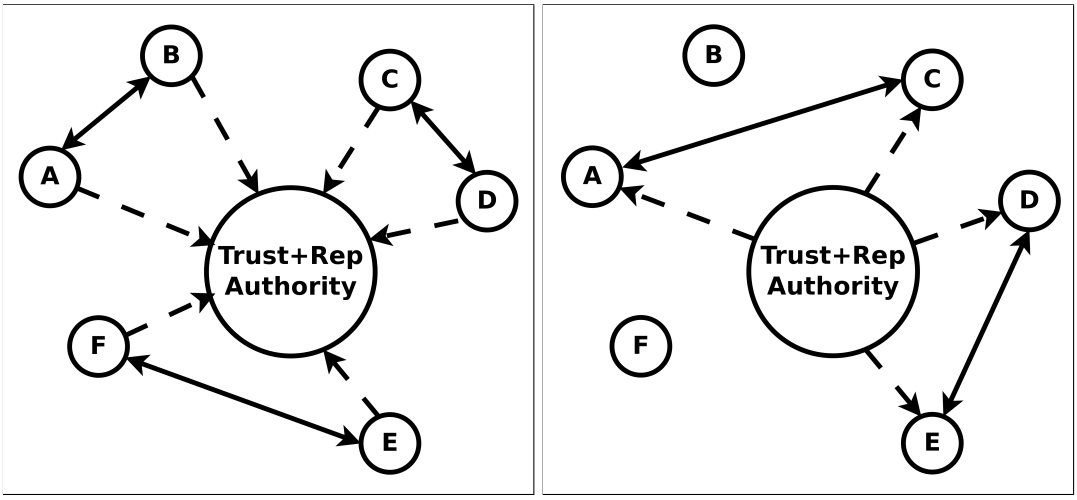
Rasmusson and Jansson [114] have used the terminology hard security and soft security to distinguish the differences in types of security mechanisms. Hard security is considered security techniques and mechanisms designed to prevent malicious users from accessing certain resources. These typically take the form of access controls or privacy in the form of encryption. However, in some instances, adversarial agents may be acting deceitfully by offering false and misleading information or resources. In these instances, social control mechanisms like trust and reputation systems can help prevent these scenarios. These control mechanisms are considered soft security [74].

2.2.1 Architecture

Trust and reputation systems can be categorised by architecture into centralised and decentralised systems. Both are explored to provide justification of their application in a DTN.

A **Centralised** trust and reputation system is where the performance of users are collected by fellow users and aggregated up to a central authority. Users are registered to the central authority, and feedback on user behaviour is submitted to the trust and reputation authority. The central authority or reputation centre collates the feedback and re-distributes the data publicly for other nodes to access. Figure 2.2 depicts the architecture of a centralised trust and reputation system with nodes undergoing transactions. In Figure 2.2a at time τ, nodes A and B are conducting a transaction as denoted by the bi-directional solid line, and each feedback to the central trust and reputation authority as denoted by the uni-directional dotted line. Nodes C and D, and E and F are also performing a transaction, and also respectively feedback to the central authority. At a later time denoted by $\tau + \delta$ in Figure 2.2b, nodes A and C wish to perform a transaction. They each individually poll the public results of the other party from the central authority. Node A pulls the reputation score of node C, and node C pulls the reputation score of node A. Nodes D and E also perform a similar process to nodes A and C. The reputation scores will then help determine whether the nodes will undertake a transaction.

A centralised architecture for trust and reputation systems provides a convenient and manageable method for enforcing rules and users. However, centralised



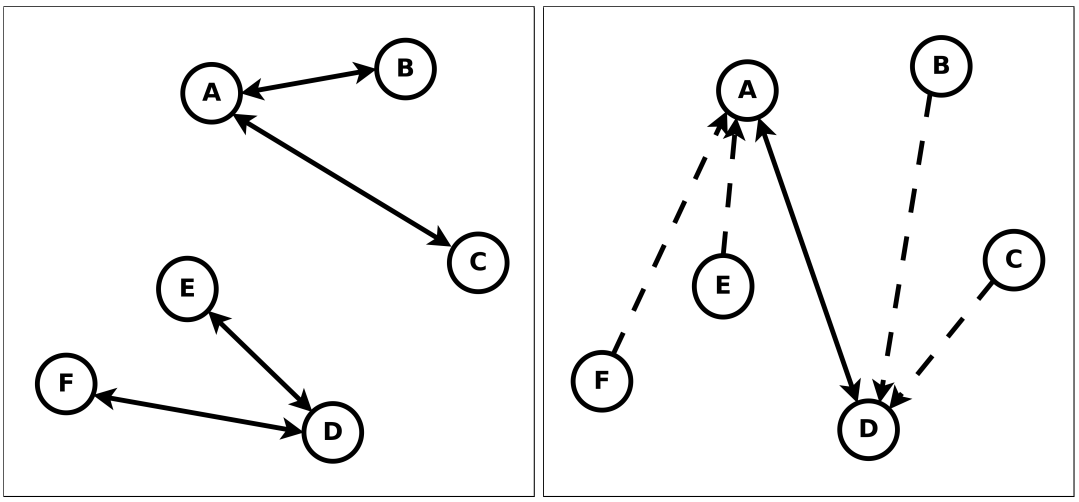
(a) τ (b) $\tau + \delta$

Figure 2.2: Centralised Trust and Reputation System.

systems form a potential single point of failure for both reliability uptime and security. Scalability is another issue, and can be difficult to implement in networks with a large number of nodes.

Unlike centralised reputation systems, which rely on a central node for computation and distribution of scores, **Decentralised** reputation systems rely on several nodes to measure, rate and individually aggregate scores of other nodes. A fully distributed reputation system requires each node to maintain scores of other nodes and make it publicly available to others on request. Each node formulates feedback on another node and retains the information. Another node may then request this feedback to formulate the reputation of another node, which may impact a pending transaction. Figure 2.3 depicts a decentralised trust and reputation system. At time τ in Figure 2.3a, node A performs a transaction with nodes B and C. Node A then holds the feedback of B and C, and nodes B and C each hold feedback on node A. Node D is also performing a transaction with nodes E and F, with D holding feedback on E and F, while nodes E and F holding feedback on node D. At a later time $\tau + \delta$ as depicted in Figure 2.3b, nodes A and D wish to perform a transaction. Since nodes B and C at time τ performed a transaction with node A, they provide feedback to node D about the reputation of node A. Nodes E and F provide the feedback reputation of node D to node A.

A decentralised trust and reputation system provides a more resilient architecture than a centralised model, as there is no central authority to exploit.



(a) τ (b) $\tau + \delta$

Figure 2.3: Decentralised Trust and Reputation System.

Unlike a centralised architecture, it is possible that nodes may refuse to provide feedback on other nodes. Due to the decentralised nature of nodes collecting and forming ratings of other nodes, it may be difficult to acquire all ratings of past transactions from the various nodes, therefore a reputation score in decentralised models is usually a subset of ratings acquired from neighbouring nodes [74]. In a DTN environment, neighbouring nodes are determined by the routing and mobility model of the nodes, resulting in the lack of a persistent end-to-end communications channel. Therefore, a decentralised architecture is more suited to a DTN deployment.

2.2.2 Trust Computation Engines

The computation engine is the algorithm or method used to derive a trust or reputation rating from feedback [74]. Several different theoretical and commercial computation engines that are relevant to DTN applications are reviewed in this section.

The simplest form of trust system is a dual state **discrete** trust system. In this system, either *the node is trustworthy* or *the node is not trustworthy*. Public key authentication is a form of a dual state discrete trust system, as keys are either *trustworthy* or *untrustworthy*. Additional discrete state models have been proposed by Manchala [94] and Abdul-rahman and Hailes [1]. Abdul-Rahman and Hailes [1] proposed a trust reputation model that determines the trustwor-

thiness of entities based on four discrete trust degrees; Very Trustworthy (VT), Trustworthy (T), Untrustworthy (U), and Very Untrustworthy (VU). Referrals or recommendations are also used in this model. Entities can use their own perception of the recommender to scale the recommendation of a third party entity. If the entity has had experience with the recommended entity, they may use the difference of trust to rate the recommender and adjust the trust settings accordingly. This method of trust computation engine is effective in applications where humans are involved, as the categorisation into the discrete trust states requires reasoning.

The **summation of scores or averaging** of scores is a simple form of computation engine employed for reputation systems. Summation can take the form of adding all the positive scores and subtracting all the negative scores. eBay's reputation system follows this principle. Resnick and Zeckhauser [118] provide a detailed explanation of how the eBay reputation system operates. Registered users with eBay can buy and sell anonymously, and prior to February 1999, buyers and sellers were given the opportunity to leave feedback after transactions. Comments and a numerical ratings of +1 positive feedback, 0 neutral feedback, and -1 negative feedback were allowed. After February 1999, eBay modified the system to tie any -1 negative feedback to a particular transaction, and in February 2000, eBay again modified the system to only allow any feedback tied to the transaction. The final metric score of buyers and sellers is the number of distinct users who left positive feedback, minus the distinct users who provided negative feedback. Past history is also considered, with scores from the past 7 days, 1 month, and 6 months publicly displayed. The eBay reputation system is a centralised system with eBay collecting and aggregating the feedback and distributing them to users of the website. An extension to summation of scores, is to average scores. Product review and commerce sites such as Epinions and Amazon average scores. Both use a similar system of a rating 1 to 5 stars, which is averaged to provide an overall score. Epinions further segregates the 1 to 5 star rating into categories such as delivery, customer service and ease of use [74]. Summation or averaging computation engines are effective implementations for a DTN trust system (Section 2.4), as nodes can manage the trust ratings of other nodes independently. An independent trust system provides nodes with information on whether to proceed with a transaction with another node without additional communications overhead that comes with a reputation system.

Flow models are systems that compute trust through iterations or looped through long chains [74]. Flow model systems such as PageRank [105], assign a constant trust weight for the entire system. The system score is distributed throughout the nodes, such that the trust and reputation score of a node can only increase at the cost of a decrease in trust value of other nodes [74]. PageRank works on the principle of looking at the number of incoming hyperlinks that point to a webpage, which increase the ranking, and the number of outgoing hyperlinks from the webpage, which decrease the ranking. The combination of PageRank scores for all the websites total the system score. Even as new webpages are created, the total system score remains constant with individual PageRank scores being re-adjusted. PageRank is a centralised trust and reputation architecture, with the central authority re-adjusting scores to maintain the constant system score. It requires a holistic view of the entire network to derive values for trust and reputation.

An extension of PageRank is the flow model trust and reputation system EigenTrust by Kamvar et al. [75]. EigenTrust is a reputation system implemented on decentralised Peer to Peer (P2P) networks. It assigns a unique global trust value to each node on the P2P network, derived from the past history of uploads. The system normalises and aggregates scores from various other nodes, either centrally or distributed amongst nodes. The applicability of this computation engine in DTNs is hindered by scalability. The scheme requires the distribution of a collation of local trust values into a combined table, which is distributed amongst nodes. This results in high communications overhead as the quantity of nodes in the DTN grows.

This thesis explores a trust scheme that uses a combination of both threshold, and summation or average computation engine to effectively form a discrete dual state trust system for autonomous applications, where there is no human involvement. Trust scores resulting from the summation or average computation engine that are below the threshold would be considered untrustworthy, whilst scores above the threshold result in a trustworthy state. Since the trust scores are local and relevant to the individual node, this scheme provides scalability and no additional communications overhead. As a result, this scheme is further explored in this thesis to provide the dual state outcome of public key authentication in a DTN with autonomous nodes, where there is no human intervention.

2.3 Background on Key Management through PKI

PKI is a procedure of attesting the binding between the identity of a person or entity and the corresponding public key. In some instances, the entities involved may have never physically met or interacted with one another. It also provides a method of distributing public keys to entities for confidentiality, data integrity and message authentication operations [44], as well as revocation of public keys. This is called a public key management lifecycle, which is illustrated in Figure 2.4. The stages of key management that are relevant to this thesis are *distribution*, and *revocation* and *replacement*, which are highlighted in Figure 2.4. In this section, two methods of public key authentication are categorised based on trust. The first is a *Hierarchical* trust approach such as X.509 relying on a Trusted Third Party (TTP) called a CA. The second is a *Transitive* trust approach such as OpenPGP and Web of Trust, where trust is distributed.

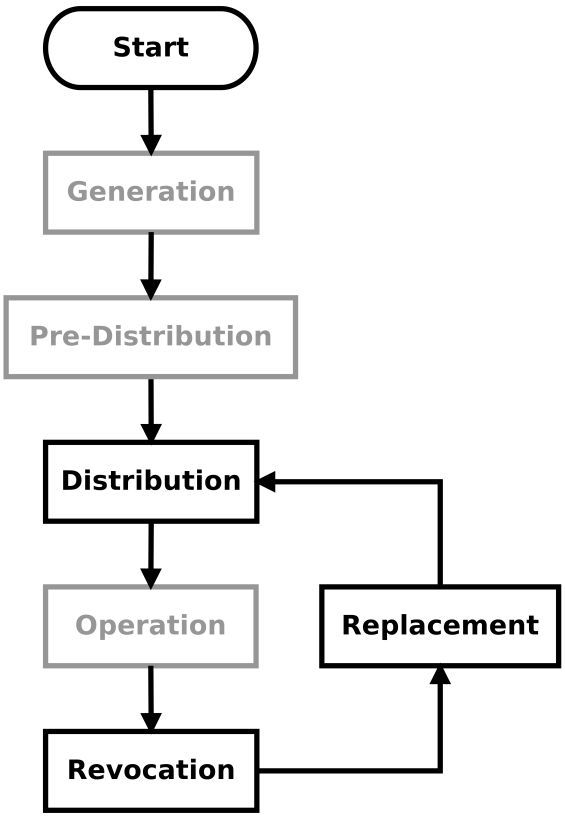


Figure 2.4: Public Key Lifecycle.

2.3.1 Hierarchical PKI (with Hierarchical Trust)

The most common form of hierarchical PKI is a centralised TTP CA model. The CA is able to provide a binding between the public key and the identified user, since it is a mutually trusted entity in the network. There may be either a single entity or multiple entities providing this role. The application of a CA for the Transport Layer Security (TLS) protocol in the X.509 system will be described.

A *public key certificate* or more commonly *certificate*, is a digital document that proves ownership of a public key. In a typical X.509 PKI system, the public key contains information about the identity of the owner and is also signed by an entity who has verified the integrity of the information provided.

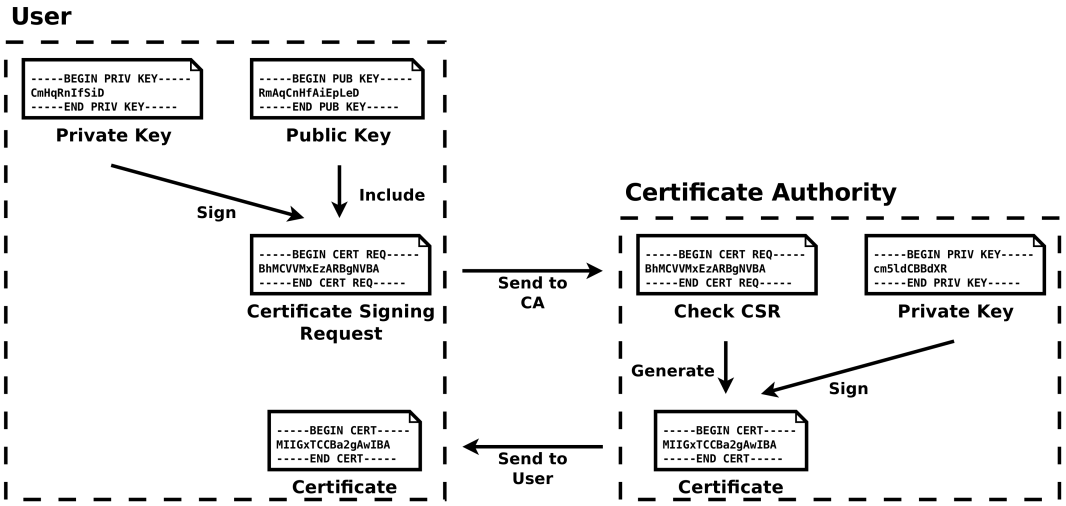


Figure 2.5: Certificate Registration Process.

The certificate registration process is shown in Figure 2.5. The user generates a Certificate Signing Request (CSR), which includes their public key, and is signed by their private key. The CSR is sent to the CA, where it is checked. The identity of the user is checked with the identity stated on the public key within the CSR. If CA is satisfied with the validity of the CSR, a certificate is generated and signed with the private key of the CA signifying their assertion of the identity with the public key. The certificate is then sent back to the user. Certificates provide an assertion by the CA that the identity-public key binding is authentic. These public keys are then used to provide additional security operations such as confidentiality, data integrity, and message authentication.

Because the public and private keys are critical to providing additional security properties, public key authentication during the process of key exchange

and distribution is important. The use of a centralised CA provides public key authentication during the key exchange process. The CA provides assertion that a users public key actually belongs to the user by signing the public key with the CA private key to generate a certificate. The identity-public key binding represented by the certificate can be verified using the CAs public key and returns a boolean value of either *trustworthy* or *untrustworthy*.

Reliance on a single CA to provide public key authentication can lead to a single point of failure in trust establishment. Due to this, the X.509 system utilises multiple intermediary CAs, which are able to issue certificates whilst a Root CA remains offline. The use of intermediate CAs is due to the fact that if a CA is compromised, every certificate issued is also compromised. For a single tier architecture, where there is only a single CA (the Root CA) issuing certificates, compromise of this CA would require a complete revocation of all valid certificates issued. By utilising intermediate CAs, only the certificates issued by the compromised CA has to be revoked. A new intermediate CA with the trust establishment from the Root CA can be formed to replace the compromised CA.

Figure 2.6 depicts the hierarchies and chain of trust in a CA based PKI. At the top of the hierarchical tree, the Root CA issues certificates for intermediate Tier 1 CAs. The Tier 1 CAs issue certificates for additional intermediate CAs that make up Tier 2. It is these Tier 2 CAs that issue certificates to users Alice and Bob. Public key authentication is achieved using the Certification Path Validation process defined in RFC 5280 [24]. If Alice does not hold the public key of the Tier 2 CA that issued Bob's certificate, a chain of multiple certificates may be needed up to the Root CA to provide public key authentication of Bob's public key. Because Alice and Bob have certificates issued from different Tier 2 CAs, and also Tier 1 CAs, they would only need to have the public key of the Root CA to be able to perform public key authentication on each others public keys.

A CA architecture based PKI has the benefit of providing hierarchical public key authentication. A multi-tiered PKI allows intermediate CAs to issue certificates to end users, whilst keeping the Root CA offline. This results in a single trusted authority that can be common to all entities of the network. Conversely, CAs present a single point of failure and the are capable of being compromised [41], with large quantities of certificates being compromised. The specifications

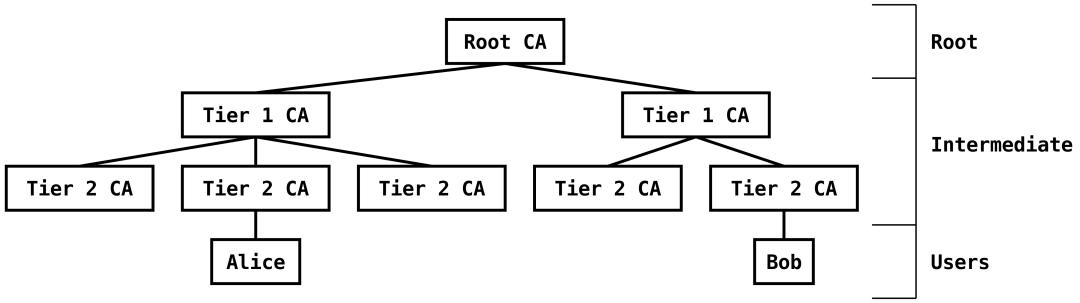


Figure 2.6: Hierarchies and Chain of Trust in a CA based PKI.

also do not specify certain scenarios such as the revocation of root certificates [54, 47, 70].

Certificate Revocation in CA architecture based PKI is achieved using one of two methods of blacklisting. The first is the distribution of a CRL to all users. The CRL is a list of serial numbers of revoked certificates that have been revoked. The size of the CRL file is a weakness of the CA based PKI. With a large number of entities in the network, the CRL file can grow quite large, creating difficulty in distributing the file [54]. Currently, the SANS institute uses the Global Sign CRL list [49], which as of July 2015 contained approximately 136,000 serial numbers of revoked certificates totalling to a CRL file size of 5.0 Mb. These include certificates revoked over a 4-year period since June 2011. Assuming the conservative estimate of a Vehicle to Vehicle (V2V) CRL file size to be 5 MB, under the Dedicated Short Range Communications (DSRC) protocol [85], each car would require 1.5 to 13 seconds to receive the CRL. Therefore, PKI implementations using CRLs for certificate revocation are communications heavy [101]. The availability of a CRL is also an issue for the X.509 system. The client web browser may only trust a certificate if a CRL is available. This potentially provides an attacker with DoS capabilities. CRLs typically have a short validity period to ensure freshness of data. When they expire, the client or user will poll the CRL distributor for a new or fresh CRL to prevent a replay attack using an old list. However, if a new CRL cannot be obtained before the previous one expires, the entity will be subject to a DoS attack. The CRL in this instance provides the only method of determining the validity or authenticity of an unexpired certificate. As a result, operations involving that certificate, such as verifying the authenticity of a message signed by that certificate cannot take place. Conversely, to prevent this loss of functionality when offline, the

client may be configured to automatically trust certificates regardless of whether a CRL is available. An attacker controlling the communications channel may prevent the distribution of the CRL to force a client to automatically trust a certificate. These two scenarios highlight the failing of a certificate revocation process dependent on CRLs, particularly when clients are offline. The frequent disconnected state of DTN nodes makes CRL distribution infeasible.

The second method of certificate revocation is Online Certificate Status Protocol (OCSP), which is outlined in RFC 6960 [121]. OCSP requires the client to poll a server to verify the validity of a certificate. Taking the example of Alice and Bob, Alice suspecting the Bob's private key may be compromised, Alice polls the CA who issued Bob's certificate with an OCSP request. The OCSP request contains data including protocol version and the target certificate identifier in the form of the certificate serial number. The OCSP server responder checks the request and returns the status of the certificate with either *good*, *revoked*, or *unknown*. OCSP certificate revocation has the advantage over CRLs as the OCSP request and response contains less information then a CRL. However, OCSP based certificate revocation requires clients to have a connection to centralised infrastructure such as the CA to query the status information of a certificate. Due to the requirement of an online, easily accessible and available OCSP server, this solution becomes unsuitable for application in a DTN or MANET.

Another issue with OCSP revocation is the possibility of two forms of a MITM attack. The first is a replay attack by an intermediate malicious party. Because the OCSP request and respond channel is not authenticated, this implementation is vulnerable to a replay attack by someone who replays an certificate valid OCSP response before the expiration date, but after a certificate has been revoked. This can be mitigated by the inclusion of a cryptographic nonce. However, with CA OCSP responses typically having a validity period of several days, and most OCSP infrastructure not supporting cryptographic nonces, the replay attack is still a vulnerability in the OCSP revocation implementation. The second MITM attack is compromising a server's private key, and the attacker taking the position of a MITM. This position enables them to interfere with OCSP queries. DTNs are particularly susceptible to MITM attacks, due to the store, carry, and forward method of routing employed.

The use of a centralised PKI such as a CA provides an efficient method of public key authentication. However, many of the efficiencies and benefits from

having a centralised TTP are negated by characteristics of a DTN. The decentralised nature of nodes in a DTN, along with the lack of a persistent end-to-end communications channel hinders centralised public key authentication and certificate distribution. Many DTN applications such as battlefield networks rely on no pre-existing infrastructure, thereby entirely ruling out centralised schemes. While the role of a CA can be delegated to a specific node, the potential for compromise is high due to the threat model of a persistent adversary. The compromise of the centralised TTP in a DTN undermines the entire security of all network entities. Similar issues extend to the certificate revocation phase. The distribution of an ever-growing CRL may become prohibitive in severely restricted bandwidth networks. The reliance on the freshness of the CRL, along with the highly disconnected and fragmented nature of a DTN where nodes may be offline for long periods of time, would make using a CRL for certificate revocation difficult. The alternative certificate revocation method of OCSP addresses the size of CRLs, but is susceptible to the two forms of a MITM attack due to DTN routing. Along with the highly disconnected and fragmented nature of DTNs, dependence on a OCSP response for the validity of a certificate may be problematic and unreliable. Although various proposals in Section 2.5.1.1 and 2.5.2.1 attempt to specifically address DTN key distribution and revocation issues respectively, decentralised PKI models such as OpenPGP are better suited for the application of DTNs.

2.3.2 Distributed PKI (with Transitive Trust)

OpenPGP and Web of Trust is the most common form of transitive trust PKI. It is a decentralised method of providing public key authentication during key distribution, and revocation. This makes it highly suitable for application in a DTN environment. The OpenPGP protocol [14] or more commonly known as PGP is an open standard for providing privacy and authentication for data communications. It employs core technologies such as digital signatures, encryption, compression and Radix-64 conversion [14]. Created in 1991 by Phil Zimmermann, it is predominantly used to secure e-mail communications.

Similarly to the CA model, the public and private keys in PGP are critical for providing additional security properties. Therefore public key authentication during the process of key exchange and distribution is important. In an PGP system, the key exchange process consists of distributing the public keys of the users

in the network who wish to communicate. The issue in any key exchange scheme is public key authentication. The lack of public key authentication provides an adversary with the capability of performing a key spoof attack, particularly between users who have no a-priori knowledge of each other. An attacker Eve may perform a key spoof attack between Alice and Bob, exploiting the lack of public key authentication. Eve pretends to Alice that she is Bob, whilst pretending she is Alice to Bob. Any messages passed through Eve protected with the spoofed keys is subject to a MITM attack.

PGP provides a framework for providing confidence between users and their corresponding public keys. A PGP identity certificate containing the public key and owner information can be signed by other users. This endorses the binding between the public key and owner information, creating a Web of Trust of keys and users. For two users to exchange public keys as well as verify their identities, the following process occurs. The two users Alice and Bob obtain each other's public keys either directly from each other or from a public key repository. Alice meets Bob so that Bob can verify that the public key fingerprint Alice has matches his public key. Alice also verifies Bob's identity with a government issued photo identification. The reciprocal steps are conducted for Bob to verify Alice's public key.

For larger number of users, key signing parties where public keys are presented in person, verified, and then signed by the participants can be held. Key signing parties with a large number of participants do not scale well using the single verification method due to the multiple signing of keys from multiple parties, as well as verifying multiple identities [122]. To provide a more efficient mechanism of key signing, key signing parties employ the Zimmermann-Sassaman key-signing protocol [122]. Prior to the key signing party, the key signing party co-ordinator will collect public keys of the participants. A fingerprint (unique identifying string) database of public keys is compiled and distributed to the participants. At the party, the fingerprints are displayed, with the owner attesting to the correctness. After all keys have been displayed, users individually verify the identity of each owner by using forms of government issued photo identification. After the party, participants sign the public keys they verified during the party and distribute these instances back to a keyserver or the user.

The Web of Trust model is built on the transitive trust principle. Public keys of each node are exchanged directly with users forming the highest level of trust

- Direct Trust. The public keys are easily verifiable as they were transferred by the user owning the corresponding private key. Users may receive public keys of other users through keyservers or indirectly through friends. This forms indirect trust relationships between users much like the principle of: *If Alice trusts Bob, and Bob trusts Carol, then Alice can indirectly trust Carol.* Figure 2.7 depicts the trust relationship between users. Alice (A) has a direct trust relationship between Bob (B), Carol (C), Dan (D), and Erin (E) as it has previously directly met these users. An indirect trust relationship is formed between Alice and Xavier (X), as Carol, Dan and Erin provide enough endorsement between Alice and Xavier. Conversely, Alice does not form an indirect relationship with York (Y) as there is insufficient endorsement from Bob and Carol.

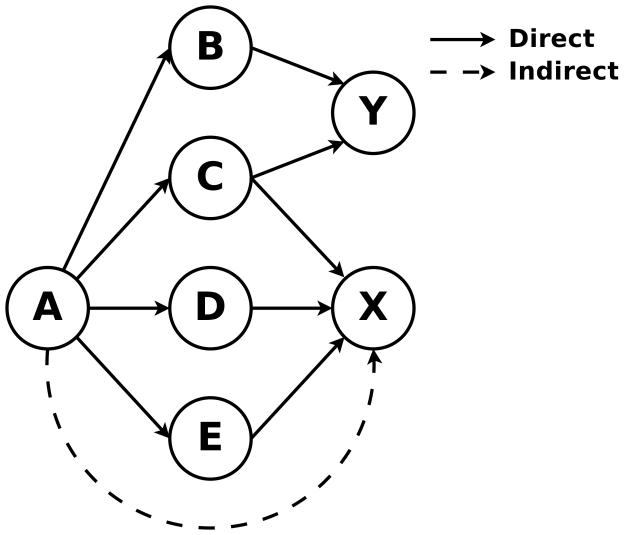


Figure 2.7: Trust relationship model between nodes.

The level of trust a user has in the confidence of another PGP users key is expressed through discrete states. There are four levels of trust in ascending order; Do NOT Trust, Trust Marginally, Fully Trust, and Ultimate Trust. An additional discrete state of Not Applicable is also available. These trust settings require human intervention to define how each individual differentiates between the categories such as Trust Marginally and Fully Trust. Rules can be set to allow any number of marginally trusted key endorsers be required to trust a key.

PGP implementations have a provision for a vote or threshold based trust scheme [14]. This implies the transitive trust principle. An example is when Alice is determining whether to trust Bob's public key. Bob's public key has been

signed by Carol, whom Alice has designated as fully trust. Since Carol is a fully trusted endorser, and has endorsed Bob's key, Alice may through the transitive trust principle trust Bob's public key. Partial or marginal trusted endorses can also be used. Alice, determining whether to trust Dan's public key, sees that it has been endorsed by Erin, Frank, and Gina, all whom Alice has assigned as partially trusted endorsers. Having three partially trusted endorsers, Alice may now trust Dan's public key. In this specific OpenPGP implementation, the fully and partially trusted threshold is 1 and 3 respectively. The full and partial thresholds are flexible and adjustable, with the GNU Privacy Guard (GPG) implementation using 1 and 3 [25], while PGP 2.6 using 2 and 4 [137].

Key revocation in PGP implementations is achieved using revocation certificates. Revocation certificates signify a message to cease use of the specified public key in the future. However, the public key can still be used to verify the authenticity of a message sent prior to the distribution of the revocation certificate. PGP Revocation certificates are generated whilst the user still retains sole control of their private key, and needs to be stored in a secure place for future use. The PGP standard provides several reasons for revocation, they include; No reason specified, Key has been compromised, Key is superseded, and Key is no longer used. The issue with pre-generated revocation certificates is secure storage. This is a particular issue for DTN nodes, where the threat model makes them susceptible to *physical tampering* by an adversary, resulting in the potential for false revocation. A revocation certificate also provides no trust transferral between the old and new keys. Unlike the centralised PKI model, where the new key is issued by the CA and is deemed trustworthy, the entire Web of Trust attached to the old key is lost when the revocation certificate is released. The new key would require the Web of Trust to be manually rebuilt.

PGP implementations provide a flexible trust framework to achieve public key authentication during key distribution and revocation. Unlike a centralised PKI architecture, the individual users are responsible for their trust decisions on whether to certify or trust a key. Each user effectively becomes their own CA. The decentralised nature of this architecture also distributes the risk to the individual users and is better suited for a DTN threat model where persistent adversaries exist. An adversarial agent undertaking a key spoof attack would have to compromise a significant number of users to get the spoofed key falsely certified. Public key authentication through thresholds of fully or marginally

trusted keys is attractive for DTNs as it provides network entities, who have never met, the ability to establish end-to-end communication channels. This also assists public key distribution.

However, a PGP implementation detailed in this section will not work in a DTN environment. PGP is heavily dependent on human intervention for assigning initial trust to public keys. The strength of the Web of Trust results in an issue pertinent to DTNs. DTN applications such as MANETs and VANET are open and dynamic, allowing nodes to leave and join freely. New nodes joining the DTN will have difficulty in joining an already existing Web of Trust, as existing users using the vote or threshold based trust scheme [14], will require several fully or partially trusted endorsers before trusting a new unknown public key. Public key authentication operations are also computationally and communications intensive, which are restricted on DTN nodes. Similar problems for public key authentication during key distribution are also present for key revocation. Public key authentication during revocation is achieved by using a decentralised method of distributing revocation certificates. These certificates signify the cessation of public key use. Something that is particularly important when the key has been compromised. Due to this, the revocation certificate has to be pre-generated, whilst the user still retains sole control of the private key. The certificate has to be stored securely. The threat model of a DTN makes nodes susceptible to *physical tampering*. As a result, an attacker may obtain the pre-generated revocation certificate of a node, and trigger a false revocation. Although various proposals in Sections 2.5.1.2 and 2.5.2.2 attempt to specifically address DTN key distribution and revocation issues respectively, many schemes are computationally expensive, rely on nodes with elevated privileges to perform special key management duties, or rely on a-priori knowledge for initial trust establishment.

2.4 Existing DTN Trust and Reputation Systems

This thesis explores the use of a trust or reputation system to provide public key authentication for all stages of key management. However, many previous applications of trust and reputation systems for DTNs specifically focus on past historical behaviour of nodes to provide *adversary detection* [5, 6, 135] or *optimal message routing* [83, 21, 22]. In particular, many assume an adversary model

where there is a persistent attacker in the network, with the intention of excluding them.

Proposals such as Ayday et al. [5], and Ayday and Fekri [6] utilised an iterative trust and reputation mechanism to detect a physically captured and controlled legitimate node. The capabilities of the captured node include dropping or modifying packets from other nodes, and injecting packets to cause a DoS attack on the network. In the proposed scheme, nodes known as raters, evaluate peers who perform a service (service providers) in the DTN based on past behaviour. The trustworthiness of rater nodes are also evaluated to prevent gaming of the trust and reputation system. Using an iterative process of determining reputations of both rating nodes and service providing nodes, the proposed scheme does not rely on a central authority. A major shortcoming of this proposal is the sole reliance on the history of Quality of Service (QoS) of routing. An adversary could be capable of behaving in a way to avoid detection such as not injecting or modifying packets. However, they would remain undetected during a key spoof attack, which leads to compromising the security of the network.

To assist with QoS dynamic message routing in a DTN, Chen et al. [21, 22] proposed the use of a trust and reputation system that combines past historical data of both QoS and social trust to obtain a composite trust metric. The composite trust metric is used to assist the DTN to facilitate dynamic trust management for routing, particularly in changing conditions at runtime [22]. The social trust component is comprised of honesty and unselfishness, and is considered for node trustworthiness for message delivery. While the QoS component is comprised of connectivity is considered for the node capability of delivering the message in a timely manner. The trust relationship between two nodes is assumed to be linear and computed using a summation of weighted averages of connectivity (QoS component), honesty, and unselfishness (social components). The trust score node i assigns of node j at time t is given by Equation 2.1 [21]:

$$T_{ij}(t) = w_1 T_{i,j}^{e-connect}(t) + w_2 T_{i,j}^{d-connect}(t) + w_3 T_{i,j}^{honesty}(t) + w_4 T_{i,j}^{unselfishness}(t) \tag{2.1}$$

Where, w_1, w_2, w_3, w_4 are weightings assigned to each trust component. Trust components $T_{i,j}^{e-connect}$ and $T_{i,j}^{d-connect}$ are associated with node i connectivity to node j, and node j connectivity to destination node d respectively. $T_{i,j}^{honesty}$

represents node i opinion of the honesty of node j, and $T_{i,j}^{unselfishness}$ is the node i opinion of the co-operation of node j.

Two versions of the proposed trust management systems were evaluated in [21] to the epidemic flood routing proposed by Vahdat and Becker [127]. The first was an equally weighted QoS and social trust component trust management system. This version provided ideal performance for the delivery ratio between nodes. The second was entirely QoS component, with no social trust component, which provided ideal performance for message delay. The linear trust computation engine employed in [21], shows that it is suitable for application in a decentralised DTN.

Various trust management systems specifically for the application of VANETs have also been proposed. The issue of scalability is a significant issue to VANETs, as the potential number of nodes is high, and the deployment environment is large. As a result, many rely on infrastructure in the form of RSUs or appointed cluster head nodes. The reliance on pre-existing deployed infrastructure and knowledge of the network is also assumed. This may be valid in densely populated areas, however, would not be valid in remote locations, or developing countries where infrastructure is non-existent or unreliable.

Golle et al. [50] proposed a data based trust model to detect and correct malicious data in VANETs. Nodes hold a model of the VANET, which is used to provide data validation. Data is collected with the possibility that adversary nodes are present, this is then compared to the VANET model held by each node and a score is produced. Data is accepted as valid if the data is consistent with the model. Although this proposal does not rely on infrastructure, it still does not satisfy the definition of self organising, as there is a global knowledge shared amongst nodes [129]. Raya et al. [116] also proposed a data based trust model for data and event reporting. This proposal was focused on the trustworthiness of data reporting rather than the trustworthiness of the entity. As a result, trust was formed based on events and was ephemeral for the data generated by the event, not the entity reporting it. Although this provides a trust management system for event based data reporting, entity based trust management is also important. Both proposals presented focus on event based reporting and the history of service provision. However, they do not attempt to address entity based trust management or public key authentication.

Dotzer et al. [33] proposed a self-organising trust establishment scheme for

VANETs, which can be extended for public key authentication. They proposed the distributed Vehicular Ad-Hoc Network Reputation System (VARS) that combined both data and entity trust. They make use of opinion piggybacking for validating received messages. Nodes in the network append their own opinion of the trustworthiness of the forwarded data, while also deciding whether to trust the information based on the opinions of previous forwarding nodes. Location and distance of the node reporting the event, and the node receiving the information also contributes to the decision of the trustworthiness of the data. Several shortcomings of VARS are outlined in [77, 68, 3]. Even though VARS will consider the previous opinions that are appended to the data bundle being forwarded, the opinion of earlier nodes will be repeatedly and recursively considered as it continues to be forwarded [77]. VARS does not address the initial bootstrapping and updating of trust values. The forwarded messages are susceptible to tampering, with no authentication of the node initially reporting the message. The cost of communications bandwidth is high in large ephemeral networks with numerous nodes, as nodes have to forward many opinions [68, 3]. This shortcoming affects the scalability of deploying VARS.

To address the scalability of deploying VARS, Chen et al. [20] proposed a VANET trust modelling framework that improved network scalability by reducing communications bandwidth utilisation. They utilise a cluster based data routing mechanism that collects and sends peer trust opinions regarding a message sent by the originator and the message itself. Nodes are geographically grouped into clusters, and from each cluster, a node is randomly selected to act as the cluster leader. They assume that there is a pre-established co-operative link between the cluster leaders to provide an intra-cluster link. The peer-to-peer trust opinions aggregated by cluster leaders are a combination of role-based and experience based metrics. Role-based trust are fixed by an offline central authority and is assigned to a small number of nodes that have specific responsibilities in the traffic system (police cars, traffic controllers and public services). Experienced-based trust is for nodes without a role and is based on the behaviour of the node as evaluated from other nodes in a range of [-1, +1]. Each node holds a list of trust in their local repository. Experience trust is dynamic and scalable and can increase for correct decisions, and can decrease for incorrect decisions. The computation of experience based trust is linear with the number of times receiving trust opinions from a node. Although this proposal improves the net-

work scalability by using a cluster based data routing mechanism, it heavily relies on the cluster leaders that are selected in each geographical clusters. As VANET nodes are highly mobile, the system would need to ensure that cluster leaders were distributed geographically to ensure sufficient dispersement of leaders. Mobile cluster leaders leaving the geographical clusters may require other nodes to be randomly selected to act as cluster leaders, thus re-establishing the intra-cluster link proposed. This scheme is infeasible in a network deployment without *a-priori* knowledge.

Patwardhan et al. [108] proposed a data intensive reputation management scheme. It provided a bootstrapping process to build trust relationships between nodes with self-generated persistent identities using data validation. They consider data to be trustworthy either if the data source is considered trustworthy, or if there are multiple copies from distinct multiple sources. Data received from a primary source directly is considered trustworthy, while as the device moves further away from the primary source of data, it becomes more unreliable. The availability of data from multiple copies from distinct multiple sources mitigates the risk of corruption and fabrication [108]. They also assume the system model of pervasive environments in urban or metropolitan areas, particularly constrained mobility. They also assume that persistent identities are crucial for association of reputation between nodes. Their proposal was simulated in a 700m by 900m area around DuPont Circle in Washington DC, with nodes generating a constrained mobility movement model to the Cartesian co-ordinates of the simulation space. They varied the number of nodes from 50 to 200 and used a communications range of 100m. The speeds of the nodes ranged from 15 to 25m/s, with pause times of 0 to 30 seconds and a total simulation duration of 30 mins. At 38 major intersections, anchor nodes were placed that are assumed to be pre-authenticated and providing trustworthy data, while the other nodes do not have any trust relationships with each other. Patwardhan et al. utilises an important concept of multiple copies of data from distinct multiple sources mitigates corruption and fabrication. This is a desirable concept for a decentralised and distributed trust and reputation system. However, the scheme was evaluated with a realistic vehicle simulation model on a small scale, and relied on a few pre-authenticated anchor nodes to bootstrap the trust management process. These results question the scalability of the proposal due to the reliance on infrastructure-like anchor nodes.

Trust and reputation scores in a majority of trust management systems reflect on past experiences such as behavioural and data validation. This creates the issue of how to bootstrap newly deployed and joining nodes into the network. Current bootstrapping processes either set an arbitrary trust value for unknown nodes or attempt to build a trust score around the actions of the node [3]. Abidin and Kolberg attempt to address the bootstrapping process by using social networks to establish trust [3]. Their proposal uses the social connections and links in social networks such as LinkedIn, Twitter and eBay between the drivers of vehicles to provide an initial trust level when a node joins a pre-existing VANET. Although this proposal assists in addressing the bootstrap problem when a node joins a VANET, it assumes and requires human driver interaction and the human social connections to provide the initial trust level, and therefore is not applicable to an autonomous VANET.

The unique characteristics and wide variety of DTN deployments results in a high probability that a persistent adversary will operate in a DTN. It is assumed that the removal of adversarial agents is infeasible. Therefore, a method for detecting and limiting the effect an adversary is capable is required. Trust and reputation systems provide a methodology for securing a DTN against such a threat. However, many proposals assume an application for established networks as they focus on past historical behaviour of nodes to provide adversary detection [5, 6, 135] or optimal message routing [83, 21, 22]. Schemes that deal with trustworthiness of data [33] are more concerned with the forwarding of data, and are also vulnerable to message tampering as there is no authentication mechanism provided for the forwarded messages. This leads back to the initial bootstrapping problem, which was attempted in [108] and [20]. However they were also dependant on pre-established links [20] or infrastructure [108]. As a result, they are unsuitable for an environment void of any infrastructure, and where nodes are deployed with no a-priori knowledge. Therefore, this thesis investigates the research gap of utilising a trust or reputation system for initial trust establishment during the key deployment phase of a DTN, without any *a-priori* knowledge.

2.5 Existing DTN Key Management Schemes

Past proposals for providing security in DTNs involves a form of cryptographic key implementation to provide confidentiality, data integrity, and message authentication. Many of these proposals have evolved from Wireless Sensor Networks (WSNs), and have focused on the use of symmetric keys [16]. Symmetric key implementations are suitable for devices with constrained resources as computation needs are significantly less than asymmetric key implementations [66]. Solutions include using common shared keys [7], pair-wise probabilistic [37, 19, 29], pair-wise deterministic [89, 15], and group keys [67]. However, given the mobility model of DTNs, neighbouring nodes change frequently making asymmetric key implementations more desirable. With DTN nodes in VANETs being cars, nodes may be considered to have significantly more resources than static WSN nodes.

Due to the differences specific in characteristics between DTNs and WSNs, public key management implementations are better suited for DTNs. Two stages in key management (Figure 2.4, Page 24) are important to the security of a DTN; *distribution*, and *revocation and replacement*. Distribution is primarily the dissemination of public keys or the issuance of certificates. Revocation and replacement is divided into two separate steps. Revocation, is concerned with the removal of public keys from use. This may be a planned revocation, where public keys have a finite validity period, or may be an unplanned revocation, where the private key has been compromised. Replacement, is concerned with the re-distribution of a new public key directly after a revocation event. There is an intermediary step in key management between distribution, and revocation and replacement known as *operation*. This is where the public keys are used to provide confidentiality, integrity, and message authentication for network traffic. Although this usually constitutes a long time period in the key management life cycle of a DTN, this thesis is primarily concerned with the key distribution, and revocation and replacement stages. In particular, the provision of public key authentication during these two stages.

2.5.1 Distribution

The provision of public key authentication during key distribution can be categorised into two main approaches [104]. The first is a *centralised* model where

a TTP such as a CA issues certificates, keys and other materials as outlined in Section 2.3.1. This is similar to the use of PKI as outlined in Perlman [109] and Zhou and Haas [134]. The CA may be a single authority or multiple authorities distributed in the network, and provides public key authentication for the keys. The second is a *decentralised* model where nodes are self organising, and key distribution is implemented through direct and indirect contact as well as trust propagation similar to the OpenPGP Web of Trust model as outlined in Section 2.3.2 [138, 2]. However, due to the nature of DTNs, some decentralised schemes proposed attempt to provide public key authentication by distributing the CA role. These schemes are not considered fully decentralised, but are included due to classification by various literature.

2.5.1.1 Centralised

Traditional CAs utilised for the Internet are not applicable for DTNs due to the ephemeral and opportunistic connections between nodes. Because availability of the CA cannot be guaranteed, centralised CA based implementations in DTNs take the form of multiple entity CA, where multiple nodes distributed in the network perform the role of the CA. The use of threshold cryptography [123] as a (t, n) threshold signature scheme [48] allows the CA signing key to be divided into n shares, and recovered from t components.

Zhou and Haas [134] proposed a coalition of designated nodes called server nodes who carried out certification operations in a DTN. The server nodes are configured by a trusted entity with the threshold values (t), and distributing the shares (n). When a node registers for a certificate, at least t of n server nodes must collaborate and combine the CA signing key with one of the server nodes. Extending the proposal of Zhou and Haas, Kong et al. [79] and Luo and Lu [91] proposed that all nodes of a network may act as a member of the distributed CA. This proposal increases the availability a node has to a distributed CA. However, it still relies on a trusted entity to configure thresholds and initial credentials during the bootstrapping process.

The necessary threshold number of server nodes (t) required for CA operations may not be available for all parts of the DTN. Therefore availability for CA operations may be intermittent [46]. Another security consideration is that a (t, n) threshold signature scheme only tolerates up to the threshold number of corruptions before CA operations are hindered, thereby causing a DoS attack.

Although this would be mitigated in the proposal of Kong et al. [79], where all nodes are a member of the distributed CA, a malicious node may still contribute falsified shares. The proposal from Kong et al. [79] also means that an attacker would only have to compromise the number of nodes equal to the threshold, instead of targeting specific certificate server nodes. CA operations are considered more communications intensive. Each certificate request requires communications with a minimum t nodes, with some communications being completed using multiple hops as characteristic of a DTN. This can significantly increase communications overheads as well as cause delays with certificate issuance. The computation required in implementing threshold cryptography schemes is also an issue [102]. The proposed scheme by Luo and Lu [91] found that generation of partial RSA signatures using a (t, n) threshold signature scheme is significantly more computationally expensive than standard RSA signing [26]. The increase in computational overhead also contributes to the delay and availability for certificate issuance.

Although VANETs are a type of DTN, they may rely on static infrastructure and more traditional CA schemes. Infrastructure based VANET trust establishment solutions such as [115] relied on static RSUs to allow both V2V and Vehicle to Infrastructure (V2I) communications. This proposal utilised certificates to provide identity and credential management with the RSUs acting as a gateway to a trusted third party CA. The use of a CA also allows for infrastructure based revocation of malicious nodes from the network. Kim et al. also proposed an infrastructure based trust management scheme. It utilises government organisations as a central server, and is reliant on trusted third parties for generating security keys and certificates [77]. The keys are used for authentication in the case of V2V communications, with identities requiring a renewal annually. Revocation of certificates is handled using the IEEE P1609.2 Standard CRL. Liao et al. also proposed a VANET trust model for incident reporting relying on V2V communications, while also taking advantage of V2I communications to static stations [86]. Although infrastructure based proposals such as [115, 77, 86, 55, 56] provide ease of management of identity, credentials and revocation process, they are heavily reliant on static infrastructure such as RSUs and trusted third parties such as CAs.

The key management framework proposed by Hao et al. [59, 60] relies on the distributed nature of RSUs to act as group private key generators for vehi-

cles within coverage. The proposed scheme when compared to a centralised CA scheme [77] provides several benefits. The first is key maintenance operations such as public key updating is easier and more flexible [59], as well as a more efficient revocation scheme as RSUs store certificate revocation lists. Location privacy of vehicles is also improved as group private keys are changed as vehicles change issuing RSUs. Due to the important role of group key generator, and distributed deployment of RSUs, they are attractive and susceptible to compromise. To enable a vehicle to determine whether it is being issued a group key from a compromised RSU, Hao et al. [60] propose the reliance on global long term public and private key pairs issued by a CA.

All centralised key management schemes proposed for DTNs and its application in VANETs rely on some form of CA to provide public key authentication for the keys distributed. The CA may be a single entity [115, 61, 130, 9], or multiple entities distributed amongst nodes [134, 79, 91]. However, they all require some form of trusted authority to empower and configure the centralised authority [18]. The inherent characteristics of a DTN make centralised key management schemes unsuitable. Even by distributing the CA role to multiple nodes through threshold cryptography will not work effectively in a DTN. The necessary threshold of nodes required for CA operations may not always be available. While threshold schemes (t,n) can only tolerate up to threshold (t) corruptions before CA operations are hindered. Both instances would result in either a temporary or more permanent DoS attack. Computational overhead with processing partial RSA signatures is also higher [26]. All these issues contribute as to why centralised key management schemes are unsuitable for autonomous DTNs. As a result, decentralised schemes that are not reliant on infrastructure are better suited to DTNs. However, some decentralised schemes still rely on some form of infrastructure, and thus cannot be considered fully decentralised.

2.5.1.2 Decentralised

Decentralised key management solutions assume nodes are self-organising, that create, store, distribute and revoke public keys without the dependence on a trusted authority [18]. Decentralised schemes are typically based on the OpenPGP model for PKI. Nodes distribute their own public keys whilst signing other nodes public keys to provide public key authentication in a Web of Trust model.

Capkun et al. [18] proposed a Web of Trust based self-organising public

key management scheme for MANETs. Public keys are distributed and trust is formed through physical contact between nodes. Each node generates a public and private keypair with a limited time validity period. They sign and issue public key certificates to other nodes that it trusts. An issuing node will hold in their local repository certificates they have issued and certificates issued to that node. From this process, a certificate graph is created. The next step is certificate exchange, where nodes will periodically exchange certificates. Nodes multicast to neighbouring nodes a subset of the certificate graph created during the certificate issuing process. This information is stored in the non updated certificate repository, which contains expired certificates that are not updated [18]. The non updated certificate repository provides the node with an estimate of the certificate graph. This is suitable for the scheme as it is reasonable to assume most certificates will be renewed by issuers rather than be revoked. Finally, the updated certificate repository is constructed by either communicating with the nodes certificate graph neighbours or by the repository graph construction algorithm [18]. Public key authentication is achieved through chains of trust paths established from the certificate graphs. The scheme proposed by Capkun et al. allows certificate generation, exchange and authentication to be conducted automatically, whilst certificate issuing and revocation operations need to be conducted consciously by a user [18].

The scheme proposed by Capkun et al. [18] presents significant disadvantages as the scale of the DTN increases. Computation and communications efficiency is an issue. Verification of a certificate in Capkun et al. [18] requires the verification of multiple certificates to establish the chain of trust. On constrained devices, each verification step is computationally intensive. The communications overhead for certificate distribution and issuing significantly increases as the number of nodes increase. To reduce the overhead, Li et al. [84] proposed a localised key management scheme, which eliminates the certificate graph of Capkun et al. [18]. Although this reduces the storage requirements, Li et al. [84] still depends on signed chains of trust between nodes, which does not reduce the computational overhead in verifying signatures. In large scale DTN deployments, an increase in the certificate chain length also increases the number of verification operations required to establish the chain, thereby increasing computational overhead. The chains of trust also present a security issue. As the chain is increased, the trust of the public key is diminished [46, 14].

The application of both Capkun et al. [18] and Li et al. [84] for a DTN with fast moving dynamic nodes such as a VANET is infeasible. The certificate update process for both proposed schemes requires time to sign and issue certificates. Li et al. [84] mentions that node movement speed should be moderate in relation to both certificate issuance and packet transmission latency. Fast moving dynamic nodes in a VANET would cause frequent certificate update instances. Coupled with the low latency between vehicles potentially travelling at high speeds, the time required for certificate issuing would not be sufficient.

Ngai and Lyu [103] proposed a cluster based decentralised PKI scheme. Nodes self-organise themselves into clusters, where each node in a cluster is assumed to know each other. The cluster of nodes monitors and retains trust values of all other nodes in the same cluster group. Continuous trust values between 0 and 1 are assigned based on node behaviour on security collaboration and data forwarding. When two nodes from differing clusters need to communicate, the other nodes in the cluster will reply to the node requesting the public key with the public key and a trust value of that node. This scheme utilises a trust and monitoring system to provide public key authentication. The disadvantages of this scheme are that it assumes nodes have some *a-priori* knowledge of the network prior to deployment to organise themselves into clusters. The mobility model of a DTN also hinders the trust and monitoring of other nodes in the same cluster group.

An extension of the cluster model proposed in [103] is Rachedi and Benslimane [113]. Nodes form clusters, and monitor each other nodes behaviour and trust. The addition of a cluster head acting as the CA for that particular cluster, enables introductions by nodes from other cluster groups to occur more easily than the aggregated introductions of cluster nodes in [103]. However, the elevation of a specialised node as cluster head creates a single point of failure for adversary attack for each cluster, and potential DoS attack. To mitigate this issue, Rachedi and Benslimane [113] also propose a sub-ordinate class of dispensable confident nodes called Registration Authoritys (RAs). RAs receive certificate requests on behalf of the cluster head CA and filtering them before passing them to the CA. This creates the necessity for both the cluster head CA and sub-ordinate RA to be in close communications range, which is infeasible in a DTN due to the sparse mobility model. Schemes that rely on clusters and cluster heads are still reliant on infrastructure-class nodes, and therefore cannot be considered as fully

distributed.

Omar et al. [104] proposed a distributed trust model for MANETs, allowing generation, storage, and distribution of public certificates without the need for a central authority. The key management is distributed as all nodes contribute to administering the network. This work improves the proposal of Capkun et al. [18] where nodes store and distribute certificates, by adding a transitive trust threshold cryptography scheme. This prevents false public key certificates being distributed by malicious nodes. The transitive trust relationship in PGP is extended by adding resilience in threshold cryptography through the idea that *if Alice trusts Bob, and Bob trusts Carol, then Alice can trust Carol if some other* $(t-1)$ *entities also trust Carol.* They propose a (t, n) threshold cryptography scheme where n is the number of nodes and t is the threshold where $t < n$.

The proposal by Omar et al. [104] at initialisation has a system dealer or service provider that is common to the member nodes. This system dealer has pre-established trust with all the member nodes. The system dealer distributes to each node a private share. An example of this is using the threshold scheme by Shamir [123]. Each node then generates a partial certificate for other nodes that it trusts in the system. Once this is complete, the system dealer is made redundant and is no longer needed to administer the system. When a new node wants to join the system, it requests a neighbouring node (the delegate node) to process the join request. The joining node sends its own public key, and trust evidence such as a password to the delegate node. The delegate node then broadcasts the request to other neighbouring nodes. After reaching a threshold of signings from neighbouring nodes, the delegate node authenticates the joining node by sending back the signed certificate. The joining node then generates its own private share to allow trust establishment of other new nodes.

Omar et al. [104] evaluates this proposal by introducing a malicious node. The authors outline that an internal malicious node is capable of issuing different types of false certificates. The first is where a certificate that binds the public key of node I to identity node J is issued. The second is to issue a certificate that binds node J to a forged public key. The third is to generate fake nodes and bindings, and generate appropriate keys to match identities. They evaluate the performance of their proposal in a simulation based in MATLAB. This experiment used 100 nodes to form a MANET. The movement of each node was defined by the random way-point model of Johnson and Maltz [73]. The proposed scheme

has shortcomings in the threshold signing, which from past proposals presents an additional computation overhead. Additional computation overhead is also attributed to the reliance on certificate chaining for public key authentication.

The increased communications and computation associated with establishing a certificate chain to verify a public key is still an open issue. Maity and Hansdah [92] proposed an on-demand certificate-less public key management scheme. Since the public keys of nodes are generated, stored and distributed by the nodes themselves, the authenticity of the public key cannot be ascertained. Hamouid et al. [57] extended the certificate-less Web of Trust model from [92], whilst also verifying the authenticity of public keys without the reliance on a CA. The use of certificate chains, which constitute the PGP Web of Trust imposes high overhead for key management, particularly in memory, bandwidth and power. Because of this, Hamouid et al. proposed a self-certifying identity based cryptography key scheme. They also separate Key Authenticity Trustworthy and Node Trustworthiness as independent, and state that the trust of public keys are absolute boolean states. Keys are either trustworthy or untrustworthy. Their proposed scheme relies on a Trustor Node N_i trusting another node N_j. Instead of signing or endorsing the public key of N_j, the trustor node N_i issues a Witness (W_{ij}) associated with the node identifier. The Witness is sent over an assumed authentication channel to Node N_j, where N_j uses the Witness to generate its private key SK_j. This scheme means that the Witness is part of the private key of the N_j and seen as the signature of the issuing node. This scheme binds the Private key of N_j (SK_j) to its identity. Because trust is assumed to be transitive, any node that trusts the Witness Issuer (N_i), can compute the self certified public key PK_j. This public key authentication scheme removes the problematic Private Key Generator (PKG) in traditional Identity Based Cryptography (IBC) schemes, which can lead to key escrow and key revocation issues.

Although this proposal provides an improvement in the memory, bandwidth and power of the nodes, the proposal still has some shortcomings. The key generation process is still dependant on another party, although not a TTP, it is distributed to the Witness issuer. As a result, there still has to be some level to trust between the Trustor Node and the node requesting keys generation, and is subject to potential malicious trustor nodes acting as bad witnesses between the identity of the node. The bootstrap of the network still requires some form of ring issuing root nodes to initiate the key generation process, essentially relying

on another party for the private key. The memory, bandwidth and power savings this scheme provides is mainly from the lack of public keys nodes store. Nodes do not store public keys instances, but rather generate reactively the public key of the node they wish to communicate and verify with. This is based on the assumption they can establish a trust chain back to that node, which also results in an issue with the reliability of key authenticity.

Decentralised key management schemes are better suited for DTNs. The distributed and decentralised nature of nodes means that a PGP Web of Trust PKI model is more favoured over a centralised CA model. However, the issues of *scalability* are still present. Public key authentication is computationally expensive on devices with constrained resources [66]. The creation of chains of trust requires nodes to exchange-sign-exchange certificates, thereby also increasing the communications overhead. Establishing a chain also presents a security issue, as the trust of the public key is decreased as the chain is increased [46, 14]. Other implementations to remove the overhead of the chain of trust has included the organisation of nodes into clusters. Cluster heads are appointed in [103], and public key authentication is achieved using a trust and monitoring scheme. However, this scheme relies on a-priori knowledge for cluster organisation, and results in the reliance on infrastructure-class nodes. The proposal in [57] removes the chain of trust, but is dependant on trustor nodes, another form of infrastructure-class nodes during the key generation process. Since DTNs are likely to have a *persistent adversary* in the network, the scheme is susceptible to adversarial nodes acting as malicious trustor nodes. Additional shortcomings in the form of bootstrapping issues during initial deployment, as well as revocation issues are discussed further in Section 2.5.2.2. Many of the schemes claim to be distributed, but are actually hierarchic, as they depend on a TTP prior to deployment or cluster structures. As a result of reviewing these schemes, a fully decentralised and transitive trust key distribution scheme is required to provide public key authentication in autonomous DTNs.

2.5.2 Revocation and Replacement

The distribution and exchange of public keys is an important stage in any DTN key management lifecycle (Figure 2.4, Page 24). However, just as important to distribution, is revocation and replacement of keys. In this section, the related prior works for key revocation and replacement are presented. The key revocation

and replacement process is an important part of the key management lifecycle of any PKI. Once keys have been distributed, it may be necessary to revoke those keys if they are compromised. Replacement keys are also essential to maintain the security of the network when the current key have been compromised, or when older keys expire. Previously proposed key revocation and replacement schemes for DTNs can be categorised in two groups: *Centralised* schemes based on traditional PKI schemes utilising a CA, and *Decentralised* schemes based on monitoring and reputation such as IBC and threshold cryptography based proposals. From these related works, the technical gaps are discussed and presented.

2.5.2.1 Centralised

Many DTN and VANET key revocation schemes are based on traditional centralised PKI implementations [117, 115, 116, 99]. These key revocation schemes rely heavily on CRLs [124]. The CRL is a list that identifies a revoked certificate, which is signed by the CA and made available to nodes of a network from public distribution points [44].

Kong et al. [79], proposed a CA model where all nodes are a member of the distributed CA. Utilising threshold cryptography to divide the CAs private key for signing, Certificate revocation was also managed by a group of nodes. Dissemination of revoked certificates was done through the distribution of a CRL list. Raya et al. [117] proposed a CA based certificate revocation model for vehicular networks. They assume a VANET where a trusted third party manages the identities, credentials and cryptographic keys of the nodes. They also assume certificates are not valid for an unlimited duration, and the revocation should occur in a timely manner as to avoid exploitation by adversary nodes [117]. The CA is responsible for revoking certificates and can do so through two methods. The first is through the use of CRLs. Raya et al. [117] proposed the Revocation using Compressed Certificate Revocation Lists (RC^2RL) scheme. To address the problem of a large CRL as more nodes are revoked, Raya et al. [117] utilised Bloom filters to compress the CRL. Bloom filters have the property that they can return a configurable rate of false positives but never a false negative [117]. The compressed CRL is then distributed to all nodes within the network. The second method of revocation is the Revocation of Tamper Proof Device (RTPD). This is used when an adversary node is detected and the CA wishes to remove all the keys in the tamper proof device of the node. The CA generates a revocation message

for the node to be revoked consisting of node metadata. As the CA administers the keys and certificates of all node entities in the network, it encrypts the revocation message for the specific node to be revoked. The CA then signs the message to provide authenticity of the revocation message before distributing it to the node to be revoked.

These two methodologies of revocation are initiated solely by the CA. Raya et al. [117] also proposed the Distributed Revocation Protocol (DRP), which acts more as a warning system to nodes. This allows nodes to temporarily revoke another node until the CA is capable of actually revoking the malicious node. This is further extended by Raya et al. in [115], where they introduce a misbehaviour detection system and the Local Eviction of Attackers by Voting Emulators (LEAVE) protocol. The LEAVE protocol still relies on a centralised CA to provide permanent revocation. The schemes proposed by Raya et al. all rely on a CA to act as the sole authority in revoking a node. They also utilise revocation in the context of the removal of an offending or malicious node. Although these schemes may be sufficient for a VANET where a CA can be assumed, it does not apply to networks with intermittent or no access to the centralised CA.

Lin et al. [87] extended the proposal of vehicular certificate revocation in Raya et al. [117] by proposing a RSU based certificate revocation mechanism. Lin et al. [87] noted that certificate revocation events although rare, require timely notification to the nodes. They leverage the assumed existing infrastructure of the RSUs to handle the majority of revocation tasks. The RSUs are distributed fixed points of trustworthy infrastructure, to which the CA broadcasts a certificate revocation notification. It is the responsibility of the RSU to check the status of certificates of messages passing within the network with the certificate revocation notification from the CA. Lin et al. [87] also utilise the predicted vehicle movement to allow neighbouring RSUs to co-operate in revoking an adversarial node. This proposal distributes the responsibilities of the centralised CA to the RSUs. However, the RSUs are still dependent on the CA to provide the revocation notification.

In an attempt to move the revocation tasks from the centralised CA or RSUs, Kumar et al. [80] proposed a secure decentralised PKI for VANETs where the vehicles themselves maintain and perform the revocation tasks. However, they still rely on a CA to monitor all communications and act as a Key Distribution

Centre (KDC). Kumar et al. assume that the CAs are the most trusted party in the network, and cannot be compromised by any type of attack. Keys are initially distributed by the CA, whilst key revocation for misbehaving nodes in the network is performed in a decentralised manner by the respective Learning Automata on each node. The Learning Automata is responsible for re-keying the vehicles with fresh keys during an update and revocation process. The assumption of an uncompromisable CA is unrealistic. Kumar et al. simulated their proposal with 550 nodes and a communications range of 100m. The simulation space was 500x500 metres, with RSUs deployed in 500m distributions.

A major shortcoming of using a CA for key revocation is scalability, because CRLs can grow very large in size for large domains and networks even with compression. There is a large network overhead in downloading such a large CRL for all clients in the network [44]. Schemes that utilise ΔCRLs, where only the differences or discrepancies in CRLs are distributed may be more suitable [44]. Proposals such as [80] also show that CRLs need to be constantly updated. However, due to the high mobility of nodes in a DTN and VANET, and ephemeral network connections, this is difficult to achieve.

Another certificate revocation scheme proposed for the Internet is OCSP. Both OCSP and CRLs are forms of blacklisting. However, they differ in how a certificate validity is checked. CRLs require the dissemination of a blacklist of revoked certificates, whilst OCSP requires a connection to an OCSP Responder or centralised infrastructure such as the CA to query the status information of a certificate. Although OCSP provides a communications overhead advantage over the distribution of CRLs, they do require an online, easily accessible and available OCSP responder. As such, this solution becomes unsuitable for application in a DTN and MANET, due to bottlenecking to the central server and reliance on a single OCSP Responder [95].

In an attempt to provide an OCSP service to MANETs, Marias et al. [95] proposed an Ad-hoc Distributed OCSP for Trust (ADOPT) scheme for certificate validation. They attempt to distribute the deployment of OCSP by utilising cached pre-signed OCSP responses. Three categories of nodes are required to support the proposed scheme: Server, Caching, and Client nodes. Server nodes are directly connected to the CAs and act as the OCSP responders. They announce revocation status for certificates and issue pre-signed OCSP responses to Caching nodes. Caching nodes receive the pre-signed OCSP responses and act

as OCSP responders to Client nodes. Caching nodes are distributed throughout the DTN. When a Client node requests the status of a specific certificate to a Caching node, the Caching node will search for a pre-issued and pre-signed response. The benefit of this scheme is that the Caching nodes do not need to execute expensive computational operations in signing and validating the response, as they only cache signed responses from Server nodes. If a Caching node does not have a response, it then re-broadcasts the request. Although this scheme distributes the operation of the OCSP through Caching nodes, it is heavily dependant on a reliable connection between Server and Caching nodes for up to date pre-signed OCSP responses. Issues such as cache management for nodes in a DTN as well as time thresholds affect the deployability of such a scheme in DTNs.

Another issue is the potential for a replay attack. Because messages in a DTN take the form of a store, carry, and forward bundle, a certificate status message from the OCSP server would be passed through multiple nodes. Because the channel is not authenticated, this implementation is vulnerable to a replay attack by someone replaying a certificate valid OCSP response before the expiration date, but after a certificate has been revoked. These issues, and the highly disconnected and fragmented nature of DTNs makes OCSP difficult to implement in a DTN.

Centralised TTP solutions are capable of providing efficient key revocation schemes for traditional networks such as the Internet. However, as with key distribution, the characteristics of a DTN mean that the deployment may not have reliable access to the TTP for authentication and key revocation tasks. Many schemes presented solely rely on the CA to perform authentication and key revocation.

All centralised key revocation schemes proposed for DTNs and its application in VANETs rely on some form of centralised authority to provide public key authentication for the revocation of keys. The centralised authority may be a single entity [115, 61, 130, 9], or multiple entities distributed amongst nodes [134, 79, 91]. However, they all require some form of trusted authority to empower and configure the centralised authority [18]. The inherent characteristics of a DTN make centralised key management schemes unsuitable. Many schemes presented solely rely on the CA to perform authentication and key revocation. The use of CRLs to inform nodes of revoked certificates presents issues with communications

overhead as the scale of the network increases. The use of a ΔCRL is more suitable. OCSP schemes are also unsuitable for DTN applications as the store-carry-and-forward routing makes OCSP responses susceptible to replay attacks. The issues associated with the centralised key revocation schemes presented in this section suggests that decentralised schemes are more suitable.

2.5.2.2 Decentralised

Decentralised key revocation schemes that are not reliant on a TTP have been proposed in [18] and [84]. However, many schemes for DTNs focus on the removal of misbehaving or malicious nodes from a network, instead of the removal of a key from operational use [90, 4, 100, 65]. Such schemes focus on node monitoring to determine whether a node is misbehaving. Accusations or a weighted report is then used to initiate the process of node revocation. Many proposals fall back on a CA to provide authentication during the revocation process. In addition, few investigate the key revocation and replacement process, particularly when a node decides to self-revoke.

The self-organising public key management scheme presented by Capkun et al. [18] outlines two scenarios where certificate revocation may occur. The first is if a node believes that the identity-key binding is no longer valid. The second is if a node believes their private key has been compromised. Capkun et al. [18] provides two methods for certificate revocation. The first is using an explicit revocation statement. A particular node will have a list of nodes that it feeds certificate updates to. As a result, the explicit revocation statement only has to be distributed to these nodes. The second is implicit revocation, which is dependant on the certificate expiration date. A certificate is considered implicitly revoked if it is not updated past the expiry date.

Li et al. [84] also provides a certificate revocation process for their decentralised PKI scheme. They propose the use of a two-hop revocation mechanism where the node initiating the revocation broadcasts the request a distance of two-hops directly. All other nodes outside the two-hop distance are informed through distribution of a blacklist similar to a CRL. The scheme proposed utilises certificates that signify a relationship between two nodes of one-hop distance. As a result, only the two nodes who form the certificate are allowed to initiate the revocation process. No other nodes may be involved. This has the disadvantage that if either of the nodes who can initiate revocation of that certificate is an

adversary, they may be able to trigger a false revocation of a legitimate node. With enough adversaries performing a false revocation attack, it would be possible to achieve a DoS attack. This applies for both revocation scenarios outlined in Capkun et al. [18] where the identity-key binding is no longer valid, or the private key has been compromised.

Revocation schemes focusing on the removal of misbehaving nodes can mitigate a DoS attack on the network. Assisted with a node monitoring scheme as well as a trust and reputation system, nodes may expel adversarial nodes before they inflict additional damage. Luo et al. [90] proposed a node neighbour monitoring scheme. Nodes disseminate signed accusations of other nodes to a predefined number of neighbours (m-hop). Nodes independently receive the accusation, and upon verifying the trustworthiness of the accusing node, add the information into a ticket revocation list. Using a threshold (t, n) based certificate scheme to distribute the duties of a centralised CA, a node is revoked if the threshold is exceeded. Trustworthiness of the accusing node is based on past behaviour judged by the local neighbourhood monitoring scheme. Nodes are also assumed to be pre-authenticated using the threshold based distributed CA.

Similarly, Arboit et al. [4] also presented a certificate revocation scheme based on weighted accusations from nodes. The weighted accusations are based on prior behaviour and reliability of the node. This system provides protection against wrongful revocation of certificates by malicious nodes. The authors still stipulate that prior to joining, nodes must hold a valid certificate from a recognised CA as well as CA public keys [4]. These certificates are used for network authentication and the CA is responsible in verifying the identity of the node before certificate issuance. When a malicious node is detected, the network is capable of 'self healing' by revoking or excluding the malicious node. Although effective, the scheme is still dependant on the CA to provide public key authentication, particularly if a legitimate node was required to update compromised keys.

Moore et al. [100] proposed two strategies for revoking misbehaving nodes. The first is using a re-election process, where nodes are required to secure a majority approval from peers over regular time intervals. This is in contrast to other proposals [90, 4], which rely on negative vote based systems to remove a misbehaving node. Moore et al. [100] proposed that honest nodes must demonstrate that it still has authority to remain in the network, thereby excluding

misbehaving nodes that fail to be re-elected. The second is using a suicide attack. The accusation of a misbehaving node should be done at a large cost as to deter malicious nodes from falsely accusing legitimate nodes. As a result, the node reporting a misbehaving node will also be expelled from the network along with the misbehaving node. When a central authority is available, nodes will report the accusation, however the central authority will handle authentication and the revocation process. The suicide attack is also extended to work in the absence of a central authority. Assuming nodes have pre-authenticated and pre-distributed private and public key pairs, the accusation is signed by the accusing node and then distributed to neighbouring nodes who verify the signature and independently handle the removal of keys for both the accusing and accused node. Although these proposals are effective and scale well, they do not address the issue of key renewal, as nodes are simply rejected from the network.

A similar proposal to Moore et al. [100] is Hoeper and Gong [65], which not only proposed a revocation scheme for the removal of misbehaving nodes, but also included a provision for a node to self-revoke their private keys. Hoeper and Gong proposed a monitoring based key revocation scheme for MANETs. They proposed the identification of malicious nodes in the network through monitoring neighbouring nodes by also considering false positive and false negative rates. They employ pairing based IBC schemes [11, 64]. The scheme, much like other IBC implementations rely on a Key Generation Centre (KGC), but also provide an extension to distribute the role of the KGC by using a distributed online KGC. Thus public key authentication is achieved by some form of KGC. The key revocation focusses on the misbehaviour detection and removal of a node. They present aspects steps for the key revocation scheme. The first is a local monitoring scheme where each node monitors all other neighbours one-hop distance. Any suspicious behaviour of neighbouring nodes are reported. The second is a Harakiri or suicide message. This is when a node realises that its private key has been compromised and creates a message informing other nodes to cease using the key pair. The message, either from the reporting of suspicious node behaviour in step 1 or a Harakiri message in step 2, is propagated to neighbouring nodes. Finally, each node updates their own key revocation list based on the validity of the receiving messages.

The key distribution proposal of Hamouid et al. [57], which provides a certificate-less Web of Trust model for public key authentication for MANETs,

presents some issues that make key revocation and update a problem. Due to the setup of the scheme, and their reliance on the Trustor Node and issuance of Witness, when a node wishes to update their key, the entire subordinate chain would need to be regenerated. The subordinate uncorrected key is needed in an authentication invoked by one of the predecessor nodes. This is due to the fact that the self-certified key generation scheme, generates a key from the Trustor Nodes key. Hamouid et al. mitigate this by proposing a key correction operation, which addresses the need to regenerate all subordinate keys. However, the key correction operation may result in extra resource consumption if the same node frequently updates their key, thereby causing a repeated key correction of the entire subordinate chain. Hamouid et al. further address the repeated key correction issue by implementing a lazy key correction, which only occurs when required. Another issue with key correction is when Trustor Node keys are tied to other nodes. The trustor chain in a DTN application is highly likely to be cyclical or heavily connected in a mesh. This creates an infinite loop of key correction. Although this scheme focuses on the public key authentication aspect of key management, there are elements of key revocation and key renewal presented. However, due to the dependant nature that the private key of a node is to a neighbouring node, issues such as infinite key correction still present an issue for key revocation and replacement.

Many of the prior works demonstrate the challenge of providing public key authentication during revocation. The CA based models rely on the TTP to authenticate the revoking node and perform the key revocation and replacement process. Decentralised schemes such as [18] and [84] provide a mechanism for key revocation and replacement, however, are subject to adversarial nodes triggering a false revocation of a legitimate node. Many proposals focus on the removal of misbehaving nodes, which may help mitigate DoS attacks, but does not resolve the problem of an unplanned self-revocation event. In a decentralised scheme relying on IBC or threshold cryptography, public key authentication is difficult as some proposals such as [4] and [93] fall back on a CA, and others such as [65] rely on private key material in the revocation message, or the addition of multiple key pairs or pre-distributed keys [64]. As a result, a fully decentralised and distributed public key authentication scheme for an unplanned revocation event is still an open problem.

2.6 Research Challenges

The focus of this thesis is to address the challenges outlined in the previous sections of this chapter. In particular, the issue of public key authentication for public keys of autonomous nodes in a DTN, during all stages of the key management lifecycle. Figure 2.8 identifies the areas and categories identified as research gaps from this section, as well as the subsequent chapters that will address these gaps.

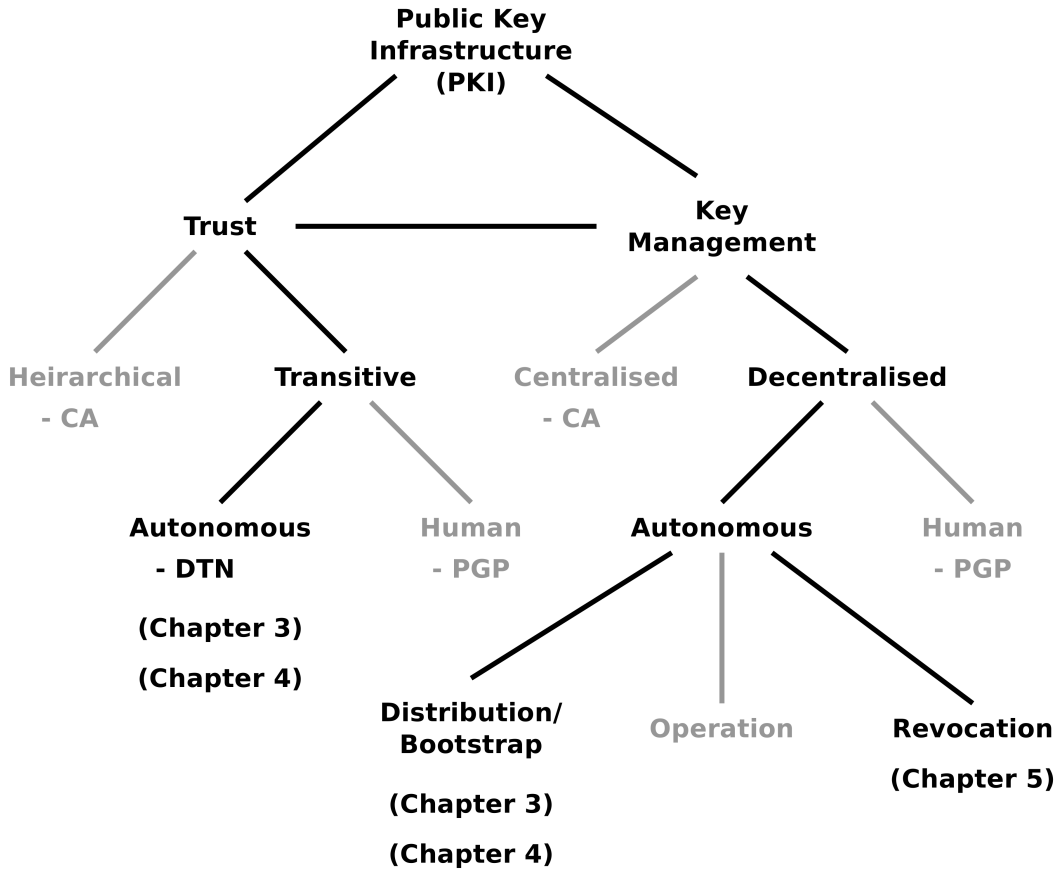


Figure 2.8: Research Categories and Area gaps identified and addressed in this thesis.

The requirements identified include a *fully distributed* and *decentralised* model, based on a *transitive trust* principle. This is suitable for environments *void of any infrastructure* or reliance on infrastructure-class nodes, as availability of service cannot be guaranteed. Therefore a *trust system* is more advantageous than a reputation system to reduce communications overhead. The ability to mitigate a key spoof attack by the adversary attempting to exploit the identity-public

key binding is an important requirement for a DTN public key authentication scheme. Particularly as DTNs are *large open networks*, and nodes are *pervasive*. A *persistent adversary* is assumed to be present. The pervasiveness of nodes and a persistent adversary threat model results in nodes being highly susceptible to *physical tampering*. Therefore, the challenges are:

1. **Public Key Authentication for public keys during Key Distribution**: The issue of public key authentication during key distribution is still an open issue. Traditional methods rely in either a CA or a Web of Trust model. The reliance on a CA is unsuitable due to the decentralised nature of a DTN. While the signing and chain of trust in a Web of Trust model is computationally expensive and problematic. Therefore, the question of providing a decentralised and fully distributed public key authentication for key distribution in autonomous applications, leads to Research Question 1: *Can a trust or reputation system be utilised to assist in DTN Key Distribution such that Public Key Authentication can be achieved without a trusted third party, but by automatically including mobility parameters, behaviour, and levels of collaboration into trust?* Chapter 3 describes the design, implementation, and evaluation of a trust system to address this research question.

2. **Application of a trust and reputation system in a large scale DTN**: Autonomous DTNs may be deployed on a large scale. This includes a large number of nodes deployed over a large geographic area. Past research has focused on a small number of nodes (100) deployed in a small geographic area (500m-1km). The DTN application of VANETs is an example of a large scale deployment. Therefore, the question of providing a scalable, fully distributed, public key authentication key distribution scheme in a large open autonomous application leads to Research Question 2 and 3. Research Question 2: *Is it possible to apply a trust or reputation system for DTN Key Distribution for a large scale realistic DTN application?* With the addition of Global Positioning System (GPS) capabilities and location tracking on vehicles this leads to Research Question 3: *Is it possible to leverage location data to assist a trust or reputation system for DTN Key Distribution?* Chapter 4 addresses both these research questions.

3. **Key Authentication for public keys during Key Revocation**: The issue of public key authentication during key revocation is still an open issue. Similarly to key distribution, traditional methods rely in either a CA or a Web of Trust model. Past research focuses on planned revocation events such as key expiry and renewal. However, unplanned revocation events such as when a private key has been compromised is still an open problem. Therefore, the question of providing unplanned key revocation in a fully distributed transitive trust based PKI for autonomous applications leads to the following two research questions are therefore posed. Research Question 4: *Is it possible to utilise a trust or reputation system to assist in DTN Key Revocation such that Public Key Authentication can be achieved without a trusted third party?* and Research Question 5: *Is it possible to provide trust transferral of an old compromised public key to a newly generated public key without a trusted third party during an unplanned key revocation event?* Both research questions are addressed in Chapter 5.

2.7 Summary

This chapter covered background concepts in DTNs, trust and reputation systems, and PKI. More specific DTN applications of trust management and key management were discussed. It examined the characteristics, routing, applications, and challenges specific to DTNs. Different Trust and Reputation systems were also examined, outlining differing architectures and computation engines. Two common models of PKI were reviewed in general covering how public key authentication is achieved for public keys. Key distribution and revocation was also covered. Finally, key management schemes specific to DTNs were covered for the two critical stages of distribution, and revocation and replacement. Based on this survey of previous related work, three research challenges were identified for providing a complete decentralised key management scheme for autonomous DTNs. These challenges are explored and the research questions posed are addressed in the following chapters.

Chapter 3

DTN Key Distribution

Securing communications and data in a Delay Tolerant Network (DTN) involves satisfying general security properties such as confidentiality, data integrity, and message authentication. Cryptographic technologies such as cryptographic hash functions and public key cryptography are some essential tools in providing security. However, the use of public keys to secure networks gives rise to a critical foundational issue of public key authentication. This issue is essentially the verification of the identity-public key binding. The lack of public key authentication exposes networks to adversarial agents who are capable of exploiting the identity-public key bindings. This provides them the capabilities of eavesdropping on sensitive communications, modification of safety critical messages, and identity deception by impersonating other entities. Traditional networks such as the Internet achieve this through a Certificate Authority (CA), a centralised form of Public Key Infrastructure (PKI). The decentralised and distributed nature of DTNs make such schemes unsuitable. Compounded with the fact that DTN deployments are now utilising autonomous nodes, decentralised trust establishment without humans is an open research problem.

The research presented in this chapter specifically address Research Question 1 (Chapter 1): *Can a trust or reputation system be utilised to assist in DTN Key Distribution such that Public Key Authentication can be achieved without a trusted third party, but by automatically including mobility parameters, behaviour, and levels of collaboration into trust?* This chapter reviews and extends prior works in this area by proposing a combined trust system and key distribution

mechanism to achieve public key authentication, in particular in autonomous DTNs. This proposed scheme is called the *Leverage of Common Friends (LCF)* trust system and is verified in the results of the simulation and evaluation comparison of prior work.

The structure of this chapter is organised as follows. Section 3.1 outlines a similar past proposal of providing public key authentication in DTNs. Section 3.2 provides a detailed outline of the System Model and Security Properties of the network, along with definitions, threat model and adversary capabilities. Section 3.3 presents the newly proposed LCF trust system to assist in the provision of public key authentication during key distribution. Trust weighting values and the criteria are discussed and selected. Section 3.5 outlines the experimental methodology including experimental and adversary setup, the experiments conducted as well as the security and performance metrics used to evaluate the LCF trust system. Section 3.6 presents the experimental results and evaluation of how the proposed LCF trust system provides public key authentication. Section 3.7 discusses the implications and issues related to the proposed scheme. Finally, Section 3.8 summarises the research and contributions presented in this chapter.

3.1 Background and Related Work

Jia et al. [72] outlined that key distribution in a DTN where PKI is unavailable is still an open problem. The authors proposed the use of a similar key distribution scheme to PGP with varying levels of trust and utilising two channel cryptography techniques to prevent key spoofing during transfer. Nodes generate their own public and private key pair similar to Rivest et al. [119], and move in close geographic proximity to each other. Each node exchanges public keys with one another, and stores, carries and forwards public keys. The two channel cryptography scheme provides security during the key exchange phase.

The public keys of each node are exchanged when they are in close proximity to each other similar to the Resurrecting Duckling Scheme by Stajano and Anderson [126]. This scheme allows two nodes in close proximity, to exchange keying material over an opportunistic link using imprinting. This keying material can be stored and used later by nodes to establish a confidential channel. Utilising the Resurrecting Duckling Scheme for the key distribution of public keys would be suitable for a DTN, particularly as there is no reliance on a centralised

authority to manage the nodes. It is a fully distributed scheme, where there is no pre-established trust.

The public keys of each node are exchanged by meeting other nodes, forming the highest trust level, direct trust. Keys in the direct key list are assumed to be trustworthy as they were received from the node that owns the public key. As nodes are highly mobile, they receive the public keys of various other nodes, becoming carriers. These carried keys, belonging to other nodes, are also distributed forming indirect trust relationships between nodes. This follows the Web of Trust principle: *If Alice trusts Bob, and Bob trusts Carol, then Alice can indirectly trust Carol.*

Because the ownership of carried keys cannot be easily verified in a distributed system when compared to a centralised architecture, Jia et al. [72] proposed the use of an approval system. The receiving node may approve or reject the carried key based on the trust value of the carrier node it received the key from. For example, Bob may have received many instances of Carol's key from various other carriers. Bob trusts these carriers with varying degrees of trust. Bob assigns a trust value to each carrier, and if the total trust of the combined carriers is above the threshold, Bob approves Carol's key into the approved key list. Since human reasoning is required to provide the initial trust value of each carrier, Jia et al. utilised randomly generated trust values in the simulation. The key distribution scheme was simulated using randomly generated values of initial trust in the NetLogo [131] simulator.

Jia et al. [72] utilised the spread of carrier keys to effectively distribute keys in large scale DTN systems upon deployment. However, the issue of public key authentication is an open problem, particular when there is no Trusted Third Party (TTP). The problem is further complicated for DTN applications consisting of autonomous nodes, where there is no human involvement. Initial trust establishment between autonomous nodes is difficult, as such a scheme is heavily dependant on human intervention or a centralised management.

The unresolved issues in [72], along with the research gaps identified in Chapter 2 presents an open research problem of whether a trust or reputation system can be used for initial trust establishment in autonomous nodes. With addition to the use of a trust or reputation system to provide public key authentication during key distribution. Many of the trust and reputation schemes presented in Chapter 2 were utilised for adversary detection or optimal message

routing. Schemes that attempted to address the issue of initial bootstrapping were dependent on either pre-established links or infrastructure. These schemes are not suitable for a DTN deployment where there is no *a-priori* knowledge. Past proposals for key distribution in DTNs attempt to provide a distributed and decentralised key management scheme. However, due to the difficulty in providing a truly fully distributed key management scheme, many fall back on pre-established infrastructure-class nodes, or a hierarchical based trust model. As a result, the open research problem of whether a trust or reputation system can assist autonomous DTN key distribution such that public key authentication can be achieved without a trusted third party and *a-priori* knowledge remains.

3.2 Key Distribution System Model and Security Properties

In this section, the System Model and Security Properties of the application environment identified from relevant literature presented in Chapter 2 are outlined and identified. In addition, the likely application and landscape of the network is also discussed in system model. Terminology and notations such as types of nodes and keys are defined. The system model is susceptible to attack by adversaries that will attempt to exploit a threat model, which is defined. The capabilities of the adversary are defined along with the extent of their attacks. Having defined the system model of the network, threat model and adversary capabilities, the security properties that the proposed key distribution scheme should achieve is outlined. Throughout this chapter, the notations in Table 3.1 are used to refer to nodes, keys, trust, and black hat nodes.

3.2.1 System Model

The system is assumed to be a closed DTN, spanning a small geographic area. The deployment environment has no other entities except nodes themselves. There is no centralised PKI or any form of TTP, and there is no public communications infrastructure. Nodes are considered fully autonomous and mobile, requiring no human intervention. They self-initialise on deployment with no a-priori knowledge of the network or their neighbours. There is no pre-deployment initialisation phase by an offline authority. The key management phases of

Table 3.1: Notations

Notation	Description
Node ID Notations	
i	Unique Persistent Identifier i
N_i	Node i
Key Notations	
K_i	Keypair of N_i
S_i	Secret (Private) Key of N_i
P_i	Public Key of N_i
D_i	Direct Key List of N_i - List of public keys received directly from another node
A_i	Approved Key List of N_i - List of trusted public keys received from carrier nodes
U_i	Untrusted Key List of N_i - List of untrusted public keys received from carrier nodes
Trust Notations	
T_i^j	Trust Value N_i has of N_j
t_n	Trust component weighting given to each contact
n	Number of contacts
t_c	Trust component weighting given to each common contact between two nodes
c	Number of common contacts between two nodes
t_d	Trust component weighting given to each key discrepancy instance
d	Number of key discrepancies
$t_{neutral}$	Initial starting value of trust
Black Hat Notations	
$S_{(m,i)}$	Spoofed Private Key with identity of N_i, generated by malicious N_m
$P_{(m,i)}$	Spoofed Public Key with identity of N_i, generated by malicious N_m

Moore et al. [100] are adopted for this thesis. However, given the lack of a pre-distribution phase, the key management phases are; initial bootstrapping (distribution), operation, and revocation.

Nodes retain a persistent unique identity [108], and generate Elliptic Curve Cryptography (ECC) public and private key pairs [98, 78]. These keys are used to perform security based tasks such as providing confidentiality, data integrity, and message authentication. It is also assumed that the key pairs have a long but finite time period of validity, similar to Pretty Good Privacy (PGP) where keys

may last 1 to 2 years. In this chapter, it is assumed that nodes communicate to each other through close wireless communications using Bluetooth [10]. Due to their mobile nature, they create ephemeral or opportunistic bi-directional connections between neighbours [65]. These communications connections are used to exchange the public keys they own, as well as the public keys of other nodes they have met and carry using the Resurrecting Duckling Scheme. Public key exchange is considered to be a low-cost communications procedure as the exchange occurs at one-hop distances between nodes [13]. The public key exchange is completed over a two channel or side channel scheme as outlined in [72]. It is assumed that adversary nodes will always exist in the system, and that it might not be feasible to expel such a node.

3.2.2 Trust Model

Using a transitive trust model outlined in Section 2.3.2 [138], the public keys of each node are exchanged when nodes are within communications range. These keys form the highest level of trust - Direct Trust. The public keys (P_i) are easily verifiable as they were transferred by the node owning the corresponding private key (S_i). Due to the mobility of nodes in a DTN, nodes will also receive the public keys of other previously met nodes, becoming carriers. These carried keys belonging to other nodes are also distributed in the key exchange process. As in the PGP Web of Trust model, indirect trust relationships are formed between nodes much like the transitive trust principle.

3.2.3 Definitions

The following terminology used throughout this chapter is defined:

1. *Key Distribution* is the process where a node distributes their public key (P_i) to another entity in the network for the use of providing end-to-end secure communications.

2. *Public Key Authentication* - Is the verification of the identity-key binding of a public key. In a decentralised public key distribution scheme such as PGP and the Web of Trust, public key authentication is achieved by the human user confirming the entity's claimed identity is associated with their corresponding public key. It is measured as a boolean (Y or N).

3. *Public Key Confidence* is proposed in this thesis as having *confidence* in the identity-key binding of a public key. In an autonomous DTN, without a centralised PKI, or any other infrastructure but the nodes, verification that the public key being distributed belongs to the associated node is difficult. In a DTN application, public key confidence is how confident the autonomous nodes are that the multiple instances of the same public key they are receiving is actually owned by the node identity. It is a continuous value consisting of the culmination of trust values. When the confidence in a public key exceeds a threshold, public key authentication is achieved.

4. *Trust Value* is a real numerical value within a predefined range, that is assigned to a single key or node by an entity describing the level of trust it has in that key or node.

The following terms on how nodes categorise the receipt of keys are defined. These reflect how the node came into possession of the public key.

1. *Direct Key* is a public key that a node has received from the owner - N_A has received the public key of N_B (P_B) directly from N_B. Direct Keys are stored in the Direct Key List (D_A).

2. *Carrier Key* is a public key that a node has received from another node who has previously met the owner of that public key - N_A has received the public key of N_B (P_B) from the carrier N_C. A Carrier Key can either be one of the following:

   (a) *Approved Key* is a public key that was distributed by a carrier node that has exceeded the public key confidence threshold to be trusted - Using the example from Figure 2.7, N_A has received the public key of N_X (P_X) from various carrier nodes (N_C, N_D, and N_E) and is confident that P_X actually belongs to N_X. Approved Keys are stored in the Approved Key List (A_A).

   (b) *Untrusted Key* is a public key that was distributed by a carrier node that has not exceeded the public key confidence threshold to be trusted - Using the example from Figure 2.7, N_A has received the public key of N_Y (P_Y) from various carrier nodes (N_B, and N_C) and is not confident that P_Y actually belongs to N_Y. Untrusted Keys are stored in the Untrusted Key List (U_A).

Table 3.2: Classification of Nodes and their Keys

Nodes	White Hat (N_A)	Key pair (K_A)	Private Key (S_A)
			Public Key (P_A)
	Black Hat (N_M)	Key pair (K_M)	Private Key (S_M)
			Public Key (P_M)
		Spoofed ID	Private Key ($S_{(M,A)}$)
			Public Key ($P_{(M,A)}$)

Table 3.2 provides a summary of the different classifications of nodes, keys and revocation materials. They can be broadly categorised into two types of nodes. *White Hat Nodes* (N_A) are nodes that are legitimate, and have not been compromised by an adversary or attacker. They generate a key pair (K_A), which consists of a *White Public Key* (P_A) and a *White Private Key* (S_A). *Black Hat Nodes* (N_M) are nodes that have been compromised by an adversary or attacker. Like White Hat Nodes, they possess their own Key pair (K_M), which consists of a *Black Public Key* (P_M) and a *Black Private Key* (S_M). N_M as part of their malicious nature may create Spoofed ID material.

The first form is a *Spoofed ID Public Key* ($P_{(M,A)}$). This is when N_M generates a key that has the identity association of a White Hat node N_A, but the key of a Black Hat Node N_M. It is created when N_M changes the identity association of its public key (P_M) from itself (M) to the identity of a White Hat Node (A). This results in a public key that other nodes think belongs to N_A (that is N_A has the corresponding private key) however, N_M holds the corresponding private key.

3.2.4 Threat Model

Given the system model outlined in Section 3.2.1, the following threat model is assumed:

1. *Lack of Infrastructure* - With no infrastructure to assist in key management operations such as public key distribution, nodes will have to handle these operations independently. An adversary node may take advantage of the environment where nodes have to independently distribute their own keys to assume the identity of another node. The lack of infrastructure also affects the implementation of a trust and reputation system as there is difficulty in the aggregation of trust scores to form a reputation system.

2. *Lack of Public Key Authentication* - There is no trusted third party, and therefore no assurance of public key authentication. There is no authenticity between the public key and the identity of the owner [12]. An adversary node is capable of associating its own public key to the identity of another node, distribute this key, and perform a Man In The Middle (MITM) attack.

3. *Physical Tampering* - DTN nodes may be subject to physical tampering [106]. Attackers may gain physical access to a node, modifying the behaviour, public key bindings, and distribute Spoofed ID Keys from tampered nodes. The physical tampering of nodes may be mitigated but cannot be prevented. As such, nodes will need a mechanism to detect and protect the network against potentially compromised nodes.

4. *Eavesdropping and Modification of Communications* - Since the connection between nodes in a DTN are ephemeral, and with no static routing, nodes may be required to pass on messages between nodes. This provides an adversary the capability to overhear as well as the potential to modify wireless communications between nodes [107]. Due to this threat, nodes will need to establish secure end-to-end communications.

3.2.5 Adversary Capabilities

Adversarial nodes will attempt to exploit the threat model outlined for this network. With a Spoofed ID Key, any adversary node is capable of impersonating another node. This has consequences for the security of communications, and future key management activities such as key revocation. Due to the characteristics of a DTN, an adversary is capable of eavesdropping on communications, and modifying data. These networks require communications to be passed on between nodes in a store and forward method [40]. A message or bundle encrypted with a Spoofed ID Key that an adversary has generated, means that if the adversary is capable of intercepting the message, they can successfully perform a MITM attack [12]. An insider attack model was used, with the adversarial nodes assumed to have similar abilities as outlined by [32] with the following capabilities:

1. Adversarial nodes can obtain any message passing through the network between two other nodes within communications range [40, 12, 107].

2. They are a member of the network and can therefore initiate and receive communications with other nodes in the network.

3. They are able to generate new Black Private and Public Keys (S_M, P_M) as defined in Section 4.2.3 and distribute P_M. S_M and P_M between adversary nodes are independent to prevent White Hat nodes from blacklisting a common P_M between all adversary nodes [32].

4. They are able to generate and distribute Spoofed ID Keys ($P_{(M,i)}$) as defined in Section 4.2.3. This is when they associate their own private and public key pair with another node's identity in their D_M or A_M Lists [12].

3.2.6 Security Properties

Given the system and threat model of the network, this section outlines the desired security properties the key distribution scheme should achieve. Ultimately, the aim is to provide public key authentication in an autonomous DTN node during key distribution.

Property 1: A White Hat node (N_A) should be able to generate a key pair (K_A), and distribute the public key (P_A) to establish a secure end-to-end communications channel with other nodes in the network.

Property 2: Any White Hat node should be able to utilise indirect relationships through carried keys to allow a greater number of nodes to communicate with.

Property 3: The effect of a Black Hat node (N_M), wishing to modify the identity-public key binding of a White Hat node (N_A) by distributing a spoofed ID key ($P_{(M,A)}$) should be mitigated.

3.3 Leverage of Common Friends (LCF) Trust System

A new linear computation trust system called LCF is proposed to provide public key confidence, and by extension public key authentication, to help establish secure autonomous communications. The trust relationship between two nodes is assumed to be linear, which is similar to many trust and reputation systems for online retail sites such as eBay and Amazon [74]. Weighted scores similar to

[21] were used to provide different trust components to formulate a trust score between two nodes.

The proposed trust system leverages common contacts between two nodes that are meeting for the first time. Nodes meet and exchange keys similar to the PGP model. They also build a Web of Trust as they move around in the community [138]. It is assumed that the more nodes (or in a social context "friends") that the node has met (n), the more trustworthy and well established it is in the community. Although the absolute number of nodes met is important, the number of nodes shared in common (c) provides a more substantial metric for establishing initial trust. The number of nodes shared in common (c) is the quantity of mutual node interactions the two meeting nodes have previously encountered. A comparison of the common node meetings between two nodes meeting for the first time mitigates the effect an adversary node fabricating a large list of friends to falsify a higher trust rating. These two properties, the number of nodes met (n) and number of common nodes (c), both increase the trust value.

Adversary nodes, exploiting the lack of public key authentication may modify the identity-public key binding of another node, and distribute this key to perform a MITM attack. A legitimate node who has established the correct identity-public key binding will become aware of this discrepancy in identity-public key binding. As a result, the decreasing trust value (d) is the discrepancy between two nodes over the binding of an identity and the public key. If there is a discrepancy, both nodes will decrease trust value with respect to each other. Nodes will take the default position of completely trusting themselves, and assume their version of the public key is the correct key, whilst the neighbouring node has a spoofed key. It is assumed that trust is diminished significantly faster than increasing trust.

The linear relationship between common contacts and trust can be represented by the equation below:

The trust of N_A assigns to to N_B (T_A^B) is given by Equation 3.1:

$$T_A^B = t_{neutral} + (t_n * n) + (t_c * c) + (t_d * d) \tag{3.1}$$

where:

t_n is the trust weighting given to number of contacts.

n is the number of nodes N_B has met, where $0 \leq n \leq$ Node Population.

t_c is the trust weighting given to the common contacts of N_A and N_B.
c is the number of nodes in common with N_A and N_B, where $0 \leq c \leq$ Node Population.
t_d is the distrust weighting upon discovering a potential spoofed key.
d is the number of discrepancy keys between N_A and N_B, where $0 \leq d \leq$ Node Population.
$t_{neutral}$ is the starting value of trust.

Nodes use the LCF trust system independently to establish initial trust with a neighbouring node. The trust system scores are not aggregated to form a reputation system, but are calculated, and used solely by the individual node. Each node will independently generate trust scores of other nodes based on Equation 3.1. The components n and c are positive trust components, which increase the trust score a node will assign to another node. The component d, measuring identity-public key binding discrepancies is a negative trust component, decreasing the trust score of a node that is apparently distributing falsified public keys. The trust score a node assigns to a neighbouring node can increase or decrease after each meeting over time, as the components n, c, and d will continually change over time.

3.4 Trust Weighting Selection

The values for the trust weightings t_n, t_c, and t_d were selected to satisfy several criteria for key distribution. The process involved first choosing a weighting for t_n, and subsequently setting t_c and t_d to satisfy the criteria.

The trust weighting t_n was selected based on Dunbar's Number [35], which suggests that for humans, there is a cognitive limit that a person can sustain stable social relationships. The number of relationships is typically between 100 to 200 relationships [112, 63]. Studies into user relationships on online social networking sites such as Twitter have also correlated the range of 100 to 200 users [51]. Using 100 as a starting value for upper limit of relationships between nodes to mimic human behaviour, this number coincides with the experimental node population size. Therefore t_n was selected so that if a single node were to have directly met the entire node population, their overall trust weighting should be the upper bounds of trust ($T_A^B = 1.0$). Using this criteria, the trust weighting

t_n is set at 0.01.

To determine the criteria for selecting the common trust weighting t_c, concepts from the PGP Web of Trust scheme were adopted. The limitation of using a PGP Web of Trust scheme in an autonomous system is the requirement for a human to assign the initial trust establishment of a public key using discrete levels (Full, Marginal). However, in the proposed scheme, the LCF trust system provides the initial trust establishment between two nodes. This criteria, can therefore be applied to the acceptance process of public keys in autonomous systems, as the criteria provides a guide on how autonomous nodes are to establish trust. The status of a key is determined as valid if the two following conditions is met for the GNU Privacy Guard (GPG) implementation of PGP [25].

1. Key is signed by a enough valid keys that satisfies one of the following conditions:
   (a) The user has signed it personally
   (b) The key has been signed by 1 Fully trusted key
   (c) The key has been signed by 3 Marginally trusted keys
2. The path of signed keys between the two keys is less than 5 steps.

The modern OpenPGP reference RFC4880 [14] does not define the number of Fully or Marginally trusted keys required before a key is considered valid, as this relationship is dependant on the implementation of the OpenPGP reference. Although the GPG implementation uses a Fully and Marginally trust relationship of 1 and 3, the PGP 2.6 implementation uses a 2 and 4 Fully and Marginally trust relationship. The number of Marginally trusted keys for a key to be considered valid is useful to help set t_c, such that nodes will have 3 to 4 instances of a key from various carrier nodes (marginally trusted nodes). Using the various PGP implementations as a guide, and having set $t_n = 0.01$, the trust weighting t_c is determined to be in relation to t_n to satisfy the following criteria:

1. $t_c > t_n$ as to reflect that nodes in common should be considered a higher trust weighting compared to the number of known nodes.
2. The number of carrier key instances required before trusting a key instance should be on average between 3 and 4 instances.

3. The minimum number of key instances before a node trusts a key should be at least 2.

4. The maximum number of key instances before a node trusts a key should be limited to twice the number of average key instances. Using these numbers that limit is 8. Having a higher number would unnecessarily prolong the key distribution.

5. The majority number of key approvals should require 3 or 4 key instances.

Figure 3.1 depicts the five number summary (minimum, 1st quartile, median, 3rd quartile, maximum) and average results of the number of key instances that lead to approval whilst varying the Common Trust Weighting t_c. The results show that setting t_c to 0.025 and 0.035 would satisfy the 5 criteria for number of key instances. When $t_c = 0.025$, the average number of key instances for approval is 3.65 keys, while when $t_c = 0.035$, the average is 3.39 keys. The median of results for $t_c = 0.025$ and $t_c = 0.035$, is 4 and 3 respectively. From these results, the common trust weighting t_c was set to 0.025.

Finally the distrust key weighting t_d was set to be a significant disadvantage for distributing a black hat key, having it set at a penalty of 10 times t_c. Therefore, t_d was set at -0.25.

3.5 Experimental Methodology

An open source DTN simulator was developed in Python called *Traffic Djam* [30] that models the decentralised distribution of public keys between nodes using a Web of Trust model to provide a fully distributed and decentralised key distribution scheme. Random movement models for a predefined number of nodes and the size of the simulation space was generated. This movement model was recorded and re-used for each repeat of the experiment to provide a controlled movement and node connection model. Nodes were initialised with a random starting XY co-ordinate, a node ID, and a randomly generated public key signature. The simulation space was divided into squares. For each node at each time step, the simulator rolls a nine-sided die to determine whether the node should move into the eight adjacent squares or stay in the current square. Nodes within a predefined communications range of another node may then connect to each other to engage in public key exchange.

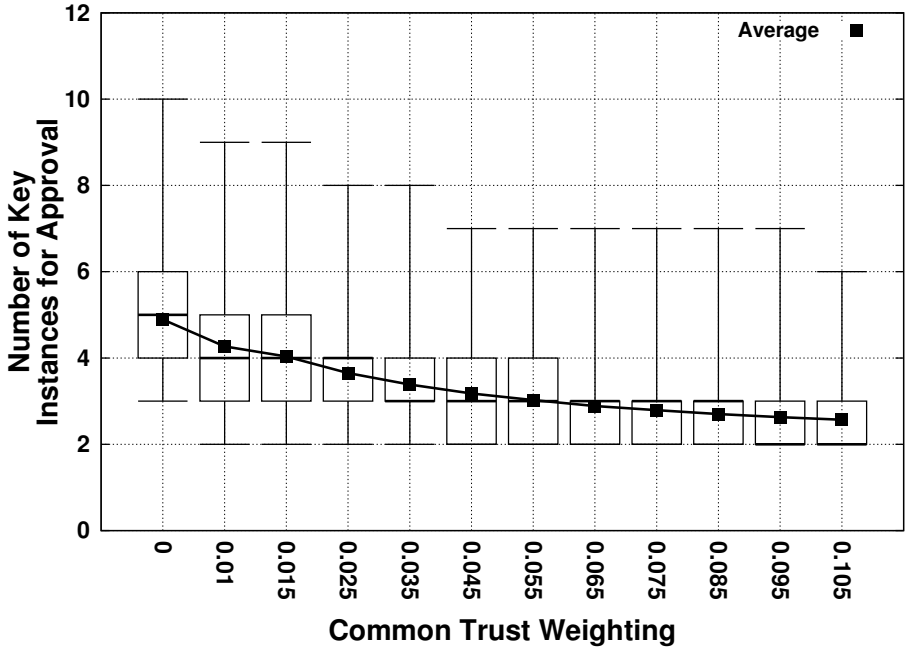


Figure 3.1: Common Trust Weighting (t_c) variations.

3.5.1 Experimental Setup

It is assumed that there is no public infrastructure, no trusted third party, and no central point where nodes are initialised. Nodes self-initialise when they are deployed in the simulation area. They generate a public and private key pair similar to [119] during initialisation. Nodes are assumed to be resource constrained and deployed in a small network area. They then move around randomly in the simulation area and connect to other nodes within a defined communications distance. When nodes connect, they exchange their public keys for a pre-determined time. Jia et al. [72] used a time delay to accommodate for the two channel cryptography steps in key exchange. Assuming that the nodes in this simulation are resource limited, the nodes form temporary connections only in close proximity. Using a low powered, close range wireless connection such as Bluetooth, a time delay of 60 seconds was assumed. This was to be a worst case scenario for nodes to establish a secure channel, handshake, exchange public keys and transfer additional data such as messages. Initial experiments of varying the time delay that nodes stop and transfer data bundles is shown

in Figure 3.2. As expected, reducing the time delay for data bundle transfer between nodes results in a faster key distribution as it allows more opportunistic connections to be made than in a longer time delay scenario.

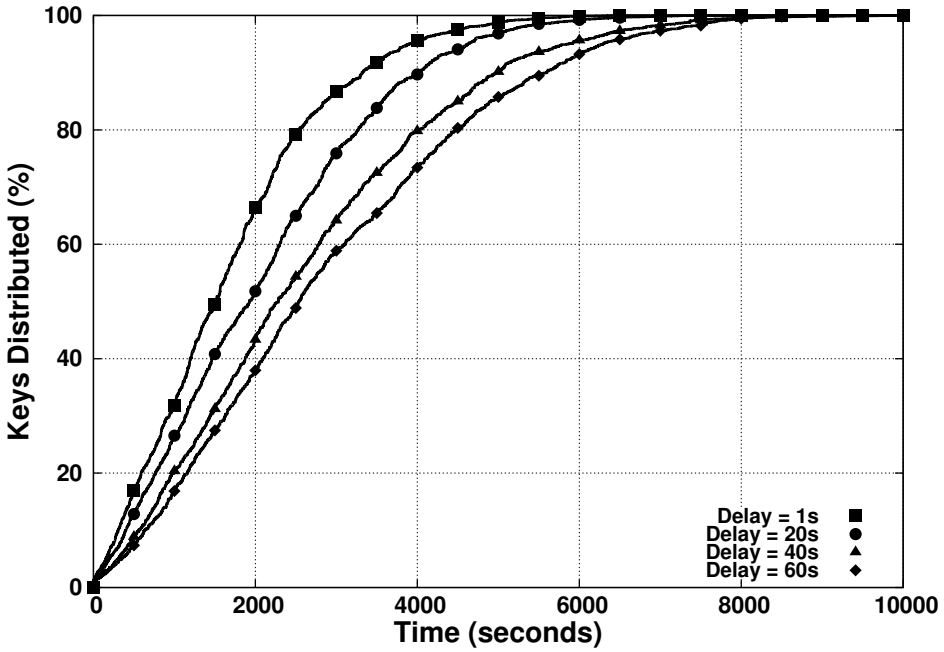


Figure 3.2: Key distribution over time with varying connection time delay.

The public keys of each node are exchanged by meeting other nodes, forming direct trust, the highest trust level. Keys in D_i are easily verifiable as they were received from the node that owns the public key. As nodes are highly mobile, they receive the public keys of other nodes, thus becoming carriers. These carried keys, belonging to other nodes are also distributed forming indirect trust relationships between nodes using the transitive trust principle.

Because the identity-public key binding of carried keys cannot be easily verified in a distributed system in comparison to a centralised architecture, an approval system similar to [72] was used. The receiving node may approve or reject the carried public key based on the trust value of the carrier node it received the key from.

For example, N_A may have received many instances of P_B from various other carriers. N_A trusts these carriers with varying levels of trust, and assigns a trust value to each carrier. If the total trust (or collective trust) of all carriers is above

the public key confidence threshold, N_A approves P_B into A_A. A rejected key that is below the public key confidence threshold is placed in U_A until the threshold is met. The proposal in [72] required human reasoning and intervention to provide the initial trust value of each carrier in this situation. Three trust establishment methodologies are simulated and compared.

The first methodology is shown in Figure 3.3. It depicts the public key exchange of two nodes when connected using no trust system. This scenario is the control scenario of the experiments and is an absolute trust scenario. Upon connection, both nodes flag a connected status and stop moving. In the direct key exchange phase of Figure 3.3, N_A sends P_A to N_B, and N_B reciprocates by sending P_B. Each node then adds the directly received public key to their respective D_i. The next phase is the carrier key exchange as shown in Figure 3.3. This is when N_A sends the list of nodes it has met in the past, essentially D_A. N_B also sends its respective list of nodes D_B. N_A may potentially provide false information about D_A to N_B. This can be mitigated by sending a cryptographic hash or Hash-Based Message Authentication Code (HMAC) as a challenge-response of D_i prior to sending the list. Each node scans the list and finds public keys that are not in D_i or A_i and adds them to their respective A_i.

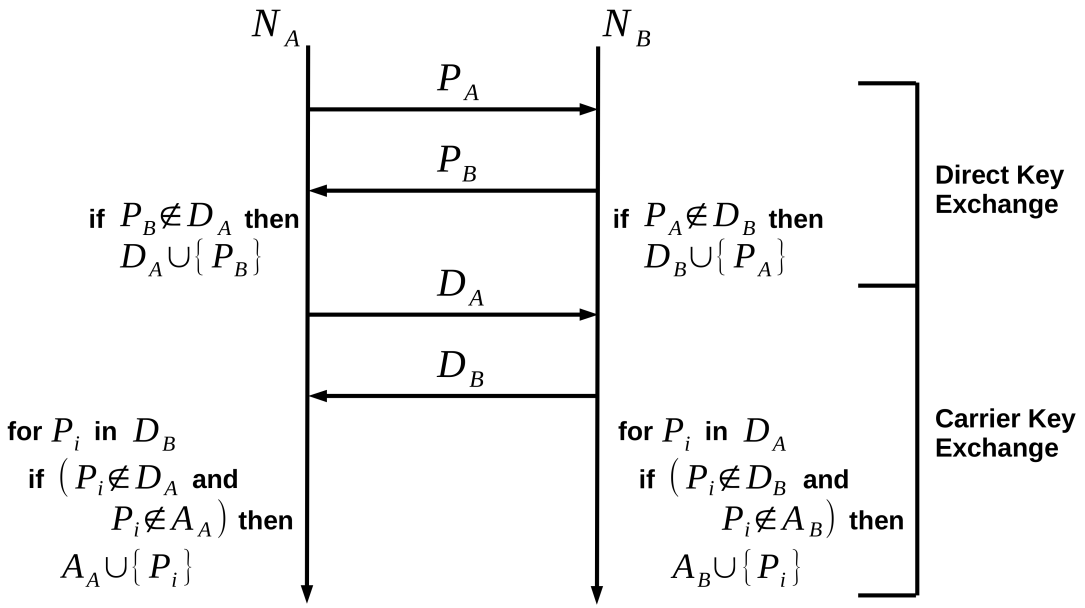


Figure 3.3: Public key exchange with no trust system (absolute trust).

The second method is the random trust assignment outlined in [72]. Because the approval process requires human reasoning and intervention to provide the

initial trust value of each carrier, the initial trust value was randomly generated for their simulation. This leads to an asymmetric trust relationship between the two nodes. Figure 3.4 depicts the public key exchange with random trust assignment from [72]. It shows additional steps in the carrier public key exchange phase. The carrier key exchange process now includes an approval process utilised by [72], where the trust of carrier nodes is accumulated, and if above the threshold, the carried key is approved. The initial trust establishment is randomly generated.

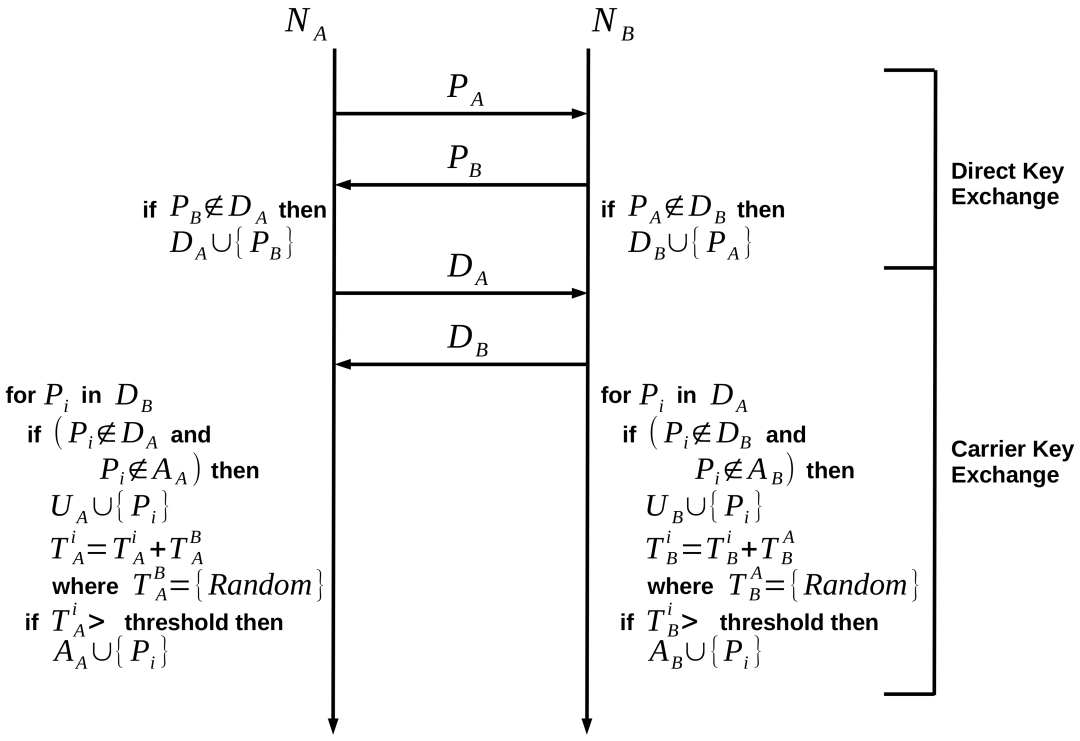


Figure 3.4: Public key exchange with random trust assignment.

The third methodology is the proposed LCF trust system. Figure 3.5 depicts the public key exchange process using the proposed LCF trust system for autonomous node applications. It shows that the establishment of initial trust stage is more comprehensive compared to [72], as it requires both nodes to send a list of direct contacts to each other. Trust is then computed based on these lists, following which, the usual direct and carrier key exchange is carried out before disconnecting.

This trust system is scalable in comparison to the Web of Trust model. The use of a Direct and Approved Key List allows the introduction of new nodes into

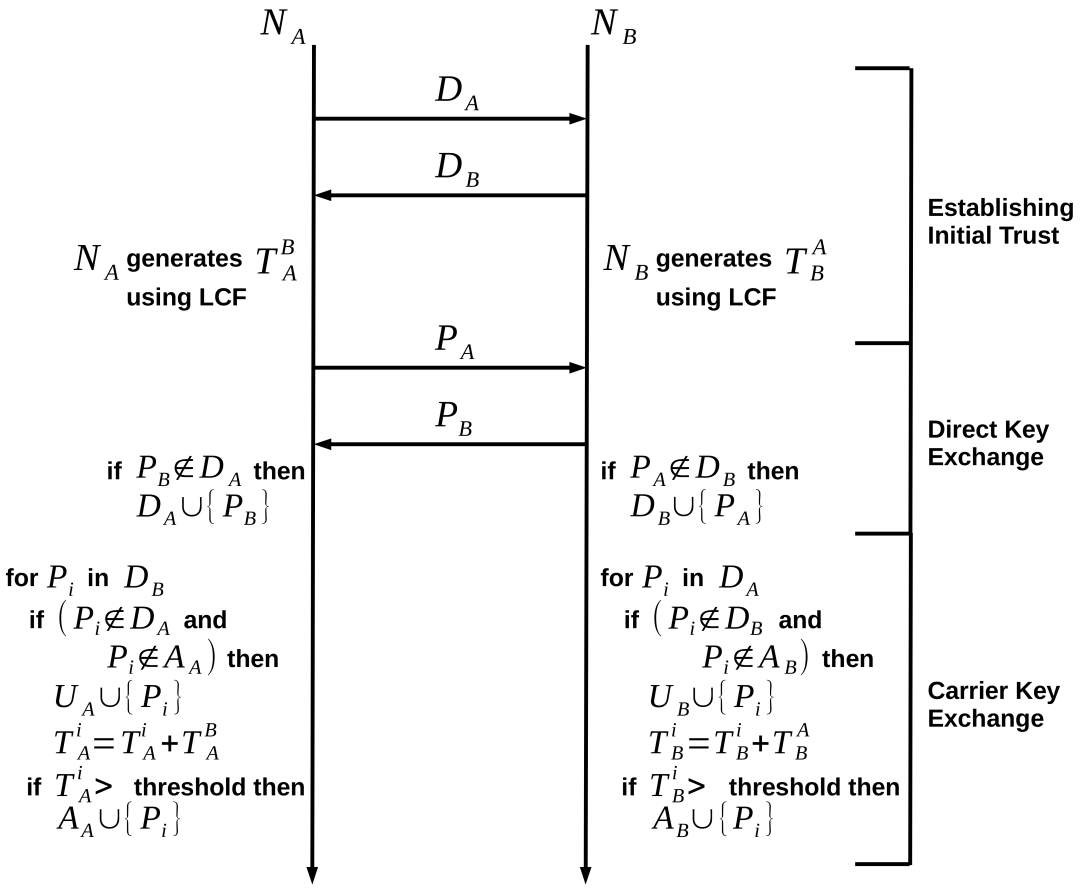


Figure 3.5: Public key exchange with LCF trust assignment.

the community at a later time without creating segregated networks. Initially, new nodes introduced into the network will have a low trust rating. However, as they move, meet, and interact with already established nodes their trust rating will increase over time.

The efficient distribution of public keys in large scale DTN deployments is a desirable property. The use of carrier nodes to assist with public key distribution allows nodes to communicate with nodes they have not met. This provides a scalable network that is not dependant on any other infrastructure for public key distribution.

3.5.2 Adversary Setup

A single Black Hat node was introduced into the simulation. This simulated the event of an attacker physically compromising a node in the network. A single node was selected to observe the singular effect that one Black Hat node would

have on the network. The designated node would perform a key spoof attack by changing all the keys in D_M and A_M to the Black Hat Public Key P_M. It would then distribute these spoofed ID keys. Upon meeting another node (N_A), the Black Hat node would transfer the Black Hat public key (P_M) and other spoofed ID keys ($P_{(M,i)}$) whilst accepting the public key of the neighbouring node (P_A), and then proceed to generate a spoofed ID public key ($P_{(M,A)}$).

3.5.3 Experiments

An experiment consisted of three scenarios, each using different methods of establishing initial trust. The first scenario called *Control*, utilised an absolute trust method. The second scenario called *Random*, utilised a random trust system similar to the one proposed by [72]. The third scenario called *LCF*, utilised the new LCF method described in Section 3.3. For each experiment, a random movement model was generated and replayed for all the three scenarios. This allowed the same movement and connections to nodes to be replayed for every different method of establishing initial trust. The way each methodology of establishing initial trust changed the rate of key distribution was observed.

Each scenario was simulated for a total of 10,000 seconds with 100 nodes assigned in a 100m by 100m square grid. Nodes travelled at 1m/s with a communications range of 1 metre. When detecting a neighbouring node, it would engage in key exchange for a total of 60 seconds to simulate a worse case scenario time for nodes to handshake, exchange keys using various key exchange protocols and transfer additional data such as messages. During this period, the two nodes would generate an initial trust value using one of the three methods, exchange their own public keys, and public keys in D_i, which would become approved keys provided they exceeded the trust threshold. Upon completion of the public key exchange process, nodes would disconnect and resume movement.

Using the Bluetooth communications standard [10], with a conservative Class 2 radio device range of 1 metre, the estimated baud rate provided is between 3 to 24 megabits per second (Mbits/s). Using a 256 bit ECC public key, and assuming a public key package size of 300 bytes to include additional key metadata and information, a node transferring a full keyring of 100 public keys could be completed in 0.01 to 0.08 seconds. This provides ample time during the 60 seconds for two meeting nodes to establish initial trust, handshake, and exchange public keys and other messages.

Table 3.3 provides a summary of the simulation parameters. Jia et al. [72] used a trust threshold value that was 0.1 above the highest trust value that could be assigned to a node. Since the Random scenario generated a trust value between 0 and 1, the trust threshold value was set at 1.1 for both Random and LCF scenarios. The threshold value was also set such that a public key could not be approved by just one trustworthy node, but had to be received by a minimum of two nodes for approval. The threshold was set to match the threshold in [72] to allow comparison of results.

Table 3.3: Experimental Simulation Constants

Parameter	Value
Experiment Environment	
Environment Size	100 x 100 Metres
Duration	10,000 Seconds
Movement Model	Random
Node	
Number of Nodes (N)	100
Node Speed	1 m/s
Node Wait Time	N/A
Communications Standard	Bluetooth
Communications Range	1 Metre
Key Exchange Time	60 Seconds
Trust	
Trust Range [lower, upper]	[0, 1] Continuous
Initial Trust Value	0.5
Trust Threshold	≥ 1.1
Black Hat Nodes	
Number of Black Hat Nodes	1

In total, six experiments were conducted, each with three scenarios. The Random scenario, was run three times for each experiment, and an average was taken. Figure 3.6 depicts the placement of the nodes in the simulation space. The X and Y axis depicting the XY co-ordinate position of the nodes, and the Z-axis depicting the number of keys in the Direct and Approved Key List. It shows a uniformly distributed placement and movement of nodes in the simulation. However, due to the randomness of the initial node placement, this may not always occur.

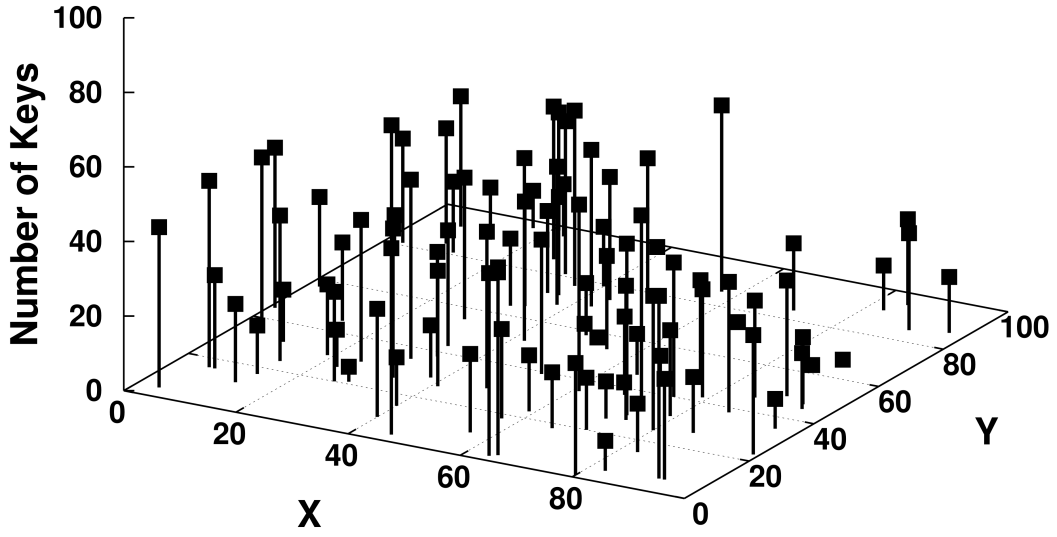


Figure 3.6: Placement of nodes in simulation space and number of keys for each node.

3.5.4 Security and Performance Evaluation Metrics

Three security and performance evaluation metrics were measured in this simulation. These metrics are used to determine whether the security properties identified in Section 3.2.6 are met by the proposed key distribution scheme. The security and performance evaluation metrics outlined are *Public Key Distribution Efficiency*, *Black Hat Public Key Distribution*, and *Spoofed ID Key Distribution*.

1. *Public Key Distribution Efficiency* is a measure of the speed at which public keys are distributed in the DTN. The more instances of a node's public key in the network increases the confidence that the node actually owns the public key. This makes it difficult for a Black Hat node to distribute a Spoofed ID Key tied to the identity of another node. Therefore, it is desirable for public keys to be distributed quickly amongst the nodes in the bootstrapping process. Using a two tiered key system with Direct Key and Approved Key Lists, two sub-properties of Public Key Distribution Efficiency can be measured. The Direct Key List provides a list of other nodes that a given node has met. While the Approved Key List provides a list of nodes that the node is aware of exists, but has not yet directly met. After the deployment process, nodes meet and exchange keys. It is expected that the number of public keys in the Direct and Approved

Key Lists will continue to increase over time. At a certain point, all nodes will become aware of all other nodes, where the number of nodes in both the Direct and Approved Key List equal the total number of nodes in the network. As time approaches infinity, it is expected that the Approved Key List will decline as nodes meet the other nodes they knew existed but had not met previously. In this instance, the approved keys are promoted to direct keys.

2. *Black Hat Public Key Distribution* is the measure of how widespread the Black Hat Public Key (P_M) as defined in Section 3.2.3 is distributed amongst the nodes. The provision of public key confidence should be able to mitigate the distribution of such keys. This metric includes two sub-metrics that are measured.

   (a) The number of public keys the Black Hat node is able to collect from neighbouring nodes. The more public keys and identities the Black Hat node can collect from legitimate nodes, the more spoofed ID keys $P_{(M,i)}$ it can generate.

   (b) The number of White Hat nodes that have the P_M in either D_i or A_i. When a White Hat node receives P_M they also become carriers of this key, and may pass it to other nodes it may meet.

   Better security is achieved then the distribution of P_M is mitigated, whilst assisting the distribution of legitimate public keys.

3. The final evaluation metric measured is the number of *Spoofed ID Keys Distributed* ($P_{(M,i)}$) throughout the system. Spoofed ID Keys differ from the Black Hat Public Keys in the second evaluation metric, in that they explicitly exploit the identity-key binding as defined in Section 3.2.3. The robustness of the key management and distribution scheme is measured by the Spoofed ID Key Distribution. Public key confidence should also be able to mitigate the distribution of $P_{(M,i)}$. These keys allow the Black Hat node to eavesdrop on communications intended for the legitimate node, and allow impersonation of the victim node.

3.6 Results and Analysis

This section presents and discusses the experimental results on the three evaluation metrics discussed in Section 3.5.4. They are evaluated to determine whether the key distribution scheme fulfils the security properties identified in Section 3.2.6.

3.6.1 Public Key Distribution Efficiency

The first metric measured is the Public Key Distribution Efficiency. An efficient distribution of public keys directly, as well as an efficient distribution of carried (indirect) keys fulfils Security Properties 1 and 2 from Section 3.2.6. Figure 3.7 depicts the percentage of keys in the system over time for the Direct, Approved and combined (Direct+Approved) key lists in the LCF scenario. Full key distribution results can be found in Appendix A. The results show the keys distributed directly increase linearly over time. At the end of the simulation, only 47% of keys distributed were directly exchanged. The Approved key distribution results allow an additional number of nodes to communicate using indirectly trusted keys, thereby fulfilling Security Property 2. The results showed the percentage of Approved keys exceeding Direct keys after 2,500 seconds into the simulation where it peaks at around 7,000 seconds before declining. The decline in approved keys is due to nodes directly receiving a public key currently in their A_i, thereby upgrading the approved key to a direct key. The system and mobility model of the simulation would result in all nodes eventually meeting every other node, and the approved key distribution declining to 0%. However, in a large and open system model, the approved key distribution would be useful to facilitate secure communications to an additional number nodes, as not all nodes will meet every other node.

Figure 3.8 compares the public key distribution of both direct and approved keys for each scenario. It shows that the Control scenario provides the most efficient key distribution with each node averaging 100 keys after 10,000 seconds. This indicates that each node has either met or is aware of the other 99 nodes in the simulation, and is capable of establishing secure end-to-end communications. The Random and LCF scenarios show a slower, but still effective, public key distribution. It is interesting to note that the Random scenario was slower than the LCF scenario. The LCF trust system, resulted in a more relevant initial trust

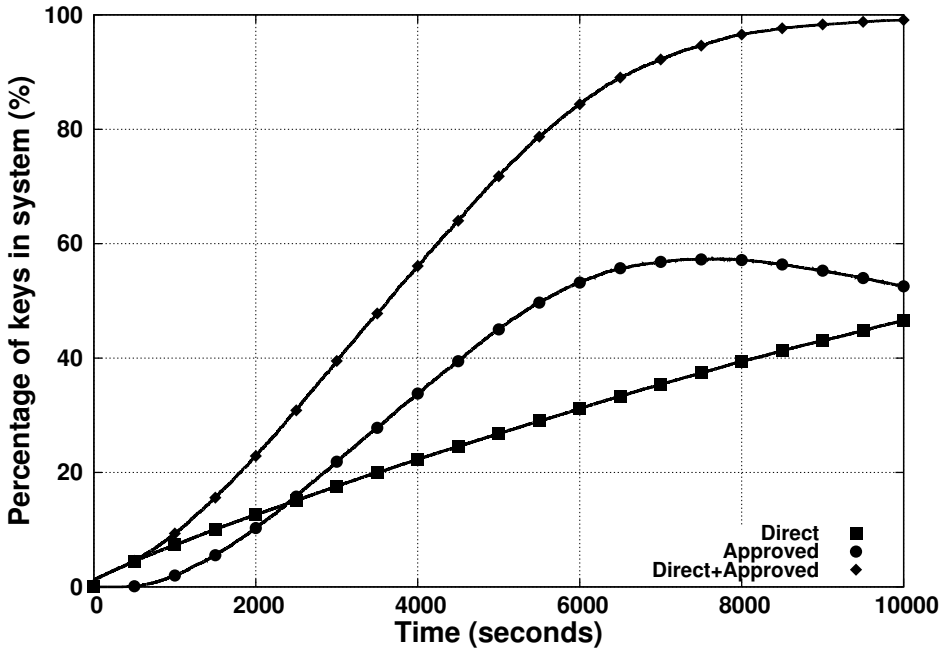


Figure 3.7: Direct and Approved key distribution over time for LCF Scenario.

value and a trustworthy distribution of keys in comparison to a randomly generated trust value in the Random scenario. This is demonstrated when analysing the time taken for each scenario to distribute 50% of the keys. The Control scenario distributed 50% of the keys in approximately 2,500 seconds, the fastest of all three scenarios. The LCF scenario distributed the same amount of keys in approximately 3,600 seconds, whilst the Random scenario took the longest requiring approximately 4,200 seconds to distribute 50% of the keys.

When compared to Jia et al. [72], the Random scenarios indicate similar trends in experimental results, with minor differences in public keys distribution. This is likely due to the difference in simulation engines. Jia et al. [72] utilise the NetLogo [131] simulator, whilst this chapter utilises a fully customisable DTN simulator. Furthermore, experimental setup information necessary to replicate the random movement model in [72], was also not defined. Random movement could either be random direction, or random waypoint (direction and speed) [73], where nodes randomly generate a destination, path and speed. In this experiment, the direction was randomly generated, but the speed was kept constant at 1m/s.

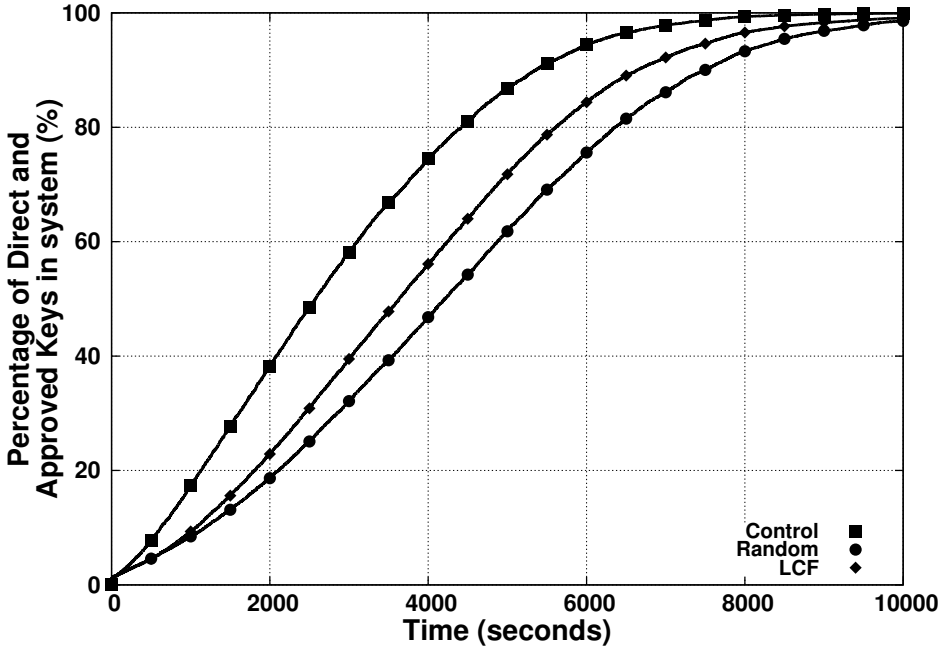


Figure 3.8: Direct and Approved key distribution over time for the different trust systems.

3.6.2 Black Hat Public Key Distribution

The second evaluation metric is the Black Hat Public Key Distribution. Mitigating the effect a Black Hat nodes has in the network fulfils Security Property 3 identified in Section 3.2.6. This metric can be further divided into two sub-metrics, *Black Hat node with Black Hat Keys*, and *White Hat nodes with Black Hat Keys*.

Table 3.4 shows that using the Random method to establish initial trust, provides little or no additional security compared to the Control scenario, for the prevention of Black Hat public keys being distributed. Some experiments (see Appendix B) show that more Black Hat public keys are approved in the Random scenario at the end of the simulation than the Control scenario.

Black Hat node with Black Hat Keys is the number of public keys N_M was able to obtain in both the D_M and A_M. For each experiment, the contents of D_M between all scenarios were identical. This was due to the same movement pattern being replayed for all three scenarios, resulting in the same node meetings. However, the contents of A_M are different due to different trust systems

Table 3.4: Experiment 1 Results

	Key List	Control	Random			LCF
			1	2	3	
Black Hat Node with Black Hat Keys	Direct	42	42	42	42	42
	Approved	58	58	58	58	3
White Hat Nodes with Black Hat Keys	Direct	40	40	40	40	40
	Approved	50	51	54	50	53
Spoofed ID Keys	Approved	30	16	25	29	2
Totals	Direct	82	82	82	82	82
	Approved	138	125	137	137	58
	All	220	207	219	219	140

Table 3.5: Averaged Experimental Results

	Key List	Control	Random	LCF
Black Hat Node with Black Hat Keys	Direct	46.33	46.33	46.33
	Approved	53.67	53.33	3.83
White Hat Nodes with Black Hat Keys	Direct	43.67	43.67	43.67
	Approved	44	47.78	48.17
Spoofed ID Keys	Approved	50.67	53.22	6
Totals	Direct	90	90	90
	Approved	148.33	154.33	58
	All	238.33	244.33	148

between the scenarios. In Tables 3.4 and 3.5, this is called Black Hat node with Black Hat Keys. In Table 3.5, the Control and Random scenarios have similar results with 53.33 and 53.67 respectively, averaged between all conducted experiments. Investigation of the Approved Key List for the Random scenario, found that many of the keys were approved late in the simulation. Even with a random trust value being assigned to a carrier node, if the carried key was received from enough sources to exceed the trust threshold, the key could still be approved. For the Random scenario, this typically occurred late in the simulation. In the LCF scenario, nearly all experiments showed poor results in obtaining carrier keys from other nodes. Table 3.5 shows an average of 3.83 keys. This is due to the trust system employed.

In the LCF scenario, during the process of examining common contacts, a node will check both the identities and public keys in D_i of the neighbouring node. If they are identical, the trust value is incremented. However if the public key of the node is different, the trust value is decreased significantly at a rate

of t_d per discrepancy. This is designed to establish a low initial trust value for nodes that carry false keys. Due to the lack of a centralised key manager or trust and reputation system, the only method of checking whether the node is carrying spoofed keys is to compare public keys of nodes in common to both nodes. Since the node assumes itself trustworthy, if there is a discrepancy in the compared public key, it will assume the other node is carrying a spoofed key. The Black Hat node carries multiple spoofed public keys ($P_{(M,A)}$, $P_{(M,B)}$, ..) that it regards as correct and fully trustworthy. Therefore, if it meets another node with legitimate public keys, it will assume that the other node is spreading spoofed public keys and degrade the trust rating. With the introduction of multiple Black Hat nodes, it is expected to segregate the network into the two groups of White Hat and Black Hat nodes.

The second sub-metric, White Hat nodes with Black Hat Keys is the number of White Hat nodes that have received P_M, in either D_i or A_i. Again, as the movement model is the same for all the three scenarios, it was expected that D_i would be identical between the scenarios. However, the contents of A_i would be different based on the trust system. In Tables 3.4 and 3.5, this is called White Hat nodes with Black Hat Keys. The results indicate that both the Random and LCF scenarios provide little security in preventing this. Table 3.5 shows the average of all six experiments. The Control scenario had 44 Black Hat public keys distributed as a baseline. The Random and LCF scenarios distributed slightly more Black Hat public keys averaging 47.78 and 48.17 respectively. In particular for the LCF scenario, the Black Hat public key is distributed by White Hat nodes that have directly met the Black Hat node. They then receive the Black Hat public key in D_i, and distributed them to other White Hat nodes through the approval process. Since the LCF scenario is designed to establish a more appropriate initial trust value, the Black Hat key is still distributed slightly better than both the Control and Random scenarios. Practically, this is an acceptable result as it allows the Black Hat key owned by the Black Hat node to be disseminated through the network.

3.6.3 Spoofed ID Public Key Distribution

The third evaluation metric measures the Spoofed ID Public Key Distribution. Limiting the distribution of Spoofed ID Public Keys in the network fulfils Security Property 3 identified in Section 3.2.6. These are public keys that have a White

Hat identity but have the Black Hat public key ($P_{(M,i)}$). $P_{(M,i)}$ keys pose a larger threat to secure communications in the DTN as it allows the Black Hat node to eavesdrop and modify messages being routed through a store-carry and forward scheme [53]. Table 3.4 shows the penetration of the Black Hat Public key in the system for one of the experiments conducted, and Table 3.5 depicts the averaged results for all six experiments. Results for all experiments can be found in Appendix B. The summation of these three metrics was also measured over time. Figure 3.9 depicts the average of all six experiments showing the distribution of Black Hat keys over time.

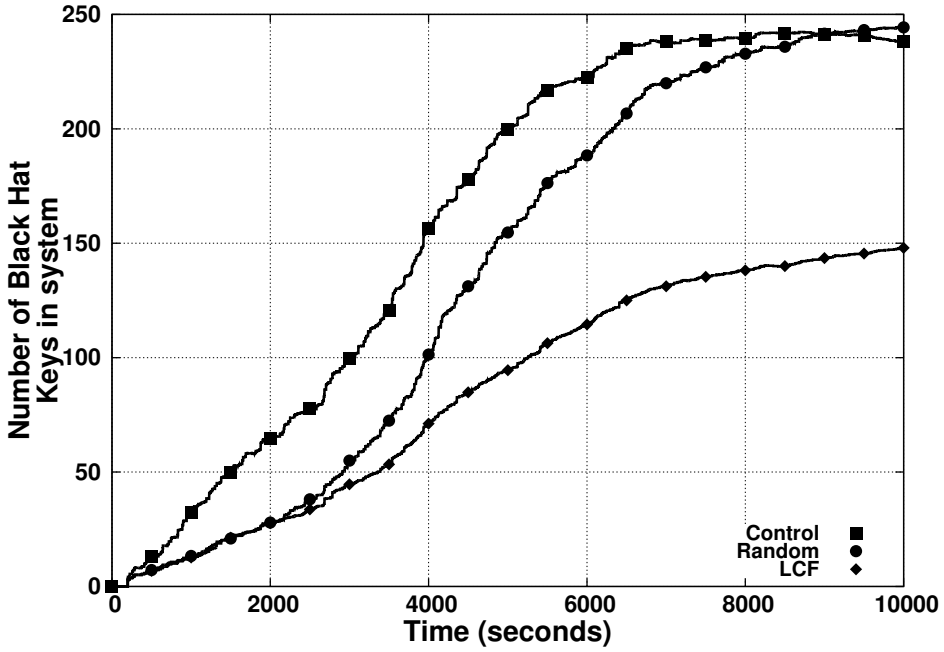


Figure 3.9: Black Hat key distribution over time for the different trust systems.

From the experiments, it is evident that only the LCF trust system successfully mitigates the distribution of $P_{(M,i)}$ keys, averaging 6 instances as shown in Table 3.5. This is an effective solution as the system segregates the network into White Hat and Black Hat nodes. The Random scenarios show little to no effect on mitigating the distribution of such keys, and in some experiments, performed worse than the Control scenario. The Random scenarios averaged 53.22 falsified keys, which was higher than the Control scenario of 50.67 Spoofed ID keys. The mitigation of distributing Spoofed ID public keys by a Black Hat node is required

to provide secure communications in such a network. Distribution of such keys allow the Black Hat node to eavesdrop and modify communications, and allow the impersonation of a White Hat node. The results show that the Black Hat node is segregated through trust by other White Hat nodes in the LCF scenario, with a low acceptance of Spoofed ID keys.

The total Black Hat key distribution over time in Figure 3.9 shows that the LCF scenario mitigates a significant amount of Black Hat and Spoofed ID public keys introduced by a single Black Hat node. It shows that the LCF scenario distributed 40% less Black Hat keys at the end of the simulation than the Control and Random scenarios. It also shows that the distribution of Black Hat keys plateaus out around 7,000 to 8,000 seconds. Variations in the results are evident in the Random scenarios for each experiment. Although the movement model was replayed for each scenario, the initial trust value when two nodes meet in the Random scenario was randomly generated. Replaying each simulation would result in variations in the content of A_i due to the random trust value.

From the three security and performance evaluation metrics, it is evident that the LCF trust system provides the best public key confidence in comparison to the random trust system (Random) and no trust system (Control). With a 38% reduction, the LCF trust system is capable of mitigating the distribution of Black Hat public keys in the scenarios presented in this experiment. However, this comes at the cost of Public Key Distribution Efficiency. The LCF trust system takes 44% longer than no trust system to distribute 50% of the public keys. In comparison to the random trust system, the proposed trust system is 16% faster at distributing 50% of the public keys, whilst also more effective at mitigating the distribution of Black Hat public keys by 40%. The results demonstrate that the LCF trust system sufficiently provides public key authentication in a simulated DTN environment.

3.7 Discussion

This section discusses the implications and issues related with the proposed scheme, and results. The implementation of a trust and reputation system to provide public key confidence, and by extension public key authentication for key distributions provides a few interesting issues.

1. The implication of trust, key management are all integrated and cyclical.

Traditional public key authentication schemes such as a CA must first trust in the identity-public key binding before signing it to assert their authority. Subsequently, when an entity wishes to authenticate the public key, they must first trust the signer who asserted the identity-public key binding. Having verified the signature, the entity can trust the public key. In the proposed key distribution scheme, trust and key distribution are parallel. The trust a node (N_A) has of another node (N_B) is dependent on the public keys being distributed. Concurrently, the public keys being distributed are trusted at a confidence level of the trust N_A has of N_B and vice-versa. Thereby, the Quality of Service (QoS) a node provides in distributing public keys, forms the trust value of that node, which the public keys inherit.

2. Trust and reputation systems are not perfect. The implementation of a trust and reputation system addresses the autonomous nature of the DTN environment described. However, there will be issues of White Hat nodes being labelled as Black Hat nodes (false positive), and Black Hat nodes labelled as White Hat nodes (false negative). As a result, the scheme provides some assurances or confidence of the identity-public key binding, and public key authentication is achieved by exceeding a threshold. Additionally, the distributed nature of the proposed scheme does not qualify as a reputation system, but rather a trust system as there is no aggregation or collation of trust scores. Each node retains and manages independent trust ratings of all other nodes it has met. The use of a reputation system may reduce false positives or negatives, but will also be susceptible to gaming from specific adversaries.

3. The proposed LCF scheme can be further applied to the CA model used for the Internet. With over 600 CAs [36] performing certificate operations on the Internet in an attempt to remove the single point of failure issue with only a single CA, it becomes difficult in trusting all of them. Three methodologies to provide trust in CAs could be applied using the LCF scheme. The first is a trust establishment scheme amongst the CAs themselves. The second is a trust establishment scheme amongst end users, and the third is a composite trust establishment scheme consisting of both CAs and end users.

CA trust establishment amongst themselves: CAs could apply the distributed LCF trust establishment scheme to provide a trust score rating for each other CA. Metrics including quantity of certificates issued over a variety of measurements such as time and location could be applied to form an aggregated reputation score for each CA.

CA trust establishment amongst end users: End users could apply the distributed LCF trust establishment scheme to provide a trust score rating for each other CA. Aggregation of trust scores from a large quantity of end users could provide an accurate reputation score of CAs and the certificates issued.

Composite trust establishment scheme: Combining both CA trust establishment amongst themselves and end users would result in a composite trust establishment scheme. Scores of both would provide accurate reputation scores of CAs and the certificates issued, whilst mitigating gaming of scores by one of the parties.

4. The proposed LCF scheme and the results obtained have broader implications in DTN applications, particularly in Vehicle Ad-Hoc Networks (VANETs), where autonomous vehicles are becoming more prevalent in society. The application of the LCF trust system to assist key distribution could be used in Internet of Things (IoT) autonomous networks involving location networks, flash networks, or expendable networks. These are networks where they are created for a specific or particular purpose, deployed as necessary, and with no prior planning.

3.8 Conclusion

Providing public key authentication is a current problem, which is further complicated by the distributed and decentralised characteristics of DTNs and its application with autonomous nodes. Public key authentication is important to assert that the identity-public key binding is valid and has not been modified by an adversary. The consequences of this include the potential to eavesdrop on communications, modify messages, and deceive their origin. Therefore the provision of public key authentication in DTNs is critical and important, particularly in autonomous system networks. The research presented in this chapter

has addressed this critical problem by combining an initial trust establishment system with a key distribution scheme similar to PGP to provide confidence in the identity-public key binding. The main contributions of this chapter are:

- The investigation, development, and evaluation of the LCF trust system: A public key distribution scheme that utilises a trust system to provide public key authentication. Some of the sub contributions include:
  - A key distribution scheme that is efficient in the distribution of public keys using a store, carry, and forward exchange.
  - A key distribution scheme that mitigates the effect an adversary has in distributing spoofed public keys that modifies the identity-public key binding.

The LCF trust system provided a more useful initial trust value between nodes with no prior history, and without using a centralised trust and reputation manager. It also provided an effective mitigation in the distribution of adversary keys by 40%. In particular the distribution of spoofed public keys was significantly reduced from an average of 50.67 keys in the Control scenario to an average of 6 keys in the LCF scenario. These results indicate the combining of a trust system and key distribution scheme can provide public key authentication in autonomous DTNs. This has implications for autonomous VANET applications.

The limitations of this work is acknowledged in that area of selection criteria for trust weightings (t_n, t_c, and t_d). The criteria for t_n was adopted from the cognitive limit of human stable social relationships and correlated with additional research into online social relationships. OpenPGP implementations of GPG and PGP 2.6 criteria determined the criteria for t_c. Although the values selected for these trust weightings was purposely chosen to fulfil the above criteria, variations in values is an area for future experimentation.

Chapter 4 extends the LCF trust system presented in this chapter to also include location based data to assist public key authentication during key distribution. This is to further improve the Spoofed Key Distribution metric against multiple and stationary adversaries. The experimental environment is expanded to a realistic large scale geographic DTN deployment involving vehicular nodes.

Chapter 4

DTN Location Based Key Distribution

The previous chapter described in detail the design, development, and evaluation of a Delay Tolerant Network (DTN) public key distribution scheme that utilised the Leverage of Common Friends (LCF) trust system to provide public key authentication. The proof-of-concept scheme utilised a common and effective linear trust computation engine to provide trust in the identity-public key binding in autonomous nodes in a completely decentralised and distributed environment with no other infrastructure. The scheme was evaluated in a small controlled simulation environment. However, many DTN applications include large scale deployments such as Vehicle Ad-Hoc Networks (VANETs), where large quantity of nodes may be deployed over large geographic areas.

Many prior works in VANET trust and reputation systems, and key management utilise a centralised or infrastructure based approach. The reliance on fixed Road Side Units (RSUs) may be a valid assumption in densely populated areas such as Central Business Districts (CBDs), however may not be valid in more rural or under-developed areas. The simulation and evaluation of the proposed schemes are conducted in small controlled simulation environments over a short period of time. The work presented in this chapter addresses these issues by extending the work presented in Chapter 3, by applying and evaluating the LCF trust system in a realistic VANET environment, and adding location based information to the trust computation. Therefore, this chapter addresses the two

research questions identified in Chapter 1. Research Question 2: *Is it possible to apply a trust or reputation system for DTN Key Distribution for a large scale realistic DTN application?* and Research Question 3: *Is it possible to leverage location data to assist a trust or reputation system for DTN Key Distribution?* Specifically, it focuses on:

- The evaluation of the LCF trust system in a large scale VANET application under attack from a variety of adversaries.
- The design, development, and evaluation of the Location based Leverage of Common Friends (LLCF) trust system to include co-localisation data to assist public key authentication and key distribution.

The structure of this chapter is organised as follows. Section 4.2 provides a detailed outline of the System Model and Security Properties of the network, along with definitions, threat model and adversary capabilities. Section 4.3 presents the newly proposed LLCF trust system to assist in the provision of public key authentication during key distribution. Section 4.4 outlines the experimental methodology including experimental and adversary setup, the experiments conducted as well as the security and performance metrics used to evaluate both the LCF and LLCF trust system. Section 4.5 presents the experimental results and evaluation of how both trust systems provide public key authentication when under attack from a variety of adversaries. Section 4.6 discusses the implications and issues related to the proposed scheme. Finally, Section 4.7 summarises the research and contributions presented in this chapter.

4.1 Background and Related Work

Many past evaluations of trust or reputation systems for DTNs have focused on closed and small scale simulations [108, 59, 60, 104], typically covering small geographic areas, a small deployment of nodes, unrealistic movement models, and a closed system. However, many DTN applications such as autonomous VANETs are open and large scale. These deployments can span large geographic environments with a large deployment of nodes. The vehicles or nodes, have vehicular movement models, and can enter and leave the network freely and at any time, making the deployment an open network. Therefore, there is a need

to evaluate the LCF trust system on a realistic VANET application environment to provide security in a VANET.

Several security requirements are necessary for securing Vehicle to Vehicle (V2V) VANETs. These requirements are distilled from the European Union (EU) [9] and United States (US) [61, 130] V2V Public Key Infrastructure (PKI) proposals. They are Certification and Validation, Scalability and Efficiency, Revocation, and Privacy.

Certification and Validation are the key properties of a PKI and are closely linked to the cryptographic algorithms and protocols. A certificate is essentially the public key that belongs to an entity in the system that is signed by the certificate authority. The certificate is used to ensure that the public key that is received with a message does belong to the entity that it claims. Without certification, an attacker may be able to substitute their own public key for another entity and send messages claiming to be from them. If a PKI possesses the certification property, the system should be able to successfully distribute public key certificates to the entities that they belong to.

Scalability and Efficiency. The PKI for VANETs are large scale infrastructures that are intended to span continents in terms of geographic distances. The means that the number of vehicle, known as nodes, are expected to run into the hundreds of millions. This is a challenge as the only other PKI system that is similar in magnitude is the one provided on the Internet. As a result of the massive scale of the PKI, all computations, communications and storage usage must be carefully considered. Part of this requirement is that these performance metrics does not place a burden on the system. The vehicle communication system must be functional despite the PKI system used to secure it.

Revocation. One of the key functions of a PKI system is the ability to maintain trust and security by notifying nodes of invalid or corrupted certificates. A mechanism must exist that allows the nodes to recognise that the central authority no longer accepts a particular public key. This is a necessary function as it cannot be assumed that secret keys and their associated public keys can be kept secure indefinitely.

Privacy. One of the key requirements of a V2V PKI is that the privacy of the car device is maintained. There are two aspects of privacy. First is operational privacy. There are many situations where the need for cars to be anonymous from other cars and external entities to the system while they are sending and

receiving messages in normal operations. Second is the desire for car devices to be anonymous from the Certificate Authority (CA) and the other components of the system that make up the certificate distribution system.

Both the EU and US V2V PKI standards attempt to provide these requirements with centralised infrastructure-based PKI. However, scenarios exist such as remote and isolated areas where the reliance on centralised infrastructure cannot be guaranteed. Therefore, this chapter will specifically address two of the requirements necessary for securing VANETs; Certification and Validation, and Scalability and Efficiency. More specifically, the provision of these two requirements in a VANET environment where there is no infrastructure, thereby requiring a fully decentralised and distributed key management scheme.

Furthermore, as the nodes in a VANET are autonomous vehicles, they also maintain Global Positioning System (GPS) capabilities for navigation. This co-localisation data can be further extended and input as an additional trust component to the LCF trust system to form a location based trust system to further enhance the capabilities of the LCF trust system.

4.2 Location Based Key Distribution System Model and Security Properties

In this section, the System Model and Security Properties of the application environment identified from relevant literature presented in Chapter 2 are outlined and identified. In addition, the likely application and landscape of the network is also discussed in System Model. Terminology and notations such as types of nodes and keys are defined. The system model is susceptible to attack by adversaries that will attempt to exploit a threat model, which is defined. The capabilities of the adversary are defined along with the extent of their attacks. Having defined the system model of the network, threat model and adversary capabilities, the security properties that the proposed key distribution scheme should achieve is outlined. Throughout this chapter, the notations in Table 3.1 are used to refer to nodes, keys, trust, and black hat nodes.

Table 4.1: Notations

Notation	Description
Node ID Notations	
i	Unique Persistent Identifier i
N_i	Node i
Key Notations	
K_i	Keypair of N_i
S_i	Secret (Private) Key of N_i
P_i	Public Key of N_i
D_i	Direct Key List of N_i - List of public keys received directly from another node
A_i	Approved Key List of N_i - List of trusted public keys received from carrier nodes
U_i	Untrusted Key List of N_i - List of untrusted public keys received from carrier nodes
Trust Notations	
T_i^j	Trust Value N_i has of N_j
t_n	Trust component weighting given to each contact
n	Number of contacts
t_c	Trust component weighting given to each common contact between two nodes
c	Number of common contacts between two nodes
t_d	Trust component weighting given to each key discrepancy instance
d	Number of key discrepancies
t_l	Trust component weighting given to each location
l	Normalised trust score of a particular area
$t_{neutral}$	Initial starting value of trust
Black Hat Notations	
$S_{(m,i)}$	Spoofed Private Key with identity of N_i, generated by malicious N_m
$P_{(m,i)}$	Spoofed Public Key with identity of N_i, generated by malicious N_m

4.2.1 System Model

The system is assumed to be a large scale open VANET, spanning a large geographic area. The deployment environment has no other entities except nodes themselves. There is no centralised PKI or any form of Trusted Third Party (TTP), and there is no public communications infrastructure. Nodes are considered fully autonomous and mobile, requiring no human intervention. They

self-initialise on deployment with no a-priori knowledge of the network or their neighbours. There is no pre-deployment initialisation phase by an offline authority. The key management phases of Moore et al. [100] are adopted, but given the lack of a pre-deployment phase, the key management phases are; initial bootstrapping (distribution), operation, and revocation.

Nodes retain a persistent unique identity [108], and generate Elliptic Curve Cryptography (ECC) public and private key pairs [98, 78]. These keys are used to perform security based tasks such as providing confidentiality, data integrity, and message authentication. It is also assumed that the key pairs have a long but finite time period of validity, similar to Pretty Good Privacy (PGP) where keys may last 1 to 2 years. It is an open network, where nodes freely join or leave, with a large number of nodes in the system at any time. VANET nodes solely rely on V2V communications as there is no other existing infrastructure. They communicate through wireless communications using the IEEE 802.11 specification suite [71]. Due to their mobile nature, they create ephemeral or opportunistic bi-directional connections between neighbours [65]. When nodes connect, they require a defined amount of time to stay within communications range to allow key exchange. Using the Dedicated Short Range Communications (DSRC) specification [71], the time required can be calculated based on the transmission rate and the data bundle size. If the nodes are within communications range for the required time to allow the key exchange process it is performed successfully. However, if there is insufficient time, incomplete data bundles are transferred, and the exchange fails. Partial bundles are assumed to be dropped. These communications connections are used to exchange the public keys they own, as well as the public keys of other nodes they have met and carry. Public key exchange is considered to be a low-cost communications procedure as the exchange occurs at one-hop distances between nodes [13]. The public key exchange is completed over a two channel or side channel scheme [72]. Due to the large number of nodes and geographic size of the system, it is assumed that adversary nodes will exist in the system, and that it might not be feasible to expel such a node. Nodes in a VANET are assumed to be less resource constrained than nodes in a DTN such as in [69, 76, 18, 28]. This is assumed as VANET nodes have greater significant capacity in energy, computation, and memory.

4.2.2 Trust Model

Using the same transitive trust model outlined in Section 3.2.2 [138], the public keys of each node are exchanged when nodes are within communications range. These keys form Direct trust, and are easily verifiable as they were transferred by the node owning the corresponding private key. Carried keys belonging to other nodes are also distributed in the key exchange process. Indirect trust relationships between autonomous nodes are formed if the collective trust values are sufficient similar to the trust model in Chapter 3.

4.2.3 Definitions

The following terminology used throughout this chapter is defined:

1. *Key Distribution* is the process where a node distributes their public key (P_i) to another entity in the network for the use of providing end-to-end secure communications.

2. *Public Key Authentication* - Is the verification of the identity-key binding of a public key. In a decentralised public key distribution scheme such as PGP and the Web of Trust, public key authentication is achieved by the human user confirming the entity's claimed identity is associated with their corresponding public key. It is measured as a boolean (Y or N).

3. *Public Key Confidence* is proposed in this thesis as having *confidence* in the identity-key binding of a public key. In an autonomous DTN, without a centralised PKI, or any other infrastructure but the nodes, verification that the public key being distributed belongs to the associated node is difficult. In a DTN application, public key confidence is how confident the autonomous nodes are that the multiple instances of the same public key they are receiving is actually owned by the node identity. It is a continuous value consisting of the culmination of trust values. When the confidence in a public key has exceeded a threshold, public key authentication has been achieved.

4. *Trust Value* is a real numerical value within a predefined range, that is assigned to a single key or node by an entity describing the level of trust it has in that key or node.

The following terms on how nodes categorise the receipt of keys are defined. These reflect how the node came into possession of the public key.

1. *Direct Key* is a public key that a node has received from the owner - N_A has received the public key of N_B (P_B) directly from N_B. Direct Keys are stored in the Direct Key List (D_A).

2. *Carrier Key* is a public key that a node has received from another node who has previously met the owner of that public key - N_A has received the public key of N_B (P_B) from the carrier N_C. A Carrier Key can either be one of the following:

   (a) *Approved Key* is a public key that was distributed by a carrier node that has exceeded the public key confidence threshold to be trusted - Using the example from Figure 2.7, N_A has received the public key of N_X (P_X) from various carrier nodes (N_C, N_D, and N_E) and is confident that P_X actually belongs to N_X. Approved Keys are stored in the Approved Key List (A_A).

   (b) *Untrusted Key* is a public key that was distributed by a carrier node that has not exceeded the public key confidence threshold to be trusted - Using the example from Figure 2.7, N_A has received the public key of N_Y (P_Y) from various carrier nodes (N_B, and N_C) and is not confident that P_Y actually belongs to N_Y. Untrusted Keys are stored in the Untrusted Key List (U_A).

Table 4.2: Classification of Nodes and their Keys

Nodes	White Hat (N_A)	Key pair (K_A)	Private Key (S_A)
			Public Key (P_A)
	Black Hat (N_M)	Key pair (K_M)	Private Key (S_M)
			Public Key (P_M)
		Spoofed ID	Private Key ($S_{(M,A)}$)
			Public Key ($P_{(M,A)}$)

Table 4.2 provides a summary of the different classifications of nodes, keys and revocation materials. They can be broadly categorised into two types of nodes. *White Hat Nodes* (N_A) are nodes that are legitimate, and have not been compromised by an adversary or attacker. They generate a key pair (K_A), which

consists of a *White Public Key* (P_A) and a *White Private Key* (S_A). *Black Hat Nodes* (N_M) are nodes that have been compromised by an adversary or attacker. Like White Hat Nodes, they possess their own Key pair (K_M), which consists of a *Black Public Key* (P_M) and a *Black Private Key* (S_M). A Black Hat Node as part of their malicious nature may create Spoofed ID Material.

The first form is a *Spoofed ID Public Key* ($P_{(M,A)}$). This is when the Black Hat node generates a key that has the identity association of a White Hat Node, but the key of an Adversarial Node. It is created when N_M changes the identity association of its public key (P_M) from itself (M) to the identity of a White Hat Node (A). This results in a public key that other nodes think belongs to N_A (that is N_A has the corresponding private key) however, N_M holds the corresponding private key.

4.2.4 Threat Model

Given the system model outlined in Section 4.2.1, a similar threat model to Chapter 3 was assumed:

1. *Lack of Infrastructure* - With no infrastructure to assist in key management operations such as public key distribution, nodes will have to handle these operations independently. An adversary node may take advantage of the environment where nodes have to independently distribute their own keys to assume the identity of another node. The lack of infrastructure also affects the implementation of a trust and reputation system as there is difficulty in the aggregation of trust scores to form a reputation system.

2. *Lack of Public Key Authentication* - There is no trusted third party, and therefore no assurance of public key authentication. There is no authenticity between the public key and the identity of the owner [12]. An adversary node is capable of associating its own public key to the identity of another node, distribute this key, and perform a Man In The Middle (MITM) attack.

3. *Physical Tampering* - VANET nodes may be subject to physical tampering [106]. Attackers may gain physical access to a node, modifying the behaviour, public key bindings, and distribute Spoofed ID Keys from tampered nodes. The physical tampering of nodes may be mitigated but cannot

be prevented. As such, nodes will need a mechanism to detect and protect the network against potentially compromised nodes.

4. *Eavesdropping and Modification of Communications* - Since the connection between nodes in a VANET are ephemeral, and with no static routing, nodes may be required to pass on messages between nodes. This provides an adversary the capability to overhear as well as the potential to modify wireless communications between nodes [107]. Due to this threat, nodes will need to establish secure end-to-end communications.

5. *Open and Dynamic Network* - Nodes may be deployed, join, or leave the network at any time. With no TTP to provide public key authentication or detection of adversaries, it is easy for an attacker to deploy new adversarial nodes into the network. Therefore, it is possible for the network with a large population of nodes to always consist of a population of adversarial nodes, which cannot be expelled.

4.2.5 Adversary Capabilities

Adversarial nodes will attempt to exploit the threat model outlined for this network. With a Spoofed ID Key, any adversary node is capable of impersonating another node. This has consequences for the security of communications, and future key management activities such as key revocation. Due to the characteristics of a DTN, an adversary is capable of eavesdropping on communications, and modifying data. These networks require communications to be passed on between nodes in a store and forward method [40]. A message or bundle encrypted with a Spoofed ID Key that an adversary has generated, means that if the adversary is capable of intercepting the message, they can successfully perform a MITM attack [12]. An insider attack model was used, with the adversarial nodes assumed to have similar abilities as outlined by [32] with the following capabilities:

1. Adversarial nodes can obtain any message passing through the network between two other nodes within communications range [40, 12, 107].

2. They are a member of the network and can therefore initiate and receive communications with other nodes in the network.

3. They are able to generate new Black Private and Public Keys (S_M, P_M) as defined in Section 4.2.3 and distribute P_M. S_M and P_M between adversary nodes are independent to prevent White Hat nodes from blacklisting a common P_M between all adversary nodes [32].

4. They are able to generate and distribute Spoofed ID Keys ($P_{(M,i)}$) as defined in Section 4.2.3. This is when they associate their own private and public key pair with another node's identity in their D_M or A_M Lists [12].

5. They are able to remain stationary in an area with high node density to increase the impact of their attack.

Two variations of adversaries with two differing movement models specific to VANET applications [20, 108] are introduced:

1. *Static adversary* movement model is when the adversary nodes are placed in a common location together. The nodes are unable to move, and remain stationary. These take the form of long-term parked vehicles, or fixed devices introduced by the adversary.

2. *Dynamic adversary* movement model is when the adversary nodes are mobile. These take the form of mobile vehicles, which have been compromised to behave as adversarial nodes.

4.2.6 Security Properties

Given the system and threat model of the network, this section outlines the desired security properties the key distribution scheme should achieve. Ultimately, the aim is to achieve public key authentication in an autonomous VANET node during key distribution.

Property 1: A White Hat node (N_A) should be able to generate a key pair (K_A), and distribute the public key (P_A) to establish a secure end-to-end communications channel with other nodes in the network.

Property 2: Any White Hat node should be able to utilise location information to assist the approval process of carried keys to form indirect relationships.

Property 3: The effect of a Black Hat node (N_M), wishing to exploit the lack of public key authentication of a White Hat node (N_A) by distributing a spoofed ID key ($P_{(M,A)}$) should be mitigated.

Property 4: Any White Hat node should be able to utilise location information to assist the identification of spoofed ID keys ($P_{(M,A)}$) by a stationary Black Hat node (N_M) in a high node density location.

Property 5: The trust system should create a segregation between Black Hat nodes and White Hat nodes, thereby making it easy to distinguish between legitimate and adversarial nodes.

4.3 Location based Leverage of Common Friends trust system

A new linear computation trust system called LLCF is proposed to provide public key confidence, and by extension public key authentication, to establish secure autonomous communications. The proposed trust system extends the LCF trust system of Chapter 3 with the inclusion of co-localisation data. The trust relationship between two nodes is assumed to be linear, which is similar to many trust and reputation systems for online retail sites such as eBay and Amazon [74]. Weighted scores similar to [21] were used to provide different trust components for formulate a trust score between two nodes.

The two properties, the number of nodes met (n) and number of common nodes (c), both increase the trust value. Whilst the decreasing trust value (d) is the discrepancy between two nodes over the binding of an identity and the public key. If there is a discrepancy, both nodes will decrease trust value with respect to each other. Nodes will take the default position of completely trusting themselves, and assume their version of the public key is the correct key, whilst the neighbouring node has a spoofed key. It is assumed that trust is diminished significantly faster than trust is increased. Location information can increase and diminish trust based on the reputation of the particular location the key exchange occurs. With the inclusion of location to the calculation of trust between nodes, the linear relationship between common contacts and location to trust can be represented by Equation 4.1. The trust of N_A assigns to N_B (T_A^B) is given by:

$$T_A^B = T_{neutral} + T_{number} + T_{common} + T_{key\_disc} + T_{location} \tag{4.1}$$

Where $-1 \leq T_A^B \leq 1$. $T_{neutral}$ is the starting value of trust. T_{number} is the trust value assigned for the number of nodes N_B has met. T_{common} is the trust

value assigned for the common contacts of N_A and N_B. T_{key_disc} is the trust value assigned for key discrepancies between N_A and N_B. $T_{location}$ is the trust value assigned for the location where N_A received the key of N_B. T_{number}, T_{common}, T_{key_disc}, and $T_{location}$ can be calculated from Equations 4.2 to 4.5.

$$T_{number} = t_n \times n \tag{4.2}$$

$$T_{common} = t_c \times c \tag{4.3}$$

$$T_{key\_disc} = t_d \times d \tag{4.4}$$

$$T_{location} = t_l \times l \tag{4.5}$$

Where t_n is the trust weighting given to number of contacts, and n is the number of nodes N_B has met. The term t_c is the trust weighting given to the common contacts of N_A and N_B, and c is the number of nodes in common with N_A and N_B. The term t_d is the distrust weighting upon discovering a potential false key, and D is the number of discrepancy keys between N_A and N_B. The relationship between t_n, t_c, and t_d is set by initial experiments conducted in Section 3.4

The addition of the term $T_{location}$ provides location based information to establish a trust value between N_A and N_B, with t_l the upper value of trust assigned to a location, and L is the value of trust assigned to a location. Due to the method by which l is calculated, t_l was set as the maximum value the location trust term (Equation 4.5) can contribute to Equation 4.1.

The trust value assigned to a location term l of Equation 4.5, is the value that N_A assigns to the location it received P_B from N_B. As nodes move, they keep a history log of locations they have visited. For each location, nodes log the number of Direct Keys received from other nodes, and the number of key discrepancies during Carrier Key exchange between other nodes. The number of Direct Keys provides a metric of the number of potential White Keys transferred in a particular location. The number of key discrepancies in the Carrier Key exchange process provides a metric of the number of potential Black Keys transferred in a particular location. These keys can only be detected if there are two or more different public keys bound to one node identity. For each location,

nodes independently calculate K_{diff} by Equation 4.6, the difference of Direct Key count (DK_{count}) or White Keys, and number of key discrepancies (KD_{count}) or Black Keys.

$$K_{diff} = DK_{count} - KD_{count} \tag{4.6}$$

$$l = \begin{cases} \frac{K_{diff} - Key_{White(Min)}}{Key_{White(Max)} - Key_{White(Min)}} & \text{if } K_{diff} > 0 \\ -\left(\frac{K_{diff} - Key_{Black(Min)}}{Key_{Black(Max)} - Key_{Black(Min)}}\right) & \text{if } K_{diff} < 0 \\ 0 & \text{if } K_{diff} = 0 \end{cases} \tag{4.7}$$

Equation 4.7 represents how l is calculated. For the locations where K_{diff} is positive, it represents that the number of White Keys exchanged in this location exceeded the number of Black Keys exchanged. The location trust value l is a normalised value of K_{diff} for all locations, such that $0 < l \leq 1$. The location a node assigns $l = 1$ is the potential location where the most number of White Keys are being exchanged. For the locations where K_{diff} is negative, it represents the number of Black Keys exchanged in this location exceeded the number of White Keys exchanged. The location trust value l is a normalised value of K_{diff} for all locations, where $-1 \leq L < 0$. The location a node assigns $l = -1$ is the potential location where the most number of Black Keys are being exchanged. For all locations where $K_{diff} = 0$, meaning the number of White Keys equalled the number of Black Keys exchanged in this location, the location trust value $l = 0$, resulting in the original LCF trust computation engine. The location weighting term t_l was set at 0.5 to ensure that $-0.5 \leq T_{location} \leq 0.5$.

The addition of $T_{location}$ to the computation engine allows nodes to detect locations where adversary nodes congregate and exchange Spoofed ID Keys as defined in Section 4.2.4. This provides additional security against a static adversary, as nodes would compute the adversarial location to be untrustworthy, which would affect the trust value of received keys in that area.

The trust score of each location is calculated independently by each node. It is relative only to the other locations that the node has calculated a trust score of, and is independent of the trust score a differing node will assign to the same location. Since there is no aggregation of trust score by collective nodes, nodes cannot game or influence the location trust of another node. Each

node independently determines the location relative to every other location. At deployment, a node will only be aware of a small number of locations. However, as the time a node operates in a network increases, they will visit an increasing number of locations and become aware of a larger number of locations. The use of a normalised location trust component given by Equation 4.7 takes this into consideration, as nodes will form their own independent view of the world.

4.4 Experimental Methodology

An open source DTN and VANET simulator was developed in Python called *Traffic Djam* [30] that models the decentralised distribution of public keys between nodes using a Web of Trust model. Traffic Djam initialises nodes with a persistent unique node identifier (Node ID) similar to [108], and a public and private key pair signature similar to key fingerprints in PGP [138]. Nodes follow a movement path based on city vehicle movement models. Nodes within a predefined communications range of another node connect and engage in public key exchange whilst in motion. The simulator engine focuses on the security of the application layer of a DTN or VANET.

4.4.1 Movement Model

A movement model based on the Cabspotting [38] project in San Francisco was used. The Cabspotting project tracks the activity of commercial taxi cabs in San Francisco. Each taxi (or node), is assigned a unique anonymous identifying tag, and is tracked using a GPS receiver at intervals of less than 10 seconds. The GPS location in decimal latitude and longitude pairs along with the timestamp and taxi occupancy are updated to a central server. The Dartmouth EPFL Mobility Dataset [110], detailed in [111] was used, taking a 48 hour subset window of this vehicle dataset with the highest concentration of nodes. Within this 48 hour period, a maximum of 497 unique nodes are present. This data is replayed in the *Traffic Djam* simulator.

A simulation square space of approximately 20km by 20km was created around the city of San Francisco, which is depicted in Figure 4.1. Nodes within the simulation space are considered active and are able to perform key exchange with other nodes in the simulation space. Nodes outside the boundaries are considered inactive until they enter the simulation space. The reverse applies to

Figure 4.1: Boundaries of Simulation Space in relation to the City of San Francisco (http://mapbox.com).

nodes leaving the simulation space. Figure 4.2 depicts the movements of a single node in the simulation space. Certain San Francisco road features can clearly be seen, such as the financial district, Mission Street, and the Bay Area Bridge heading to Berkeley, Emeryville, and Oakland.

Nodes may travel outside the pre-defined simulation space boundaries. Over the 48 hour period of the dataset, the number of nodes in the simulation space (active nodes) were measured and compared to the number of nodes outside the simulation space (inactive nodes). Figure 4.3 depicts the number of active nodes in the vehicular movement model over the duration of the simulation. The number of active nodes peak to approximately 360 in the first 3 hours of the simulation before gradually declining to a little over 200 active nodes in the final hour of the simulation.

A location based node frequency analysis was also conducted to determine which areas nodes would congregate, and which areas nodes rarely visited. Figures 4.4a and 4.4b depict the two dimensional and three dimensional node loca-

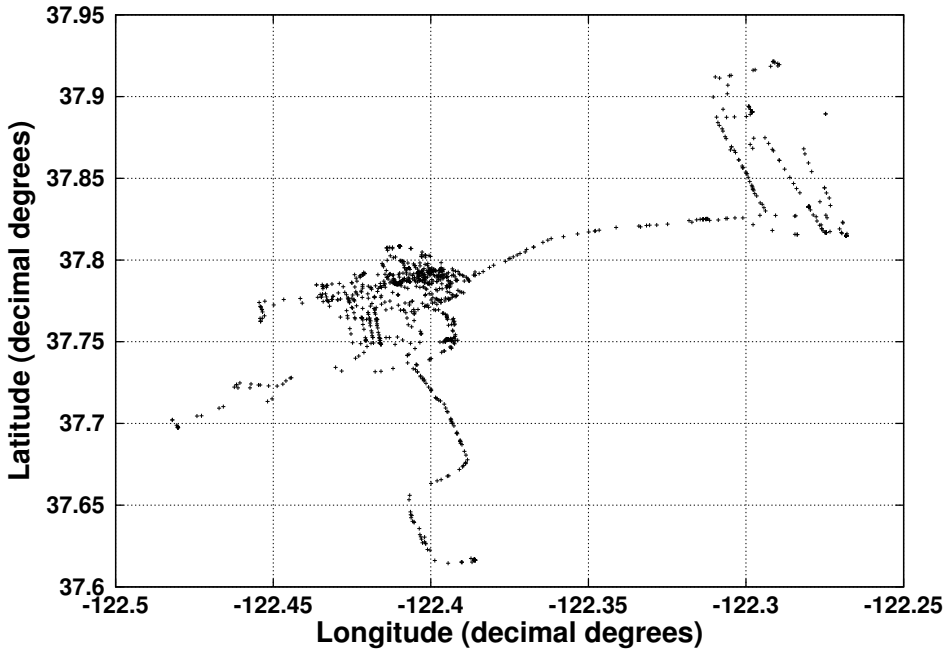


Figure 4.2: Location tracking of a single node in the City Vehicle Model.

tion representation within the simulation space.

The 20km by 20km simulation space was divided into 40,000 100m by 100m cells. For each second of the simulation a snapshot was taken of where the nodes were placed, with each cell counting the number of nodes within its boundary. Over the 172,800 seconds (48 hours) of the simulation, this node count was accumulated to provide an idea of areas where nodes congregated. The analysis found a cell with a significantly higher node count at $(X = 87, Y = 116)$. This represents a busy intersection and street corresponding to Mission Street in Downtown San Francisco.

Due to the open nature of the vehicular movement model, nodes may freely join and leave the network as they enter and exit the simulation boundaries. This movement model affects the Direct Key distribution when compared to a closed random path model. A preliminary direct key distribution analysis was conducted between the vehicular movement model and a random path movement model. Figure 4.5 shows a comparison of Direct Key distribution over time between the Random Path and Vehicular Movement models. As expected in a closed system with a fixed number of nodes such as the Random Path model, as

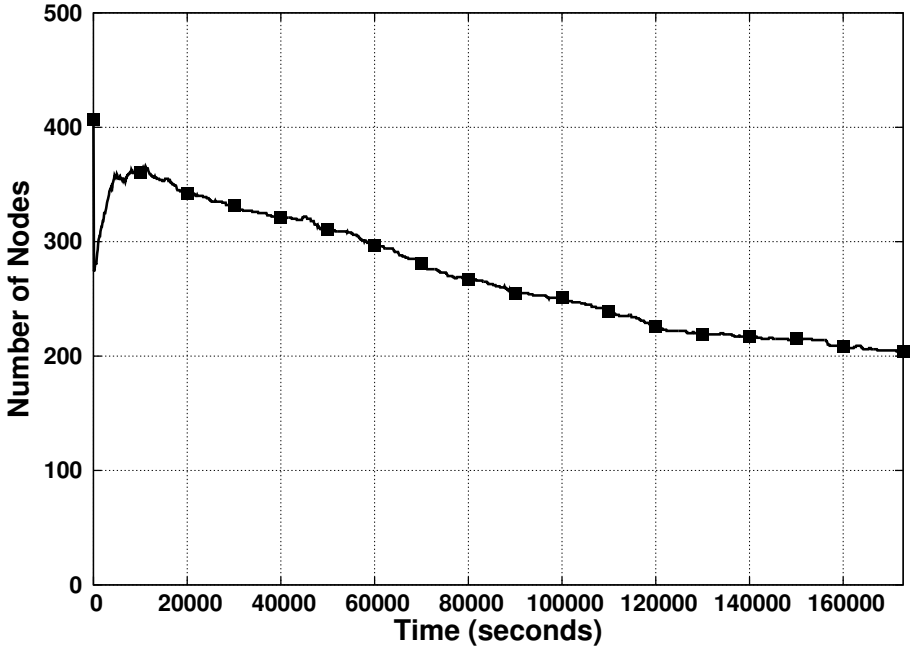
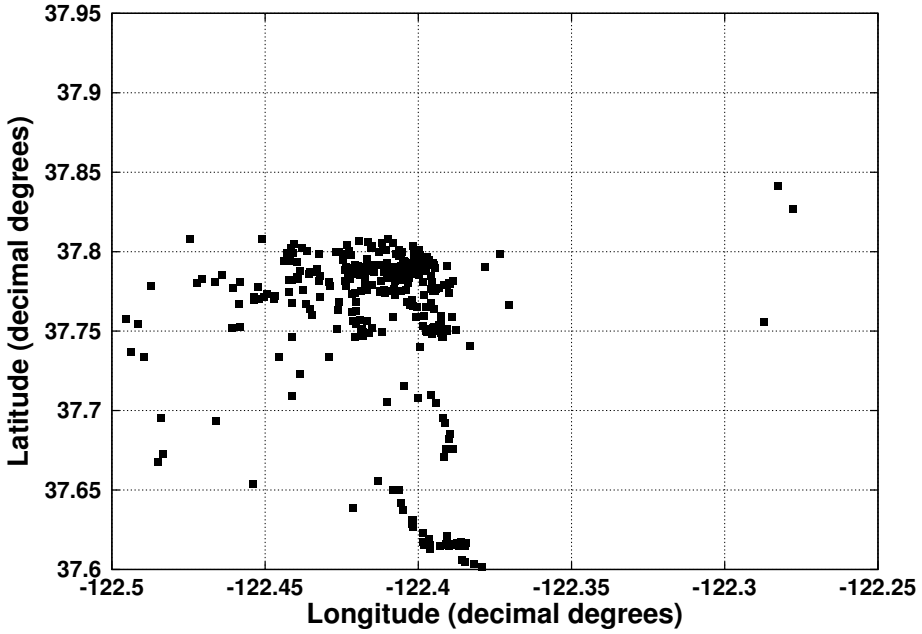
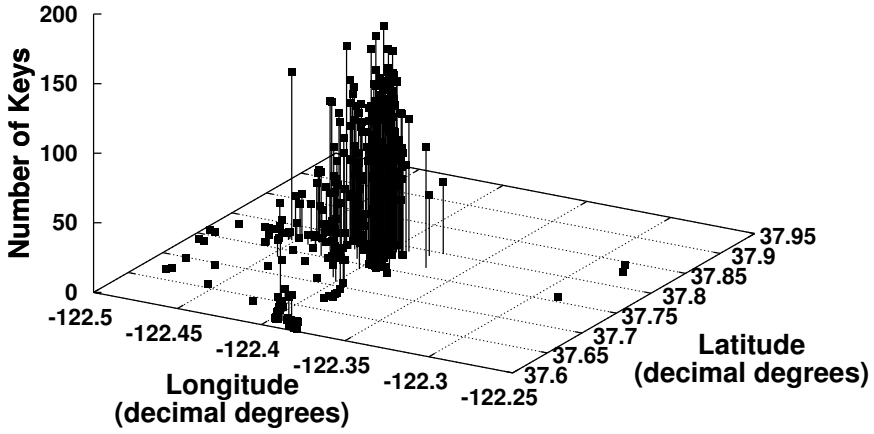


Figure 4.3: Number of active nodes over time.

time approaches infinity (or increases) nodes will eventually meet all other nodes and exchange keys they own resulting in a Direct Key list with the population of nodes. For the Vehicular Movement model, the results differ significantly from the Random Path Movement Model in that only 8.6% of possible keys are distributed. Due to the nature of the movement model where nodes leave the simulation area or become inactive, as time increases, nodes will not eventually meet all other nodes. In a VANET with distributed keys and communications, only reaching 8.6% of possible nodes would hinder communication protocols [53]. To further extend the number of communication partners in a distributed VANET, the carrier and approved key system used in Chapter 3 [31] was adopted. With the use of both Direct and Approved Keys, Figure 4.6 shows that nodes in a VANET are capable of communicating with up to 51% of the population by the end of the simulation.



(a) Two dimensional representation



(b) Three dimensional representation

Figure 4.4: Placement of nodes in simulation space.

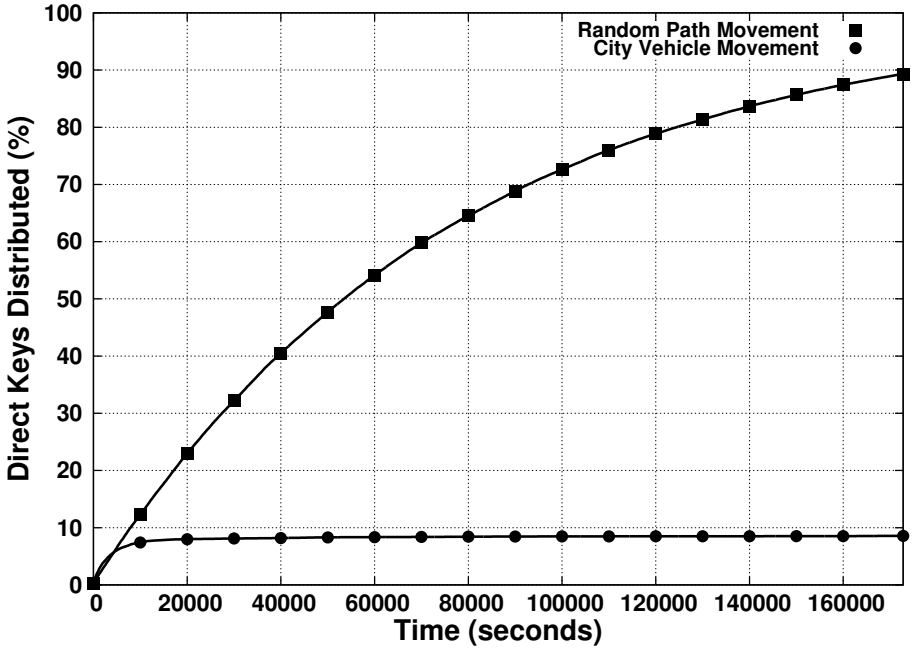


Figure 4.5: Direct key distribution over time for Random Path and City Vehicle Movement Models.

4.4.2 Experimental Setup

Using the vehicle movement model described in section 4.4.1, it is assumed a similar node deployment environment to Chapter 3, with no public infrastructure, no trusted third party, and no central point for node bootstrap or initialisation. Nodes solely rely on V2V communications and self initialise when deployed. In this experimental setup, public key authentication of carried keys cannot be easily verified in a distributed system when compared to a centralised architecture. As a result, adversaries can easily attack the system. An approval system similar to Chapter 3 [31] was used to provide a three-tiered key trust system: Direct, Approved and Untrusted Keys. The receiving node may approve or reject the carried public key based on the trust value of the carrier node it received the key from and the trust algorithm.

The first method implemented is *Control* scenario. The Control scenario is a public key exchange between two nodes using no trust system. When two nodes detect each other within communications range, they initiate key exchange. N_A sends P_A to N_B, and N_B also sends P_B to N_A. These are then added to D_A

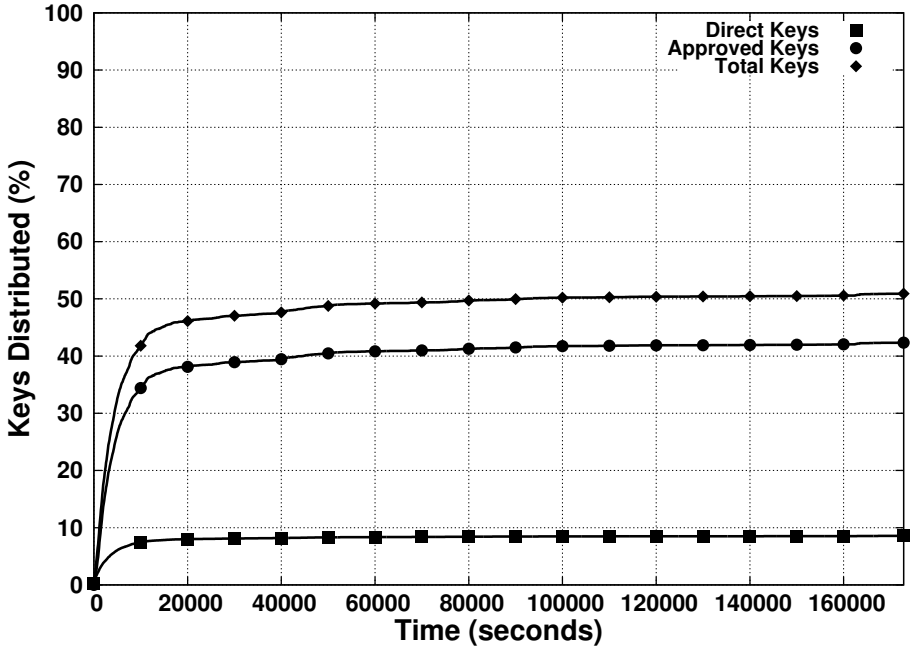


Figure 4.6: Key Distribution over Time for City Vehicle Movement Model.

and D_B respectively. N_A may potentially provide false information about D_A to N_B. This can be mitigated by sending a cryptographic hash or Hash-Based Message Authentication Code (HMAC) as a challenge-response of the D_i prior to sending the list. When N_A sends D_A to N_B, N_B also sends D_B to N_A in the Carrier Key exchange process. Each node scans the list and finds nodes that are not in the respective D_i or A_i and adds them to their respective A_i. This key exchange process occurs whilst the nodes are moving. If the nodes fall out of communications range whilst the key exchange process is occurring, then a failed exchange occurs and the nodes discard the partial data bundles.

The second method of public key exchange for autonomous VANET applications is using the LCF trust system as detailed in Chapter 3. The process is similar to the Control scenario, with the exception of an initial trust establishment phase prior to the Direct Key exchange phase. This is where both nodes send the respective D_i to each other. The trust value of a node is then calculated using these lists. The Carrier Key exchange phase for LCF has the addition of an approval process where the trust of carrier nodes is accumulated. If the trust value exceeds the public key confidence threshold, the Carried Key is approved

and added to the respective A_i.

The third method of public key exchange for autonomous VANETs is using the proposed LLCF trust system as presented in Section 4.3. Figure 4.7 shows the establishment of initial trust phase. In the LLCF scenario, each node determines its location and generates $T_{location}$ from Section 4.3 during the establishing initial trust phase. The location value contributes to the LLCF trust value assigned to the other node. The Direct Key and Carrier Key phases occur the same as the LCF scenario, with an additional step. Before disconnecting from each other, the nodes individually update their location data l based on the number of White Keys and Black Keys detected, and normalise all the results in the location table.

4.4.3 Adversary Setup

A varying number of adversary nodes was introduced into the system at $t = 0$. For the static adversary node experiments, the adversary nodes were introduced into cell $(X = 87, Y = 116)$. From the location based node frequency analysis in Section 4.4.1, this cell has the highest density of nodes, and provides a worst case scenario for an attack on the system. The adversary nodes for this scenario remain in this cell for the duration of the simulation. For dynamic adversary node experiments, the adversary nodes were introduced based on the movement model of the individual node. As a result, the adversary nodes are mobile for the duration of the simulation.

The adversary node has the capabilities as detailed in Section 4.2.5. Upon receiving keys from a neighbouring node, the adversary node will change the key to its own private and public key pair and proceed to distribute these Black Keys to other nodes. This is essentially a key spoof attack. Each adversary node would generate its own private and public key pair instead of using a master Black Key common to all adversary nodes. This increases the difficulty for a White Hat node to detect an Adversary Node from a common master Black Key. Location spoofing is also considered out of scope for the adversary capabilities in this experiment.

4.4.4 Experiments

An experiment consisted of three scenarios, each using *Control*, *LCF* or *LLCF* methods of establishing initial trust. For each scenario, the same movement

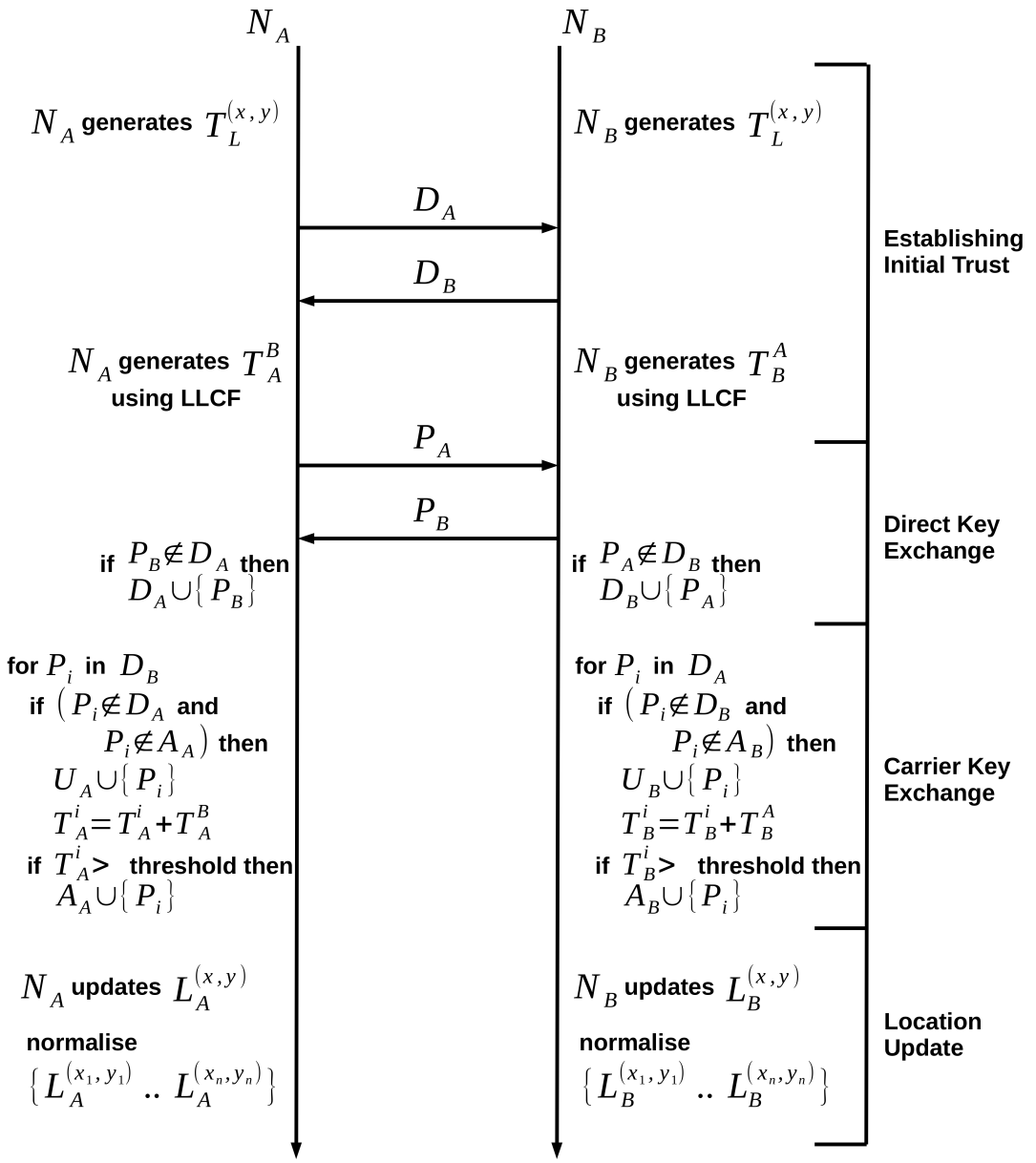


Figure 4.7: Establishing initial trust between two nodes for LLCF Scenario.

model and node connections were replayed. This allowed the observation and measurement of how each methodology of establishing initial trust affected key distribution. Table 4.3 provides a summary of the experimental simulation constants. Each scenario was simulated for a total of 48 hours or 172,800 seconds in a 20km by 20km square. A total of 497 nodes was used as constrained by the number of taxi cabs in the Cabspotting dataset. The communications range was set using a conservative assumption of 100m for DSRC communications [108].

The DSRC communications standard has a similar baud rate to the Bluetooth specification used in Chapter 3, of 3 to 27 megabits per second (Mbits/s) [85]. Using the same 256 bit ECC public key, and assuming a public key package size of 300 bytes to include additional key metadata and information, a node transferring a full keyring of 497 public keys could be completed in 0.04 to 0.4 seconds. Unlike the node connection model in Chapter 3, the nodes in this experiment remain moving, and do not stop during connection. Therefore, the simulator communication model determined whether two nodes were within the 100 metre communications range of each other for a significantly greater time period than the 0.4 seconds worst case scenario. If the two nodes had sufficient time in communications range, the data bundle was transferred successfully, otherwise partial data bundles were dropped. Since the vehicular nodes travel at differing speeds, a sub-routine in the simulator would perform a high frequency check to determine whether there was sufficient time between two nodes within communications range to transfer their respective data bundles. It is also assumed that there is minimal interference from buildings and obstacles in the communications model.

For scenarios that used a trust system to establish initial trust between nodes, the initial trust value was set as 0, and could range from a continuous value between -1.0 to $+1.0$ similar to [20]. These values also correspond to normalised trust scores, commonly utilised by trust and reputation systems [105]. The values of -1.0 and $+1.0$ represent complete un-trustworthiness and trustworthiness respectively. The public key confidence threshold was set to greater than 1.0 for both the LCF and LLCF scenarios. The threshold value was set so that a public key had to be approved by a minimum of two or more nodes. For both the LCF and LLCF scenarios, the initial starting value ($t_{neutral}$) was set at 0 to reflect a neutral trust value. The trust constants t_n and t_c were set the same as the experiments in Chapter 3. The distrust weighting (t_d) for having a false key in possession was heavily weighted at 0.05 to reflect that it is easier to distrust than a node to trust. These figures were set from initial data analysis of the model.

In total, 10 experiments were conducted, each with the three scenarios using trust establishment methods; Control, LCF, and LLCF. The number of Black Hat nodes was varied from 0, 1%, 10%, 20%, and 30% of node population. This was conducted for both the static and dynamic Black Hat movement models. The number of adversaries in the network is not a required parameter for both

the LCF and LLCF trust systems to work. The variations of Black Hat nodes is to determine the limitations of the schemes, in particular the Black Hat node population thresholds where the LCF and LLCF begin to break down. This work, associates each adversary node to a Byzantine failure in the Byzantine fault tolerance system, or a traitor in the generalised Byzantine Generals' Problem. Common solutions to Byzantine fault tolerance require the number of failures (traitors) should not exceed one third of the total nodes (generals). Hence, only an adversary population of less than 33% was analysed [81].

Table 4.3: Experimental Simulation Constants

Parameter	Value
Experiment Environment	
Environment Size	20 x 20 Kilometres
Duration	172,800 Seconds (48 Hours)
Movement Model	Vehicular Taxi Cab
Node	
Number of Nodes (N)	497
Node Speed	Dependant on movement model
Node Wait Time	Dependant on movement model
Communications Standard	DSRC
Communications Range	100 Metres
Trust	
Trust Range	[-1, 1] Continuous
Initial Trust Value	0
Trust Threshold	>1.0
Black Hat Nodes	
Number of Black Hat Nodes	[0%, 1%, 10%, 20%, 30%]
Location of Static Black Hat Nodes	$(X = 87, Y = 116)$

4.4.5 Security and Performance Evaluation Metrics

Four security and performance evaluation metrics were measured and investigated in this experiment. They are used to compare the effectiveness of different trust systems and whether they are suitable in providing public key authentication in an autonomous VANET. They are *White Key Approval and Black Key Approval*, *Spoofed ID Key Distribution*, *Key Trust Metrics*, and *Public Key Distribution Quantity*.

1. *White Key Approval and Black Key Approval* is the measure of how many White Keys and Black Keys received from carrier nodes were approved to be trusted. These keys had exceeded the public key confidence threshold for the node to trust them and have ultimately been added to the respective A_i. Because these keys were not received directly from the node that owns them, but instead received from carrier nodes, it is of interest to observe how effective a trust system is by determining which keys should be trusted and added to the respective A_i, and which keys should remain untrusted in the the respective U_i. It is the desire for the trust system to allow White Keys to be approved, while preventing the approval of Black Keys. This metric is calculated as a percentage of White Carried keys and Black Carried keys that were distributed.

2. *Spoofed ID Key Distribution* is a measure of the quantity of Spoofed ID Keys throughout the system. An adversary with the capabilities outlined in Section 4.2.5 can successfully exploit the threat of lack of public key authentication by generating Spoofed ID Keys. It is these keys that can result in the eavesdropping of communications, and modification of data. The robustness of the key distribution and management scheme is measured by the Spoofed ID Key Distribution metric. The use of a trust system to provide public key confidence, and by extension public key authentication in key distribution for an autonomous VANET should be able to mitigate the distribution of such keys.

3. *Key Trust Metrics* is a measure of the trust values assigned to each public key in the Direct Key List. The trust system will assign a trust value to the identity of the public key of a node upon meeting. This trust value determines how trustworthy or untrustworthy the node is, and provides an effective metric in determining whether the trust system is able to identify Black Hat nodes and the Black Keys they are distributing. Only White Hat nodes are included as the trust systems are designed for a White Hat node to identify Black Keys and Black Hat Nodes. Several sub-metrics are measured to determine the effectiveness of the different trust systems. The first is a histogram of the distribution of trust values assigned, the second is the average key trust value assigned for both Black and White Keys, and the third is a five number summary box and whisker plot covering the minimum, 1st quartile, median, 3rd quartile and maximum. These sub-

metrics provide a picture of the distribution of trust values that the trust system assigns to both Black and White Keys, as well as if the trust system is successfully segregating the network.

4. *Public Key Distribution Quantity* is a performance metric that is a measure of the amount of public keys that are distributed in the VANET. It is measured as a percentage of the total possible keys that can be distributed, a condition when every node has the public keys of every other node. A high Public Key Distribution Quantity is desirable as it allows nodes to create secure and authenticated end-to-end communications with a large number of nodes. Increasing the number of instances of a node's public key in the network also increases the confidence in the identity associated with the public key and the key itself.

4.5 Results and Analysis

For each of the experiments conducted, the four security and evaluation metrics described in Section 4.4.5 were used to determine the effectiveness and behaviour of the proposed trust system. The first metric is White Key Approval and Black Key Approval. The second investigates the Spoofed ID Key distribution and approval, the third metric is the Key Trust Values assigned to a key by White Hat Nodes, and the final metric is the Key Distribution Efficiency. For each experiment, the respective D_i between the Control, LCF and LLCF scenarios were identical. This was due to the same movement and connection model being replayed for all scenarios, which resulted in identical node meetings. However, the respective A_i differ due to the varying trust systems for each scenario.

4.5.1 White Key and Black Key Approval

The White Key and Black Key Approval metric investigates the acceptance of Carried public keys. Although the number of direct keys distributed will be the same between all scenarios in an experiment, the number of approved keys will differ between the Control, LCF, and LLCF scenarios. To determine whether the different trust systems are correctly approving White Hat keys, the quantities and percentages of White Carried keys that were approved were measured. Conversely, to determine whether the trust system is mitigating the approval of

Table 4.4: Static Black Hat Node Experiment Results

Black Hat Nodes	Scenario	White Carried Keys			Black Carried Keys		
		Total	App.	%	Total	App.	%
0	Control	104603	104603	100.00	0	0	N/A
	LCF	104603	70363	67.27	0	0	N/A
	LLCF	104603	71943	68.78	0	0	N/A
1%	Control	101658	101658	100.00	3919	3919	100.00
	LCF	101658	69432	68.30	3919	837	21.36
	LLCF	101658	70634	69.48	3919	810	20.67
10%	Control	78323	78323	100.00	24908	24908	100.00
	LCF	78323	52715	67.30	24908	9582	38.47
	LLCF	78323	53604	68.44	24908	8454	33.94
20%	Control	59095	59095	100.00	40407	40407	100.00
	LCF	59095	37810	63.98	40407	17122	42.37
	LLCF	59095	38675	65.45	40407	15049	37.24
30%	Control	43345	43345	100.00	52768	52768	100.00
	LCF	43345	27110	62.54	52768	22458	42.56
	LLCF	43345	27794	64.12	52768	20134	38.16

Black Hat keys, the quantities and percentages of Black Carried keys that were approved were also measured. Full results showing key quantities can be found in Appendix C. This metric provides evidence on the LCF and LLCF trust systems in this experimental environment fulfilling Security Property 1 identified in Section 4.2.6. The LCF and proposed LLCF trust systems rely on the D_i of an individual node to provide an initial trust value. However, both trust systems only affect the public key confidence of Carried Keys.

Tables 4.4 and 4.5 depict the results of White Key and Black Key approvals for the static and dynamic adversary experiments respectively. For both White Carried Keys and Black Carried Keys columns, the total number of keys (Total) of each are shown. For all scenarios (Control, LCF, and LLCF) in an experiment, the total quantities of carried keys exchanged was the same. However, the differing trust systems would affect the quantities approved (App.). The percentage of carried keys approved is also shown.

The static adversary results in Table 4.4 provides evidence that the LCF trust system is effectively mitigating the approval of Black Keys, at the minor expense of White Key approval. Using the 10% Black Hat node population as a typical experiment, 61.5% of Black Carried keys were not approved by the LCF trust system at the expense of not approving 32.7% of White Carried Keys. For all variations of Black Hat node population experiments, the LCF consistently

Table 4.5: Dynamic Black Hat Node Experiment Results

Black Hat Nodes	Scenario	White Carried Keys			Black Carried Keys		
		Total	App.	%	Total	App.	%
0	Control	104603	104603	100.00	0	0	N/A
	LCF	104603	70363	67.27	0	0	N/A
	LLCF	104603	71943	68.78	0	0	N/A
1%	Control	102539	102539	100.00	2392	2392	100.00
	LCF	102539	69688	67.96	2392	668	27.93
	LLCF	102539	70630	68.88	2392	663	27.72
10%	Control	75107	75107	100.00	29458	29458	100.00
	LCF	75107	52949	70.50	29458	9756	33.12
	LLCF	75107	52058	69.31	29458	8517	28.91
20%	Control	52215	52215	100.00	52409	52409	100.00
	LCF	52215	38243	73.24	52409	16829	32.11
	LLCF	52215	36673	70.23	52409	14217	27.13
30%	Control	35272	35272	100.00	69471	69471	100.00
	LCF	35272	27603	78.26	69471	21779	31.35
	LLCF	35272	26222	74.34	69471	17644	25.40

achieves a 63-68% approval of White Carried Keys. Black Carried key approval percentages for the LCF scenario vary from approving 21.4% of Black Carried keys at a 1% Black Hat node population, to approving a maximum of 42.6% of Black Carried keys at 30% Black Hat node population.

Comparing the results of the LLCF trust system to the LCF trust system, provides evidence that LLCF consistently out-performs LCF. For all static adversary experiments, the LLCF trust system consistently provided a higher White Carried key approval, and a lower Black Carried key approval when compared to the LCF trust system. Using the 10% Black Hat node population as a typical experiment, the 1.1% improvement in White Carried key approvals, resulted in an additional 889 White Hat Keys approved. More importantly, the 4.5% reduction in Black Carried key approvals, resulted in a reduction of 1,128 Black Hat Keys approved. At the highest Black Hat node population of 30%, the LLCF trust system achieved a 4.4% reduction in Black Carried key approvals, corresponding to a reduction of 2324 Black Hat keys approved. This trend of a marginally higher White Carried key approval, and reduced Black Carried Key approval, provides evidence of the LLCF trust system benefiting from the inclusion of co-localisation data.

The dynamic adversary results in Table 4.5 shows the LCF trust system providing a higher White Carried key approval percentage of 67-78% for all

experiments when compared to the static adversary results. The Black Carried key approval percentages for the LCF trust system also provides an improvement in approval of carried Black Hat keys, achieving around a 30% approval for most experiments. Comparing the Black Carried key results of the LLCF trust system to the LCF trust system shows a reduction in Black Carried key approvals. However, this reduction comes at the cost of White Carried key approvals in the dynamic experiments. The dynamic adversary experiments with 10% Black hat nodes or more, show that the LLCF also has a reduction in White Carried key approvals. These results provide evidence that the LLCF trust system benefits from the inclusion of co-localisation data in environments with static adversaries and small numbers of dynamic adversaries (less than 10%).

4.5.2 Spoofed ID Key Distribution

The second security metric measured is the Spoofed ID Key Distribution. This measures the number of public keys that an adversary has distributed that is a Black Key, with a White Hat Node identity ($P_{(M,i)}$). The metric provides evidence on the LCF and LLCF trust systems in this experimental environment fulfilling Security Properties 3 and 4 identified in Section 4.2.6. Although the number and percentage of total Black Keys directly distributed and approved is an important metric for the security of a network, it is the Spoofed ID Keys that provide a real threat to the secure communications of a VANET. With the ability for adversarial nodes to eavesdrop on communications and modify data, it is desired to mitigate the spread of such keys throughout the network.

Table 4.6 depicts the Spoofed ID Key results for the static and dynamic adversary experiments. It shows the quantity of Direct, Approved and totals of Spoofed ID Keys. The quantity of Spoofed ID Keys in the Direct Key Lists are constant between each scenario of each experiment. The adversary capabilities outlined in Section 4.2.5, mean that Spoofed ID Keys are distributed by adversarial nodes only. Because of this, no White Hat Node would have received a Spoofed ID Key in their respective D_i, as they would have received the legitimate key from that node. Spoofed ID keys are based on second hand confidence in the identity-public key binding, that is distributed by adversarial nodes. As a result, the Direct Key results in Table 4.6, show the quantity of Spoofed ID Keys that all adversary nodes have generated and are subsequently distributing. As the adversary population is increased, it is expected that the number of keys the

Table 4.6: Spoofed ID Key Results

Black Hat Nodes	Scenario	Static			Dynamic		
		Dir.	App.	Total	Dir.	App.	Total
0	Control	0	0	0	0	0	0
	LCF	0	0	0	0	0	0
	LLCF	0	0	0	0	0	0
1%	Control	379	2914	3293	187	1626	1813
	LCF	379	82	461	187	97	284
	LLCF	379	45	424	187	78	265
10%	Control	3601	14542	18143	1908	18769	20677
	LCF	3601	2061	5662	1908	3104	5012
	LLCF	3601	940	4541	1908	2068	3976
20%	Control	6279	22714	28993	3375	31061	34436
	LCF	6279	4851	11130	3375	4896	8271
	LLCF	6279	2884	9163	3375	3189	6564
30%	Control	8930	29189	38119	4366	37304	41670
	LCF	8930	6166	15096	4366	5798	10164
	LLCF	8930	4056	12986	4366	3557	7923

adversarial nodes are spoofing would increase. The results in Table 4.6 reflect this occurrence. The Spoofed ID Keys in their respective A_i is the metric that the trust system has influence over.

For all static adversary experiments, the LCF trust system provides considerable resilience against the approval of Spoofed ID Keys in comparison to the Control scenarios. With the addition of location data to the LCF trust system, there is a further significant reduction in Spoofed ID Keys approved for the LLCF scenarios. In the experiments with a low static adversary population, there is an additional reduction of 50% or greater in the LLCF scenarios. As the adversarial population is increased, the LLCF trust system continues to provide an additional reduction of at least 20% when compared to LCF. The dynamic adversary results show a similar result to the static adversary results. As the dynamic adversary population is increased, the quantity of Spoofed ID Keys also increases. For the Approved Key Lists in all experiments, the LCF trust system also provides considerable resilience against the approval of Spoofed ID Keys in comparison to the Control scenarios. There is also a further reduction in Spoofed ID Keys approved for the LLCF scenarios when compared to the LCF scenarios.

To further compare between experiments and adversary models, Table 4.7 shows percentage of approved keys that are Spoofed ID Keys. It is important

to note, the different trust systems will approve different quantities of keys. Therefore to facilitate comparison between experiments and adversary models, Table 4.7 provides percentages of Spoofed ID keys of the keys approved. For the static adversary experiments, the LCF trust system consistently provides a substantial reduction in the quantity of Spoofed ID keys approved. Using the 10% static adversary node experiment, in the Control scenario the Spoofed ID keys consisted of 14% of all approved keys. The LCF system provided a significant improvement with Spoofed ID keys making only 3.3% of all approved keys. The LLCF trust system provided a further improvement, with Spoofed ID keys making only 1.5% of all approved keys. This trend is consistent for all static adversary experiments.

In comparison to the dynamic adversary experiments, the Black Hat nodes were able to distribute Spoofed ID keys over a variety of locations. Similar trends were observed with the static adversary experiments, with Control having the highest penetration of Spoofed ID keys in all the approved keys, LCF providing a significant improvement, and LLCF providing a further reduction in Spoofed ID Keys. An interesting observation is that the dynamic experiments generally had a greater percentage of Spoofed ID keys in A_i than the corresponding static experiments. Exceptions include some results in the 1% and 30% experiments. These exceptions are due to the mobility model of the Black Hat nodes affecting the distribution of Spoofed ID keys.

These results, along with the results from Section 4.5.1, provide evidence that even though the LCF and LLCF trust systems may be approving less keys than the Control scenario, they are approving significantly less Spoofed ID keys, and in turn providing significant improvements in security at the cost of distribution of keys. The results from Tables 4.6 and 4.7 indicate that the LCF trust system provides additional security to the VANET by significantly reducing the quantity of approved Spoofed ID Keys. With the addition of location data to the LCF trust system to form the LLCF trust system, there is a further reduction in approved Spoofed ID Keys. When comparing the impact on the percentage of approved keys, a similar trend is established. Although the LCF and LLCF trust systems approve less keys than the Control scenario, they provide better mitigation to the approval of Spoofed ID keys. The mitigation of the distribution and approval of Spoofed ID Keys that LCF and LLCF provide, is a significant security improvement to the secure communications of a VANET, at the cost

Table 4.7: Percentage of Approved Keys that are Spoofed ID Keys

Black Hat Nodes	Scenario	Approved Keys that are Spoofed ID Keys	
		Static (%)	Dynamic (%)
0	Control	0.00	0.00
	LCF	0.00	0.00
	LLCF	0.00	0.00
1%	Control	2.76	1.55
	LCF	0.12	0.14
	LLCF	0.06	0.11
10%	Control	14.09	17.95
	LCF	3.31	4.95
	LLCF	1.51	3.41
20%	Control	22.83	29.69
	LCF	8.83	8.89
	LLCF	5.37	6.27
30%	Control	30.37	35.61
	LCF	12.44	11.74
	LLCF	8.46	8.11

of key distribution performance. Further evidence of the LCF and LLCF trust systems assisting the provision of public key authentication is discussed in the Key Trust Metric results.

4.5.3 Key Trust Value Metrics

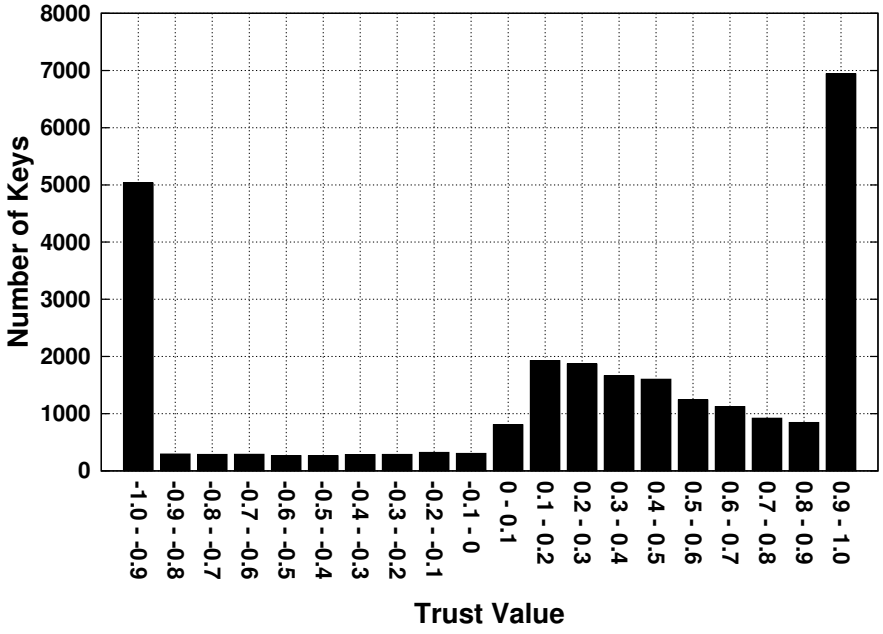
The Key Trust Value metrics were also analysed to evaluate whether the utilised trust system is effectively identifying White Keys and Black Keys. These metrics provide evidence of the LCF and LLCF trust systems in this experimental environment fulfilling Security Property 5, and indirectly fulfilling Security Properties 2 and 4 identified in Section 4.2.6. These metrics include the distribution of trust assigned by White Hat nodes for all keys, the average trust assigned by White Hat Nodes to the keys in their respective D_i, and the five-number summary of trust for both Black Keys and White Keys. These metrics only consider the D_i of White Hat Nodes, as the trust system is designed for legitimate nodes to detect the difference between White Keys and Black Keys. The adversary capabilities also skews the trust values that an adversarial node would assign to White or Black Keys.

Figures 4.8a and 4.8b show histograms for the key trust distribution in the

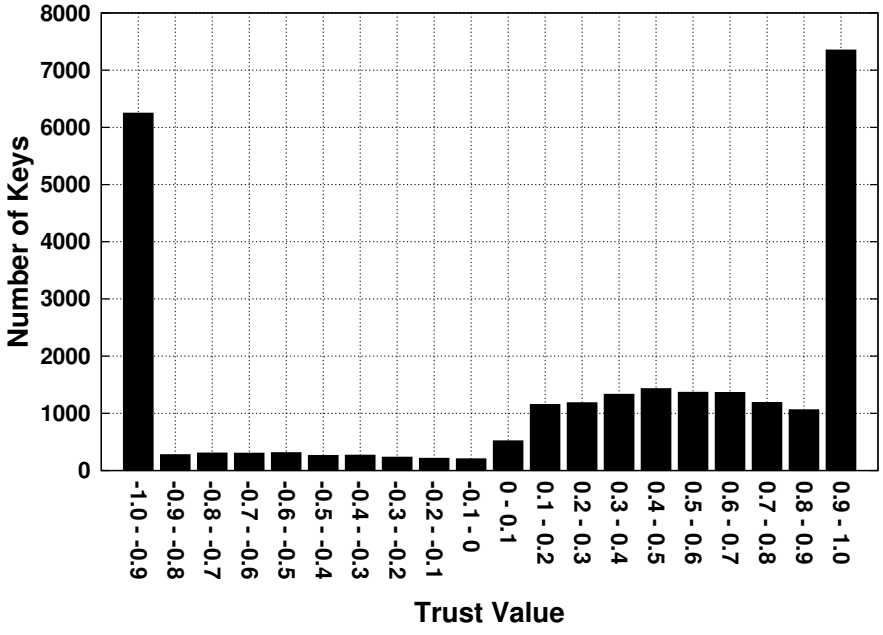
10% static adversary experiment for LCF and LLCF respectively. Similar results were achieved when the number of static adversaries was varied and the 10% experiment is presented as a representative result. Key trust distribution results for all experiments can be found in Appendix D. At the end of the simulation, the LCF trust system in Figure 4.8a depicts that a large majority of key trust values are at the extremes of -1.0 to -0.9, and 0.9 to 1.0. This indicates that LCF is polarising trust values between the Black Keys and White Keys. The proposed LLCF trust system in Figure 4.8b achieves an improvement over LCF, with greater quantities of keys in the same two extremes. The key trust distribution between the positive trust range of 0.0 to 0.9 show different results between the LCF and LLCF scenarios. The LCF scenario shows a long tail distribution skewed to 0.0, with a large quantity of keys being assigned a trust value in the indecisive range of 0.0 to 0.5. The LLCF scenario shows a more normal distribution between the ranges of 0.0 to 0.9, indicating that the trust system is assigning higher trust values to keys considered White Keys. The distribution of trust between the ranges of -0.9 to 0.0 for both the LCF and LLCF scenarios shows a relatively small quantity of keys. This is because both LCF and LLCF trust systems deduct trust from identified Black Keys more severely than the accumulation of trust to White Keys, which is an expected result from the selection of T_{key_disc} in Section 4.3.

In comparison, Figures 4.9a and 4.9b show histograms for the key trust distribution in the 10% dynamic Black Hat nodes experiment for LCF and LLCF respectively. Similar results were achieved when the number of dynamic Black Hat nodes was varied and the 10% experiment is presented as a representative result. For both the LCF and LLCF scenarios, a large quantity of keys are placed in the 0.9 to 1.0 trust range. However, the lower extreme of -1.0 to -0.9 shows a significant reduction in the quantity of keys assigned to this trust range when compared to the static Black Hat results. The LCF results in Figure 4.9a depicts a similar long tail distribution as the static Black Hat nodes between the indecisive trust range of 0.0 to 0.5. The LLCF results in Figure 4.9b depict a similar normal distribution as the static Black Hat node results in the indecisive trust range. The histograms indicate that the addition of location data in LLCF provides additional security against static Black Hat nodes over dynamic Black Hat nodes.

The average trust of all White Keys, and the average trust of all Black Keys

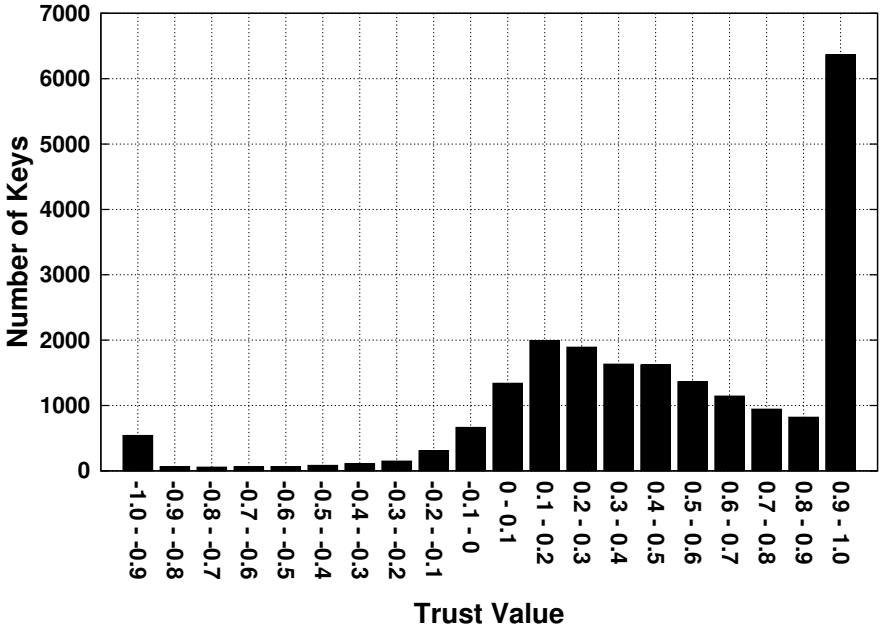


(a) LCF

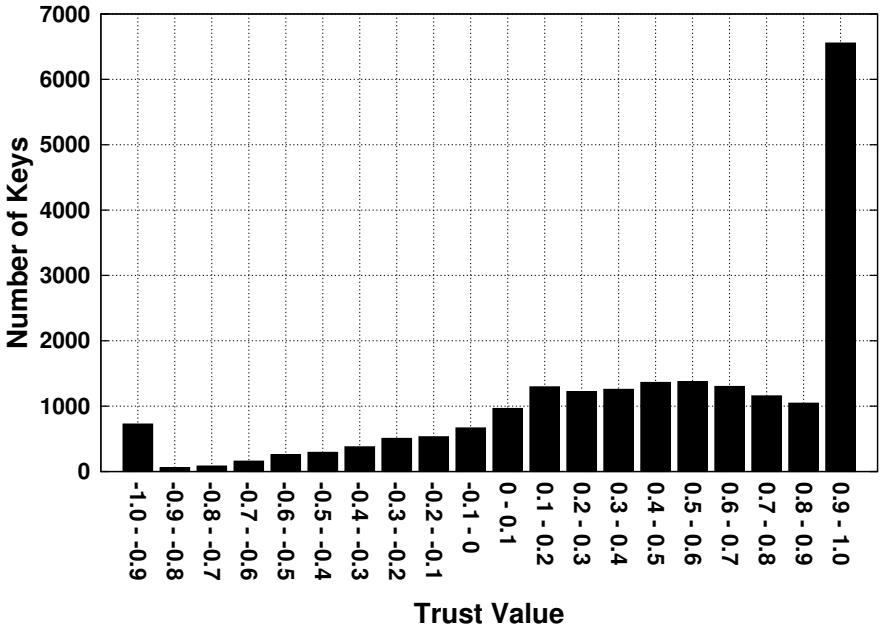


(b) LLCF

Figure 4.8: Key Trust Distribution for 10% Static Black Hat Nodes.



(a) LCF



(b) LLCF

Figure 4.9: Key Trust Distribution for 10% Dynamic Black Hat Nodes.

were also calculated for White Hat Nodes. Table 4.8 depicts the average key trust assigned by White Hat Nodes for both static and dynamic adversary experiments. The static adversary results show that the average key trust for White Keys is consistently greater for LLCF when compared to LCF, indicating that the location factor is providing a higher and more useful trust value in White Keys. For Black Keys, the average key trust for LLCF is consistently less than LCF, indicating that location contributes to a lower trust value for Black Keys. At low populations of adversarial nodes, the LLCF trust system provides a 30% improvement in average key trust over LCF. As the adversarial node population increases, the improvement drops to approximately 18%.

The dynamic adversary results show that the average key trust for White Keys is greater only for low populations of adversarial nodes. After an adversarial penetration of 20%, LLCF is marginally less effective at approximately 94% of the average trust of LCF. For Black Keys, the average trust for LLCF has been demonstrated to be less than the corresponding LCF scenario. At low adversarial populations, LLCF provides an improvement of 25% for 1% adversarial population. When increased to 30% adversarial population, there is a significant increase of 257% in distrust value for Black Hat Keys with LCF at -0.067, and LLCF at -0.239. These results show that as the dynamic adversary node population is increased, the LCF trust system has difficulty identifying Black Keys, while the LLCF trust system is still capable of identifying Black Keys, while still achieving 94% of the average key trust value for White Keys. This is a desired result, as the inclusion of location data in LLCF provides additional security in identifying Black Keys at minimal expense of White Key trust values.

The average key trust metric was also compared between the static adversary and dynamic adversary experiments. The average White Key trust values in the static adversary experiments are comparable in value to the dynamic adversary experiment. The average Black Key trust values between the two adversary movement models differ significantly. Due to the node capabilities for dynamic adversaries, multiple locations are considered untrustworthy by other nodes. This is evident with the average White Key trust value for LLCF being marginally lower than LCF. However, the dynamic adversary experiment results for Black Keys trust value indicates that both LCF and LLCF have difficulty in detecting Black Keys. An example is the 20% adversarial population experiments. The LCF trust system had an average Black Key trust value of -0.495 for the static

Table 4.8: Average Key Trust Assigned by White Hat Nodes

Black Hat Nodes	Scenario	Static		Dynamic	
		White	Black	White	Black
0	LCF	0.633	N/A	0.633	N/A
	LLCF	0.716	N/A	0.716	N/A
1%	LCF	0.630	-0.558	0.629	-0.458
	LLCF	0.685	-0.732	0.687	-0.572
10%	LCF	0.634	-0.537	0.639	-0.125
	LLCF	0.687	-0.703	0.647	-0.300
20%	LCF	0.619	-0.495	0.641	-0.085
	LLCF	0.681	-0.623	0.622	-0.259
30%	LCF	0.609	-0.518	0.655	-0.067
	LLCF	0.672	-0.612	0.621	-0.239

adversary experiments, and an average Black Key trust value of -0.085 for the dynamic adversary experiments. The proposed LLCF trust system had an average Black Key trust value of -0.623 for the static adversary experiments, and an average Black Key trust value of -0.259 for the dynamic adversary experiments. This trend is consistent as the adversarial node population increases for both adversary movement models.

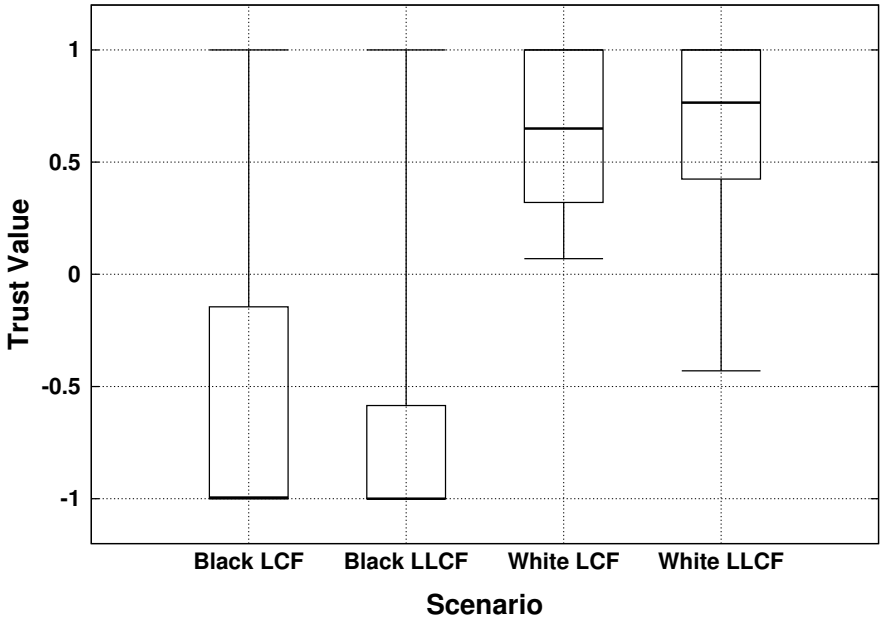
The final analysis of the key trust value metrics is the box and whisker plot of the five number summary (minimum, 1st quartile, median, 3rd quartile, maximum) for both Black Keys and White Keys in the 10% adversary experiments. Similar results were achieved when the number of static adversaries was varied and the 10% experiment is presented as a representative result. Figure 4.10a presents the results for the 10% static adversary experiment. Full results for all experiments can be found in Appendix E. The range of trust assigned for Black Keys for both LCF and LLCF span the entire trust range of -1.0 to $+1.0$, with the median and 1st quartile at the lower limit of -1.0. However, the quartile span for Black Keys under the LLCF trust system is significantly smaller and skewed towards the lower limit in comparison to the LCF results. The White Key results show a greater range for LLCF compared to LCF. However, the LLCF results show a smaller higher quartile span, and a higher median when compared to the LCF results. The White LLCF results also have some negative trust values indicating some White Keys were falsely identified as Black Keys. These are the keys received in an area with a high concentration of adversary nodes.

The dynamic adversary experiment results in Figure 4.10b show similar results for the White Keys between LCF and LLCF. However, the Black Key results depict how both trust systems have difficulty identifying such keys when the adversary is mobile. The Black Key results show that LCF and LLCF assign trust that spans the entire range of -1.0 to $+1.0$. Although the quartile span of the LLCF results is greater than the LCF results, it is skewed towards -1.0, with the 1st and 3rd quartile, and the median lower than LCF. These results show that in both static and dynamic adversary movement models, the LLCF trust system provides better identification and assignment of trust to Black Keys than the LCF trust system. The comparison between the static and dynamic adversaries also indicates that the implemented trust systems are better at mitigating the capabilities of static adversaries.

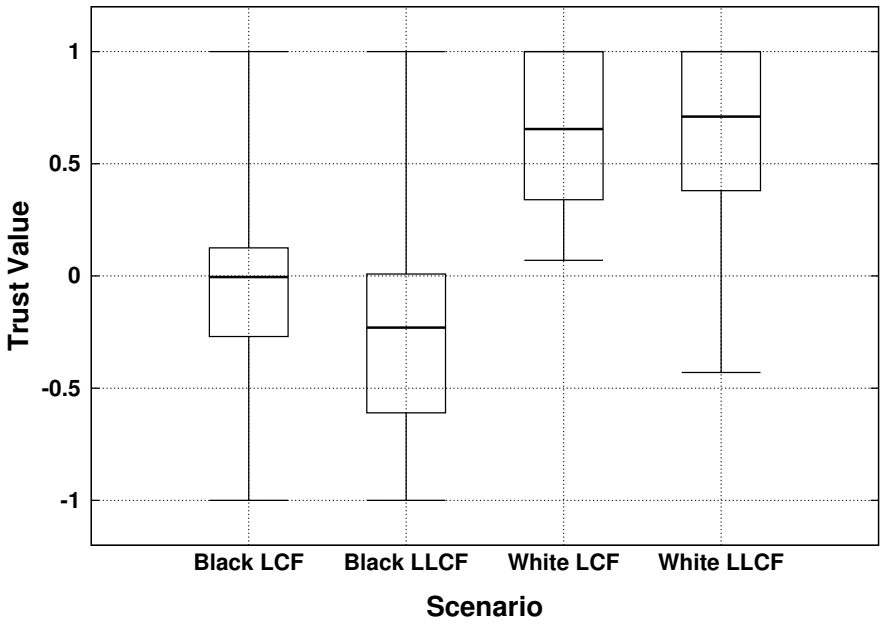
4.5.4 Public Key Distribution Quantity

Public Key Distribution Quantity is a performance metric that assesses how effective the trust systems were at distributing public keys. These metrics provide evidence on the evaluated trust systems fulfilling Security Properties 2 and 4 identified in Section 4.2.6. A Public Key Distribution Quantity of 100% was not achieved due to the open nature of the system model. With nodes able to leave and join the network freely, with some nodes spending very little time in the network, it is not possible to have a public key distribution quantity of 100% even as time approaches infinity. Figure 4.11 depicts the key distribution efficiency when there are no black hat nodes for the three scenarios; Control, LCF, and LLCF. Full results are available in Appendix F. This metric considers both Direct and Approved keys as they represent keys that have exceeded the public key confidence threshold, thereby achieving public key authentication. For all scenarios in Figure 4.11, the rate of key distribution significantly plateaus after 80,000 seconds of simulation. From Figure 4.11 it is evident that the Control scenario provides the best key distribution, with 50.9% of total keys distributed. Although the LCF and LLCF scenarios show a reduction in Key Distribution Efficiency, the LLCF scenario with 37.7% of keys distributed, has a slight improvement over LCF with 37.0% of keys distributed by the end of the simulation.

The key distribution efficiency is affected when static adversary nodes are introduced. The correlation between the increase of key distribution with the increase of adversary nodes is attributed to the placement of the attacking nodes.



(a) Static Black Hat Nodes



(b) Dynamic Black Hat Nodes

Figure 4.10: Five number summary for Black and White keys for 10% Black Hat Nodes.

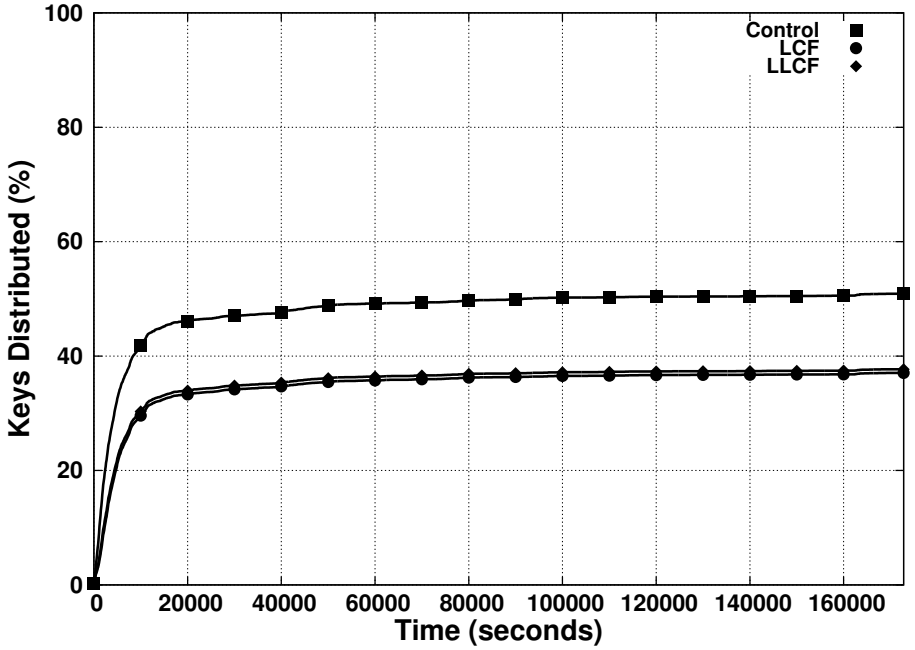
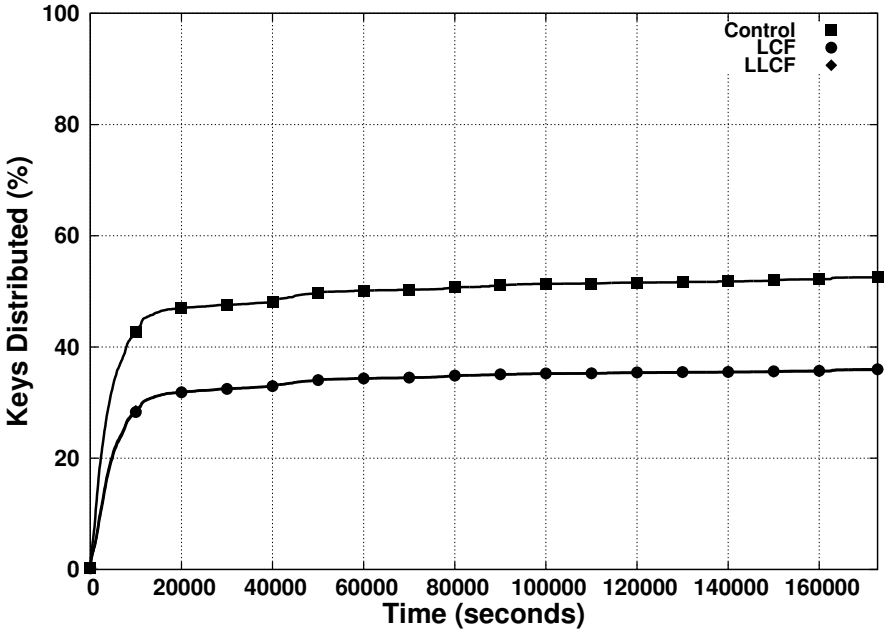


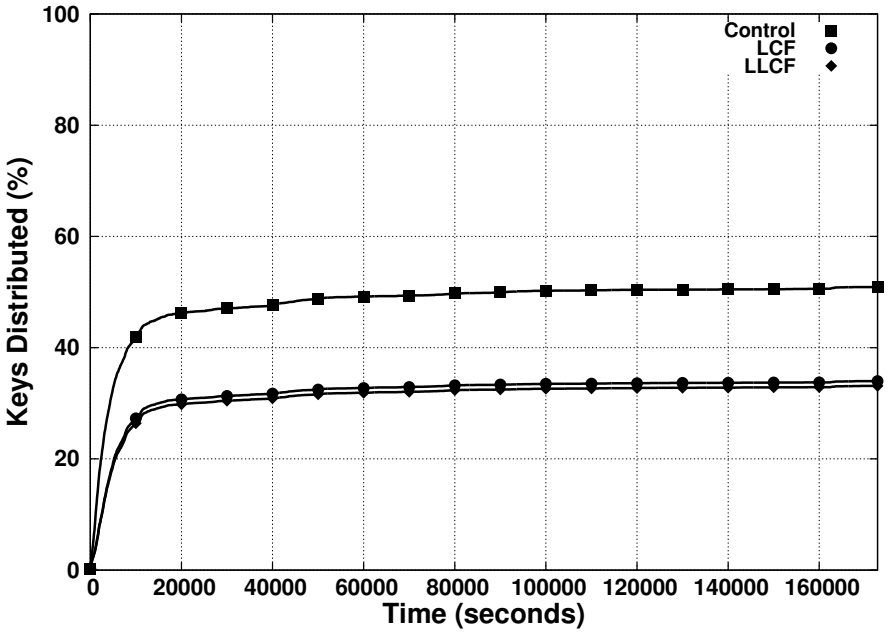
Figure 4.11: Key Distribution over Time for no Black Hat Nodes.

They are placed in a busy intersection with high node density, and thus distributing and exchanging keys more abundantly. Both the LCF and LLCF scenarios see a slight increase in key distribution as the adversary node population is increased, but remains steady in the 37-47% keys distributed range. Comparing LCF and LLCF key distribution, for all experiments with static adversary nodes, the two scenarios had key distribution percentages within 1% of each other, showing that LLCF has similar key distribution efficiency to LCF regardless of adversary node population.

For the dynamic adversary node experiments, the key distribution for all Control scenario experiments remained within the 50-51% range. This is expected as the adversary nodes are constantly moving as determined by the movement model. Therefore, the key distribution efficiency remains consistent between all experiments. Both the LCF and LLCF scenarios have a reduction in percentage of keys distributed as the population of adversary nodes increases. With an increase in adversaries, there is an increase in distribution of black keys. As a result of this, the LCF and LLCF scenarios are approving less carried keys and thus reducing the key distribution. The LLCF scenarios for all experiments show



(a) Static Black Hat Nodes



(b) Dynamic Black Hat Nodes

Figure 4.12: Key Distribution over Time for 10% Black Hat Nodes.

a trade-off in key distribution efficiency for security in comparison to the LCF scenarios. The cost is in the order of 1-2%, and can be attributed to the fact the LLCF relies on detecting and determining the location of adversary nodes to form a location trust value, whereas the LCF trust system does not. Due to the mobile movement of the adversaries in the dynamic experiments, it is difficult for the LLCF trust system to determine and assign a location as untrustworthy or trustworthy. This is reflected in the cost to key distribution percentages of the LLCF trust system in comparison to the LCF trust system. Similar key distribution results were observed as the adversarial node count was increased.

The key distribution efficiency also shows that the bootstrapping process occurs during the first 40,000 seconds, the time period with the highest number of nodes active in the simulation area. Although the total number of keys distributed is important as it allows nodes to communicate, it is the approved key metrics that assist in evaluating the effectiveness and impact of the LCF and LLCF trust systems on a VANET. It is these keys that are distributed through a carrier node (Approved and Untrusted Keys), and they are the only types of keys that the trust system has any influence in trusting or not trusting. The minor performance cost in key distribution for the LCF and LLCF scenarios is a minor trade-off in the additional security achieved in the previous evaluation metrics.

4.6 Discussion

This section discusses the implications and issues related with the proposed location based key distribution scheme, and results. The implementation of a trust and reputation system to provide public key authentication, through public key confidence for key distribution provides a few interesting issues.

1. A fully decentralised public key distribution in an open and dynamic network is difficult, as evident by the public key distribution efficiency results. An example of the difficulty for an open and dynamic network is the PGP Web of Trust. The use of centralised key servers [96, 42] are popular amongst PGP users as it assists in the distribution of keys, whilst retaining a transitive trust principle for public key authentication.

2. The LLCF scheme can be used for any DTN node that logs location, ranging from mobile phones, to other vehicles such as boats, planes, drones,

and space networks. LLCF would still provide security for nodes with a predictable movement model, such as a satellite or bus route, as the detection of adversaries would be easier than an unpredictable movement model as simulated in this contribution.

3. The LLCF scheme through the addition of co-localisation information can also provide safe locations for mobile phone users in a Mobile Ad-Hoc Network (MANET) and drivers in a VANET. The ability for the LLCF scheme to determine safe and unsafe locations could assist in crime mapping of a city to assist law enforcement agencies in pro-active crime mapping based on trustworthiness scores rather than reactive incident based scores.

4. The adoption of Internet of Things (IoT) devices is concurrent with the abundance of Internet Protocol version 6 (IPv6) addresses. Approximately 3.4 x 10^{38} addresses can be utilised to provide each IoT device a unique Internet Protocol (IP) address. An IP address in cyberspace is analogous to a location in real life. As a result, the LLCF trust system can provide location based trust and reputation management based on IP addresses. This has cyber-defence implications for trade and commerce, and national security. Using the LLCF trust scheme, ranges of addresses that are untrustworthy could be categorised, resulting in un-reputable merchants being blocked. In addition, address groupings by nation could be used to help mitigate Distributed Denial of Service (DDoS) attacks originating from particular countries. Other applications for cyber-defences that are based on originating IP addresses could benefit from the application of the LLCF trust scheme.

4.7 Conclusion

Past proposals in the area of VANET key management, and trust and reputation systems have been restrictively evaluated in small constrained closed environments. The lack of realistic, large geographic, and open environment simulations has limited the impact of scaling a decentralised key management or trust and reputation scheme to large scale deployments. The provision of public key authentication, and establishment of initial trust in a VANET application can also be assisted with the inclusion of co-localisation data. The impact of varying

numbers and varieties of adversaries are also an important consideration. The research presented in this chapter has addressed these issues by simulating and evaluating the LCF trust system on a realistic vehicle movement data from San-Francisco taxis. The evaluation consisted of nearly 500 nodes deployed in an open 400 square kilometre environment over a period of 48 hours. The main contributions of this chapter are:

- The extended scalability evaluation of the LCF trust system using realistic deployment conditions for a VANET environment.
- The investigation, development, and evaluation of the LLCF trust system: A public key distribution scheme that utilises co-localisation data with the LCF trust system to provide public key authentication in VANETs. The sub-contributions include:
  - A key distribution scheme that mitigates the effect a stationary adversary has in distributing spoofed public keys that modifies the identity-public key binding.
  - The segregation of Black and White Hat node populations.

The use of an open and dynamic simulation environment where nodes are free to join and leave the network impacted key distribution as opposed to a closed environment. However, significant results were obtained from the experiments. The LCF trust system is applicable and suitable for deployment in a large geographic scale DTN application such as VANETs. It provides a more useful initial trust value between nodes with no prior history, and without using a centralised trust and reputation manager, even when scaled to large quantities of nodes. It successfully mitigates the impact of Spoofed ID keys under various adversarial conditions and attacks, reducing the approved key list penetration by up to 70% in some experiments. The inclusion of co-localisation data to form the LLCF trust system provided further security improvements for establishing initial trust and public key authentication. It provided better security under a stationary adversary threat, where location data provided an additional trust component on whether to trust the identity-public key binding. The impact of Spoofed ID keys was further reduced by 50% using the LLCF trust system in stationary adversary conditions. The additional security performance achieved also resulted in a performance degradation of around 13%. These results indicate the

LCF trust system is scalable and effective in providing public key authentication in large scale DTNs. With the addition of co-localisation data, further security improvements are achieved. The LLCF trust system has potential application as a trust and reputation system for IP based white and black listing, in particular with the explosion of IoT devices and IPv6 address.

The limitations of this work is acknowledged in that area of movement model variation. The use of an open and dynamic simulation environment was necessary to provide a realistic application of DTNs. Different movement models including different vehicular types such as buses and private cars would provide more variety of movement. Including pedestrian movement models with vehicular models would provide a combined MANET and VANET application. The LLCF trust system could be further extended to include time and freshness of data to trust components, thereby also providing a methodology for the discarding of unused public keys.

Chapter 5 provides a public key authentication scheme for the final stage of key management, key revocation and replacement. Using a similar quantity of nodes and geographic area to the experiments conducted in this chapter, the provision of public key authentication for autonomous DTNs during key revocation is a critical aspect of DTN security.

Chapter 5

DTN Key Revocation and Replacement

In Chapter 3, an autonomous Delay Tolerant Network (DTN) public key distribution scheme utilising the Leverage of Common Friends (LCF) trust system to provide public key authentication was designed, developed, and evaluated. Additional evaluation and extension in the form of the Location based Leverage of Common Friends (LLCF) trust system was provided in Chapter 4. These previous chapters focused on the key distribution phase of key management and its subsequent application. However, more importantly, the continued operation and maintenance of any network utilising keys, also requires the ability to revoke and replace these keys. The key management phase of *Key Revocation and Replacement* is the focus of this chapter. The ability to revoke old or compromised keys as well as re-distributing or replacing them with new keys is critically important in establishing, and maintaining secure end-to-end communications in an autonomous DTN. Just as public key authentication is of foundational importance in ensuring the general security principles of confidentiality, data integrity, and message authentication in the key distribution phase, it is also integral to the key revocation phase.

This chapter completes the key management scheme through providing public key authentication for autonomous DTNs during key revocation, and the subsequent key replacement. It specifically addresses Research Question 4: *Is it possible to utilise a trust or reputation system to assist in DTN Key Revoca-*

tion such that Public Key Authentication can be achieved without a trusted third party? It also addresses Research Question 5: *Is it possible to provide trust transferral of an old compromised public key to a newly generated public key without a trusted third party during an unplanned key revocation event?* In addressing the relevant research questions, this chapter focuses on the distributed signing of revocation certificates to provide public key authentication of the old and new public keys. Subsequently, the proposed scheme includes the provision of allowing trust transferral between the revoked key and new public key.

The proposed scheme is compared, analysed and evaluated against other LCF based revocation schemes in a similar experimental environment as in Chapter 4, using a pedestrian movement model. The results obtained are evidence of the provision of public key authentication during key revocation and the subsequent key replacement.

The structure of this chapter is organised as follows. Section 5.1 describes the background and a related work specific to this chapter. Section 5.2 provides a detailed outline of the System Model and Security Properties of the network, along with definitions, threat model and adversary capabilities. Section 5.3 presents the newly proposed Distributed Signing (DS) revocation scheme to assist in the provision of public key authentication during key revocation. Section 5.4 outlines the experimental methodology including experimental and adversary setup, the experiments conducted as well as the security and performance metrics used to evaluate the various revocation schemes. Section 5.5 presents the experimental results and evaluation of how the proposed DS revocation scheme provides public key authentication and trust transferral. Section 5.6 discusses the implications and issues related to the proposed revocation scheme. Finally, Section 5.7 summarises the research and contributions presented in this chapter.

5.1 Background and Related Work

In this section, the background of key revocation specific to the motivations surrounding the process are categorised and discussed. A related work is also discussed and analysed.

The use of up-to-date keys in a DTN is integral to the security of the nodes and the network. Hence, during the key management life cycle (pre-deployment, initial bootstrapping, operation, and revocation [100], nodes may be required to

revoke and update their keys [117]. A node may perform key revocation under two circumstances. The first is considered to be a *planned* revocation. This may be for reasons such as a limited time validity on the key, enforcement of security policies, increasing key security, or planned obsolescence. As a result, a new key pair is required to provide the confidentiality, data integrity, and message authentication requirement in the DTN. This is possible under the assumption that the node still retains sole control of the old private key. However, there are instances where the private key may have been compromised, with the node no longer retaining sole control and possession. This poses a major security vulnerability as the compromised private key allows any node to decrypt messages and impersonate another node. This is the second circumstance of key revocation, and is considered to be an *unplanned* key revocation event. Unplanned revocation events are harder to prepare and plan for, and are reactive in nature.

Key Transition Messages used in Pretty Good Privacy (PGP) [17] provide a *planned* key revocation solution over conventional networks without centralised Public Key Infrastructure (PKI). The Key Transition Message informs other users of the requirement that the old key is no longer being actively used, and that users should begin using the new key. Unique key identifiers such as fingerprints, and user identity are included in the message. The message is then signed by both the old and new private keys to signify control of both, allowing another user to perform public key authentication of both keys. The dual signature also acts as a transfer of trust between the old key to the new key, effectively transferring the Web of Trust. However, in an *unplanned* key revocation scenario, where a node is revoking the old key due to the potential compromise of the private key, a Key Transition Message cannot be used.

In preparation for an unplanned revocation event, PGP provides users with the option to generate a revocation certificate immediately after they generate their key pair, whilst they still retain control of the private key [138]. However, the revocation certificate must also be secured against accidental or malicious disclosure by an adversary. There is also no trust transfer between the old key and new key when key revocation is invoked. This raises the question whether the trust associated with the old key can be transferred to the new key in an *unplanned* key revocation scenario for an autonomous DTN. Previous work in revocation for DTNs focus on node revocation, where a misbehaving node is removed from the network [100, 116, 65]. Key revocation schemes that

are presented are dependant on a centralised infrastructure such as Road Side Units (RSUs) in Vehicle Ad-Hoc Networks (VANETs) [87], along with Certificate Authoritys (CAs) to handle the trust transference [117], and Certificate Revocation Lists (CRLs). Proposals that move away from a centralised scheme to a distributed scheme still rely on the CA and CRLs as the most trusted party in the network, which are difficult to scale [80].

One such proposal of interest, is Hoeper and Gong [65], which not only proposed a revocation scheme for the removal of misbehaving nodes, but also included a provision for a node to self-revoke their private keys in the form of a *Harakiri* message. When a node realises that its private key has been compromised, it creates the Harakiri message informing other nodes to cease using the key pair. The message, is propagated to neighbouring nodes. Finally, each node updates their own key revocation list based on the validity of the receiving messages.

The scheme uses Identity Based Cryptography (IBC), and the inherent issues with using IBC such as key escrow are also addressed by using a (t, n) threshold scheme to distribute the responsibilities of the Key Generation Centre (KGC). The Harakiri message generated by a node when its private key has been compromised is a simple message containing the Node ID, both public and private keys, key metadata such as expiry date and revision, and a simple "revoke" status. The inclusion of the private key is a requirement as it provides verification for the key to be revoked by the node. This prevents an adversary from triggering a revocation event of another node as all other components of the Harakiri message are public knowledge. However, a Harakiri message is triggered on the condition that an adversary has compromised the private key, thereby allowing it to potentially craft a spoofed Harakiri message. Although this would render the compromised key useless to the adversary [65], it still provides the adversary the capability to revoke the node from the network. The inclusion of the private key in the Harakiri message also presents a backward secrecy issue. As the Harakiri message is propagated between nodes, the private key becomes publicly available, and affects the security of all prior messages signed or encrypted with the key pair to be revoked. To counter this issue, Hoeper and Gong [65] proposed the use of dedicated key pairs for different uses. However, this adds to the complexity of node key management. Another solution to this was proposed by the same authors in [64], which forgoes the inclusion of the private key. This

proposal relies on pre-shared keys for public key authentication, adding to the complexity of key management.

The provision of public key authentication during a planned revocation event can be achieved through Key Transition Messages. The use of a pre-generated revocation certificate is a mechanism for unplanned key revocation. However, this requires the user to store and secure the revocation certificate from accidental or malicious disclosure. Past proposals to provide public key authentication for DTNs has resulted in the disclosure of private key materials, or multiple key pairs for different roles. This leads to backward secrecy issues, as well as an added complexity in key management. As a result of these shortcomings, the issue of providing public key authentication during key revocation is still an open problem.

5.2 Key Revocation System Model and Security Properties

This section outlines the System Model and Security properties of the application environment. The likely application and landscape of the network is described. Terminology such as types of nodes and keys are defined. The system model is susceptible to attack by adversarial agents that will attempt to exploit a threat model, which is defined and described. The adversary capabilities and the extent of their attacks are also outlined. Given the system model of the network, and the threat model and adversary capabilities, the security properties that the proposed revocation scheme should achieve are identified. Throughout this paper, the notations in Table 5.1 to refer to nodes, keys, revocation, trust, and black hat nodes are used.

5.2.1 System Model

The system is assumed to be a large scale DTN such as a Mobile Ad-Hoc Network (MANET) or VANET, spanning a large geographic area. The deployment environment has no other entities except nodes themselves. There is no centralised PKI or any form of Trusted Third Party (TTP), and there is no public communications infrastructure. Nodes are considered fully autonomous and mobile, requiring no human intervention. They self-initialise on deployment

Table 5.1: Notations

Notation	Description
Node ID Notations	
i	Unique Persistent Identifier i
N_i	Node i
Key Notations	
K_i^n	Keypair of N_i (version n)
S_i^n	Secret (Private) Key of N_i (version n)
P_i^n	Public Key of N_i (version n)
D_i	Direct Key List of N_i - List of public keys received directly from another node
A_i	Approved Key List of N_i - List of trusted public keys received from carrier nodes
U_i	Untrusted Key List of N_i - List of untrusted public keys received from carrier nodes
Revocation Notations	
C_i	Revocation Certificate of N_i - Certificate stating the cease of operations of N_i's public key
Q_i	Revocation Request of N_i - Request intent for the cease of operations of N_i's public key
Q_i^j	Revocation Request of N_i signed by N_j
R_i	Key Revocation List of N_i - List of trusted Revocation certificates and Revoked keys
G_i	List of Signatories for N_i - Nodes that have signed Q_i
Trust Notations	
T_i^j	Trust Value N_i has of N_j
$T_{(i,j)}$	The Common Trust Value of nodes between N_i and N_j - Trust value of nodes in common between two nodes
T_i^D	The Collective Trust Value of N_i Direct List - Trust value of all trustworthy nodes ($T_i^j > 0$) in D_i
T_i^G	The Signature Trust Value of N_i Signature List - Trust value of all signatories in G_i
Black Hat Notations	
$S_{(m,i)}$	Spoofed Private Key with identity of N_i, generated by malicious N_m (version n)
$P_{(m,i)}^n$	Spoofed Public Key with identity of N_i, generated by malicious N_m (version n)
$C_{(m,i)}$	Spoofed Revocation Certificate with identity of N_i, generated by malicious N_m
$Q_{(m,i)}$	Spoofed Revocation Request with identity of N_i, generated by malicious N_m

with no a-priori knowledge of the network or their neighbours. There is no pre-deployment initialisation phase by an offline authority. The key management phases of Moore et al. [100] are adopted, but given the lack of a pre-deployment phase, the key management phases are; initial bootstrapping (distribution), operation, and revocation.

Nodes retain a persistent unique identity [108], and generate Elliptic Curve Cryptography (ECC) public and private key pairs [98, 78]. These keys are used to perform security based tasks such as providing confidentiality, data integrity, and message authentication. The key pairs are assumed to have a long but finite time period of validity, similar to PGP where keys may last 1 to 2 years. It is an open network, where nodes freely join or leave, with a large number of nodes in the system at any time. Nodes communicate to each other through wireless communications using the IEEE 802.11 specification suite [71]. Due to their mobile nature, they create ephemeral or opportunistic bi-directional connections between neighbours [65]. These communications connections are used to exchange the public keys they own, as well as the public keys of other nodes they have met and carry. Public key exchange is considered to be a low-cost communications procedure as the exchange occurs at one-hop distances between nodes [13]. The public key exchange is completed over a two channel or side channel scheme [72]. Due to the large number of nodes and geographic size of the system, it is assumed that adversary or malicious nodes will exist in the system, and that it might not be feasible to expel such a node.

Key revocation events are also considered rare [87] in comparison to other key management operations such as distribution. Although this work focuses on an unplanned key revocation event - the scenario when a key has been compromised and requires to be revoked, the proposed scheme is also applicable to expired keys and their replacement in a planned revocation event.

5.2.2 Trust Model

Using the same transitive trust model outlined in Section 3.2.2 [138], the public keys of each node are exchanged when nodes are within communications range. These keys form Direct trust, and are easily verifiable as they were transferred by the node owning the corresponding private key. Carried keys belonging to other nodes are also distributed in the key exchange process. Indirect trust relationships between autonomous nodes are formed if the collective trust values

are sufficient similar to the trust model in Chapter 3.

5.2.3 Definitions

The following terminology throughout this paper is defined:

1. *Key Revocation* is the process where a node no longer (or believes that it no longer) has sole possession and control of the private key. As a result, the node wishes to cease using the compromised key.

2. *Key Replacement* is the process where a node generates a new key pair to replace a revoked key pair.

3. *Revocation Certificate* is a message a node generates to distribute to neighbouring nodes with intent to cease the use of a public key. It also includes the new public key.

4. *Public Key Authentication* - Is the verification of the identity-key binding of a public key. In a decentralised public key distribution scheme such as PGP and the Web of Trust, public key authentication is achieved by the human user confirming the entity's claimed identity is associated with their corresponding public key. It is measured as a boolean (Y or N).

5. *Public Key Confidence* is proposed in this thesis as having *confidence* in the identity-key binding of a public key. In an autonomous DTN, without a centralised PKI, or any other infrastructure but the nodes, verification that the public key being distributed belongs to the associated node is difficult. In a DTN application, public key confidence is how confident the autonomous nodes are that the multiple instances of the same public key they are receiving is actually owned by the node identity. It is a continuous value consisting of the culmination of trust values. When the confidence in a public key has exceeded a threshold, public key authentication has been achieved.

6. *Trust Value* is a real numerical value within a predefined range, that is assigned to a single key or node by an entity describing the level of trust it has in that key or node.

The following terms on how nodes categorise the receipt of keys are defined. These reflect how the node came into possession of the public key.

1. *Direct Key* is a public key that a node has received from the owner - N_A has received the public key of N_B (P_B) directly from N_B. Direct Keys are stored in the Direct Key List (D_A).

2. *Carrier Key* is a public key that a node has received from another node who has previously met the owner of that public key - N_A has received the public key of N_B (P_B) from the carrier N_C. A Carrier Key can either be one of the following:

   (a) *Approved Key* is a public key that was distributed by a carrier node that has exceeded the public key confidence threshold to be trusted - Using the example from Figure 2.7, N_A has received the public key of N_X (P_X) from various carrier nodes (N_C, N_D, and N_E) and is confident that P_X actually belongs to N_X. Approved Keys are stored in the Approved Key List (A_A).

   (b) *Untrusted Key* is a public key that was distributed by a carrier node that has not exceeded the public key confidence threshold to be trusted - Using the example from Figure 2.7, N_A has received the public key of N_Y (P_Y) from various carrier nodes (N_B, and N_C) and is not confident that P_Y actually belongs to N_Y. Untrusted Keys are stored in the Untrusted Key List (U_A).

Table 5.2: Classification of Nodes and their Keys

Nodes	White Hat (N_A)	Key pair (K_A^n)	Private Key (S_A^n)
			Public Key (P_A^n)
		Revocation	Revocation Request (Q_A)
			Certificate (C_A)
	Black Hat (N_M)	Key pair (K_M^n)	Private Key (S_M^n)
			Public Key (P_M^n)
		Spoofed ID	Private Key ($S_{(M,A)}^n$)
			Public Key ($P_{(M,A)}^n$)
			Revocation Request ($Q_{(M,A)}$)
			Revocation Certificate ($C_{(M,A)}$)

Table 5.2 provides a summary of the different classifications of nodes, keys and revocation materials. They can be broadly categorised into two types of nodes. *White Hat Nodes* (N_A) are nodes that are legitimate, and have not been compromised by an adversary or attacker. They generate a key pair (K_A^n), which

consists of a *White Public Key* (P_A^n) and a *White Private Key* (S_A^n). In the event of key revocation, a White Hat Node may generate Revocation Materials to distribute. These include a *Revocation Request* (Q_A), which signifies the intent of Node A to revoke their key pair, and a *Revocation Certificate* (C_A), which is distributed to other nodes.

Black Hat Nodes (N_M) are nodes that have been compromised by an adversary or attacker. Like White Hat Nodes, they possess their own Key pair (K_M^n), which consists of a *Black Public Key* (P_M^n) and a *Black Private Key* (S_M^n). A Black Hat Node as part of their malicious nature may create Spoofed ID Material. The first form is a *Spoofed ID Public Key* ($P_{(M,A)}^n$). This is when the Black Hat node generates a key that has the identity association of a White Hat Node, but the key of an adversarial node. It is created when N_M changes the identity association of its public key (P_M^n) from itself (M) to the identity of a White Hat Node (A). This results in a public key that other nodes think belongs to N_A (that is N_A has the corresponding private key) however, N_M holds the corresponding private key.

The Black Hat Node may also attempt to generate a *Spoofed ID Revocation Request* ($Q_{(M,A)}$) or *Spoofed ID Revocation Certificate* ($C_{(M,A)}$). For both the Request and Certificate, the Black Hat node generates a revocation material that has the identity association of a White Hat Node, and a Spoofed ID Public Key. This results in a revocation request or certificate that other nodes think belongs to N_A (that is N_A is requesting the key revocation) with the N_M attempting to insert $P_{(M,A)}^n$.

5.2.4 Threat Model

Given the system model outlined in Section 5.2.1, the following threat model is assumed:

1. *Lack of Infrastructure* - With no infrastructure to assist in key management operations such as public key revocation and replacement, nodes will have to handle these operations independently. An adversary node may take advantage of the state of change as keys are being revoked and replaced to assume the identity of another node. The lack of infrastructure also affects the implementation of a trust and reputation system as there is difficulty in the aggregation of trust scores to form a reputation system.

2. *Lack of Public Key Authentication* - There is no trusted third party, and therefore no assurance of public key authentication. There is no authenticity between the public key and the identity of the owner [12]. An adversary node is capable of associating its own public key to the identity of another node, distribute this key, and perform a Man In The Middle (MITM) attack.

3. *Physical Tampering* - DTN nodes may be subject to physical tampering [106]. Attackers may gain physical access to a node, modifying the behaviour, public key bindings, and distribute Spoofed ID Keys from tampered nodes. The physical tampering of nodes may be mitigated but cannot be prevented. As such, nodes will need a mechanism to detect and protect the network against potentially compromised nodes.

4. *Eavesdropping and Modification of Communications* - Since the connection between nodes in a VANET are ephemeral, and with no static routing, nodes may be required to pass on messages between nodes. This provides an adversary the capability to overhear as well as the potential to modify wireless communications between nodes [107]. Due to this threat, nodes will need to establish secure end-to-end communications.

5. *Open and Dynamic Network* - Nodes may be deployed, join, or leave the network at any time. With no TTP to provide public key authentication or detection of adversaries, it is easy for an attacker to deploy new adversarial nodes into the network. Therefore, it is possible for the network with a large population of nodes to always consist of a population of adversarial nodes, which cannot be expelled.

5.2.5 Adversary Capabilities

Adversarial nodes will attempt to exploit the threat model outlined for this network. With a Spoofed ID Key, any adversary node is capable of impersonating another node. This has consequences for secure communications, data integrity, message authentication, and future key management activities such as key revocation. Due to the characteristics of a DTN, an adversary is capable of eavesdropping on communications. These networks require communications to be passed on between nodes in a store and forward method [40]. A message

or bundle encrypted with a Spoofed ID Key that an adversary has generated, means that if the adversary is capable of intercepting the message, they can successfully perform a MITM attack [12]. An insider attack model was used, with the adversarial nodes assumed to have similar abilities as outlined by [32] with the following capabilities:

1. Adversarial nodes can obtain any message passing through the network between two other nodes within communications range [40, 12, 107].

2. They are a member of the network and can therefore initiate and receive communications with other nodes in the network.

3. They are able to generate new Black Private and Public Keys (S^n_M, P^n_M) as defined in Section 5.2.3 and distribute P^n_M. S^n_M and P^n_M between adversary nodes are independent to prevent White Hat nodes from blacklisting a common P^n_M between all adversary nodes [32].

4. They are able to generate and distribute Spoofed ID Keys ($P^n_{(M,i)}$) as defined in Section 5.2.3. This is when they associate their own private and public key pair with another node's identity in their D_M or A_M Lists [12].

5. They are able to perform a Sybil Attack where they assume the identity of a White Hat Node, so that they revoke their keys, and generate false revocation certificates to insert Spoofed ID Keys as defined in Section 5.2.3.

6. Adversary nodes are capable of compromising the private key of neighbouring nodes if in close contact through side channel attacks.

5.2.6 Security Properties

Given the system and threat model of the network, this section outlines the desired security properties the key revocation scheme should achieve. Ultimately, the aim is to achieve public key authentication in the node requesting a key revocation and replacement.

Property 1: A White Hat node (N_R) should be able to self-revoke their own key pair (K^n_R) when they suspect it has been compromised. This comprises of a timely removal of the compromised public key (P^n_R) from other nodes in the network to prevent the compromise of messages by an adversary.

Property 2: The same White Hat node (N_R) should be able to generate a new key pair (K_R^{n+1}), and distribute the replacement public key (P_R^{n+1}) as part of the key replacement process to re-establish a secure end-to-end communications channel with other nodes in the network.

Property 3: A Black Hat node (N_M) should not be able to perform a key revocation event under the guise or identity of another node, causing a false revocation event with a spoofed revocation certificate ($C_{(M,R)}$)

Property 4: The effect of a Black Hat node (N_M), wishing to exploit the key revocation and replacement process of a White Hat node (N_R) by distributing a spoofed ID key ($P^n_{(M,R)}$) should be mitigated.

Property 5: During an unplanned revocation event, the revoking White Hat node (N_R) through some trust mechanism, should be able to inherit trust from the compromised public key (P_R^n) to the replacement public key (P_R^{n+1}).

5.3 New Key Revocation and Replacement Process

Assuming the system model outlined in Section 5.2.1, three applicable self-revocation schemes were analysed. The first is the Remove Only (RO) revocation scheme, where compromised keys are removed, but the new public keys are distributed using the LCF key distribution mechanism [31]. The second is the Remove and Replace (RR) revocation scheme. This is where the new public key is inserted into the key list where the compromised key resided. The third scheme is the newly proposed revocation scheme designed to provide public key authentication, called the DS revocation scheme. This is where neighbouring friendly nodes attest and vouch for a nodes' identity during the key revocation process.

In this section, the proposed key revocation and replacement scheme suitable for application in an autonomous DTN without any centralised PKI is outlined. The requirements as well as two base line alternatives based on the LCF trust and reputation schemes are also discussed.

5.3.1 Requirements

Due to the distributed and decentralised nature of the system model outlined in Section 5.2.1, the revocation schemes should avoid distributing the entire CRL to all nodes. Instead, it is more desirable to distribute only the ΔCRL. This reduces the communications overhead particularly as the CRL increases in size. Therefore, the following revocation certificate structure that a revoking node (N_R) would generate was adopted:

$$C_R = [R, P_R^n, certificate\,metadata, P_R^{n+1}] \tag{5.1}$$

Nodes should maintain their own independent version of the revoked key list, dependant on whether they trust the revocation certificate. Discrepancies in the revoked key list between nodes will exist due to the independent trust values nodes will assign each revocation certificate they receive. This is expected as the nodes manage trust independently due to the lack of a centralised trust authority. As more nodes self-revoke keys, the revoked key list for each node will continue to grow. To reduce the size of this list, the revoked key entries may be removed after a pre-determined length of time. The assumption here is that the DTN may span a large geographic area, and due to the openness of the network where nodes freely join and leave at any time, some nodes may never meet or have any contact with another node at opposite sides of a country. It can be assumed that the revocation certificate in this scenario would have little context for a node, and therefore may safely remove the entry.

Although the adversary may have compromised the private key, allowing it to compromise the security of previous messages encrypted and signed by the private key, the proposed scheme does not further compromise the security of prior messages by including the private key for other nodes in the revocation certificate such as in [65]. There is no need for confidentiality for the revocation certificate as it is considered public knowledge for all nodes, similar to an issued PGP revocation certificate [138].

5.3.2 Remove Only (RO) Revocation Scheme (without trust transferral)

The RO revocation scheme without trust transferral revocation process is an adaptation of Hoeper and Gong's Harakiri message [65]. The C_R is passed be-

tween nodes, and P_R^n is removed. P_R^{n+1} is distributed using the traditional LCF key distribution from [31]. There is no trust transferral between P_R^n and P_R^{n+1}. Figure 5.1 depicts the RO revocation process between the revoking node N_R and a neighbouring node N_A. Initially, N_R generates a new key pair K_R^{n+1}. It also generates a revocation certificate C_R, which contains information on the old public key P_R^n. N_R sends C_R to neighbouring N_A. N_A receives C_R and extracts information on P_R^n, and searches the Direct (D_A), Approved (A_A), and Untrusted (U_A) key lists for an occurrence of P_R^n. If P_R^n is found in any of the key lists, it is removed and added to a Revoked Key list R_A much like a CRL. N_A now also becomes a carrier of C_R and passes them onto other nodes who have not yet received the certificate. Any node with the certificate is a carrier and disseminates the certificate to every node they encounter. The new public key P_R^{n+1} is distributed using the LCF trust system.

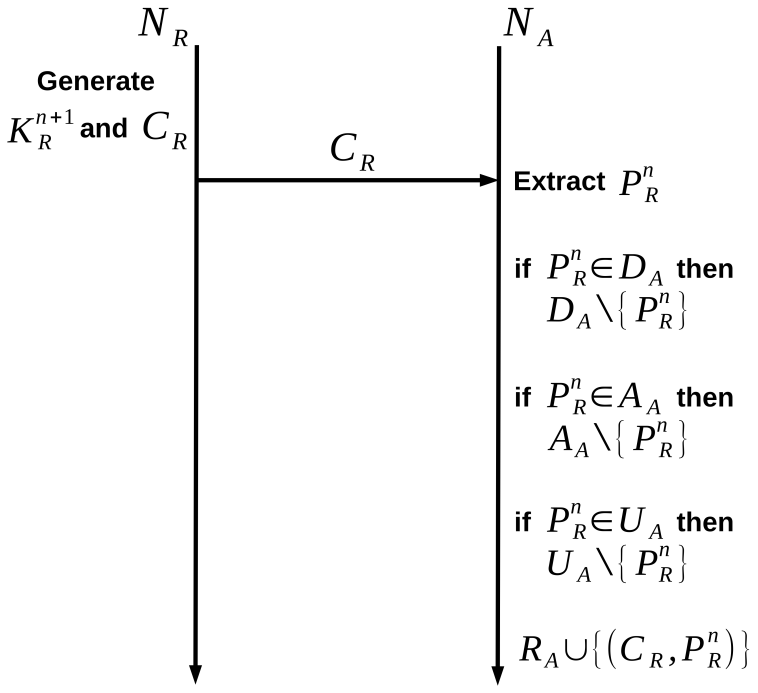


Figure 5.1: RO Revocation Process.

5.3.3 Remove and Replace (RR) Revocation Scheme (with absolute trust transferral)

The RR revocation process follows a similar process to the RO revocation process, with a variation in how P_R^{n+1} is distributed. Figure 5.2 depicts the RR revocation

process. The actions of the N_R are the same to the RO revocation process. When N_A receives C_R and searches through D_A, A_A, U_A key lists removing any instances of P_R^n, it replaces those instances with P_R^{n+1}. P_R^n is also added to R_A. This revocation process has a complete trust transferral model from P_R^n to P_R^{n+1}.

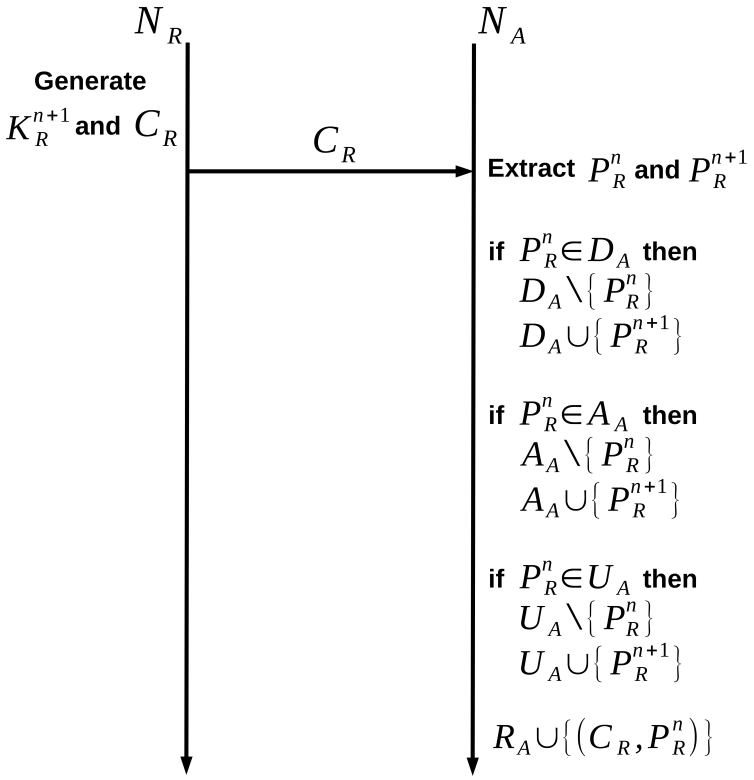


Figure 5.2: RR Revocation Process.

In the RO, and RR revocation scenarios, the revocation certificate is accepted as a 'first in, first accepted' basis. The authenticity of the revocation certificate may be in question if more than one node is requesting a key revocation for the same identity. To address this issue, and provide public key authentication during a key revocation event, the DS revocation process is proposed.

5.3.4 Distributed Signing (DS) Revocation Scheme

The DS revocation process adds security through public key authentication; by determining that the key to be revoked actually belongs to the corresponding identity. The Web of Trust property from PGP [138] of signing public keys to provide confidence in identity-key binding is utilised. The revocation request, and resulting new public key must be signed by a group of trustworthy nodes called

signatories. When a node N_R intends to revoke P_R^n, it generates a revocation request Q_R. However, instead of distributing Q_R, N_R broadcasts to neighbouring nodes to sign (or attest for) Q_R. The signing of Q_R by a neighbouring node such as N_A signifies the confidence of N_A that P_R^n is to be revoked and the newly generated P_R^{n+1} actually belongs to a trustworthy node. By extension, the most likely owner of the key and identity (public key confidence). The signing process of public keys is computationally expensive and time consuming [66]. However, its use in a revocation event is justified as key revocation would be a more scarce occurrence compared to more common key distribution events in PGP [138]. The node requesting the revocation may then choose whether to accept or reject the signature from a neighbouring node based on their trust relationship. Once a critical number of signatures are collected, the revocation request is elevated to a revocation certificate and distributed. These steps are outlined below.

The signing process is detailed in Figure 5.3. N_R after generating Q_R and K_R^{n+1}, broadcasts its intention for signatories. Neighbouring node N_A receives Q_R and extracts both P_R^n and P_R^{n+1}. It then checks whether $P_R^n \in D_A$ meaning N_A has previously met N_R. If they previously haven't met, there is no prior trust relationship between the two nodes and N_A does not sign Q_R. If there is a prior relationship, N_A requests D_R and uses this list and its own D_A to calculate the Common Trust between both nodes ($T_{(R,A)}$), which is shown in Equation 5.2. T_A^D is also calculated using Equation 5.3. The values $T_{(R,A)}$ and T_A^D are compared such that if $T_{(R,A)} > T_A^D$, N_A agrees to sign Q_R. The positive trust values of the direct key list are used to prevent Black Hat nodes from spamming and lowering the threshold of signing.

$$T_{(i,j)} = \frac{\sum_{x \in \{y | P_y \in \gamma\}} T_i^x}{|\gamma|} \quad \text{where} \quad \gamma = \{D_i \cap D_j\} \tag{5.2}$$

$$T_i^D = \frac{\sum_{x \in \{y | P_y \in D_i\} \wedge T_i^y > 0} T_i^x}{|D_i|} \tag{5.3}$$

If N_A agrees to sign Q_R, it sends the signed Q_R and P_R^{n+1} back to N_R, which has the opportunity to accept or reject the signature. Upon a successful signature by N_A, N_R must determine whether N_A is trustworthy. Figure 5.4 depicts the process of N_R accepting the signature from N_A. N_R requests D_A. From this, N_R calculates $T_{(R,A)}$. From D_R, N_R computes T_R^D. These values are compared

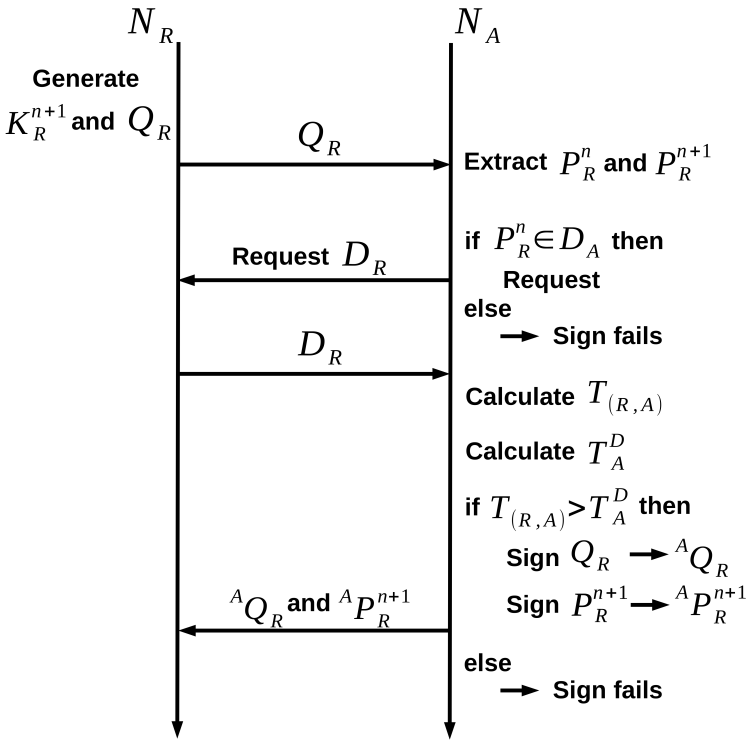


Figure 5.3: DS Revocation Request Signing Process.

such that if $T_{(R,A)} > T_R^D$, N_R accepts the signature from N_A. The positive trust values of the direct key list are used to prevent Black Hat nodes from spamming and lowering the threshold of signing.

The signing process and signature acceptance or rejection process is repeated until there are sufficient signatures on the revocation request. Each node records the number of common nodes between the two nodes in each direct key exchange. At the time of revocation, the revoking node computes the average number of common nodes from their past interactions with other nodes and this forms the required number of signatures. This average number provides an indication of the number of nodes in common (from the perspective of the revoking node) between nodes in the DTN. If the revocation process was triggered early in the bootstrapping of the network, where the revoking node had only met a very small number of nodes, the required number of signatories would reflect this, providing a realistic number of signatories required. The number of signatories required can also be limited, to prevent an excessive amount of nodes required to sign Q_R.

Once the threshold of signatures has been reached, the Q_R is elevated to a

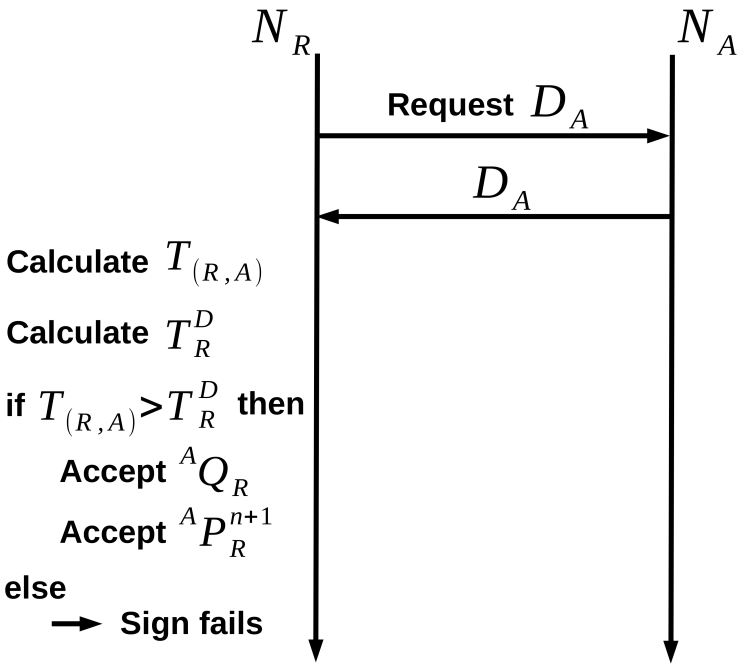


Figure 5.4: DS Revocation Request Signing Acceptance Process.

revocation certificate C_R. C_R is then distributed and carried through to other nodes using an epidemic style routing [127]. When a node receives C_R, it determines whether to trust the certificate and proceed with the key revocation process. Figure 5.5 outlines the process of accepting or rejecting the DS revocation certificate. N_A receives C_R from either N_R directly or from a carrier. From C_R, N_A extracts P_R^n, P_R^{n+1}, and G_R. The average trust of signatories T_R^G is calculated from G_R using Equation 5.4.

$$T_i^G = \frac{\sum_{x \in \{y | P_y \in \sigma\}} T_i^x}{|\sigma|} \quad \text{where} \quad \sigma = \{G_i \cap D_j\} \tag{5.4}$$

The collective signatories trust (T_A^D) is calculated using Equation 5.3. If $T_R^G < T_A^D$, then C_R and R_R^{n+1} are rejected, otherwise they are both accepted and the key revocation process continues. Each of the key lists D_A, A_A, and U_A are searched and any instances of P_R^n removed. Both the certificate and old public key (C_R, P_R^n) are also added to N_A's internal key revocation list R_A. The adoption of P_R^{n+1} is determined based on how the receiving node took receipt of C_R. N_A may have received the revocation certificate from one of two sources. The first is directly from N_R, the node that is requesting the revocation of P_R^n. In this instance P_R^{n+1} is added to D_A. The second scenario is receipt from a carrier

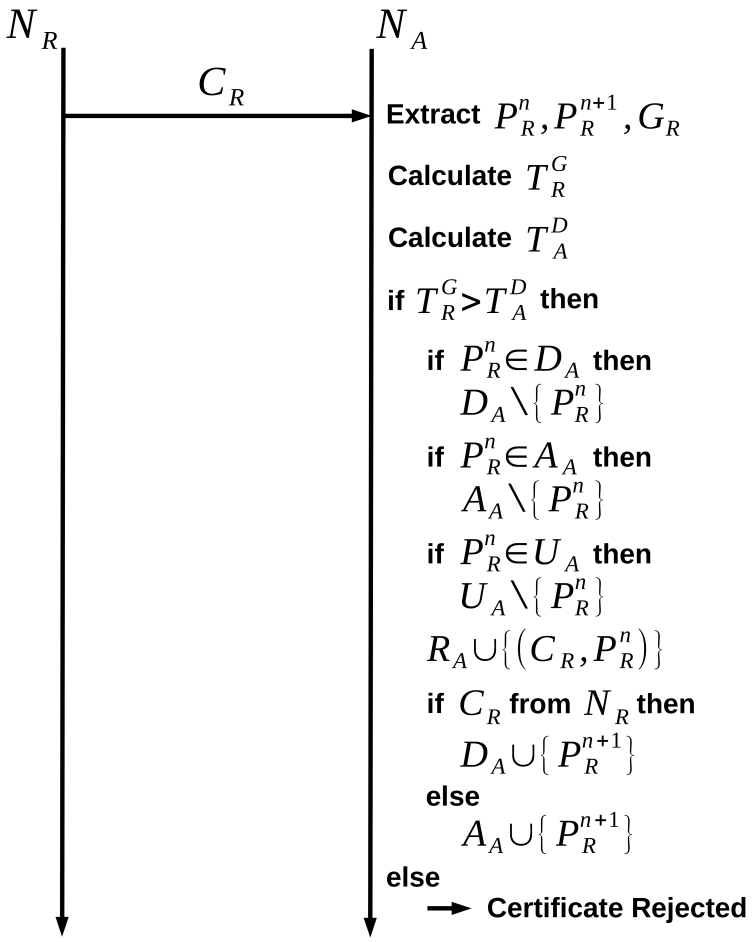


Figure 5.5: Receiving a DS Revocation Certificate Process.

node passing on C_R. In this instance, P_R^{n+1} is accepted into A_A.

5.4 Experimental Methodology

An open source DTN and VANET simulator was developed called *Traffic Djam* [30] that models the decentralised distribution of public keys between nodes using a Web of Trust model. This simulator provides complete control over the key exchange process. The simulator initialises nodes with a persistent unique node identifier similar to [108], and a public and private key pair signature similar to key fingerprints in PGP [138]. Nodes follow a random movement path model [73]. Nodes within a predefined communications range of another node connect and engage in public key exchange whilst in motion. At some defined time, a node will initiate a self key revocation and replacement process.

5.4.1 Experimental Setup

A similar node deployment environment to Chapters 3 [31] and 4 were assumed, with no public infrastructure, no trusted third party, and no central point for node bootstrap or initialisation. Nodes solely rely on peer-to-peer communications and self initialise when deployed. An approval system similar to [31] was used to provide a three-tiered key trust system: Direct, Approved and Untrusted. The receiving node may approve or reject the carried public key based on the trust value of the carrier node it received the key from and the trust algorithm.

The revocation occurs when a White Hat node meets and performs a key exchange with an adversary or Black Hat node. Because these meetings may occur in close proximity, the White Hat node suspects that their private key might have been compromised. The White Hat node then triggers a key revocation and replacement event. As such, the security and evaluation metrics were measured from the time subsequent after the key revocation event.

5.4.2 Adversary Setup

Adversary nodes in this network will exploit the lack of public key authentication and issue Spoofed ID Keys. They will bind their own public key with the identity of a White Hat Node to trick other nodes into believing this false binding. They will then distribute these keys to other nodes as the keys are used for inter-nodal communications.

The adversary nodes are introduced into the network at the very beginning of the simulation during the key distribution phase. The adversary has the opportune moment to attack the network and attempt to distribute the Spoofed ID Keys during the key distribution phase. This attack simulates the scenario of an outsider adding adversarial nodes into the network to allow eavesdropping of inter-nodal communications. The distribution of Spoofed ID Keys is detrimental to the security of the network. Nodes may transmit bundles encrypted with a Spoofed ID Key believing it to be for the desired receiver. Spoofed ID Keys provide an adversary the capability to eavesdrop on network traffic and impersonate other nodes.

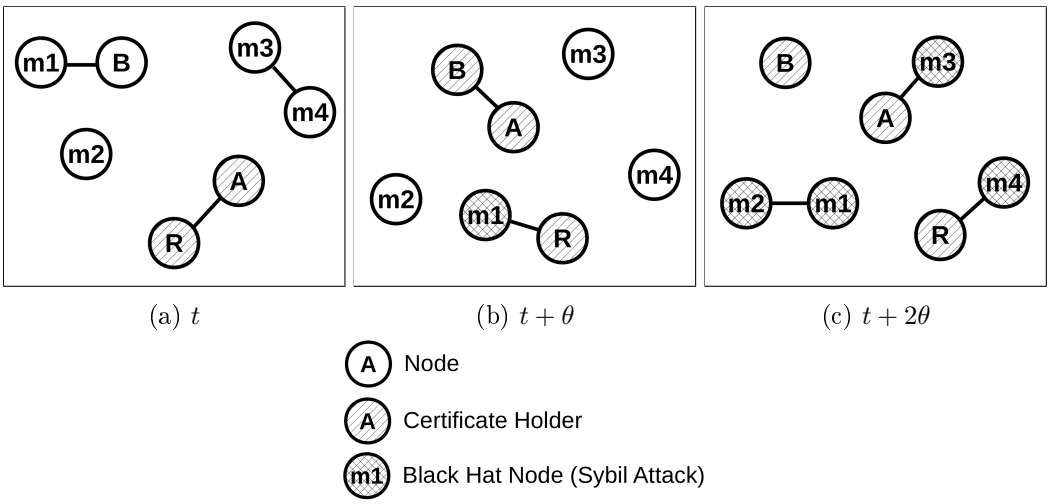
A Black Hat node will also perform a Sybil Attack in the event a White Hat node undergoes key revocation. The Black Hat node may become aware of this key revocation event, and attempt to replace the key being revoked with

a spoofed ID key instead of the new legitimate key from the White Hat node. The adversary performs a Sybil Attack to assume multiple identities and begin distributing the Spoofed ID key and a falsified revocation request under the guise of distributing the new key that is to replace the old revoked key.

Two scenarios for adversary nodes performing the Sybil Attack were investigated. The first called *Single* is where a single node performs the Sybil Attack while the other designated adversary nodes perform the Key Spoof Attack. This scenario results in only a single designated node performing the Sybil attack targeting the key revocation phase and provides insight to the effect of a single adversary node. The second called *Multiplying* is where multiple nodes begin performing the Sybil Attack as they become aware of a White Hat node revoking their keys. As described in Section 5.2.5, all adversary nodes begin the simulation performing a Key Spoof Attack. When an adversary comes into contact with a node that is requesting or conducting a key revocation action (Q_R signing, C_R distribution), they become aware of the Node ID that is attempting to revoke old keys and distribute new keys. As a result, the adversary will in addition to the Key Spoof Attack, begin a Sybil Attack on the Node ID attempting key revocation. This results in a staggered introduction of adversary nodes performing a Sybil Attack.

Figure 5.6 depicts how Black Hat nodes begin the Sybil Attack. This diagram shows an interaction of 7 nodes. Nodes A, B, and R are White Hat Nodes, with Node A the revoking node. Nodes $m1$, $m2$, $m3$, and $m4$ are Black Hat Nodes.

At time t in Figure 5.6a, N_R is the revoking node and has generated C_R and P_R^{n+1} to distribute. It is connected to N_A and passes C_R and P_R^{n+1}. N_B and N_{m1} are connected, along with N_{m3} and N_{m4}. At time $t + \theta$ in Figure 5.6b, the nodes have moved and formed new connections. N_A has connected to N_B and now distributed C_R previously obtained from N_R at time t. N_R has connected to N_{m1} and passed on C_R and P_R^{n+1}. Because N_{m1} is a Black Hat node, and it has become aware of a node performing a revocation (N_R), it begins the Sybil Attack by assuming the identity of N_R and generating a spoofed revocation certificate with a spoofed key inserted ($C_{(m1,R)}$ and $P^n_{(m1,R)}$). At time $t + 2\theta$ in Figure 5.6c, the nodes have moved again and new connections have been formed. N_R has connected with Black Hat node N_{m4}, resulting in N_{m4} beginning a Sybil Attack and assuming the identity of N_R. N_A having carried C_R, is now connected to Black Hat node N_{m3}, who also begins a Sybil Attack on N_R. N_{m1} having been



(a) t (b) $t + \theta$ (c) $t + 2\theta$

Figure 5.6: Black Hat Nodes detect key revocation process and initiate Sybil Attacks

the first Black Hat node to begin a Sybil Attack at time $t+\theta$, meets fellow Black Hat node N_{m2} to initiate a Sybil Attack on N_R.

Figure 5.7 depicts a typical staggered introduction of Black Hat nodes performing a Sybil Attack. The figure shows the percentage of the Black Hat node population that is performing the Sybil Attack. The DS revocation scenario shows a delay in the Black Hat nodes beginning to perform the attack when compared to the RO and RR scenarios. This is due to the fact that the RO and RR scenarios immediately distribute the revocation certificate, while there is a delay in signing and verifying the revocation request or certificate for the DS scenario. Even during this delay, the percentage of Black Hat nodes that begin the Sybil attack is still increasing, as they meet the node requesting revocation directly. As other adversary nodes become aware and start performing the Sybil attack, they also trigger other Black Hat nodes they come into contact to perform the Sybil attack.

This method of introducing Black Hat nodes performing the Sybil attack is a realistic method because it is assumed that there is no centralised infrastructure or authority. As a result, the Black Hat nodes do not have the capability of coordinating a systematic Sybil attack at the same time as the White Hat node requested a key revocation. The Black Hat nodes attack independently of each other and become aware of a key revocation event through dissemination through the DTN network. Each Black Hat node will generate independent $P^n_{(m,i)}$.

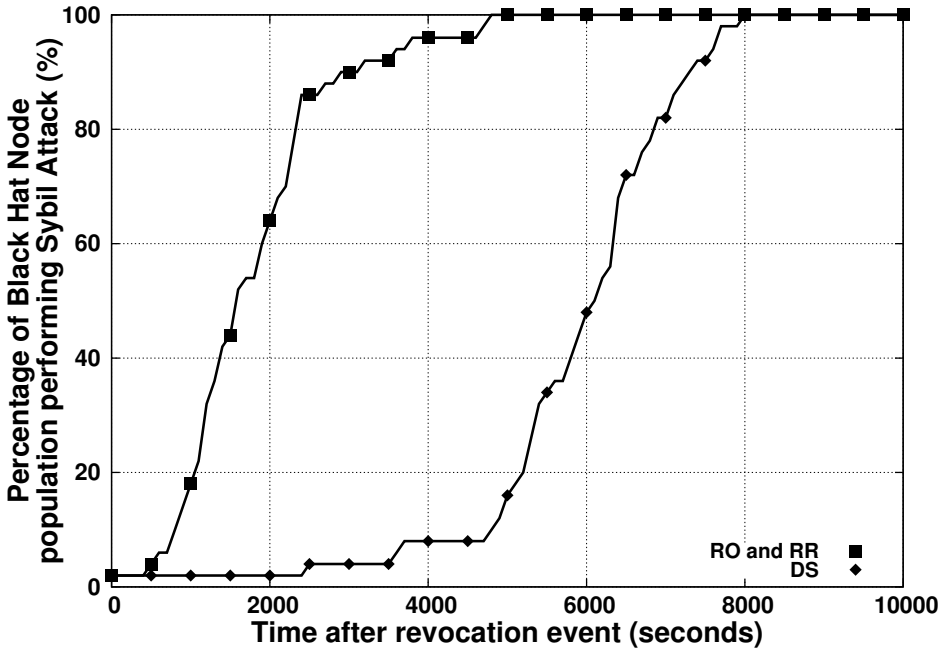


Figure 5.7: Black Hat Nodes performing Revocation Attack over Time at 10% of Node Population.

5.4.3 Experiments

An experiment consisted of three scenarios, each using different methods of public key revocation. The first scenario called *RO*, utilised the revocation method outlined in Section 5.3.2. The revoking node generates C_R and distributes it between nodes that it meets. P_R^n is removed from the keyring of other nodes and P_R^{n+1} is distributed using the LCF trust system as in Chapter 3. The second scenario called *RR* utilised the RR revocation method outlined in Section 5.3.3. The revoking node generates C_R and distributes it amongst nodes it meets. P_R^n is removed from the keyring of other nodes and replaced with P_R^{n+1}. The third scenario *DS* utilised the newly proposed revocation method described in Section 5.3.4. For each scenario, the same movement model and node connections were replayed. This allowed the observation and measurement of how each methodology of key revocation affected the removal of P_R^n and the adoption of P_R^{n+1}. Table 5.3 summarises the simulation constants used. Each scenario was simulated for a little longer than 24 hours for a total of 100,000 seconds, in a 10km by 10km square similar to a large central business district of a city. A total

of 500 nodes was used, each with a conservative communications range of 50m. Nodes move at a walking speed range of 1km/h to 9km/h between each individual waypoint. Upon reaching their destination, nodes wait for a time period between the ranges of 10 to 120 seconds before generating a new destination and travel speed.

The 802.11g communications standard has a baud rate of 6 to 54 megabits per second (Mbits/s) [71]. Using the same public key setup as in Chapters 3 and 4 of a 256 bit ECC public key, and assuming a public key package size of 300 bytes to include additional key metadata and information. Full keyring transfer of all 500 public keys could be completed in 0.02 to 0.2 seconds. The simulator communication model determined whether two nodes were within the 50 metre communications range of each other for a significantly greater time period than the 0.2 seconds worst case scenario. If the two nodes had sufficient time in communications range, the data bundle was transferred successfully, otherwise partial data bundles were dropped. Since the pedestrian nodes travel at differing speeds, a sub-routine in the simulator would perform a high frequency check to determine whether there was sufficient time between two nodes within communications range to transfer their respective data bundles. It is also assumed that there is minimal interference from buildings and obstacles in the communications model.

Experiments varying the adversary node population were conducted, each with the three revocation schemes or scenarios RO, RR, and DS. The number of adversary nodes was varied from 0, 1 node, 1%, 5%, 10%, 15%, 20%, 25% and 30% of node population. In this work, each adversary node is associated to a Byzantine failure in the Byzantine fault tolerance system, or a traitor in the generalised Byzantine Generals' Problem. Common solutions to Byzantine fault tolerance require the number of failures (traitors) should not exceed one third of the total nodes (generals). Hence, only an adversary population of less than 33% was analysed [81]. The range of percentages of adversary nodes provides observations on the system when there are low populations of adversaries operating, as well as at adversary populations approaching the Byzantine fault tolerance threshold of 33%.

Table 5.3: Experimental Simulation Constants

Parameter	Value
Experiment Environment	
Environment Size	10 x 10 Kilometres
Duration	100,000 Seconds
Movement Model	Pedestrian Waypoint
Node	
Number of Nodes (N)	500
Node Speed	1 - 9 Km/h
Node Wait Time	10 - 120 Seconds
Communications Standard	IEEE 802.11 Suite
Communications Range	50 Metres
Trust	
Trust Range	[-1, 1] Continuous
Initial Trust Value	0
Trust Threshold	$>$1.0
Black Hat Nodes	
Number of Black Hat Nodes	[0%, 1, 1%, 5%, 10%, 15%, 20%, 25%, 30%]

5.4.4 Security Evaluation Metrics

Three security evaluation metrics were investigated in this simulation. These metrics are used to determine whether the security properties outlined in Section 5.2.6 are met by the proposed key revocation scheme. These include; the timely removal of a self-revoked White Hat key from the network, along with the distribution of a replacement key to provide secure end-to-end communications, the mitigation of a Black Hat node distributing a spoofed ID key, and finally trust transferral between the self-revoked White Hat key and the replacement key. The metrics are also used to compare the effectiveness of the different trust systems, and whether they are suitable for providing public key authentication and trust transferral for key revocation in an autonomous DTN. The metrics are *Public Key Distribution*, *Revocation Request or Certificate Distribution*, and *Revocation Certificate Trust Value*.

1. The Public Key Distribution metric involves measuring several sub-metrics, they are: *Revoked Key Distribution*, *New Public Key Distribution*, and *Spoofed Public Key Distribution*. The trust based revocation process can achieve security through the timely removal of old revoked keys from other nodes, a prompt distribution of new public keys generated to replace the

old revoked keys, and the mitigation of spoofed revocation public keys, which pose a threat to MITM attacks.

(a) Revoked Key Distribution is a measure of the percentage of nodes that still hold P_R^n over time. This metric is important to determine whether the revocation process is actually removing P_R^n from circulation and use. It is expected that the proposed revocation process would remove instances of P_R^n as quickly as possible as to prevent their use in message encryption or verification.

(b) New Public Key Distribution is the measure of the percentage of nodes that have received P_R^{n+1} designed to replace P_R^n for encryption or verification purposes. The expectation is for P_R^{n+1} to be distributed promptly to facilitate the operational security of the network and prevent potential adversaries from exploiting a window of opportunity during the key transition phase.

(c) Spoofed Key Distribution measures the percentage of nodes that hold a spoofed revocation public key such as $P_{(M,R)}^n$ as defined in Section 5.2.3. This represents a successful attack by the adversary to exploit an opportunistic moment to inject a public key with falsified identity-key bindings. It is expected that the spoofed key distribution be mitigated to prevent adversaries from performing further attacks on the network such as a MITM attack.

2. Revocation Certificate Distribution involves measuring several sub-metrics, they are *White Hat Certificate Distribution*, *Spoofed Certificate Distribution*, and more specifically *Certificate True or False Positives*.

(a) White Hat Certificate Distribution is the measure of percentage of nodes that received C_R. The expectation here is that as many White Hat nodes in the population to receive C_R to allow for prompt removal of P_R^n and adoption of P_R^{n+1}.

(b) Spoofed Certificate Distribution is the measure of percentage of nodes that received a spoofed revocation certificate such as $C_{(M,R)}$ generated by an adversary node in an attempt to remove P_R^n and insert $P_{(M,R)}^n$. It is expected that the revocation scheme mitigate the distribution of spoofed revocation certificates, thus hindering a mechanism of $P_{(M,R)}^n$.

(c) Certificate True or False Positives focusses on the percentage of nodes that are White Hat nodes, in possession of a White Hat revocation certificate such as C_R (True Positive), and the percentage of nodes that are White Hat nodes, in possession of a spoofed revocation certificate such as $C_{(M,R)}$ (False Positive). The requirement is for the revocation scheme to maximise the True Positive occurrences, while minimising the False Positive occurrences. Since the proposed revocation scheme is designed to protect White Hat nodes, the analysis of this metric does not consider Black Hat nodes.

The Revocation Certificate Distribution metrics are important to determine how effective the revocation schemes are at providing security by distributing legitimate White Hat certificates, and suppressing falsified Black Hat certificates. Although the Revocation Certificate Distribution metrics are related to the Public Key Distribution metrics, they will not be the same. Public keys are still distributed through direct contact and carrier nodes so although the results may not be identical, they follow a similar trend for the individual experiments.

3. The Revocation Certificate Trust Value metric only applies to the DS revocation scenarios. It measures the average trust of the Revocation Certificate (C_R) that nodes assign as a result of the List of Signatories (G_R) attached to the certificate. Since signatory nodes will only sign C_R if they are confident in the identity-public key binding of the revoking node N_R, the trust associated with P_R^n is translated to G_R by the signing node. G_R forms the trust assigned to C_R and P_R^{n+1} for a node receiving the certificate. The Revocation Certificate Trust Value is an indicator for trust transferral between P_R^n and P_R^{n+1}.

5.5 Results and Analysis

For each of the experiments conducted, the two security evaluation metrics described in Section 5.4.4 were used to determine the effectiveness and behaviour of the proposed key revocation system.

5.5.1 Public Key Distributions

The quantity of revoked keys in the system over time indicates if the proposed key revocation system is effective in removing the revoked keys from active use. A timely removal of the revoked key (P_R^n) fulfils Security Property 1 in Section 5.2.6. Meanwhile, the quantity of new public keys (P_R^{n+1}) over time provides a metric on whether the P_R^{n+1} are actually replacing P_R^n. The distribution of P_R^{n+1} to nodes fulfils Security Property 2 in Section 5.2.6. With the constant threat of Black Hat nodes in the network, the distribution of Black Hat revocation keys ($P_{(M,R)}^n$) provides an important metric on whether the proposed key revocation system is allowing the distribution of new legitimate public keys such as P_R^{n+1}, whilst preventing the distribution of spoofed identity keys such as $P_{(M,R)}^n$. Mitigating the distribution of $P_{(M,R)}^n$ fulfils Security Property 4 in Section 5.2.6.

5.5.1.1 Revoked Key Distribution

Figure 5.8 depicts the revoked key distribution over time for the RO, RR, and DS scenarios when there are no Black Hat nodes. The time is measured in the number of seconds after the revocation event. The RO and RR scenarios follow the same trend as their revocation process is identical. This is because the only restrictions or safeguards in removing P_R^n from active use is C_R. All three scenarios initially start with 71% of nodes in the system holding P_R^n. For the RO, and RR scenarios, all instances of P_R^n were successfully removed from active use after the first 5,000 seconds post revocation event. The DS scenario, requiring signatories on the Q_R to elevate it to a C_R, has a delay, which is shown by the dotted line signifying the time when Q_R was elevated to C_R. Prior to C_R being distributed, the number of P_R^n instances actually increased. This is because the only node that has removed the key from use is the owner of the key (the revoking node N_R) and no other node has been instructed to remove P_R^n from use. Therefore, it is still being distributed by carrier nodes. When the threshold of signatories is reached and Q_R is elevated to C_R, the quantity of P_R^n begins to decline as the C_R is distributed. Unlike the RO and RR scenarios, the DS scenario does not remove all instances of P_R^n from active use and begins to plateau out. It is expected that given enough time and no adversary nodes, eventually all instances of P_R^n would be removed from active service.

With the introduction of Black Hat nodes into the simulation, there are some variations in the revoked key distribution results. The 10% Black Hat nodes

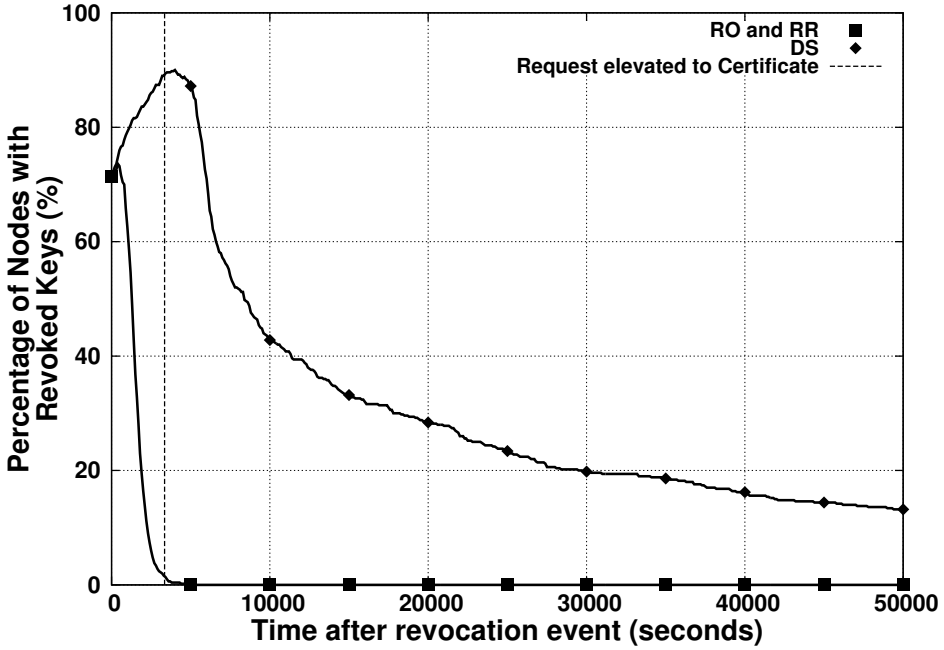


Figure 5.8: Revoked Key Distribution over time with no Black Hat Nodes.

experiment is presented as a typical experiment, with similar trends in the other experiments with varying Black Hat node populations. All revoked key distribution results can be found in Appendix G. Figure 5.9 depicts the revoked key distribution over time with a 10% Black Hat node population for a Multiplying attack. Similar results were obtained for a Single attack. Due to the presence of Black Hat nodes attacking the system prior to the revocation event, the starting percentage of revoked keys in active service is only 61%. As the Black Hat node population is increased, the starting percentage of revoked keys in service decreases. The RO and RR scenarios depict a quick removal of P_R^n, with all instances removed within 5,000 seconds after the revocation event. The DS revocation scenario, requiring signatories for Q_R to elevate it to C_R, has a delay shown by the dotted line signifying the time when Q_R was elevated to C_R. Prior to C_R being distributed, the percentage of P_R^n actually increased. This is because the only node that has removed P_R^n from use is the owner of the key (N_R) and no other node has been instructed to remove P_R^n from use. Therefore, it is still being distributed by carrier nodes. When the threshold of signatories is reached and Q_R is elevated to C_R, the percentage of P_R^n begins to decline as the C_R

is distributed. At the end of the measurement window, only 2% of nodes still retain P_R^n for the DS revocation scenario. It is expected that given enough time, eventually all instances of P_R^n would be removed from active service.

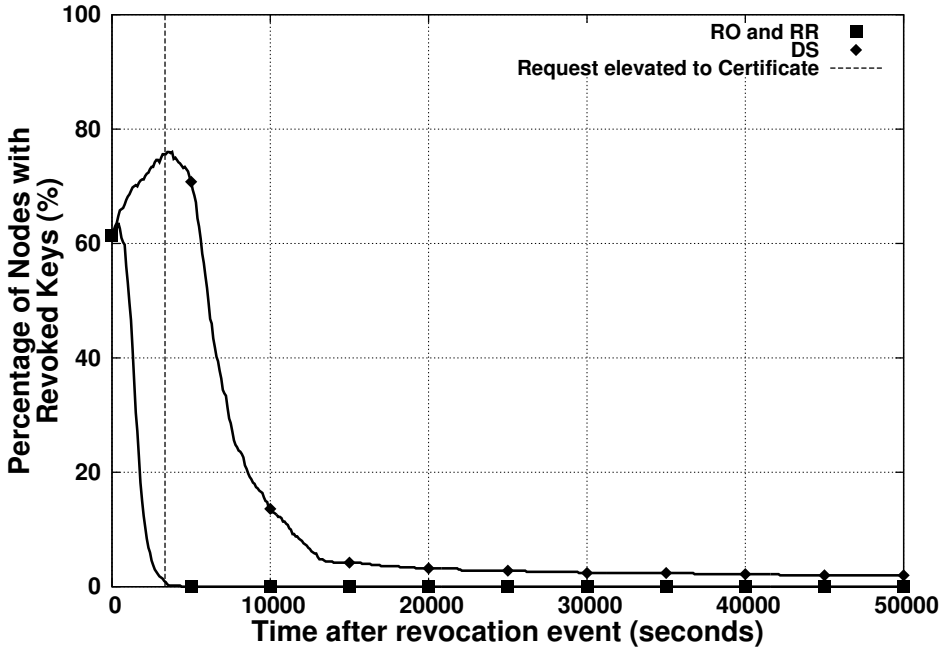


Figure 5.9: Revoked Key Distribution over time with 10% Black Hat Nodes (Multiplying Attack).

The time required for the DS scenario to gather signatories and elevate Q_R to C_R was around 3,300 seconds. The rate of P_R^n removal for the DS scenario is around 1 key every 12 seconds, while for RO and RR scenarios the removal rate is around 1 key every 5.5 seconds. The contribution of both a time delay in gathering signatories and a slower P_R^n removal rate attributes to the increasing disparity in time for P_R^n removal between the RO and RR scenarios, and the DS scenario. Taking the point where 20% of the node population still has P_R^n, both RO and RR scenarios achieve this 1,760 seconds after the revocation event, while the DS scenario only achieves this after 8,575 seconds. Although the RO and RR scenarios outperform the DS scenario in the removal of P_R^n, the distribution of replacement key P_R^{n+1} is also considered.

5.5.1.2 New Public Key Distribution

With no adversary nodes, Figure 5.10 depicts the percentage of nodes with new public keys over time after the revocation event. Unlike the revoked key distribution over time, the RO and RR scenarios differ as they distribute the newly generated keys through different methods. The RO scenario completes key distribution to all 500 nodes after 30,000 seconds, plateauing after 25,000 seconds. The RR scenario provides the most efficient new public key distribution with 15,000 seconds, half the time required for the RO scenario. The DS scenario provides the slowest response when there is no adversary nodes. In the first 10,000 seconds, the DS scenario manages to distribute the new public key to 56% of nodes, outpacing the RO scenario. However, in the next 40,000 seconds until the end of the measurement window, the DS scenario only manages to distribute to an additional 31% of nodes ending with a total of 87% of nodes with P_R^{n+1}. It is expected that given enough time and no adversary nodes, eventually all nodes in the network will have received P_R^{n+1}. Another interesting observation is the efficiency the RR scenario has for distributing the new public key. In the time taken for the DS scenario to sign and elevate Q_R to C_R, the RR scenario managed to distribute the new public key to 85% of nodes. These results indicate how efficient and effective the RR scenario is in distributing the new public key due to the absolute trust transferral. However, this only holds when there are no adversaries in the network.

Figure 5.11a depicts the new public key distribution over time with a 10% Black Hat node population for a Single Attack. All new public key distribution results can be found in Appendix H. Even with only a single node undertaking the Sybil Attack, there is a significant degradation in new public key distribution for the RO and RR scenarios. The RO scenario plateaus around 40% of nodes with P_R^{n+1} 15,000 seconds after the revocation event, and remains there with minor fluctuations until the end of the measurement window. This indicates that N_R had difficulty in distributing P_R^{n+1} to other nodes, as the single Black Hat revocation node performing the Sybil attack had already provided a spoofed key ($P_{(M,R)}^n$).

The RR scenario, which provided the best new public key distribution with no adversaries, reached a peak of 71% of nodes with P_R^{n+1} when there is only a single Black Hat node performing the Sybil Attack. After 10,000 seconds post the revocation event, the percentage of nodes with P_R^{n+1} decreases. This is due

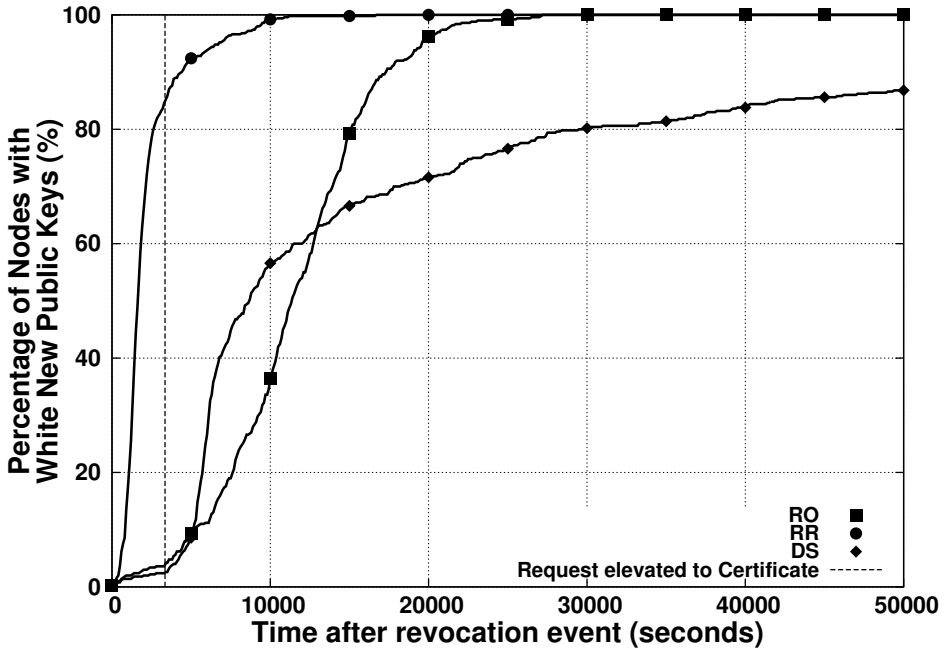


Figure 5.10: New Public Key Distribution over time with no Black Hat Nodes.

to the method of how P_R^{n+1} is distributed. The RR scenario inserts P_R^{n+1} into the keyring where P_R^n was previously found. From the experiments, many of the P_R^{n+1} instances were in fact in the Approved Key Lists of nodes. However, with a Black Hat node performing a Sybil attack, it is also distributing a spoofed revocation key directly, through direct communications. This in turn means that the spoofed revocation key is also being carried by the nodes who have met the Sybil Black Hat node directly, thereby exploiting the two tiered trust system. The key distribution mechanism means that if a node receives the key directly, it takes a higher trust value than a key received indirectly through carriers. So as seen in Figure 5.11a, the slow decline in nodes with new public keys is attributed to approved key list instances being replaced with direct key instances of the spoofed revocation key.

The DS revocation scenario provides the best new public key distribution of the three scenarios when there is only a single Black Hat node performing the Sybil Attack. In the time prior to Q_R being elevated to C_R in Figure 5.11a, there is a steady increase of nodes with P_R^n, whilst the N_R is collecting signatories. However, once enough signatories have been collected and Q_R is elevated to C_R,

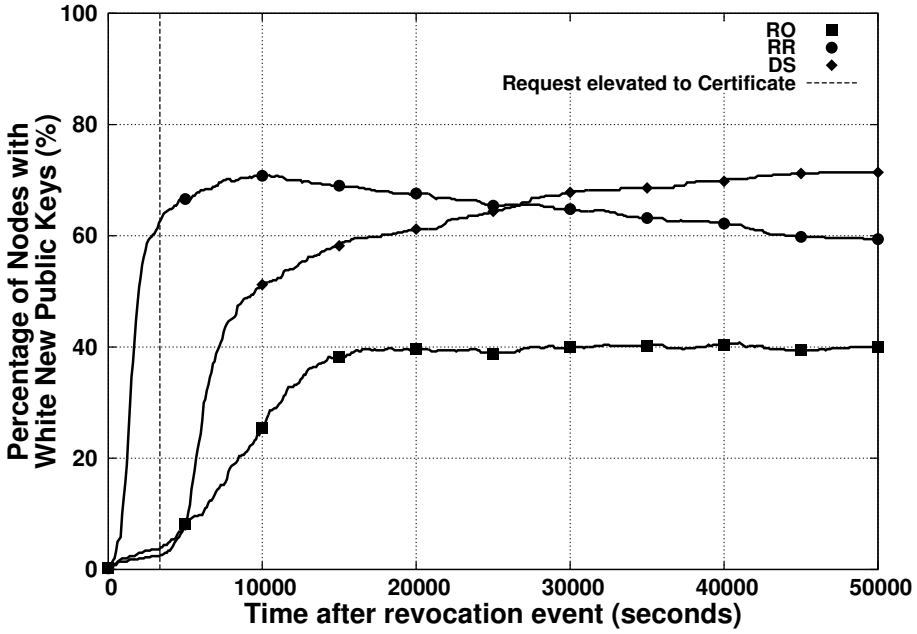
there is a large increase with over 50% of nodes possessing P_R^{n+1} in the first 10,000 seconds post revocation event. After that, a slow but steady increase of 1 new public key every 400 seconds is distributed until 45,000 seconds, where the number of nodes with the new public key begins to plateau at 71%.

With only a single Black Hat node performing the Sybil attack, an adoption of P_R^{n+1} of only 40% in the RO Scenario, 59% in RR, and 71% in the DS revocation scenario was observed. With the addition of a multiplying adversary where multiple Black Hat nodes perform the Sybil attack, there is a degradation in new public key distribution for all scenarios.

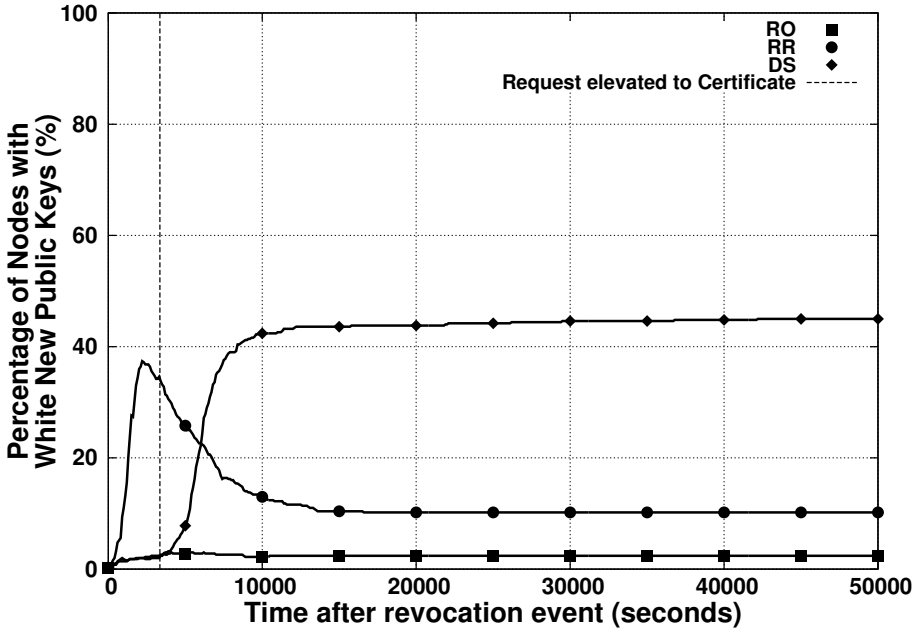
Figure 5.11b depicts the new public key distribution over time with a 10% Black Hat node population for the Multiplying attack. The RO and RR scenarios show a significant loss in adoption of P_R^{n+1}, with results of 2% and 10% respectively at the end of the measurement period. The characteristic spike in the RR scenario was observed. This is where many of the approved instances of P_R^{n+1} are replaced with direct instances of the spoofed revocation key. The rate of replacement in this scenario is significantly faster than the single Black Hat node experiment in Figure 5.11a, due to the significantly greater number of Black Hat nodes performing the Sybil Attack. The DS revocation scenario, achieved an adoption of 45%. Although this only leaves less than half the network with P_R^{n+1}, it provides a significant improvement over the RO and RR scenarios.

The percentage of nodes with P_R^{n+1} at the end of the measurement window was also measured. Figure 5.12a depicts the percentage of nodes with new public keys for each experiment with single adversary attack. Then there are no Black Hat nodes, the RO and RR scenarios distribute P_R^{n+1} to all the nodes, while the DS only achieves distribution to 87% of nodes. However, with the introduction of a Black Hat node performing the Sybil attack, the percentage of nodes with P_R^{n+1} substantially declines for both the RO and RR scenarios. Even at a Black Hat node population of 1 node, the RO scenario only manages to distribute P_R^{n+1} to 44% of nodes. In all experiments where Black Hat nodes are present, the DS revocation scenario always outperformed the RO and RR scenarios. A constant decline in percentage of new public keys is evident as the Black Hat population is increased and at 30% Black Hat node population the DS revocation scenario only manages to distribute new public keys to 50% of nodes, aligning to the Byzantium Generals Problem threshold [81].

Figure 5.12b depicts the percentage of nodes with new public keys for each



(a) Single Attack



(b) Multiplying Attack

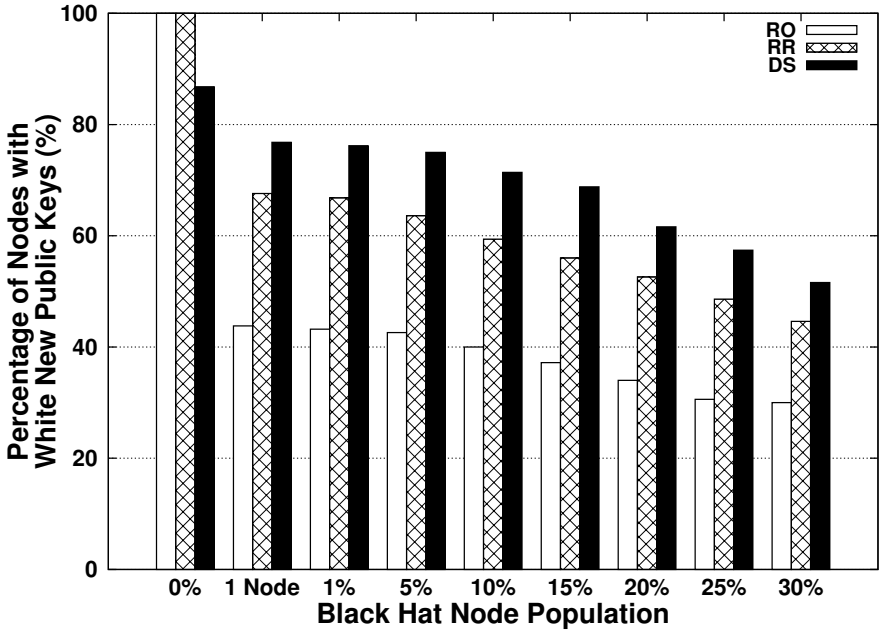
Figure 5.11: New Public Key Distribution over time with 10% Black Hat Nodes

experiment with a multiplying adversary attack. With multiple nodes performing the Sybil attack, both the RO and RR scenarios struggle to distribute the new public key. As with the results from the Single node attack in Figure 5.12a, the DS revocation scenario outperforms both the RO and RR scenarios. It is important to note that even with a 1% Black Hat node population, there is 1 White Hat revoking node, while there are potentially 5 Black Hat revoking nodes. At a Black Hat node population of 5%, there is still 1 White Hat revoking node, whilst there are potentially 25 Black Hat revoking nodes assuming the identity of a single node. With a White Hat to Black Hat ratio such as 1:5 for 1% and 1:25 for 5%, the DS revocation scenario still managed to distribute the new public key to nearly 70% of nodes for a Black Hat population of 1%, and over 50% for a Black Hat population of 5%. At 20% Black Hat node population (100 nodes), the DS scenario has difficulty to even distribute P_R^{n+1} to 20% of nodes. Reciprocal results are observed in the percentage of nodes with spoofed revocation keys.

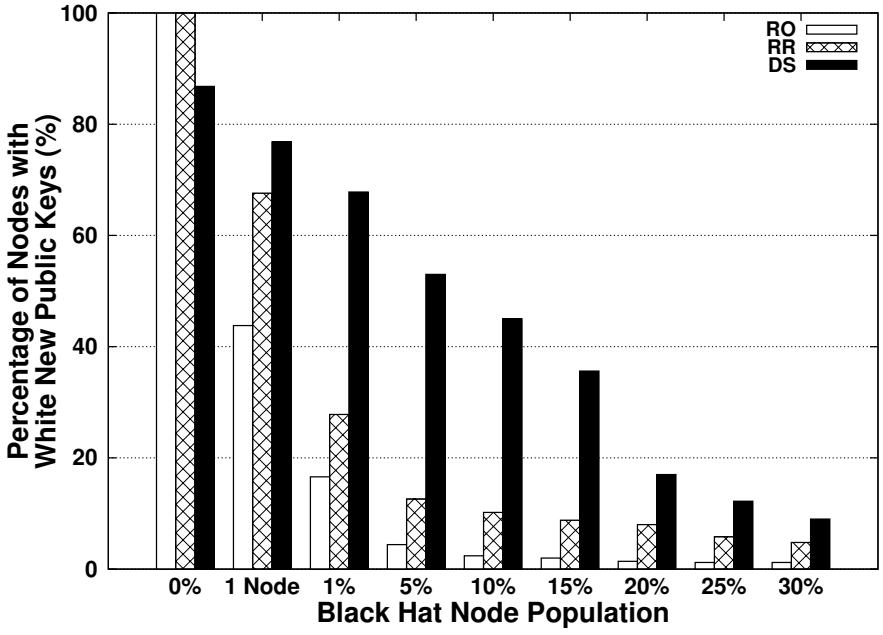
The results from the New Public Key Distribution metric show that a DTN environment with no adversaries, the RO and RR revocation schemes provide an efficient distribution of P_R^{n+1}, to a greater number of nodes than the DS revocation scheme. The provision of public key authentication for the DS scheme, results in an additional delay while Q_R is being signed before being elevated to C_R, which is then distributed. However, with the introduction of Black Hat nodes, the RO and RR schemes have difficulty in distributing P_R^{n+1}, due to the lack of public key authentication. This lack of public key authentication allows Black Hat nodes to distribute $P_{(M,R)}^{n}$, which displaces the legitimate P_R^{n+1} from other nodes. The provision of public key authentication in the DS revocation scheme is effective in mitigating the distribution of $P_{(M,R)}^{n}$, and allows the distribution of P_R^{n+1} even at high Black Hat node populations. The Spoofed Revocation Key distribution results provide further evidence supporting this finding.

5.5.1.3 Spoofed Revocation Key Distribution

Figure 5.13a depicts the spoofed revocation key distribution over time with a 10% Black Hat node population for Single Attack. All spoofed revocation key distribution results can be found in Appendix I. For the RO scenario, the number of spoofed revocation keys steadily increases at a rate of 1 key every 53 seconds until about 15,000 seconds, where the percentage of nodes with spoofed revocation keys plateaus at around 50%. The percentage of nodes with the spoofed



(a) Single Attack

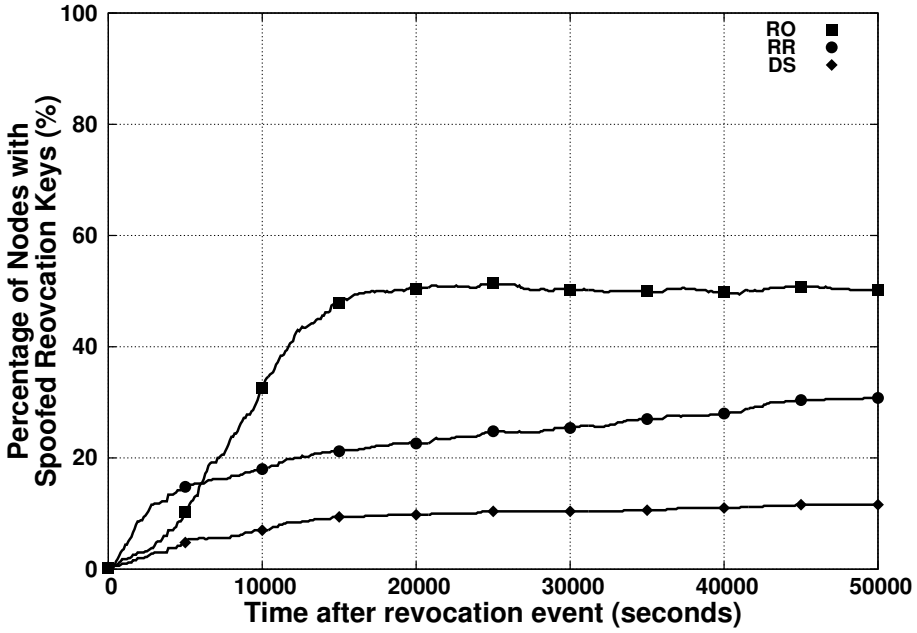


(b) Multiplying Attack

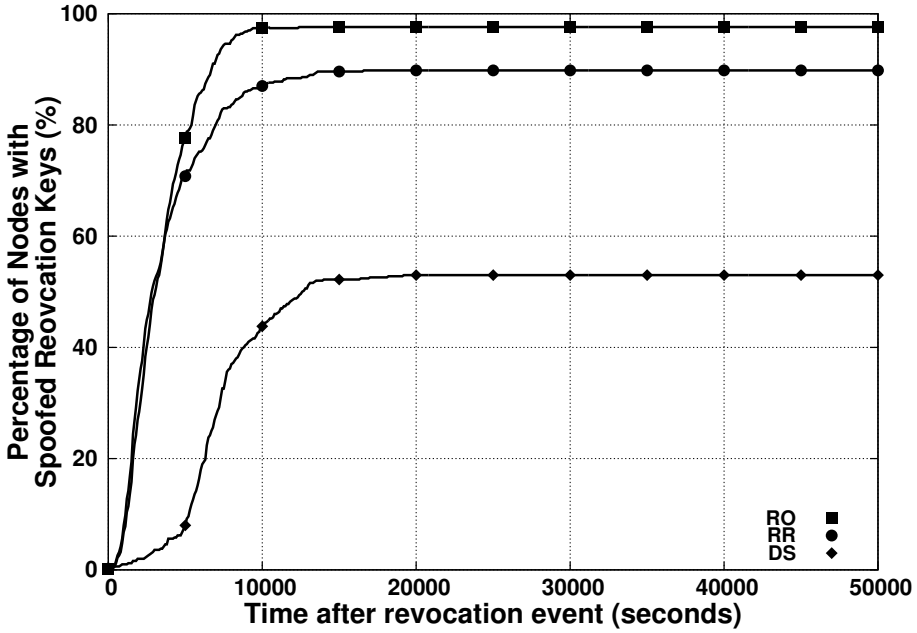
Figure 5.12: Percentage of nodes with New Public Keys for each experiment.

revocation keys in Figure 5.13a exceeds the percentage of nodes with the new public key as in Figure 5.11a. The RR Scenario also sees a steady increase of 1 key every 9 minutes after 10,000 seconds, which is similar to the rate of decrease in new public keys after the same point in time. This supports the observation that the new public keys in approved lists are being replaced by direct spoofed revocation keys. At the end of the measurement period, 31% of the nodes in the RR scenario have a spoofed revocation key. The DS scenario provided the best result in mitigating the distribution of spoofed revocation keys when there is a single adversary performing the Sybil Attack. In the first 15,000 seconds, the number of spoofed revocation keys increases at a rate of 6.3%/10000s, before plateauing at approximately 10% of nodes. In the single adversary attack experiment, the DS scenario provided a three-fold improvement to the RR scenario, and a five-fold improvement to the RO scenario in minimising the distribution of spoofed revocation keys.

Figure 5.13b depicts the spoofed revocation key distribution over time with a 10% Black Hat node population for Multiplying Attack. Both the RO, and RR scenarios fared poorly in mitigating the distribution of spoofed revocation keys. The Black Hat nodes in the RO scenario managed to penetrate 98% of nodes with a spoofed revocation key. They managed to achieve this result in the first 10,000 seconds post revocation event. The RR scenario also fared poorly in a multiple Black Hat node Sybil attack. The Black Hat nodes in this scenario managed to penetrate 90% of nodes with a spoofed revocation key, achieving that result in 15,000 seconds post revocation event. Both these scenarios demonstrate the strong adversary capabilities in flooding a network with spoofed revocation keys. The DS scenario provided the best result in mitigating the distribution of spoofed revocation keys. The Black Hat nodes in this scenario only managed to penetrate 53% of nodes with the adversary key, achieving this level after 15,000 seconds post revocation event. Although the number of nodes with spoofed revocation keys may exceed the number of nodes with legitimate new public keys, this result is still a significant improvement over the RO and RR scenarios. The DS scenario still leaves the N_R with approximately half of the network with legitimate keys. Varying the number of Black Hat nodes performing the Sybil attack demonstrates that the DS scheme still outperforms both the RO and RR scenarios. However, at higher quantities of Black Hat nodes performing the Sybil Attack, it becomes exceptionally difficult to provide a secure revocation



(a) Single Attack



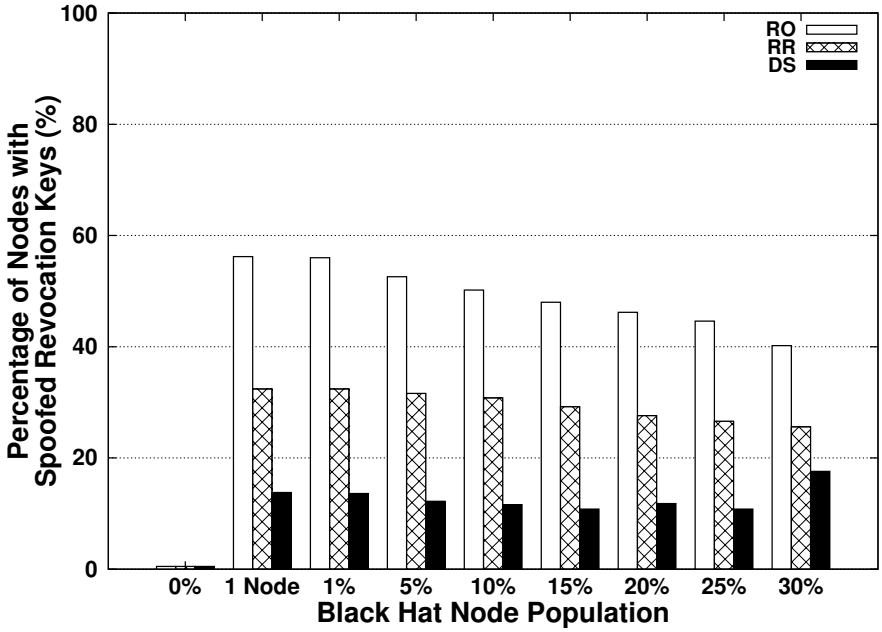
(b) Multiplying Attack

Figure 5.13: Spoofed Revocation Key Distribution over time with 10% Black Hat Nodes.

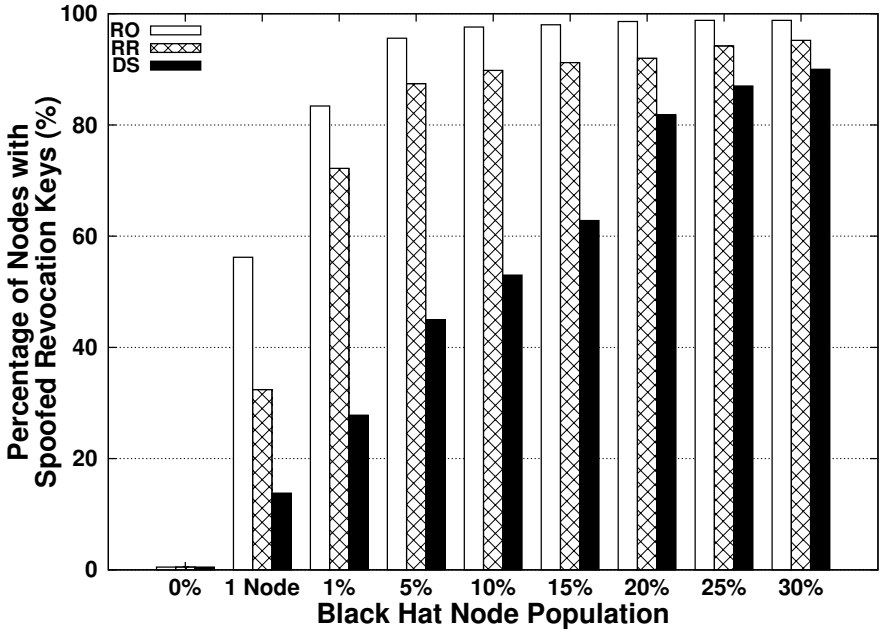
environment. This is due to the reduction in number of potential nodes to act as signatories, and the number of nodes that are willing to pass on the revocation certificate is significantly reduced.

Figure 5.14a depicts the percentage of nodes with spoofed public keys for each experiment with a single adversary attack. Reciprocal to the new public keys in Figure 5.12a, the percentage of nodes with a spoofed revocation key in the RO scenario significantly increases with the introduction of Black Hat nodes. The RR scenario trends mitigate the spoofed revocation key by half, with the DS revocation scenario achieving another half reduction in addition to the RR scenario. Interestingly, the percentage of nodes with spoofed revocation keys decreases as the Black Hat node population increases. This can be explained by the nature of the single attack. As the Black Hat node population is increased, the single node performing the Sybil attack has greater difficulty in finding potential nodes to provide the spoofed revocation key, as Black Hat nodes will spoof the revocation key with their own key. This accounts for the decline in percentage of nodes with spoofed revocation keys as the Black Hat node population increases. It is expected that when multiple nodes perform the Sybil attack, the percentage of nodes with spoofed revocation keys will be directly proportional to the Black Hat node population.

Figure 5.14b depicts the percentage of nodes with new public keys for each experiment with a multiplying adversary attack. As expected, with multiple nodes performing a Sybil attack, the percentage of nodes with spoofed revocation keys increases as the Black Hat node population increases. With the introduction of Black Hat nodes, the DS revocation scenario outperforms both RO and RR scenarios by mitigating the spread of spoofed revocation keys. Black Hat node populations between 1 node and 15% achieved a 40-60% reduction in nodes with spoofed revocation keys. While experiments with 20% and greater Black Hat nodes achieved a smaller margin in spoofed revocation key reduction. Taking the 10% Black Hat node population experiment, the Black Hat nodes in this scenario only managed to penetrate 53% of nodes with the adversary key. Although the number of nodes with spoofed revocation keys may exceed the number of nodes with legitimate new public keys, this result is still a significant improvement over the RO and RR scenarios. The DS revocation scenario still leaves the revoking node with approximately half of the network with legitimate keys. Varying the number of Black Hat nodes performing the Sybil attack



(a) Single Attack



(b) Multiplying Attack

Figure 5.14: Percentage of nodes with Spoofed Revocation Keys for each experiment.

demonstrates that the DS revocation scheme still outperforms both the RO and RR scenarios. However, at higher quantities of Black Hat nodes performing the Sybil Attack, it becomes exceptionally difficult to provide a secure revocation environment. This is because of the reduction in both; the number of potential White Hat nodes to act as signatories, and the number of nodes that are willing to pass on the revocation certificate.

5.5.2 Revocation Certificate Distribution

The distribution of revocation certificates both legitimate (C_R) and falsified ($C_{(M,R)}$) demonstrates the proposed revocation scheme achieves the Security Properties described in Section 5.2.6. More specifically, Security Property 3 is achieved by the distribution of the White Hat Certificate C_R, the prevention of distribution of the Spoofed Certificate $C_{(M,R)}$, and the distinguishing of of these two certificates. As such, three sub-metrics were measured: First is the White Hat Certificate distribution, second, is the Spoofed Certificate distribution, and finally the Certificate True or False Positives.

5.5.2.1 White Hat Certificate Distribution

Figure 5.15 depicts the distribution of revocation certificates from White Hat nodes over time when there are no adversary nodes. The dotted line indicates the time when the revocation request was elevated to a certificate for the DS scenario. In the time elapsed for the request to be elevated to a certificate for the DS scenario, both RO and RR scenarios have achieved nearly a 100% revocation certificate distribution. It achieved 100% distribution within the first 5,000 seconds of the revocation event. The DS scenario however has a large initial spike, before gradually reaching approximately 85% at the end of the measurement window. Given there are no adversary nodes, it is expected that given sufficient time and random movement of nodes, the DS scenario would also achieve a 100% distribution.

With the introduction of adversarial nodes into the system, at the level of 10% Black Hat nodes, with only a single node performing the Sybil attack, there was a reduction in White Hat revocation certificate distribution. Figure 5.16a depicts the distribution of revocation certificates from White Hat nodes over time in a 10% Black Hat node single attack scenario. All white hat certificate distribution results can be found in Appendix J. Both the RO and RR scenarios

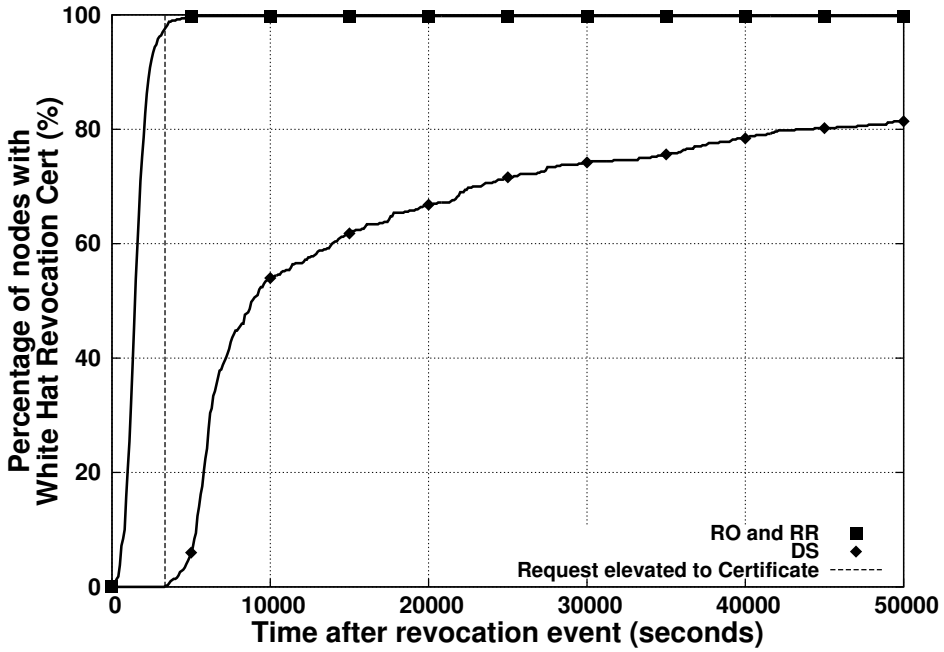


Figure 5.15: Percentage of nodes with White Hat Revocation Certificates over time with no Black Hat Nodes.

plateau at a distribution of 87% of nodes. A decrease in White Hat certificate distribution of 13% was observed, compared to the results with no Black Hat nodes in Figure 5.15. The DS scenario also followed a similar trend to the results with no Black Hat nodes. There was a slight decrease of 6% to achieve a White Hat certificate distribution of 79% compared to 85% in the experiment with no Black Hat nodes.

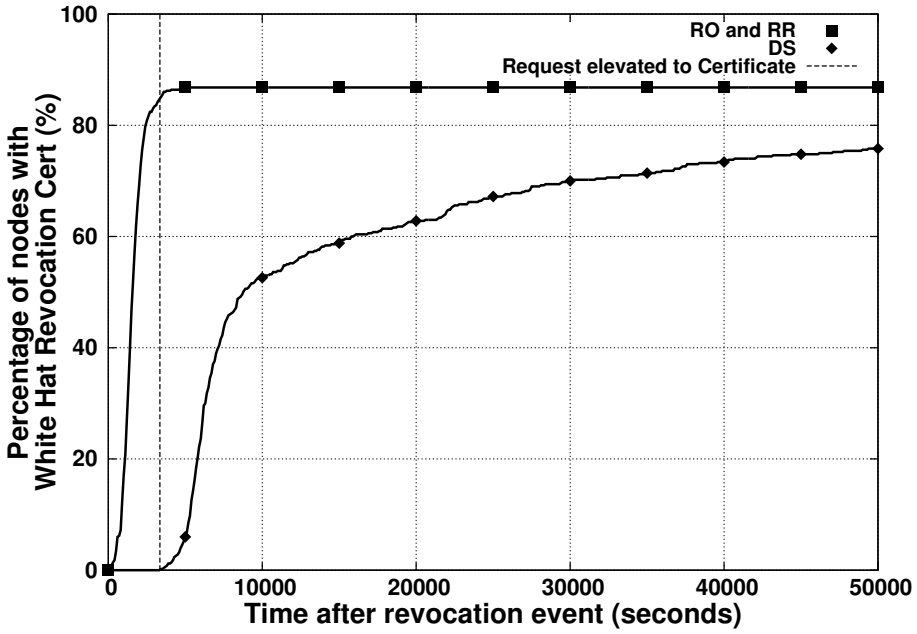
Figure 5.16b depicts the White Hat Certificate distribution over time for the multiple 10% Black Hat nodes experiment. The results of both the RO and RR scenarios indicate that all the certificates were distributed within the first 5,000 seconds post revocation event. The maximum certificate distribution is 64% of nodes. The DS revocation scenario only started distributing the certificate after Q_R was elevated to C_R, which was about 3,300 seconds post revocation event. A significant amount of certificates were distributed within the first 10,000 seconds, with the DS scenario equalling the result of the RO and RR scenarios within the first 20,000 seconds. There is a steady increase of approximately 1 certificate distributed every 4 minutes. At the end of the measurement window, the DS

scenario achieved a White Hat Certificate distribution of 79% of nodes. The DS revocation scenario had no $Q_{(M,R)}$ ever elevated to a $C_{(M,R)}$ for distribution. This provided N_R the opportunity to distribute C_R without competition from instances of $C_{(M,R)}$ from the Black Hat nodes. If even one instance of $Q_{(M,R)}$ was elevated to $C_{(M,R)}$ and subsequently distributed after the measurement period ($t = 50000 + 1$), the ability to distribute this instance of $C_{(M,R)}$ to other nodes would be hampered since 79% of nodes would already have C_R. This leaves only a potential 21% of the node population that may receive $C_{(M,R)}$. Without the particular details on the signatories of $C_{(M,R)}$, the spoofed revocation certificate may still not be accepted by other nodes due to the proposed distributed revocation framework proposed. Thus, leaving an even smaller potential sample of nodes that would accept $C_{(M,R)}$.

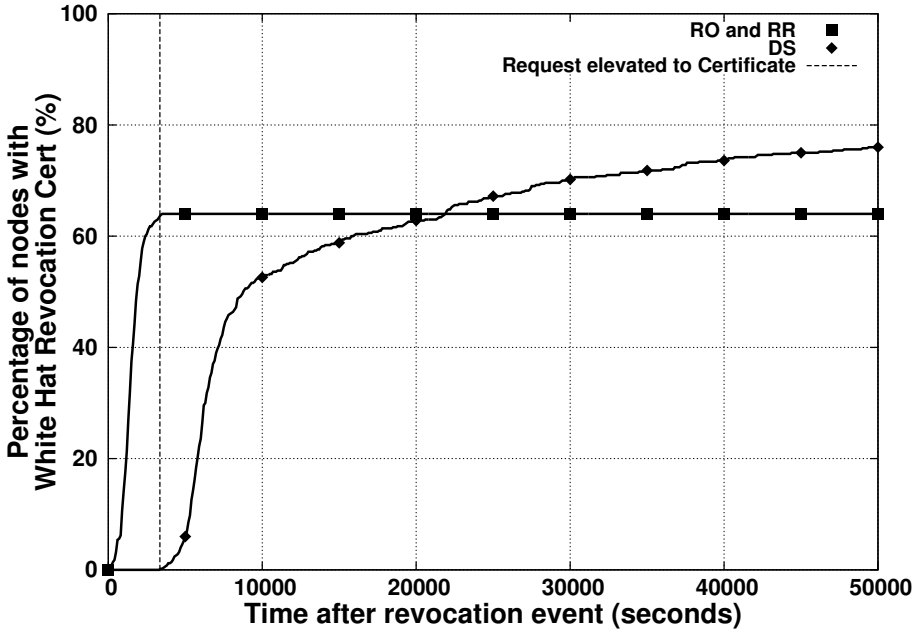
5.5.2.2 Spoofed Certificate Distribution

The percentage of nodes that received a spoofed revocation certificate was also measured. Figure 5.17b shows the results when multiple Black Hat nodes perform the Sybil attack with 10% Black Hat node population. All spoofed certificate distribution results can be found in Appendix K. In this experiment, both RO and RR scenarios saw a $C_{(M,R)}$ distribution of 26%. In contrast, the DS revocation scenario saw a complete prevention in distribution of $C_{(M,R)}$, as $Q_{(M,R)}$ was never elevated. Although only 26% of the node population received $C_{(M,R)}$ for RO and RR scenarios, the percentage of nodes who have received $P^n_{(M,R)}$ in Figure 5.12b is nearly 98% and 90% respectively. This discrepancy between certificates distributed and public keys distributed is attributed to the compounding effect of carried certificates being passed on, and the Black Hat node directly distributing $P^n_{(M,R)}$. It then becomes important for nodes to be able to distinguish between a legitimate certificate, and a spoofed certificate.

In the single attack, there is only one Black Hat node attempting to distribute a spoofed revocation certificate. Figure 5.17a depicts the results over time for the experiment with 10% Black Hat nodes single attack. The RO and RR scenarios both see a 13% distribution of the spoofed revocation certificate, while the distributed revocation sees a complete prevention in distribution of the spoofed revocation certificate, as the spoofed revocation request was never elevated to a certificate since there was not enough signatories.

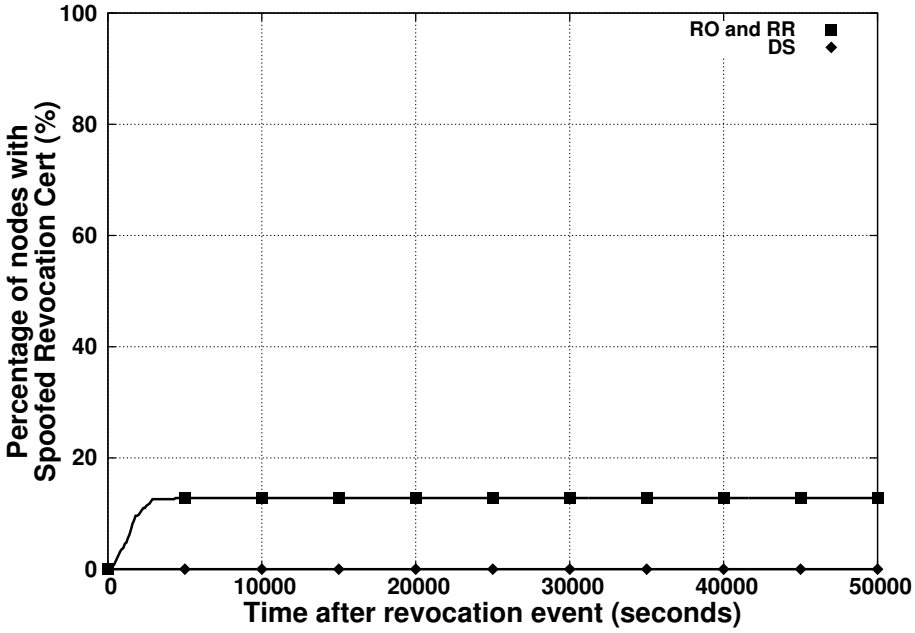


(a) Single Attack

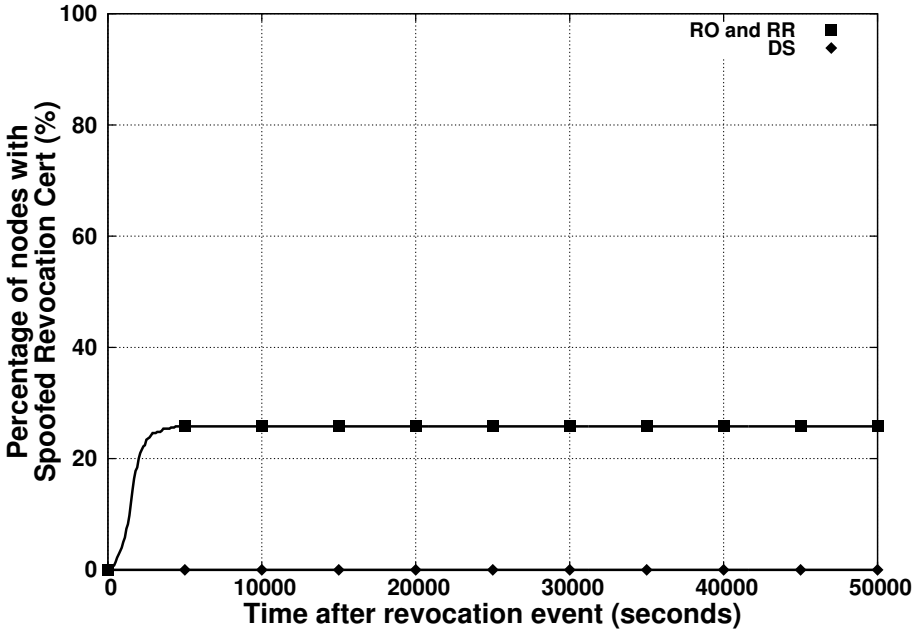


(b) Multiplying Attack

Figure 5.16: Percentage of nodes with White Hat Revocation Certificates over time with 10% Black Hat Nodes.



(a) Single Attack



(b) Multiplying Attack

Figure 5.17: Percentage of nodes with Spoofed Revocation Certificates over time with 10% Black Hat Nodes.

5.5.2.3 Certificate True or False Positives

The Certificate True or False Positives for White Hat nodes was also measured. A certificate True Positive is when a White Hat node receives a White Hat revocation certificate, while a certificate False Positive is when a White Hat node receives a spoofed revocation certificate. This metric used the multiplying attack to observe the compounding effect of multiple nodes perpetrating the Sybil attack. Figure 5.18 depicts a summary of True Positive percentages for the varying Black Hat node populations. When there are no adversary nodes, the RO and RR scenarios are successful in distributing a legitimate certificate to all nodes. The addition of public key authentication in the DS scheme results in a small percentage of nodes either not receiving C_R, or not accepting C_R due to the public key confidence is still below the threshold. However, with the introduction of adversaries, the DS revocation scenario outperformed both RO and RR scenarios. As the number of Black Hat nodes increases, the RO and RR scenarios decline at a greater rate than the DS scenario. In the extreme case where the Black Hat node population makes up 30% of the node population, The percentages of nodes with True Positives is less than 30% for the RO and RR scenarios, while the DS scenario still exceeds 50% of the node population. This indicates that over half of the node population holds a legitimate revocation certificate.

Figure 5.19 depicts a summary of False Positive percentages for the varying Black Hat node populations. As expected, when the Black Hat node population is increased, the percentage of certificate False Positive instances also increases. For all experiments, the provision of public key authentication in the DS scenario resulted in 0 False Positive certificates distributed. No instance of $Q_{(M,R)}$ was elevated to $C_{(M,R)}$ and subsequently distributed. However, for the RO and RR scenarios, even at the lowest Black Hat node population of 1 node, 13% of the node population accepted a False Positive Certificate. This gradually increases to a little over 40% as the Black Hat node population increases to 30% of the node population.

The certificate true or false positives metric provides additional evidence that the DS revocation scheme is providing public key authentication. In an environment with adversaries, the DS scheme results in a greater number of true positive certificates being distributed than the RO and RR scenarios, which provide no public key authentication. By providing public key authentication for the revo-

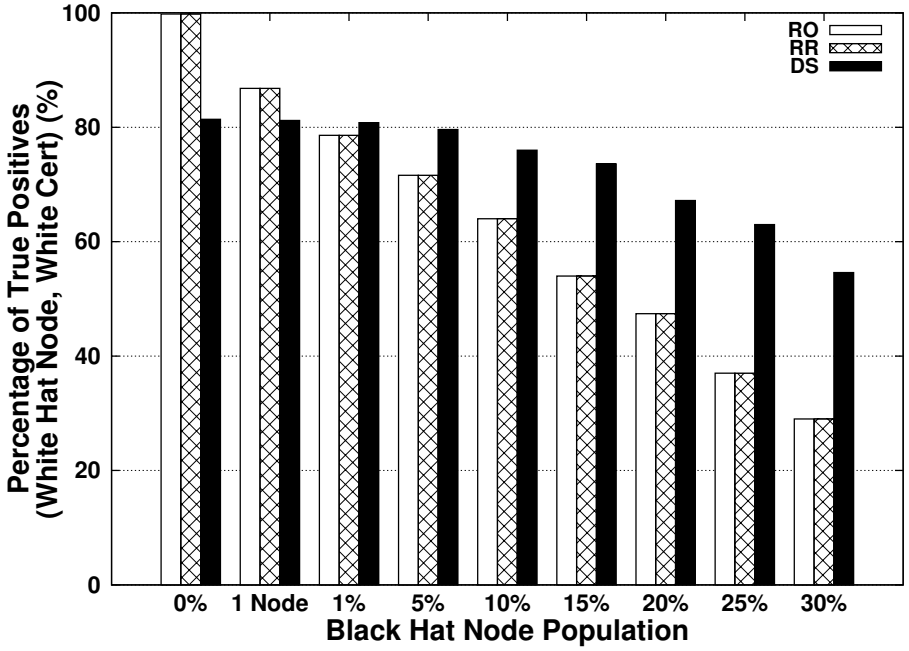


Figure 5.18: Percentage of True Positives (White Hat Node with White Certs) for Multiplying Attack.

cation certificate, the DS scheme successfully prevents a White Hat node from accepting a spoofed revocation certificate. As the results indicate, no White Hat node approved a spoofed revocation certificate in any experiment.

5.5.3 Comparison of Revocation Certificate and Key Distributions

This section will focus on the multiplying Black Hat attack. When comparing the results of the key distribution in Figure 5.12b and the results of the revocation certificate in Figure 5.18, a significant difference in results was observed. In this experimental system setup and adversary model, distributing C_R does not necessarily correspond to the distribution of P_R^{n+1}. Specifically, in the DS revocation scenarios of Figure 5.18, the percentage of nodes that are White Hat nodes and have received a White Hat certificate remains consistently above 50% of the node population for all variations of adversary node numbers. However, the percentage of nodes with new public keys in Figure 5.12b is less than the

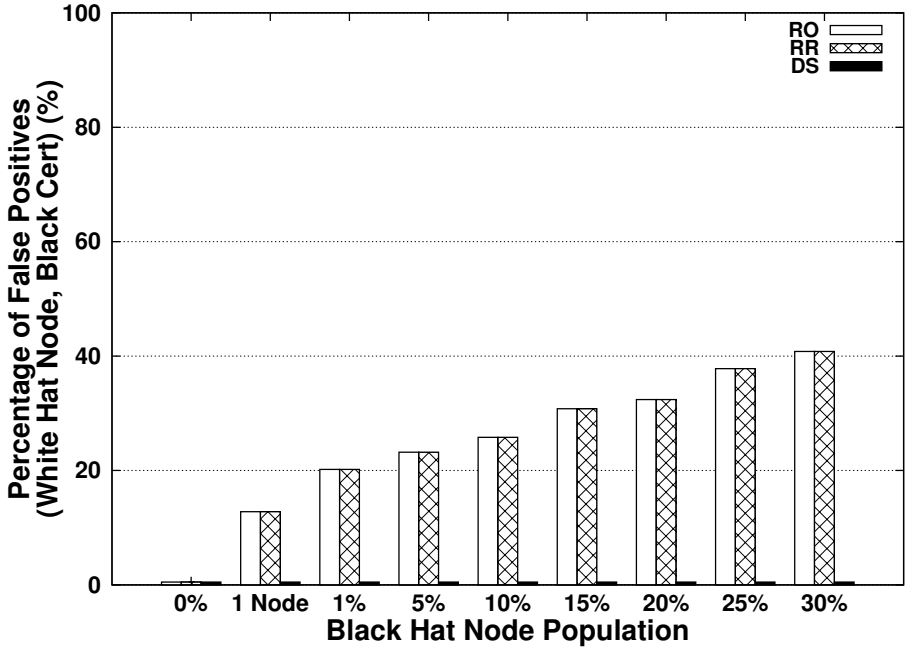


Figure 5.19: Percentage of False Positives (White Hat Node with Black Certs) for Multiplying Attack.

revocation certificate distribution. This due to the fact that receiving C_R is not the only method to receive the P_R^{n+1}. Nodes may receive P_R^{n+1} by one of two methods, the first is direct contact from the revoking node, and the second through the revocation certificate carried by other nodes.

When adversary nodes are introduced, new public key distributions decline, particularly as the adversary population is increased. The legitimate White Hat node that is requesting the revocation (N_R) is the only node that is directly distributing P_R^{n+1}. Concurrently there is a large percentage of Black Hat nodes that have assumed and claim to be the revoking node, therefore providing the opportunity to distribute spoofed revocation keys, even though G_R may have not been elevated to C_R. An example of this scenario occurring is when a node N_i has received a spoofed revocation key from the adversary directly, and added this key to D_i. At a later point in time, it receives C_R - the revocation certificate of the legitimate revoking node from a carrier. After trusting and accepting this certificate, it determines that the old key to be revoked (P_R^n) is not in the key list, and therefore the revocation process does not take place. It does however trust

the certificate, to the extent of adding it to R_i, and thus carrying and distributing C_R to other nodes. This occurrence accounts for the disparity between the results of key distribution and revocation certificate distribution in Figures 5.12b and 5.18 respectively.

The higher success rate of accepting the legitimate revocation certificate when compared to the new public keys, may mean that the only method of distributing P_R^{n+1} is through C_R. However, this creates several issues. First, is the occurrence when a node (N_i) has never met the revoking node (N_R) prior to the revocation event. N_i would have no instance of P_R^n and provide no confidence to determine whether the P_R^n is to be revoked or not since the have no history prior. This may be addressed through the signatories who have signed C_R. The second occurrence is in a system model unlike the one simulated. When dealing with a large scalable open and dynamic network where nodes may join of leave at any time, this methodology of distributing new public keys would disadvantage nodes that have joined the network at a later time.

5.5.4 Revocation Certificate Trust Value

The final metric measured was the Revocation Certificate Trust Value. This metric provides an indicator on trust transferral between P_R^n and P_R^{n+1}. The measurement of this metric along with the White Hat Certificate Distribution metric in Section 5.5.2.1 fulfils Security Property 5. The trust value attached to the revocation certificate is inherited from G_R, the list of signatories. Each node that receives C_R will compute and assign their own trust value and determine whether to accept or reject the certificate. Table 5.4 shows the average trust values that nodes assign upon receipt of C_R for varying quantities of Black Hat nodes. The results show as expected that as the number of Black Hat nodes increases, the average trust value decreases. The average revocation certificate trust values for the multiplying attack are also lower than the average trust values for the single attack. The difference in average trust values between experiments where there are no black hat nodes, and experiments with 30% black hat nodes, is due to the trust value of the signatory nodes. In all experiments conducted, C_R was only signed by another node N_A if $T_{(R,A)}$, exceeded T_A^D (Equations 5.2 and 5.3). Sufficient trust is required before N_A will agree to sign C_R. An added safeguard to prevent a Black Hat node from signing C_R to intentionally lower the trust value, is the ability for N_R to accept or reject the signature from N_A.

The signature from N_A will only be accepted by N_R if $T_{(R,A)}$, exceeds T_R^D. These two threshold requirements during the signing process, prevents the intentional lowering of trust associated with C_R and subsequent P_R^{n+1}. The high trust values in Table 5.4, even when there are large quantities of Black Hat nodes, provides evidence of the signature approval scheme providing trust transferral.

Table 5.4: Average Trust Value of Revocation Certificate

Black Hat Nodes	**Average Trust of C_R**	
	Single	**Multiplying**
0%	0.8088	0.8088
1 Node	0.8084	0.8084
1%	0.8074	0.8072
5%	0.7957	0.7953
10%	0.7867	0.7861
15%	0.7740	0.7723
20%	0.7561	0.7534
25%	0.7544	0.7392
30%	0.7277	0.7223

5.5.5 Comparison of Revocation Schemes

In this section, different related revocation schemes presented in Chapter 2 are compared with the proposed DS revocation scheme. Table 5.5 summarises the comparison between the revocation schemes. They are compared on key distribution mechanism, the communications overhead, storage overhead, and the security of the various schemes. The first two are Raya et al. [117] and Lin et al. [87]. Both these schemes rely on centralised infrastructure to perform and manage the revocation. Hoeper and Gong [65] utilises an IBC based scheme. Although distributed, it still relies on a KGC to bootstrap and initialise the network. The DS revocation scheme is entirely distributed, decentralised, and non-reliant on any party during key distribution much like the PGP model. Due to either the centralised and distributed nature of the schemes compared, the communications overheads differ. The centralised schemes of Raya et al. [117] and Lin et al. [87] provide the minimal number of messages of N. Hoeper and Gong [65] being distributed, in a worst case scenario is the order of edges between nodes. The DS revocation scheme at minimum also equals Hoeper and Gong at the order of edges. However, with the inclusion of signatories, there is an extra component which, in the worst case scenario, is the order of nodes. This

is a worst case scenario, in reality, this parameter is significantly lower. The size of the messages is also compared. Raya et al. [117] utilise the distribution of a CRL, however provide a mechanism of compressing this using bloom filters. This introduces false positives. Lin et al. [87] utilise the RSU infrastructure to filter and distribute only the differences in the CRL for that particular node, thereby only distributing a ΔCRL. The Harakiri message in Hoeper and Gong [65] is essentially a ΔCRL. This is also the case in the DS revocation scheme where the Revocation Certificate is a form of ΔCRL.

The storage requirements are consistent between all schemes. Although the proposed DS revocation scheme has provision for the removal of old entries in the CRL. The security metrics compare the infrastructure, revocation type, backward secrecy, and general adversary model. Both Raya et al. [117] and Lin et al. [87] are centralised schemes, while Hoeper and Gong [65] and the DS revocation scheme are distributed. Raya et al. [117] and Lin et al. [87] concentrate on a revocation process for node removal. However, due to the reliance on a centralised CA, there is also provision for key revocation and replacement. These two centralised schemes also take the position that a malicious or misbehaving node should be removed from operation. Hoeper and Gong [65] is only capable of node revocation in the distributed scheme. The Harakiri message only supports the removal of the node from service. As it is an IBC scheme, they are reliant on a KGC to perform the key replacement process. The DS revocation scheme is based on the PGP scheme and nodes are capable of generating their own key pairs. As a result, it provides a mechanism for compromised keys to be revoked and replaced without expelling the node from service. The proposed scheme also assumes that a DTN may cover a large geographic area, with numerous nodes, therefore making it difficult or infeasible to revoke a node from service. Instead of removing a malicious node, it mitigates or limits the damage it can cause.

Backward secrecy of the key revoked is also considered. Both centralised schemes support backward secrecy, as the compromised key is removed from service. Hoeper and Gong [65] rely on the private key to be included in the Harakiri message as a method of public key authentication. This creates a backward secrecy issue as the confidentiality of previous messages is compromised. The DS revocation scheme relies on a revocation certificate that achieves both backward secrecy and public key authentication for both old and new public keys. The centralised schemes also assumes a weak adversary model. This is when the

CA is considered to be completely trustworthy, the final authority of trust, and provides public key authentication. It is considered to be either unhackable, or outside the scope of attack. The decentralised schemes assume that nodes can be compromised, and that they form the CAs of the network, the CA can be compromised. Therefore, Hoeper and Gong [65] and the proposed DS revocation scheme assume a strong adversary model, where all entities of the network can be compromised.

In comparing revocation schemes, the proposed DS revocation scheme provides equal or better performance metrics in all categories except the number of messages communicated. The distributed nature of the revocation scheme proposed provides resilience to a stronger adversary model than centralised schemes. Backward secrecy and public key authentication is also achieved. It is also assumed that it may be infeasible to remove an adversary or misbehaving node due to the scale of the network. As a result, a distributed key only revocation mechanism that distributes a ΔCRL was presented.

Table 5.5: Comparison of Revocation Schemes

Scheme		**Raya et al.**	**Lin et al.**	**Hoeper and Gong**	**DS**
Key Distribution		CA Based	CA Based	IBC	LCF Based
Message Size		CRL[1]	ΔCRL	ΔCRL	ΔCRL
No. of Messages		N	N	O(Edges)	O(Edges)
Storage		CRL[1]	CRL	CRL	CRL
Security	**Infrastructure**	Centralised	Centralised	Distributed	Distributed
	Revocation Type	Node + Key	Node + Key	Node	Key Only
	Backward Secrecy	Yes	Yes	No	Yes
	Adversary Model	Weak	Weak	Strong	Strong

1. CRL can be compressed using Bloom Filters

5.6 Discussion

This section discusses the implications and issues related with the proposed revocation scheme, and results. The implementation of a trust and reputation system to provide public key authentication, through public key confidence for key revocation provides a few interesting issues.

1. There are delays in authentication. Public key authentication and revocation certificate authentication is subject to a trade-off in efficiency. Even with the mobility model, which increases the exposure to neighbouring network entities, there is additional delay in providing authentication in com-

parison to no authentication. For some application scenarios, this trade-off between security and efficiency might not be feasible or desirable.

2. The ability for the DS revocation scheme to provide some trust transferral between versions of keys has security implications. Although the DS revocation scheme was developed to address unplanned revocation events, the scheme can also be applied to planned events. In both cases, some trust transferral can be achieved between key versions. As a result, key usage durations may be affected. Currently, PGP users will generate keys for long durations of use (sometimes indefinitely) due to difficulty in trust transferral. However, to avoid private key compromise, the use of sub keys are also generated for various security operations [27]. The provision of authentication and trust transferral by the DS scheme may result in keys being generated for shorter durations, as revocation and replacement would provide *some* trust transferral. Thereby, improving security as keys do not remain in operation indefinitely.

3. The DS key revocation scheme can not only be applied to autonomous networks, but to human social networks. The concept of the DS key revocation scheme can provide Multiple-Entity Factor Authentication, and can be applied to the revocation of credentials (passwords) for user authentication. In particular, in 'forgot my password' schemes. It can be used as a last resort measure, for users who have forgotten their credentials, and exhausted all other challenge-response options. Users may request known and trustworthy friends to attest their identity and the associated profile. Variables such as length of service use, number of friends would contribute to the number of friends needed to provide Multiple-Entity Factor Authentication.

4. The proposed LCF trust establishment scheme and DS key revocation scheme can be further applied to the CA model used for the Internet. With over 600 CAs [36] performing certificate operations on the Internet, and a few root CAs, there is no provision of revoking a trusted Root CA Certificate as it is self signed and has no trust mechanism for verifying a CRL. Certificate revocation in this scenario would require a manual removal of the Root CA certificate from browsers.

5. Trust and reputation systems are not perfect. The implementation of a

trust and reputation system addresses the autonomous nature of the DTN environment described. However, a condition may arise where a legitimate node intends to revoke a key, however, is hindered by the DS scheme from revoking their private and public key from service.

5.7 Conclusion

Key revocation and replacement is a critical and integral part of the key management lifecycle of any network. The ability to revoke keys due to limited lifetimes, planned obsolescence, and more importantly private key compromise, is an important aspect of providing security for the ongoing operations of a network. This is even more critical in an autonomous DTN environment, where the characteristics can translate to deployment in hostile environments. The provision of public key authentication is a foundational concept to the continuance of confidentiality, data integrity, and message authentication, particularly in DTN applications, given how data is routed. Public key authentication is particularly important in the key revocation and replacement phase as it prevents an adversary from falsely revoking a legitimate node, as well as assuming the identity of a legitimate node. Past proposals to provide key revocation for DTNs have relied on a centralised TTP such as a CA, or have focused on the planned revocation event (key expiry, planned obsolescence) rather than an unplanned revocation event (private key compromise).

This chapter designed, developed and evaluated a key revocation and replacement scheme that provides public key authentication for application in an autonomous DTN. Along with Chapters 3 and 4, which presented and evaluated a key distribution scheme, the addition of key revocation and replacement completes the DTN key management lifecycle. The proposed scheme specifically addresses the provision of public key authentication during an unplanned key revocation event, and by extension can be applied to a planned revocation event. By utilising a distributed signing feature to provide public key authentication through public key confidence, autonomous nodes can be confident in the old key being revoked, while also accepting the new public key. This feature also provides trust transferral between the old key and the new key, thereby providing one solution to the issue of decentralised unplanned key revocations. The main contributions of this chapter are:

- The investigation, development, and evaluation of a Distributed Signing key revocation and replacement scheme. The scheme presented provides public key authentication for both the old revoked key and new public key during an unplanned revocation event without the requirement of a TTP. Some sub-contributions include:
  - The mitigation of effects an adversary or adversaries performing a Sybil attack has on the DTN.
  - The mitigation and prevention of an adversary or adversaries triggering a false revocation event.
- The provision of trust transferral between the old revoked public key and the new public key during an unplanned key revocation event without the dependence on a TTP.

The proposed key revocation and replacement scheme consistently outperformed the other schemes in new public key distribution for scenarios with adversarial agents. In the single attack scenarios, the DS scheme provided a White New Public Key distribution in the range of 52-77% of nodes, compared with RO and RR, which achieved 30-44% and 45-68% of nodes respectively. In the multiplying attack, the strong adversary model resulted in more varied outcomes. The DS revocation scheme still provided the best White New Public Key distribution result when compared to RO and RR schemes.

The DS revocation scheme proposed also significantly mitigated the distribution of Spoofed Revocation Keys when compared to the other scenarios. For all scenarios in the single attack, the percentage of nodes holding a Spoofed Revocation Key never exceeded 20% for the DS scheme, while the other schemes resulted in 2 to 4 times the percentage of Spoofed Revocation Keys. More varied results were obtained when the network was subjected to a multiplying adversary attack. Even with Black Hat node populations at 1%, the RO and RR schemes resulted in over 70% of nodes with a Spoofed Revocation Key. In comparison, the DS revocation scheme provided significantly less Spoofed Revocation Keys in situations with less than 20% adversarial node population.

Evidence of trust transferral of keys was obtained by the Revocation Certificate metric, in particular the White Hat Certificate Distribution. This metric showed a high adoption of signed revocation certificates for the DS scheme when compared to the RO and RR scenarios.

These results signify the provision of public key authentication during key revocation and replacement events in autonomous DTNs. In particular, during unplanned revocation events, such as the compromise of a private key. Further potential applications of the DS revocation scheme include Multiple-Entity Factor Authentication, where trusted friends are attest the identity of a locked out user.

The limitations of this work is acknowledged in that area of data routing. It is assumed that incomplete data transfers are dropped. The modelling of a truly asymmetric data bundle exchange would affect the key revocation results. A further restriction on how the new public key is distributed could be an area of further work and experimentation. The future work, which aims to provide additional security are discussed in more detail in Chapter 6, including the potential performance ramifications of the future work.

Chapter 6

Conclusion

Public key authentication, which is the ability for an entity to verify the identity-public key binding of another entity, is critical to achieving many overarching security properties such as confidentiality, data integrity, and message authentication. Without it, an adversary is capable of performing Man In The Middle (MITM) and Sybil attacks, thereby allowing it to eavesdrop on sensitive data, and compromise safety critical messages. Therefore, the ability to provide this service in any network is an important area of research.

The issues of providing public key authentication during all stages of key management in an autonomous Delay Tolerant Network (DTN) is further complicated by the inherent characteristics that define a DTN. The frequent disconnected state of the network means a reliable source to destination path is not always available. This hinders public key distribution and public key revocation operations as nodes rely on store-carry-and-forward style exchanges. The decentralised and distributed nature of DTNs means that traditional centralised Trusted Third Party (TTP) topologies are unsuitable for providing public key authentication. Autonomous DTNs also have no human intervention to provide trust and key management duties. Many past proposals that attempted to address this critical issue claimed to be distributed, but were hierarchic, as they depended on a TTP prior to deployment or cluster structures. This thesis addresses the challenges outlined by providing a fully decentralised and distributed solution. It provides a number of related contributions described in the section following.

The rest of this chapter is organised as follows. Section 6.1 summarises the

contributions presented in this thesis. Section 6.2 outlines the limitations encountered and the potential future directions for this research. Section 6.3 presents the concluding remarks.

6.1 Summary of Contributions

The challenges of providing public key authentication during all phases of public key management in an autonomous DTN have been addressed. The contributions to the research questions identified in Chapter 1 are summarised.

The first contribution of this research addressed Research Question 1: *Can a trust or reputation system be utilised to assist in DTN Key Distribution such that Public Key Authentication can be achieved without a trusted third party, but by automatically including mobility parameters, behaviour, and levels of collaboration into trust?* The contribution has been the design of a trust system to provide public key authentication during key distribution in a DTN. The trust system utilised a common and effective linear computation engine that exploits the social contacts to establish initial trust between two nodes meeting. The proof-of-concept trust system called Leverage of Common Friends (LCF) was modelled and evaluated in a controlled closed simulation consisting of a small number of nodes. The LCF trust system was evaluated by introducing an adversary performing a key spoof attack, attempting to exploit the identity-public key binding. It was also compared to two other trust establishment methodologies. The first, where trust establishment was randomly generated, and second, where trust was absolute. The experimental data indicated that the LCF trust system provided mitigation of adversary public keys and spoofed identity keys by 40%, when compared to the other two methodologies. The design and implementation of the LCF trust system addresses the issue of public key authentication between autonomous nodes during the key distribution phase. In particular, this addressed a DTN deployment environment where there is no infrastructure. To the best of our knowledge, this contribution is the first to combine an initial trust establishment system with a key distribution model to provide public key authentication for autonomous DTN nodes.

The second contribution of this research addressed Research Question 2: *Is it possible to apply a trust or reputation system for DTN Key Distribution for a large scale realistic DTN application?* The contribution has been to extend

the deployment of the LCF trust system to a realistic large scale geographic environment. The scalability of the LCF trust system was tested by modelling and evaluating public key distribution using vehicular movement models from San Francisco taxis over a 48 hour period. The movement model consisted of nearly 500 vehicular nodes in a simulation space of 400 square kilometres. The experimental data indicated that the LCF trust system was scalable at these levels and provided mitigation of adversary public keys and spoofed identity keys. However, scalability in larger orders of magnitude were not addressed, due to limitation on realistic mobility models. This contribution simulates and evaluates a combined initial trust establishment system and key distribution model to provide public key authentication at a realistic scale.

The third contribution addressed Research Question 3: *Is it possible to leverage location data to assist a trust or reputation system for DTN Key Distribution?* This contribution utilised the realistic large scale geographic environment of San Francisco taxis to leverage location to assist the LCF trust system in key distribution. The proposed Location based Leverage of Common Friends (LLCF) trust system was simulated and evaluated with the LCF trust system and absolute trust system. The effectiveness of the LLCF trust system was determined by introducing varying quantities of adversaries performing a key spoof attack. Two variations of adversaries were introduced. The first was a dynamic mobility adversary, and the second was a stationary adversary. The experimental results indicated that the addition of co-localisation data in the LLCF trust system provided better public key authentication than the LCF trust system. Trust values of legitimate keys were consistently higher in LLCF over LCF, while trust values of adversarial and spoofed keys were consistently lower in LLCF over LCF. Spoofed ID Keys were also reduced by up to an additional 50% using the LLCF trust system over LCF. However, scalability in larger orders of magnitude were not addressed, due to limitation on realistic mobility models. This contribution simulates and evaluates a combined initial trust establishment system with co-localisation data, and key distribution model to provide public key authentication at a realistic scale.

The fourth contribution designed a key revocation and replacement scheme that provided public key authentication to prevent false and erroneous revocation by an adversary. This addresses Research Question 4: *Is it possible to utilise a trust or reputation system to assist in DTN Key Revocation such that*

Public Key Authentication can be achieved without a trusted third party? The proposed scheme utilised a distributed signing scheme by social contacts for a node to self-revoke. The key revocation scheme is autonomous, decentralised, and self-organising. It provides a timely removal of revoked keys by distributing a delta Certificate Revocation List (CRL), while distributing replacement keys to re-establish an end-to-end communications channel. The effectiveness of the proposed revocation scheme was determined by introducing varying quantities of adversarial nodes performing a Sybil attack. Two variations of adversaries were introduced. The first was a singular adversary, and the second was a multiplying adversary. The experimental results indicated that the proposed key revocation and replacement scheme provided a trade-off between preventing a false and erroneous revocation by an adversary, and a delay in authentication. This contribution is the first to propose a fully distributed key revocation and replacement scheme where there is no TTP or human intervention.

The fifth contribution was the addressing of trust transferral between public keys during an unplanned revocation event, which was the premise of Research Question 5: *Is it possible to provide trust transferral of an old compromised public key to a newly generated public key without a trusted third party during an unplanned key revocation event?* Revocation of public keys may be *planned*, due to a finite lifetime of keys, and/or organisational and environmental policy. Trust transferral between the old public key, and the new public key can be transferred through various planned and organised methods (see Section 2.3 for details). However, in an *unplanned* scenario, public keys are revoked due to private key compromise, and there is no trust transferral between old and new keys. The distributed signing revocation scheme addresses the trust transferral issue by leveraging the trust of signatories. The experimental results indicated a high adoption of the signed revocation certificate and subsequent new public key by legitimate nodes when compared with other revocation schemes with no or complete trust transferral. However, trust systems are not perfect, resulting in some specific node instances where trust transferral did not occur. This contribution proposed a distributed key revocation and replacement scheme where trust transferral occurs between the old revoked key and the new public key during an unplanned revocation event.

As a result of these contributions, an additional minor contribution has been the design and development of a fully customisable DTN simulator in Python

named *Traffic Djam* [30]. This simulator was designed in particular to address the issue of simulating a trust and reputation system assisted public key exchange, as well as key revocation events. The simulator has the capability of generating random movement models, as well as using pre-generated models.

6.2 Limitations and Future Directions

A number of limitations were identified during this research. These limitations provide potential directions for future research and are presented in the following sections.

6.2.1 Trust Weighting Variation

The selection criteria for trust weightings (t_n, t_c, and t_d) for both the LCF and LLCF trust system were outlined in Section 3.4. The criteria for t_n was adopted from the cognitive limit of human stable social relationships and correlated with additional research into online social relationships. This provided a logical and sound starting assumption for the selection of t_n. The criteria for t_c was adopted from initial experiments to satisfy the OpenPGP implementations of GNU Privacy Guard (GPG) and Pretty Good Privacy (PGP) 2.6. Although these assumptions and criteria provide a logical starting place for the trust system, additional experiments could be undertaken to determine how the variation of these trust weightings affect the LCF and LLCF trust systems.

The trust weighting selections used reflect established PGP criteria. However, these weighting selections can be adjusted to reflect the deployment of the autonomous nodes for the environmental conditions, since the level of trust required reflects the risk in the network. Long term deployment environments with a high security requirement may adjust the trust weightings to reflect the necessity for more trust to be established before trusting nodes and keys. The higher trust reflects the increased risk for such a deployment. The increase in trust satisfies the higher security requirement of the deployment environment at the cost of key distribution performance as a greater number of instances of keys are required to meet the increased trust. Conversely, in a low security requirement deployment, a lower trust may be used to reflect the lower risk environment. Differing trust weightings could also be applied to nodes with different roles in the network. A long term deployment node could have a higher trust requirement, signifying an

increased risk, and compromise in key distribution and time. In comparison, an expendable node that is deployed for a short duration may have a lower trust requirement, signifying a lower risk, whilst providing an improved key distribution and time. Since risk is dependant on the application of the network, future work could include a risk factor into the trust system.

6.2.2 Movement Model Variation

The realistic movement model used to evaluate the trust systems proposed in Chapter 4 is one of many different traffic movement models available. Investigating the use of a different movement model is a potential future research direction. Vehicle movement models that include private cars, buses provide different movements compared to taxi cabs. Private cars would show specific source to destination routes for vehicles that do not congregate around areas such as airports and tourist areas, and buses would provide a periodic and set route of vehicle movements. The inclusion of pedestrian movement as well as vehicular movement in the same model would provide an interesting overlap between Mobile Ad-Hoc Networks (MANETs) and Vehicle Ad-Hoc Networks (VANETs). Additionally, varying the starting locations of Black Hat nodes is a potential future research direction, as the limitation of the movement model restricts their initial placement.

6.2.3 Inclusion of Time and Freshness of Data

Although the addition of co-localisation data to the LCF trust system to form the LLCF trust system provided an improvement in key distribution and mitigation of spoofed ID keys, the inclusion of time and freshness of data may further improve the initial trust establishment scheme. The addition of time and freshness of data may also provide a mechanism to discard and remove under utilised public keys.

6.2.4 Incomplete Data Bundle Transfer

The experiments conducted in Chapters 4 and 5, all incomplete transfers of data bundles were dumped. Modelling a purely asymmetric data bundle exchange between nodes would significantly affect key distribution and subsequent key revocation.

6.2.5 Distribution of New Public Key via Revocation Certificate Only

In Chapter 5, the distribution of the new public key post revocation could be further restricted to the revocation certificate. Theoretically, this would provide additional security in preventing spoofed ID keys from being distributed. However, this may cause issues in open and dynamic networks where nodes can join at any time. Newer nodes who have just joined the network may have difficulty in distributing replacement public keys. The effects of this provide an interesting future experiment.

6.2.6 LCF Trust and DS Revocation Schemes for CA Management

The LCF trust scheme proposed and evaluated in Chapter 3 has potential application in Certificate Authority (CA) management. Trusting the large number of Internet CAs is difficult, and the application of the LCF trust system to provide trust and indicators on the performance and security of a CA has potential in a distributed regulation of CAs. Combinations of CAs regulating other CAs, end user regulation of CA, as well as a combination of both CA and end user regulation could provide an effective method in regulating and mitigating the issuance of false certificates by rogue or compromised CAs. Furthermore, the Distributed Signing (DS) revocation scheme proposed and evaluated in Chapter 5 can be applied to provide Root CA certificate revocation in the Internet CA model. Currently the revoked Root CA certificate has to be manually removed, while the DS revocation scheme has the potential to provide a more automated revocation process.

6.3 Concluding Remarks

The provision of public key authentication for all phases of key management in a decentralised, distributed autonomous DTN has resulted in the development and evaluation of a combined trust system and key distribution scheme. In addition, the development of a combined co-localisation trust system and key distribution scheme evaluated on a realistic large geographic scale mobility model. The thesis also addresses the problem of unplanned key revocation and replacement in an

autonomous DTN without any centralised CA or TTP. Given how foundational public key authentication is to the general security properties of a DTN, it is important to provide confidence in the identity-public key binding during all phases of key management. The contribution of this thesis has been to provide public key authentication for a resource challenged network such as a DTN. Further work that extends the contribution and evaluation of the work presented in this thesis can be applied to more broad applications and varieties of DTNs.

Appendix A

LCF Key Distribution over Time Results

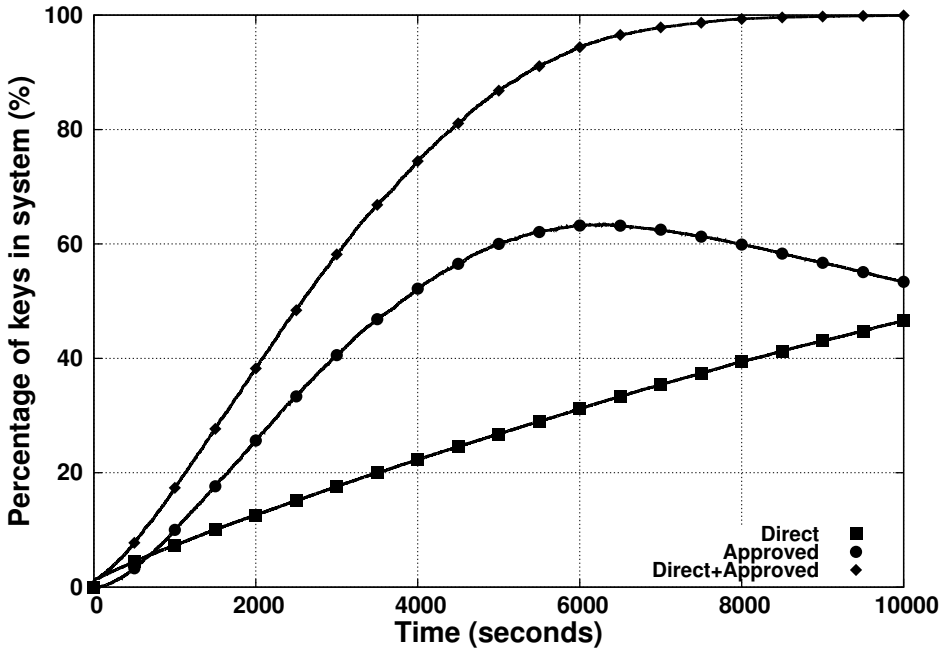


Figure A.1: Direct and Approved key distribution over time for Control Scenario.

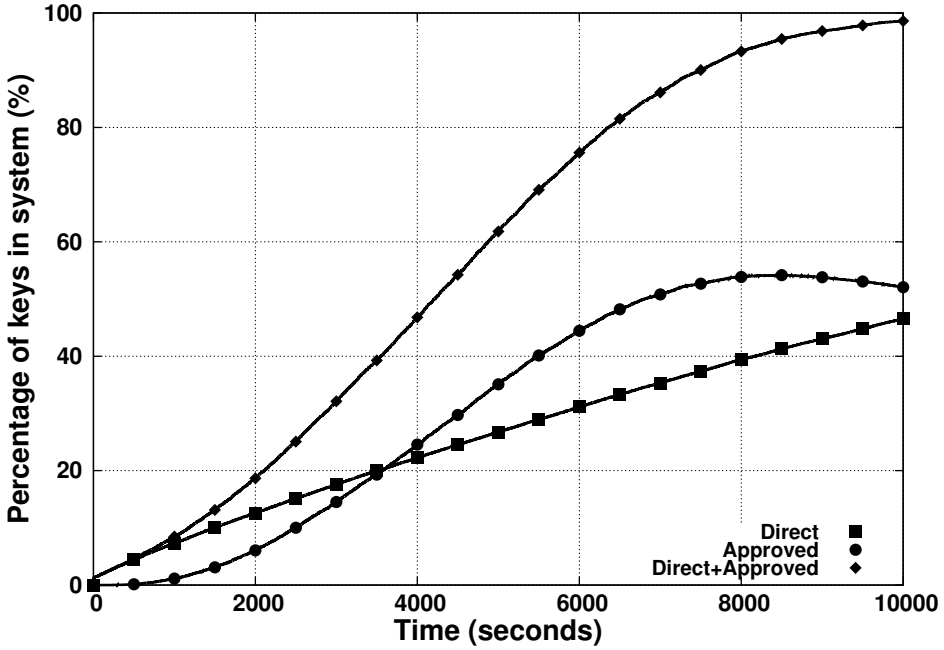


Figure A.2: Direct and Approved key distribution over time for Random Scenario.

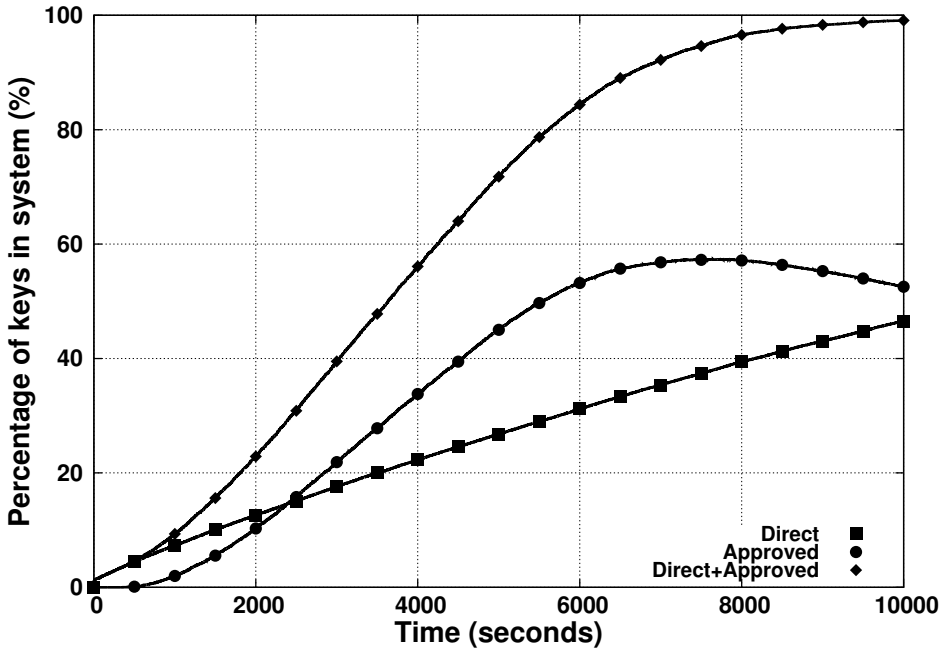


Figure A.3: Direct and Approved key distribution over time for LCF Scenario.

Appendix B

LCF Key Distribution Results

Table B.1: Experiment 1 Results

	Key List	Control	Random			LCF
			1	2	3	
Black Hat Node with Black Hat Keys	Direct	42	42	42	42	42
	Approved	58	58	58	58	3
White Hat Nodes with Black Hat Keys	Direct	40	40	40	40	40
	Approved	50	51	54	50	53
Spoofed ID Keys	Approved	30	16	25	29	2
Totals	Direct	82	82	82	82	82
	Approved	138	125	137	137	58
	All	220	207	219	219	140

Table B.2: Experiment 2 Results

	Key List	Control	Random			LCF
			1	2	3	
Black Hat Node with Black Hat Keys	Direct	48	48	48	48	48
	Approved	52	52	52	52	0
White Hat Nodes with Black Hat Keys	Direct	44	44	44	44	44
	Approved	32	47	41	43	39
Spoofed ID Keys	Approved	60	59	53	44	8
Totals	Direct	92	92	92	92	92
	Approved	144	158	146	139	47
	All	236	250	238	231	139

Table B.3: Experiment 3 Results

	Key List	Control	Random			LCF
			1	2	3	
Black Hat Node with Black Hat Keys	Direct	53	53	53	53	53
	Approved	47	47	47	47	4
White Hat Nodes with Black Hat Keys	Direct	50	50	50	50	50
	Approved	32	42	41	44	44
Spoofed ID Keys	Approved	36	79	86	81	6
Totals	Direct	103	103	103	103	103
	Approved	115	168	174	172	54
	All	218	271	277	275	157

Table B.4: Experiment 4 Results

	Key List	Control	Random			LCF
			1	2	3	
Black Hat Node with Black Hat Keys	Direct	50	50	50	50	50
	Approved	50	50	50	50	9
White Hat Nodes with Black Hat Keys	Direct	48	48	48	48	48
	Approved	50	48	49	46	49
Spoofed ID Keys	Approved	126	82	77	78	9
Totals	Direct	98	98	98	98	98
	Approved	226	180	176	174	67
	All	324	278	274	272	165

Table B.5: Experiment 5 Results

	Key List	Control	Random			LCF
			1	2	3	
Black Hat Node with Black Hat Keys	Direct	43	43	43	43	43
	Approved	57	57	57	57	4
White Hat Nodes with Black Hat Keys	Direct	41	41	41	41	41
	Approved	49	49	46	51	50
Spoofed ID Keys	Approved	34	62	59	68	10
Totals	Direct	84	84	84	84	84
	Approved	140	168	162	176	64
	All	224	252	246	260	148

Table B.6: Experiment 6 Results

	Key List	**Control**	**Random**			**LCF**
			1	**2**	**3**	
Black Hat Node	Direct	42	42	42	42	42
with Black Hat Keys	Approved	58	56	56	56	3
White Hat Nodes	Direct	39	39	39	39	39
with Black Hat Keys	Approved	51	53	52	53	54
Spoofed ID Keys	Approved	18	25	16	19	1
Totals	Direct	81	81	81	81	81
	Approved	127	134	124	128	58
	All	208	215	205	209	139

Appendix C

LLCF White/Black Key Results

Table C.1: Static Black Hat Node Raw Experiment Results

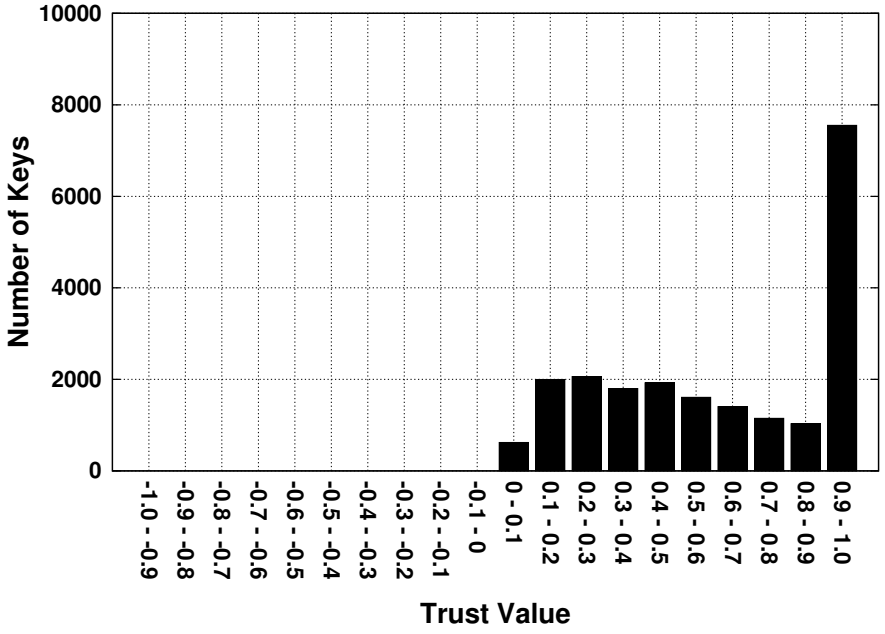
Black Hat Nodes	Scenario	Direct		Approved	
		White	Black	White	Black
0	Control	21147	0	104603	0
	LCF	21147	0	70363	0
	LLCF	21147	0	71943	0
1%	Control	20688	783	101658	3919
	LCF	20688	783	69432	837
	LLCF	20688	783	70634	810
10%	Control	17009	9576	78323	24908
	LCF	17009	9576	52715	9582
	LLCF	17009	9576	53604	8454
20%	Control	13396	22043	59095	40407
	LCF	13396	22043	37810	17122
	LLCF	13396	22043	38675	15049
30%	Control	10458	39191	43345	52768
	LCF	10458	39191	27110	22458
	LLCF	10458	39191	27794	20134

Table C.2: Dynamic Black Hat Node Raw Experiment Results

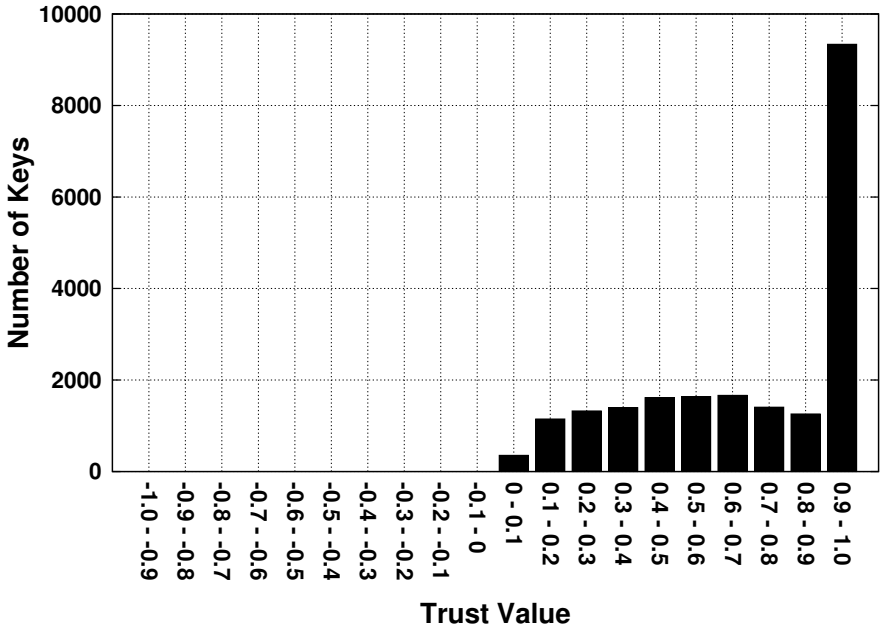
Black Hat Nodes	**Scenario**	**Direct**		**Approved**	
		White	**Black**	**White**	**Black**
0	Control	21147	0	104603	0
	LCF	21147	0	70363	0
	LLCF	21147	0	71943	0
1%	Control	20760	379	102539	2392
	LCF	20760	379	69688	668
	LLCF	20760	379	70630	663
10%	Control	17159	4062	75107	29458
	LCF	17159	4062	52949	9756
	LLCF	17159	4062	52058	8517
20%	Control	13446	7651	52215	52409
	LCF	13446	7651	38243	16829
	LLCF	13446	7651	36673	14217
30%	Control	10468	10585	35272	69471
	LCF	10468	10585	27603	21779
	LLCF	10468	10585	26222	17644

Appendix D

LLCF Key Trust Distribution Results

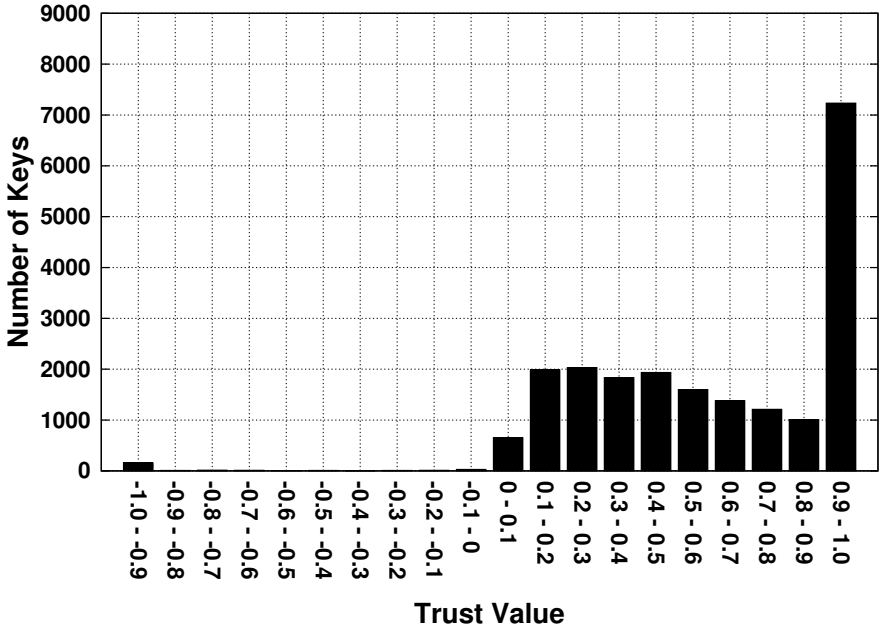


(a) LCF

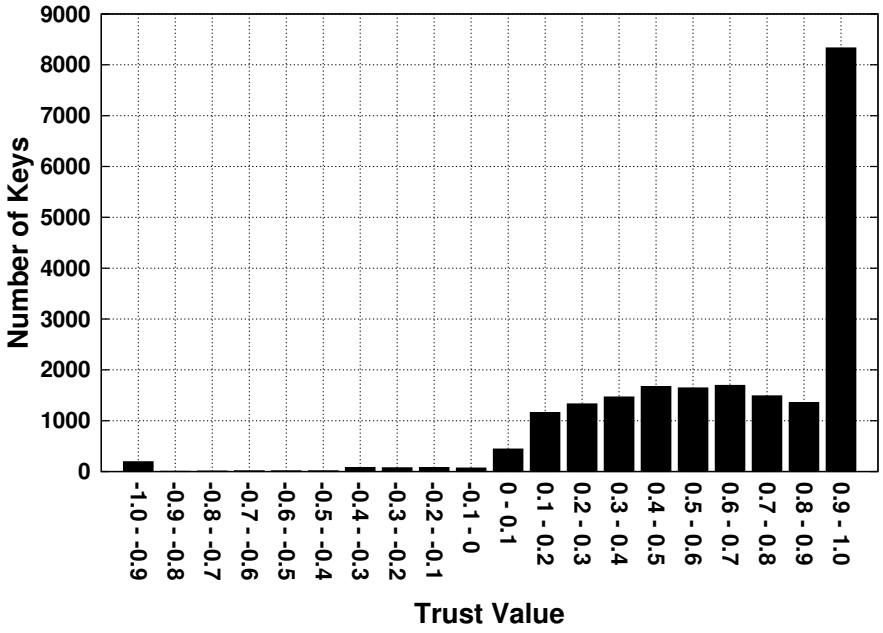


(b) LLCF

Figure D.1: Key Trust Distribution for 0% Black Hat Nodes.

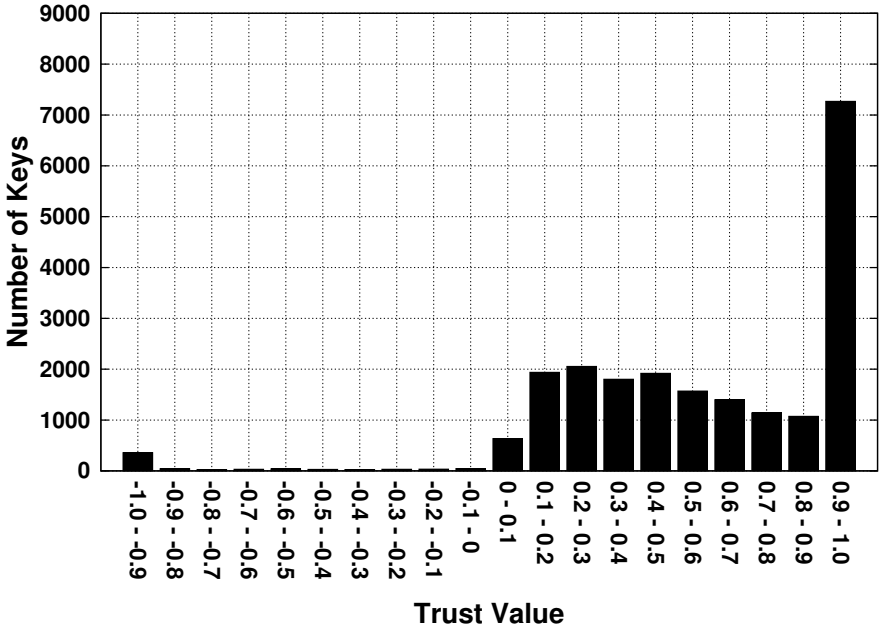


(a) LCF

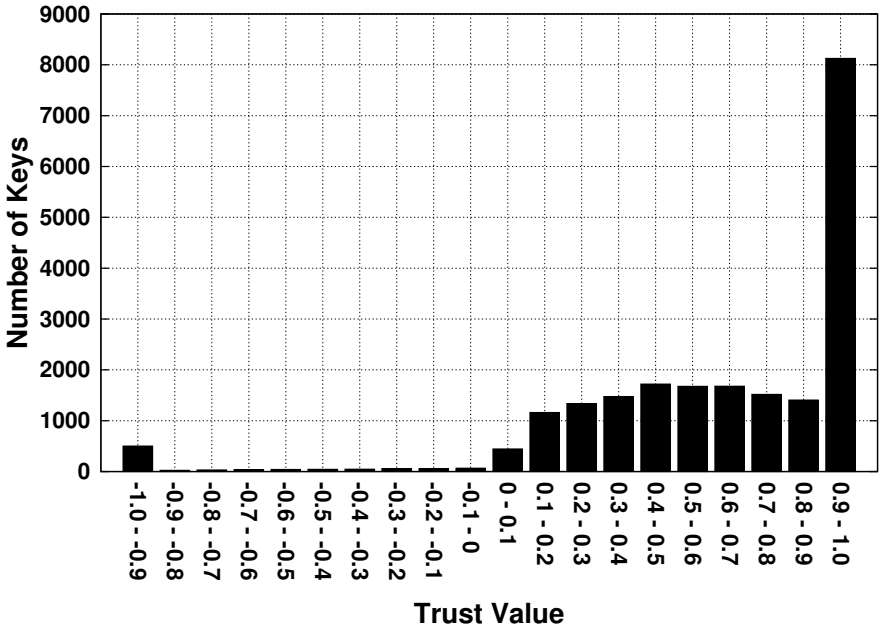


(b) LLCF

Figure D.2: Key Trust Distribution for 1% Dynamic Black Hat Nodes.

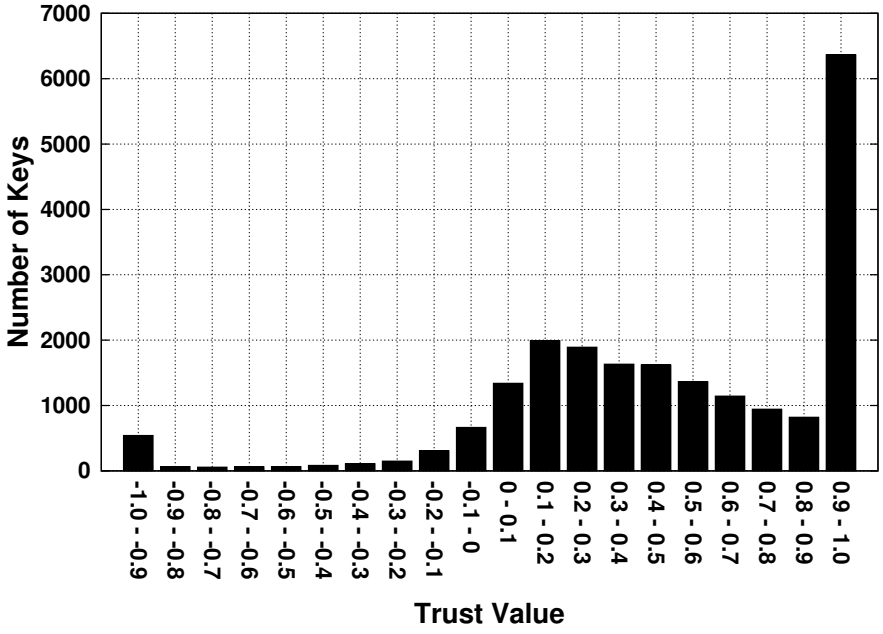


(a) LCF

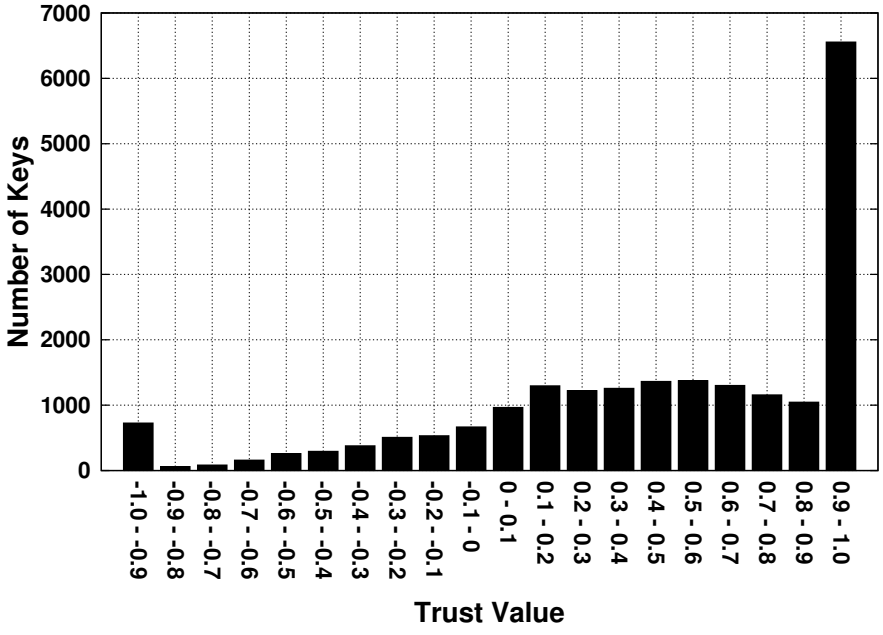


(b) LLCF

Figure D.3: Key Trust Distribution for 1% Static Black Hat Nodes.

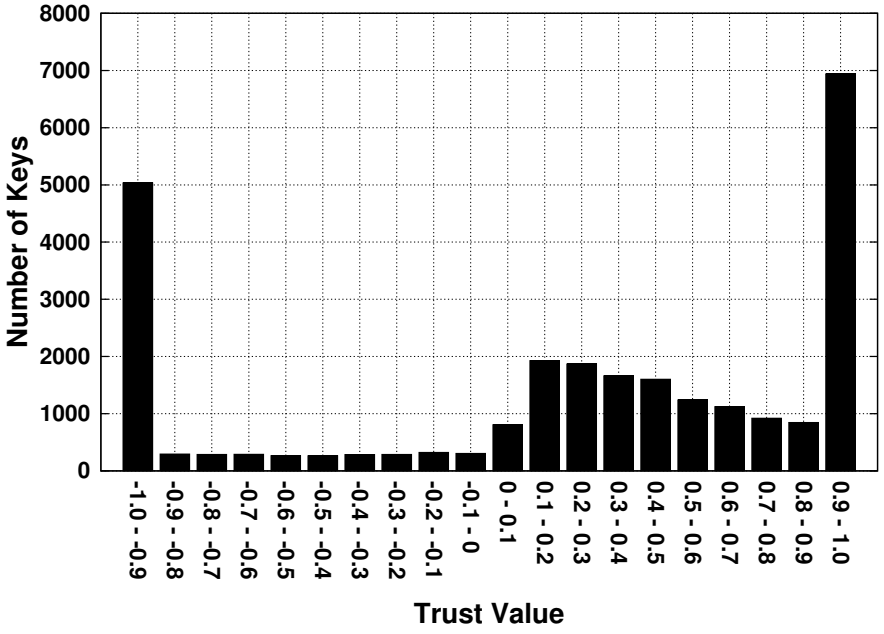


(a) LCF

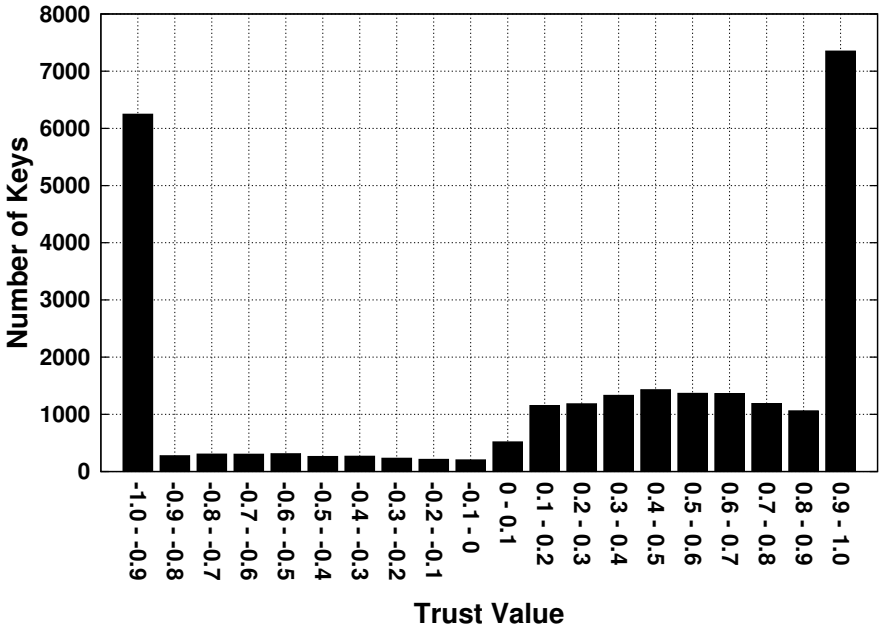


(b) LLCF

Figure D.4: Key Trust Distribution for 10% Dynamic Black Hat Nodes.

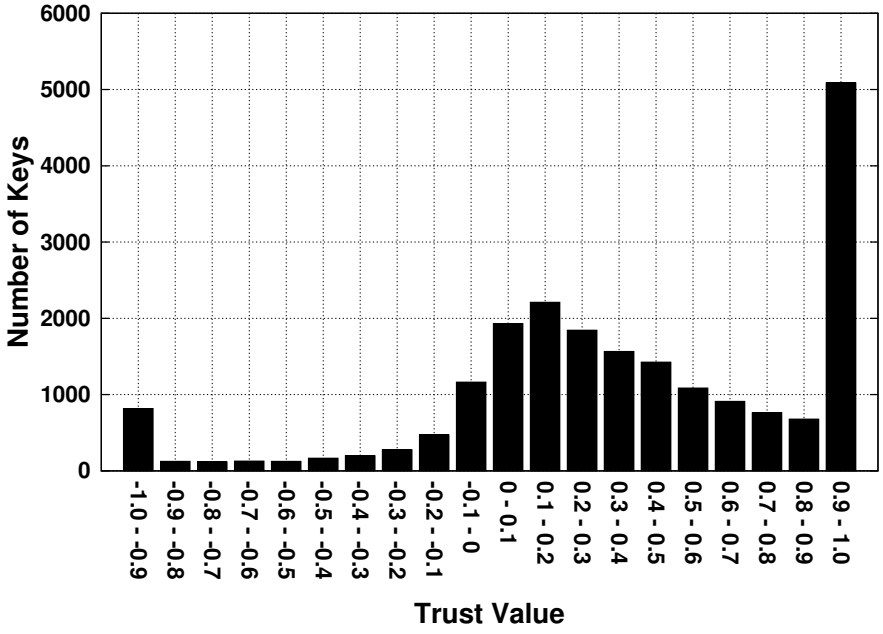


(a) LCF

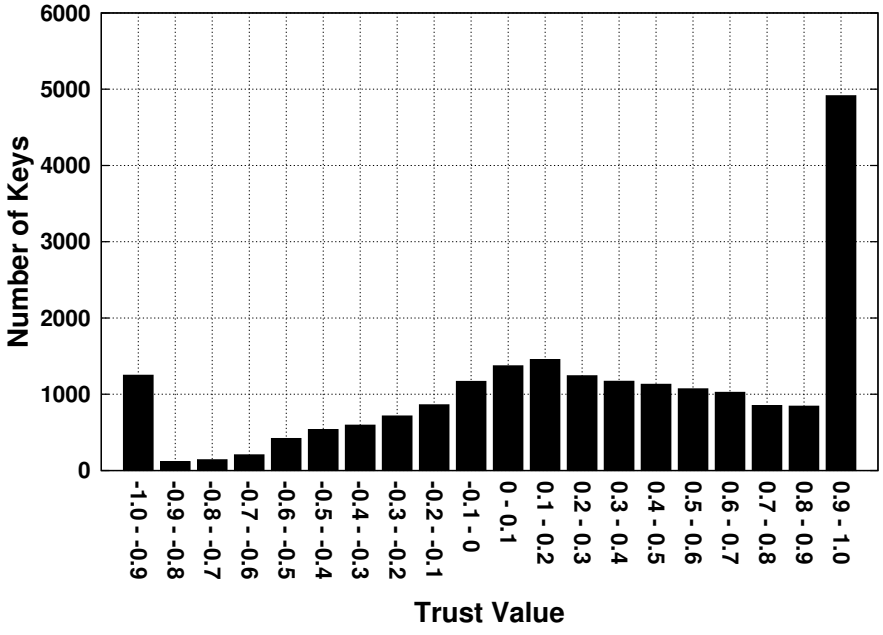


(b) LLCF

Figure D.5: Key Trust Distribution for 10% Static Black Hat Nodes.

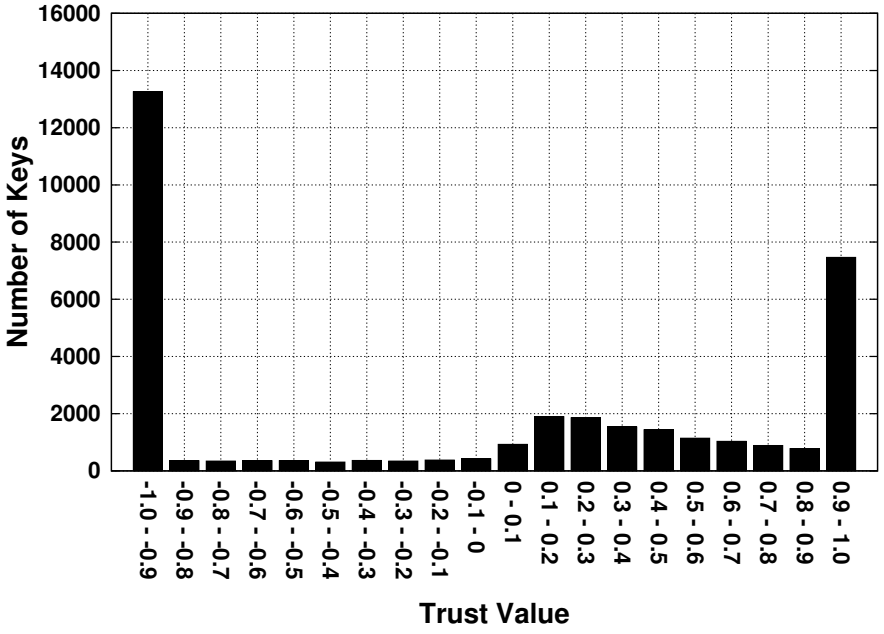


(a) LCF

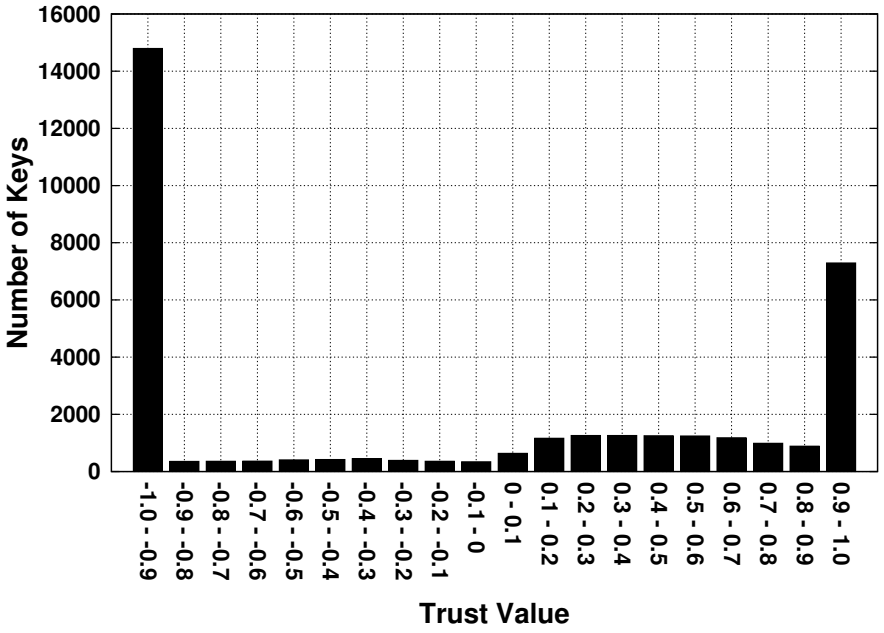


(b) LLCF

Figure D.6: Key Trust Distribution for 20% Dynamic Black Hat Nodes.

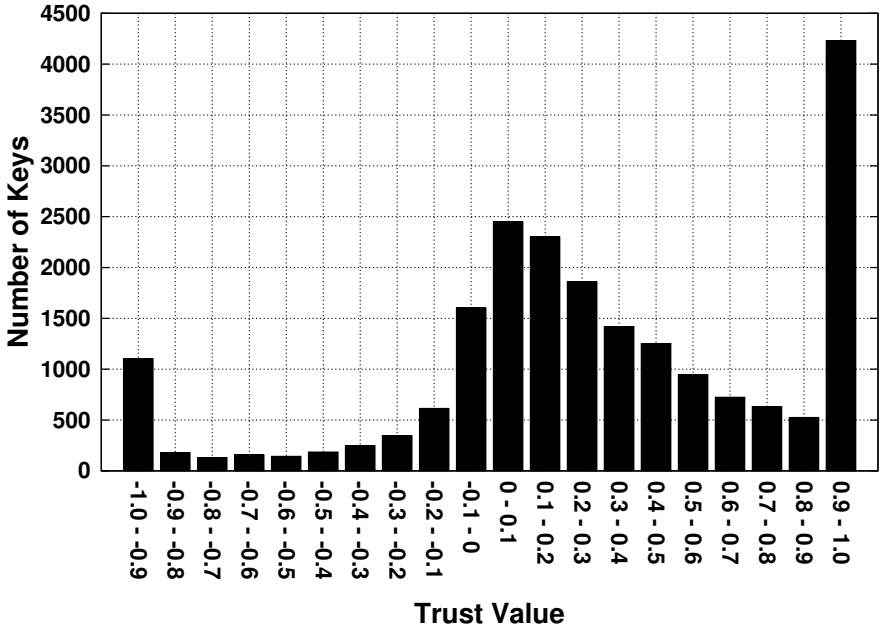


(a) LCF

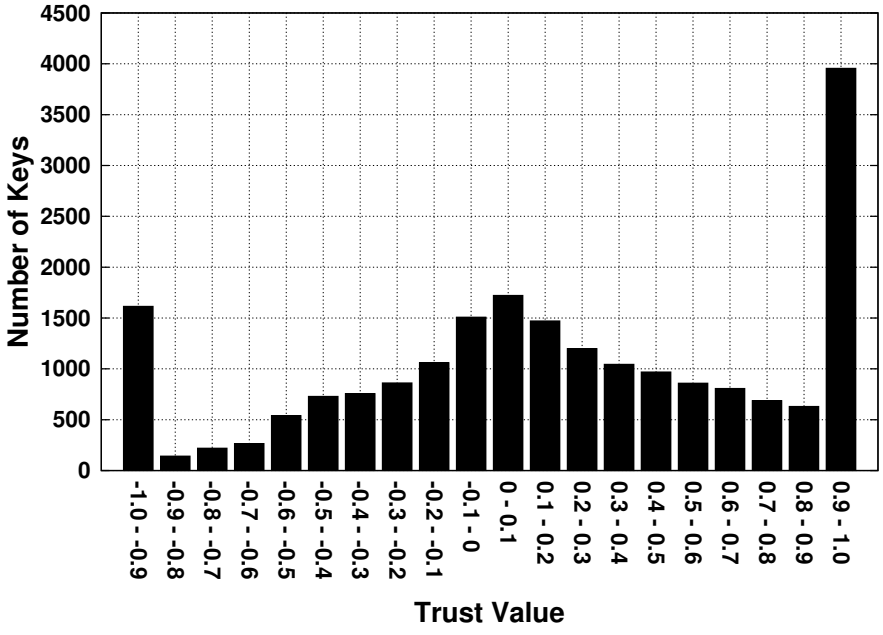


(b) LLCF

Figure D.7: Key Trust Distribution for 20% Static Black Hat Nodes.

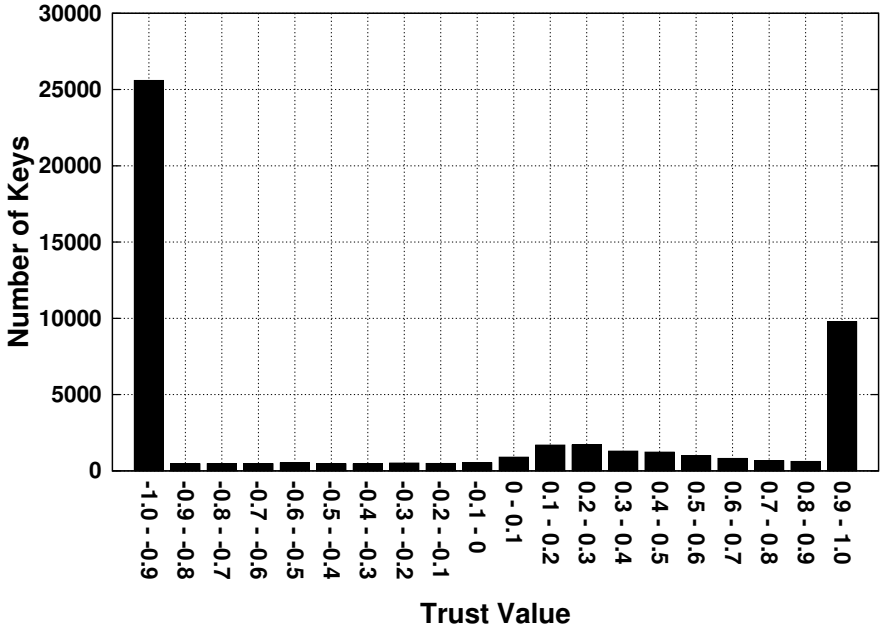


(a) LCF

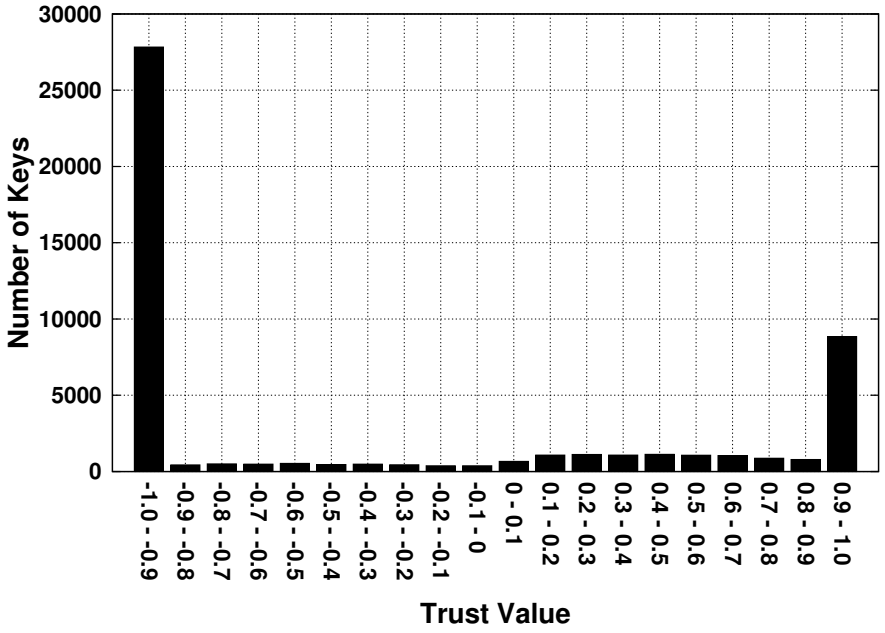


(b) LLCF

Figure D.8: Key Trust Distribution for 30% Dynamic Black Hat Nodes.



(a) LCF

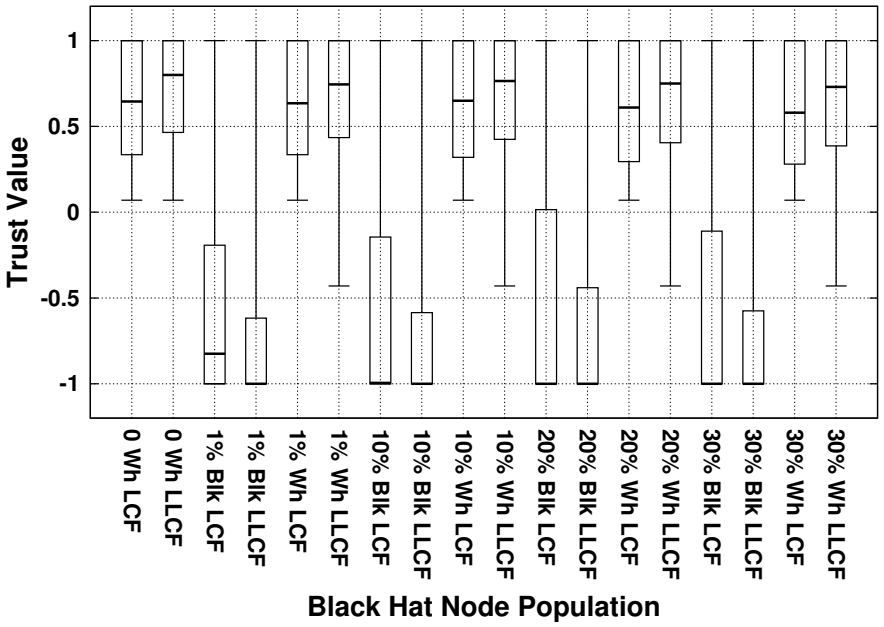


(b) LLCF

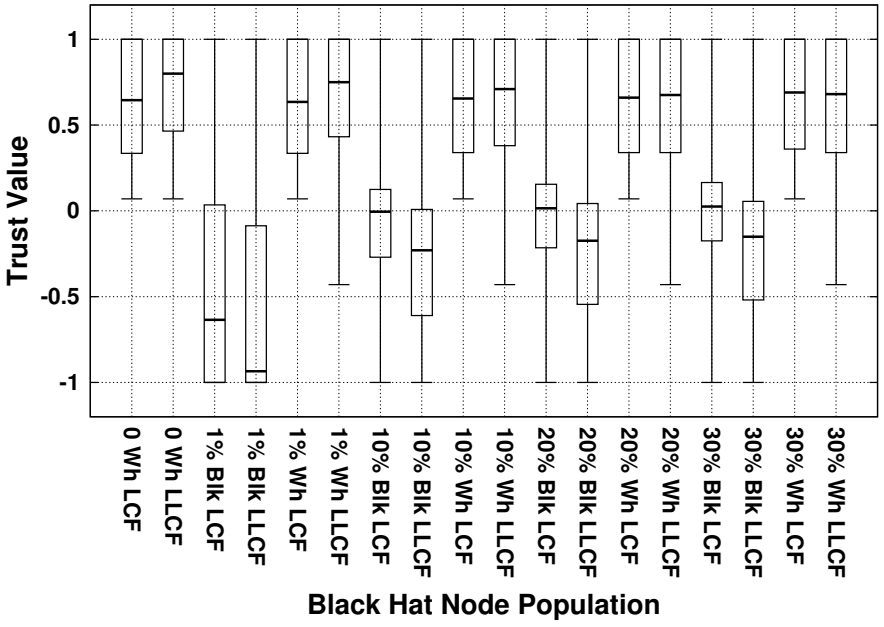
Figure D.9: Key Trust Distribution for 30% Static Black Hat Nodes.

Appendix E

LLCF Key Trust Five Number Summary Results



(a) Static Adversaries



(b) Dynamic Adversaries

Figure E.1: Key Trust Five Number Summary.

Appendix F

LLCF Key Distribution over Time Results

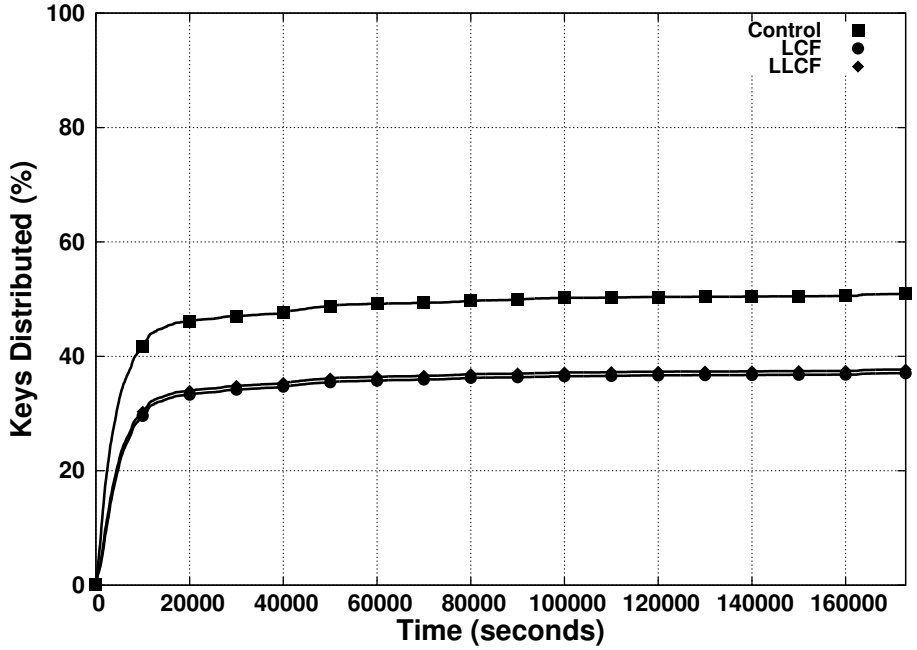
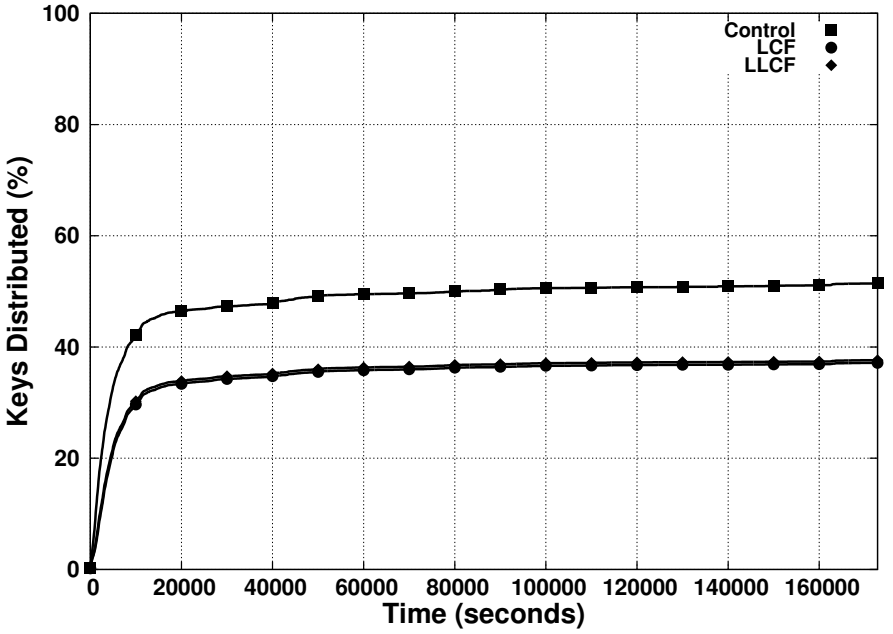
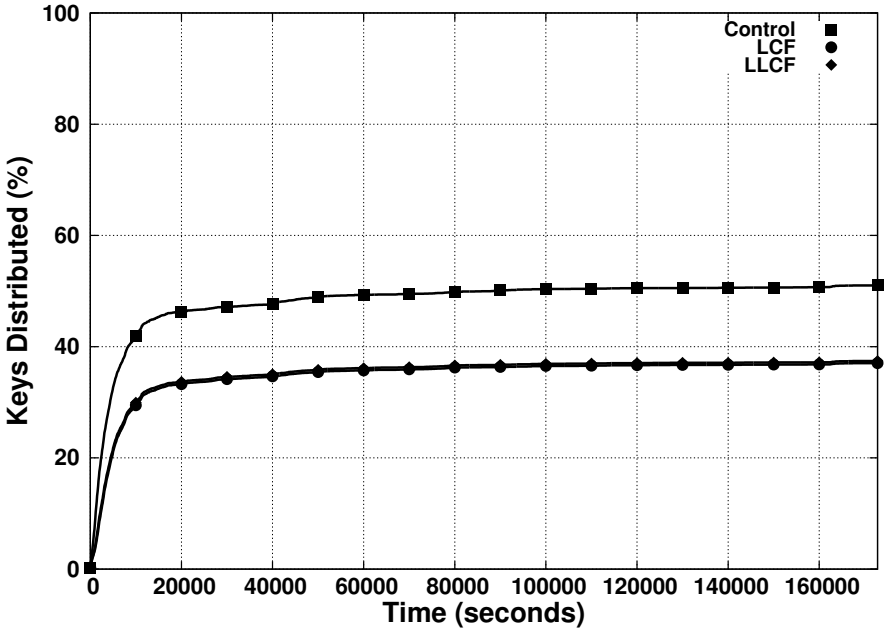


Figure F.1: Key Distribution over Time for no Black Hat Nodes.

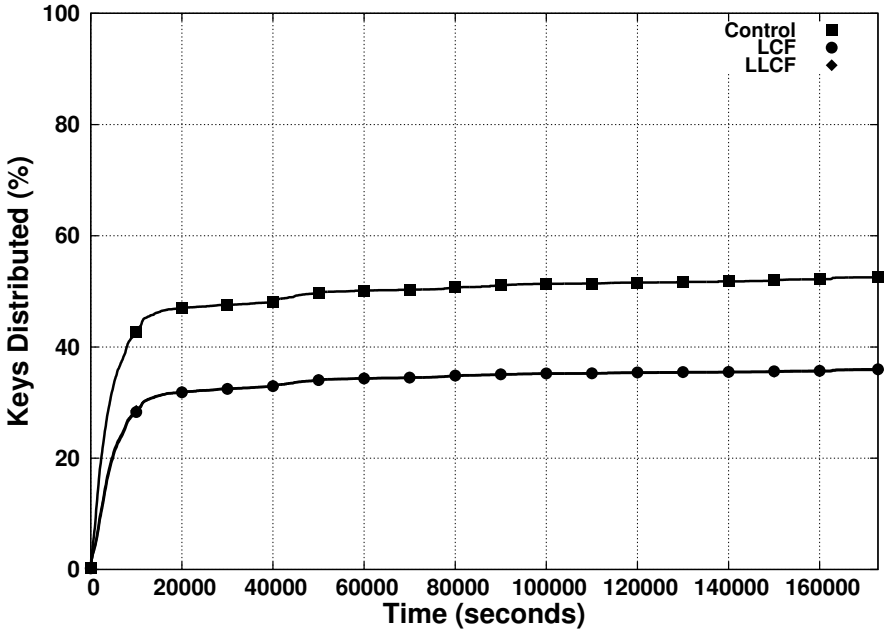


(a) Static Adversaries

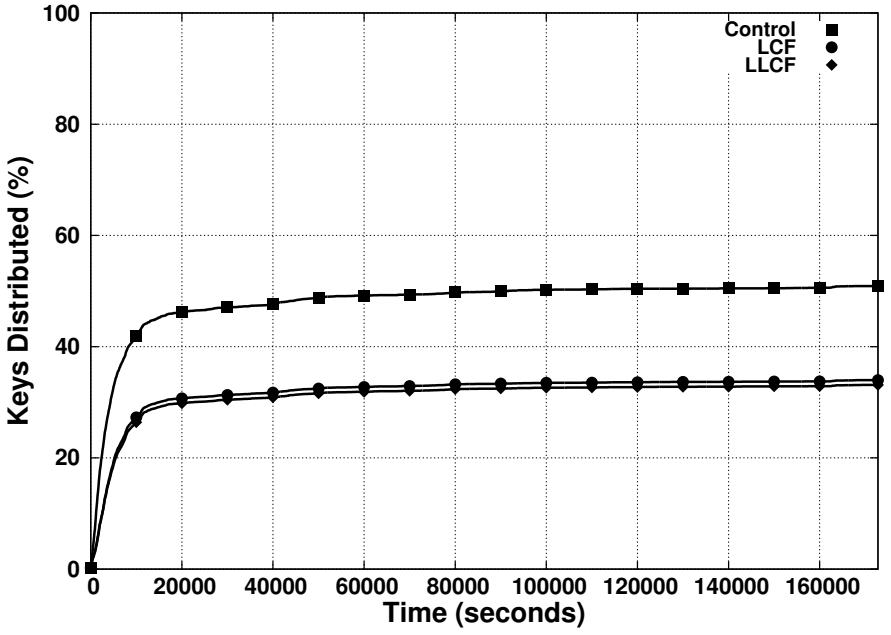


(b) Dynamic Adversaries

Figure F.2: Key Distribution over Time for 1% Black Hat Nodes.

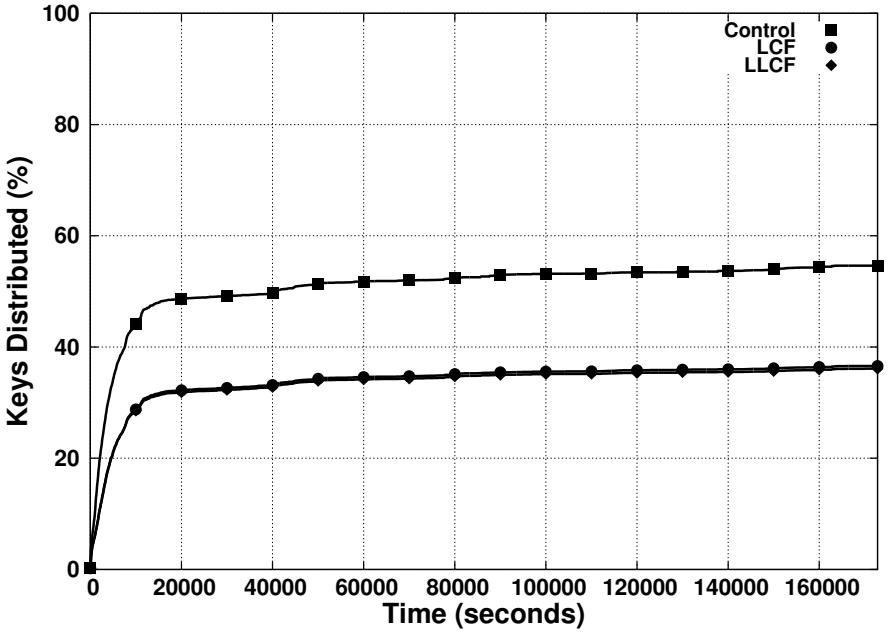


(a) Static Adversaries

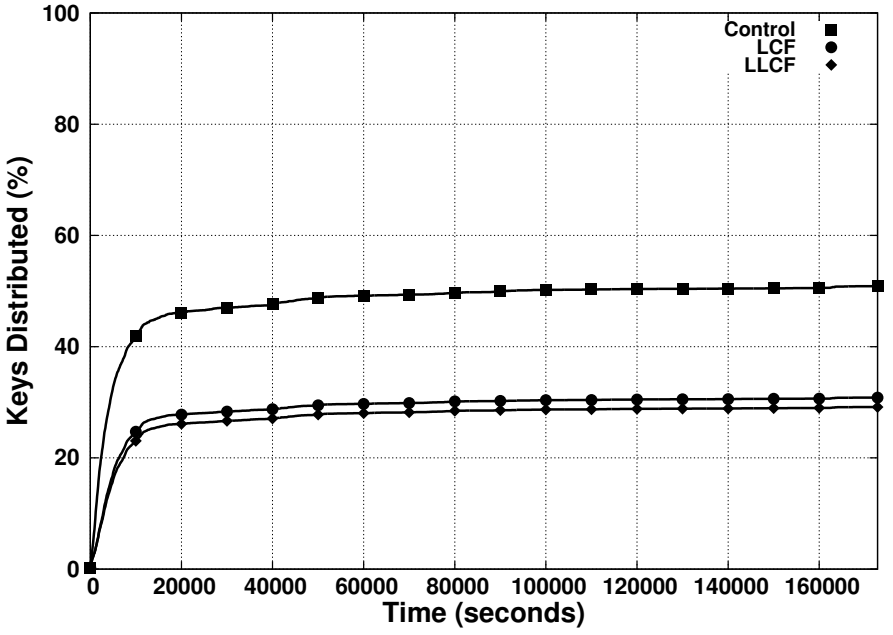


(b) Dynamic Adversaries

Figure F.3: Key Distribution over Time for 10% Black Hat Nodes.

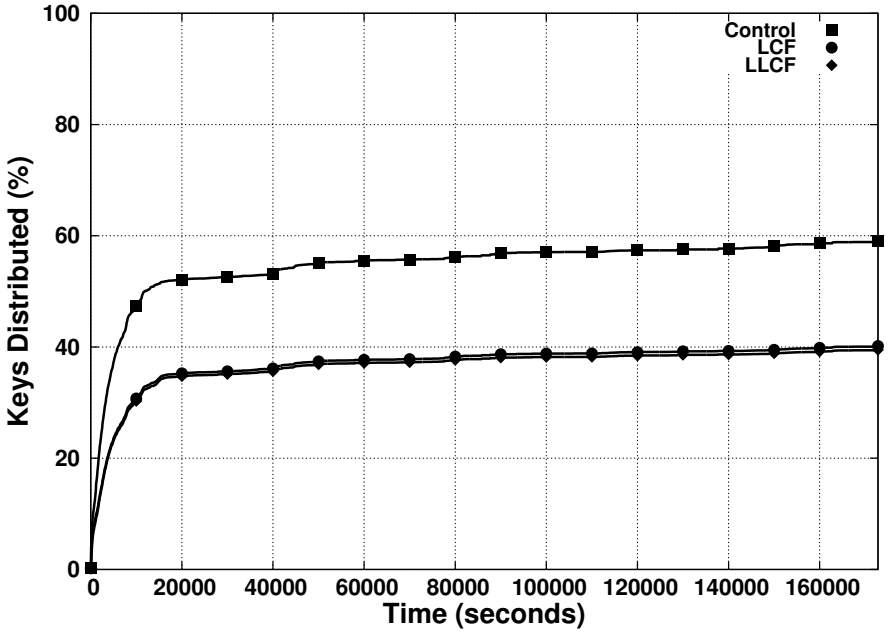


(a) Static Adversaries

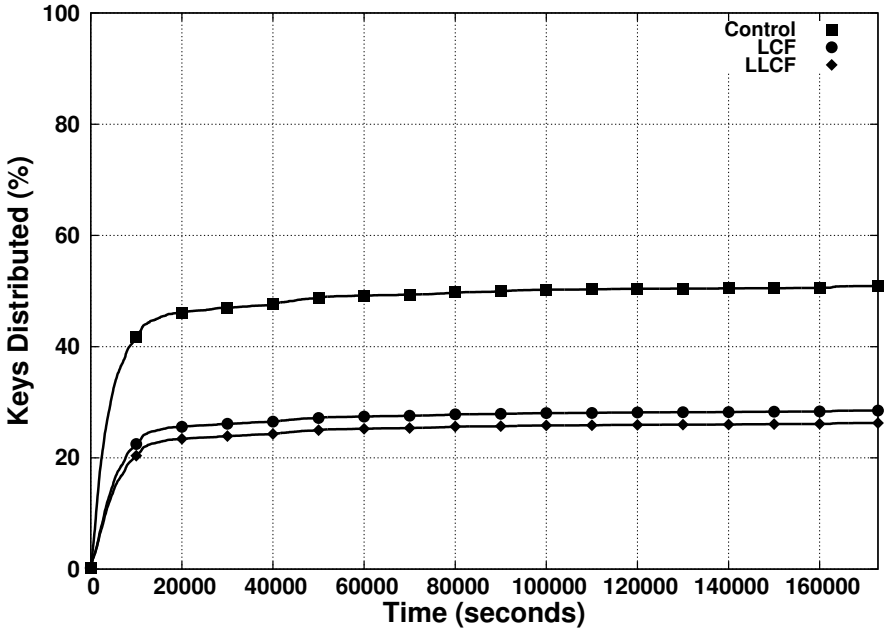


(b) Dynamic Adversaries

Figure F.4: Key Distribution over Time for 20% Black Hat Nodes.



(a) Static Adversaries



(b) Dynamic Adversaries

Figure F.5: Key Distribution over Time for 30% Black Hat Nodes.

Appendix G

Revoked Key Results

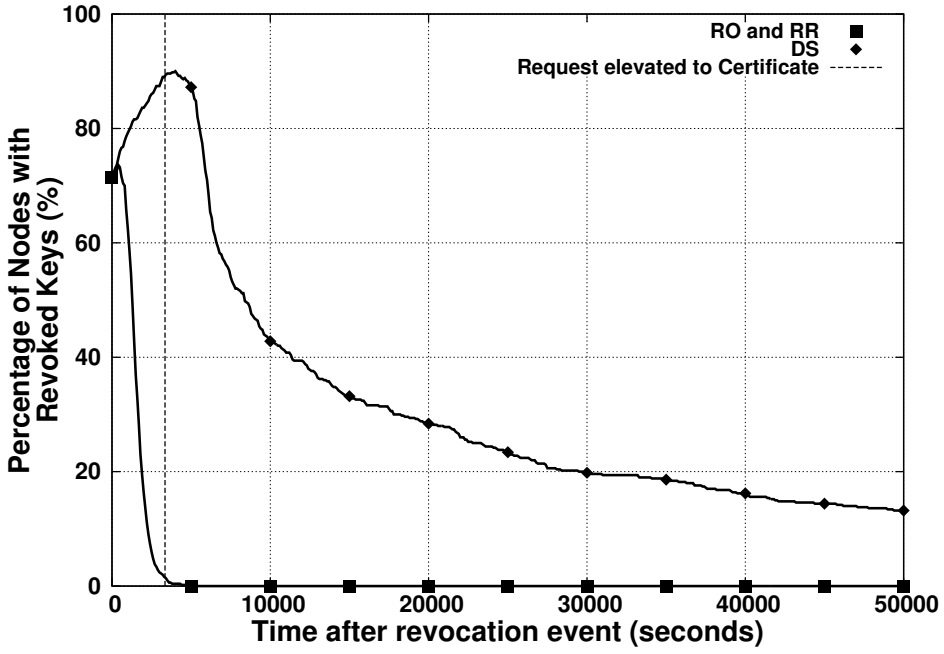


Figure G.1: Revoked Key Distribution over time with no Black Hat Nodes.

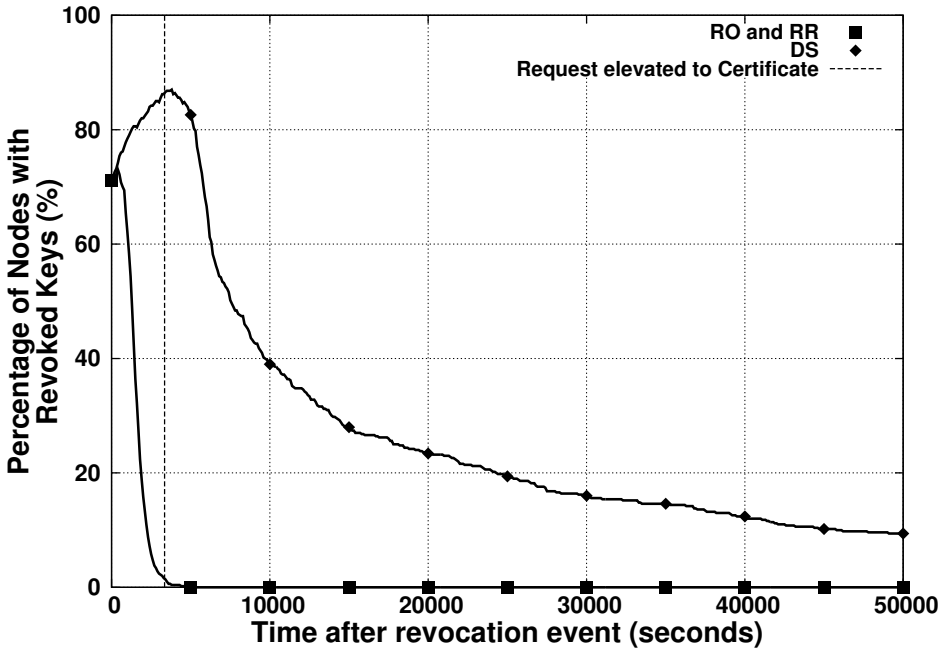
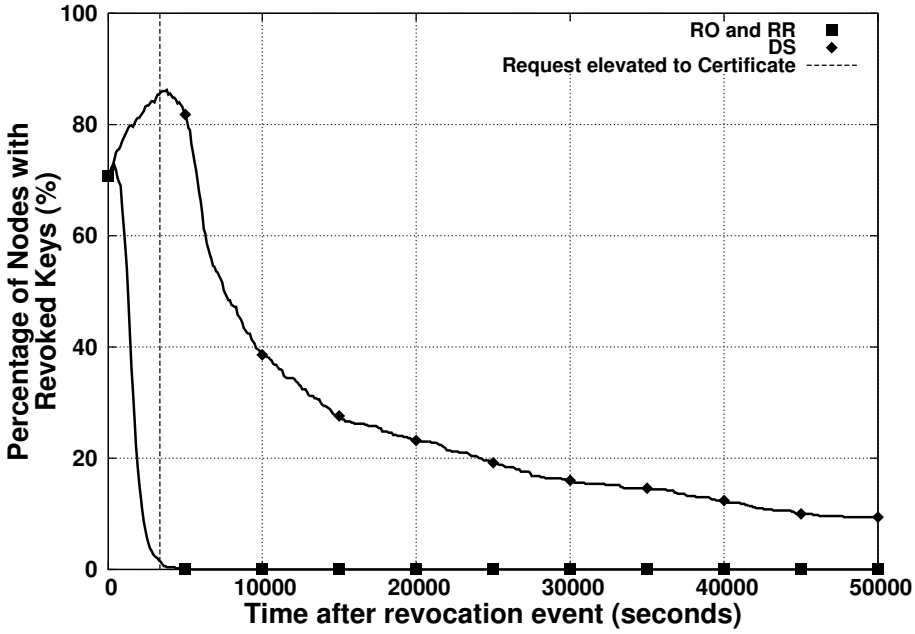
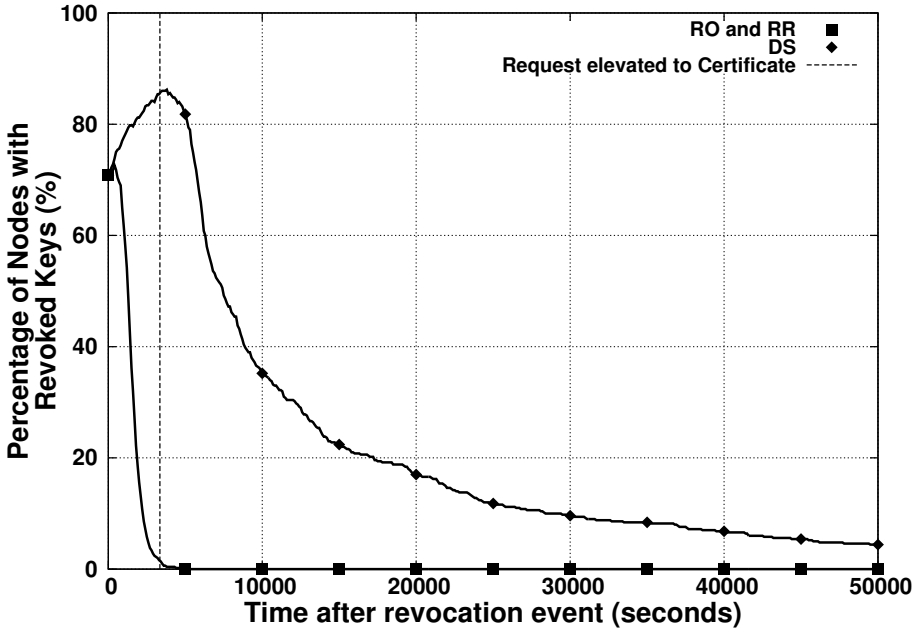


Figure G.2: Revoked Key Distribution over time with 1 Black Hat Node.

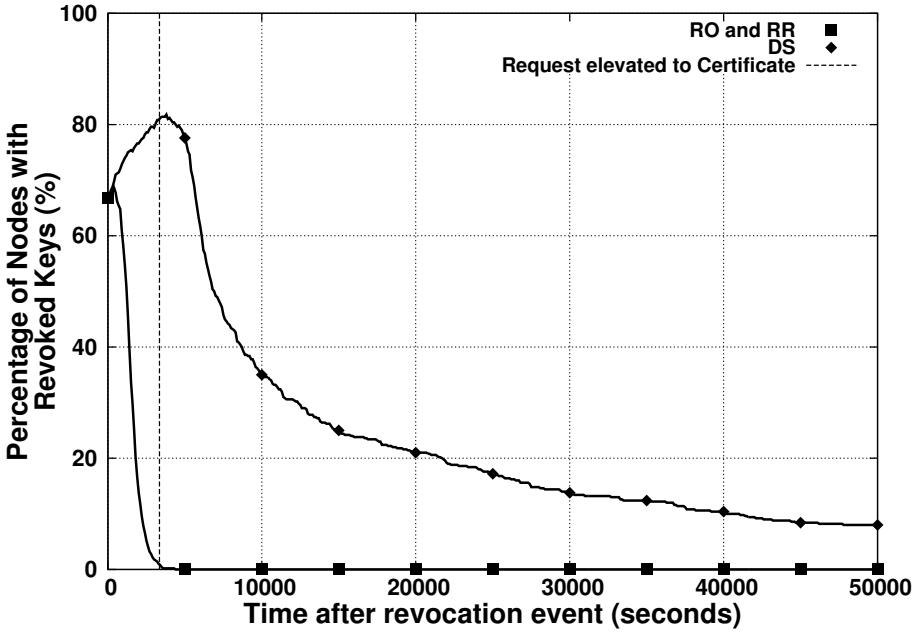


(a) Single Attack

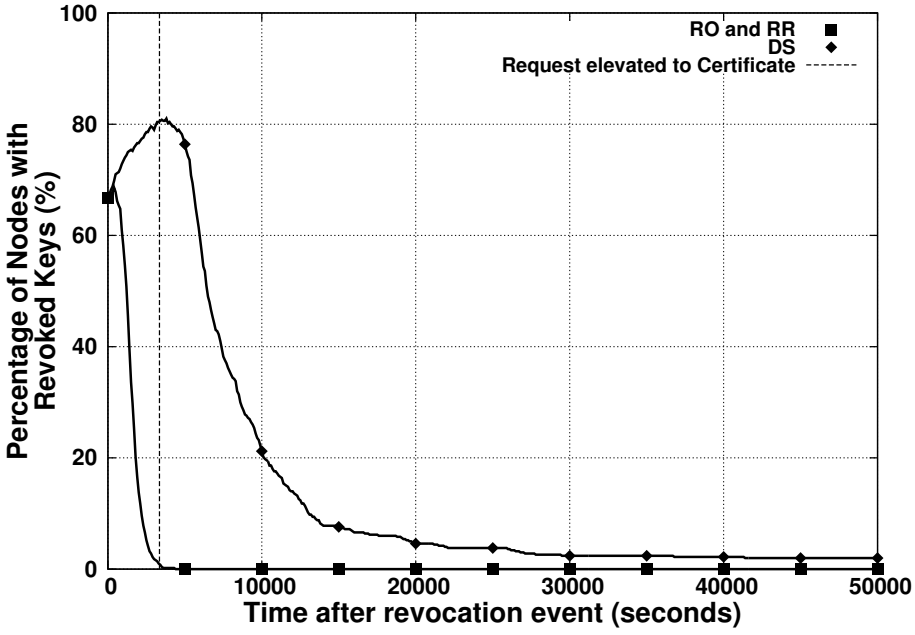


(b) Multiplying Attack

Figure G.3: Revoked Key Distribution over time with 1% Black Hat Nodes.

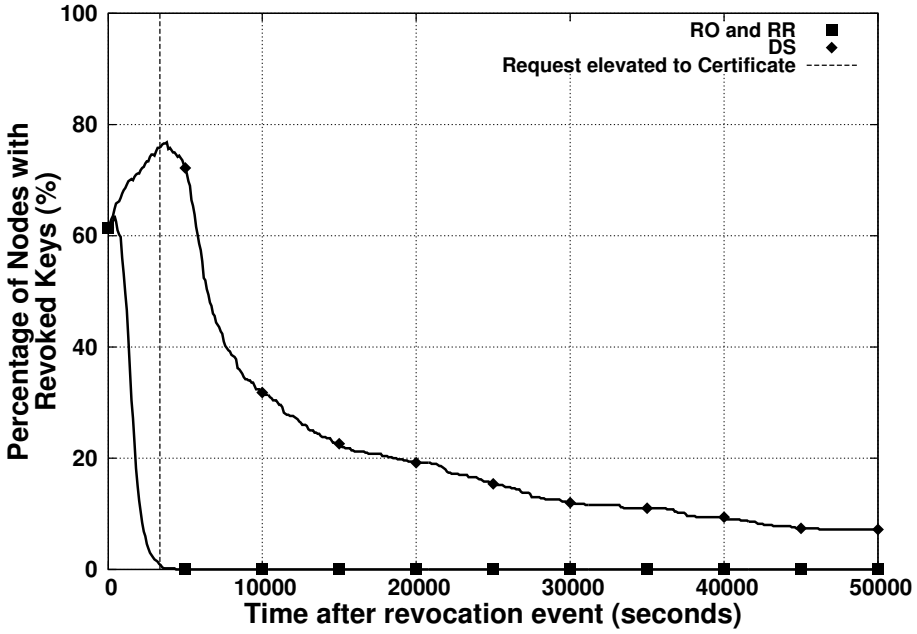


(a) Single Attack

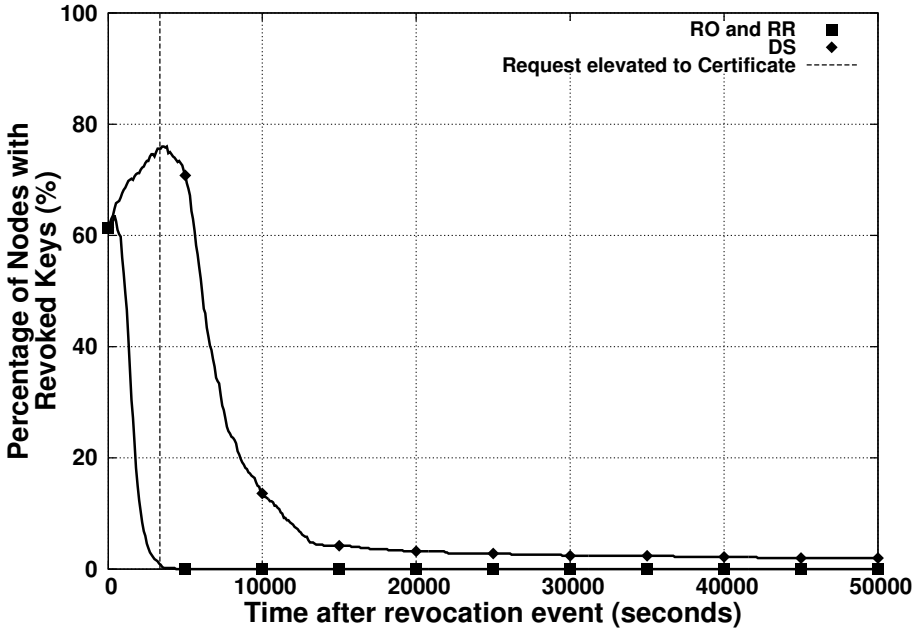


(b) Multiplying Attack

Figure G.4: Revoked Key Distribution over time with 5% Black Hat Nodes.

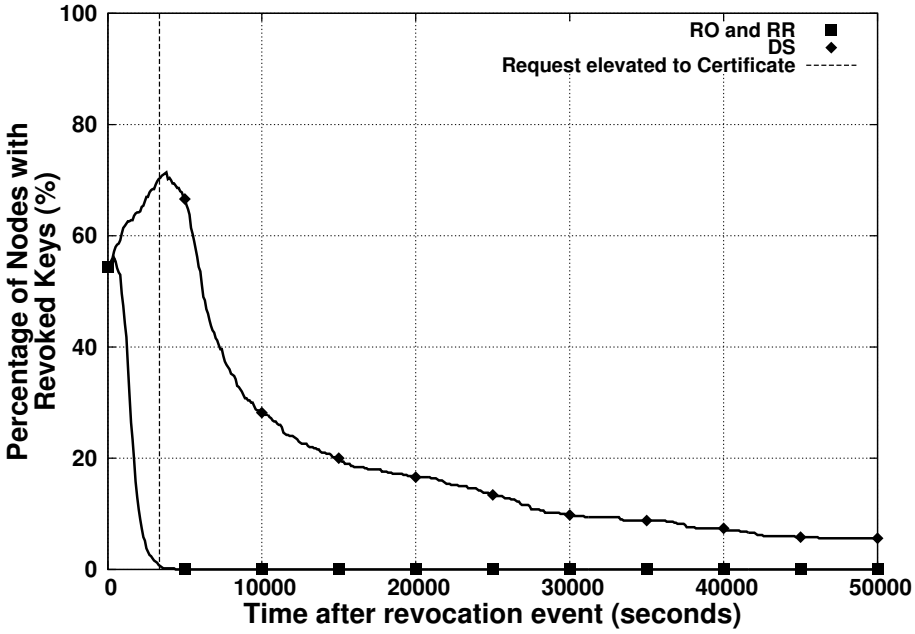


(a) Single Attack

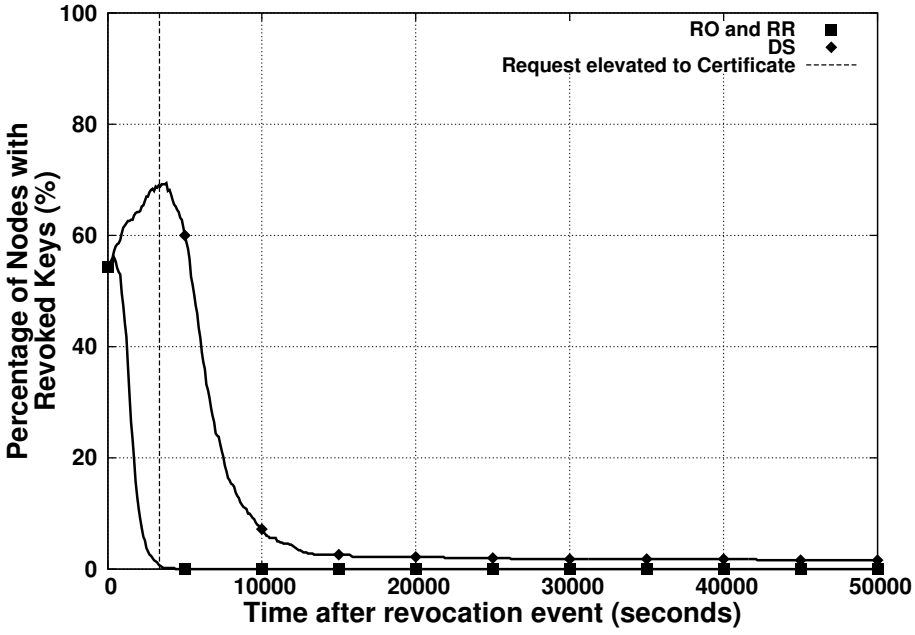


(b) Multiplying Attack

Figure G.5: Revoked Key Distribution over time with 10% Black Hat Nodes.

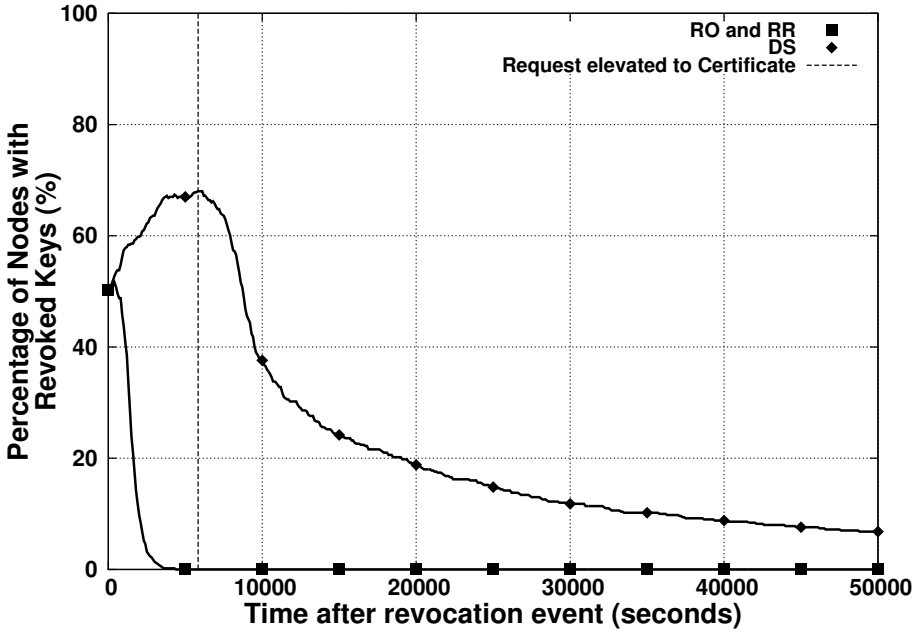


(a) Single Attack

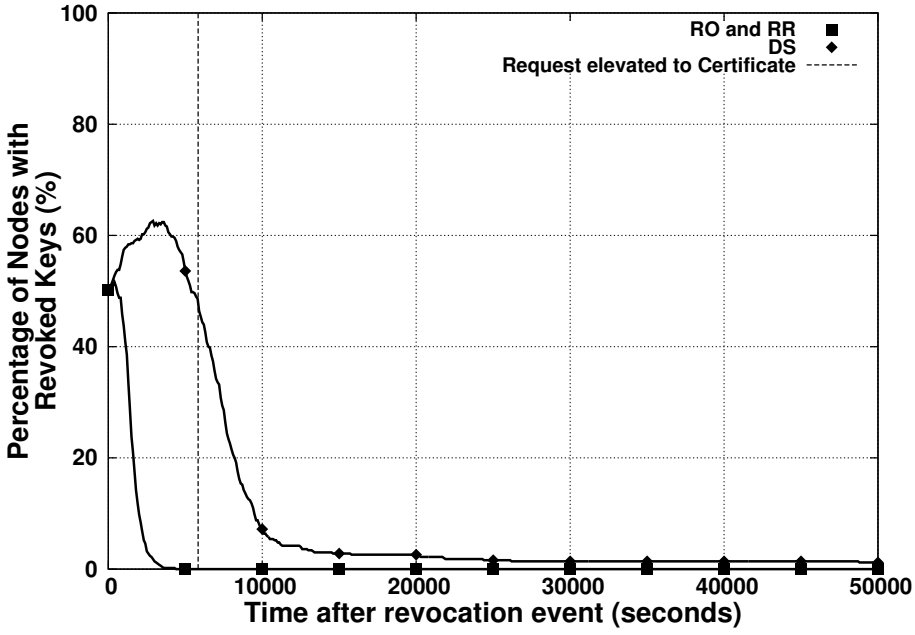


(b) Multiplying Attack

Figure G.6: Revoked Key Distribution over time with 15% Black Hat Nodes.

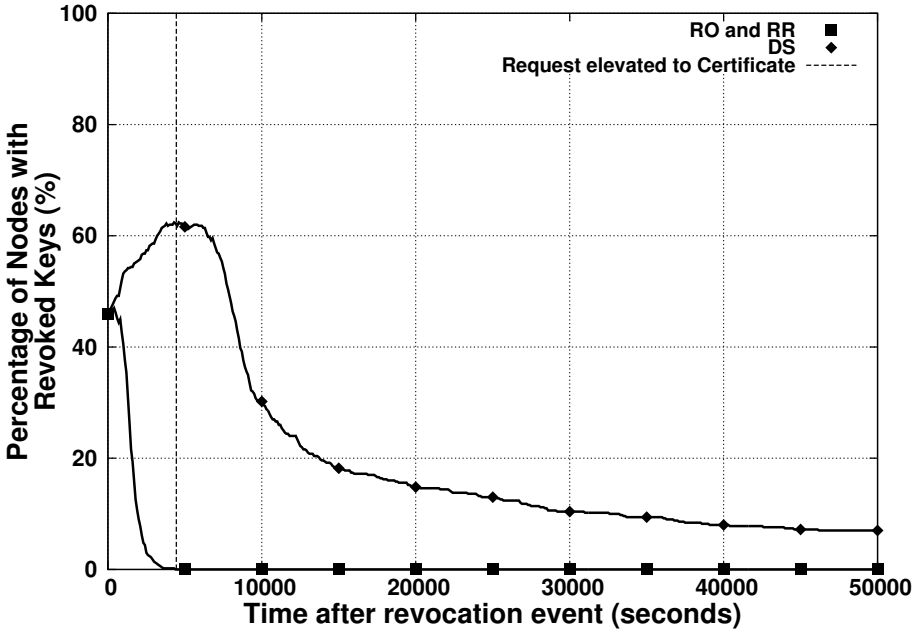


(a) Single Attack

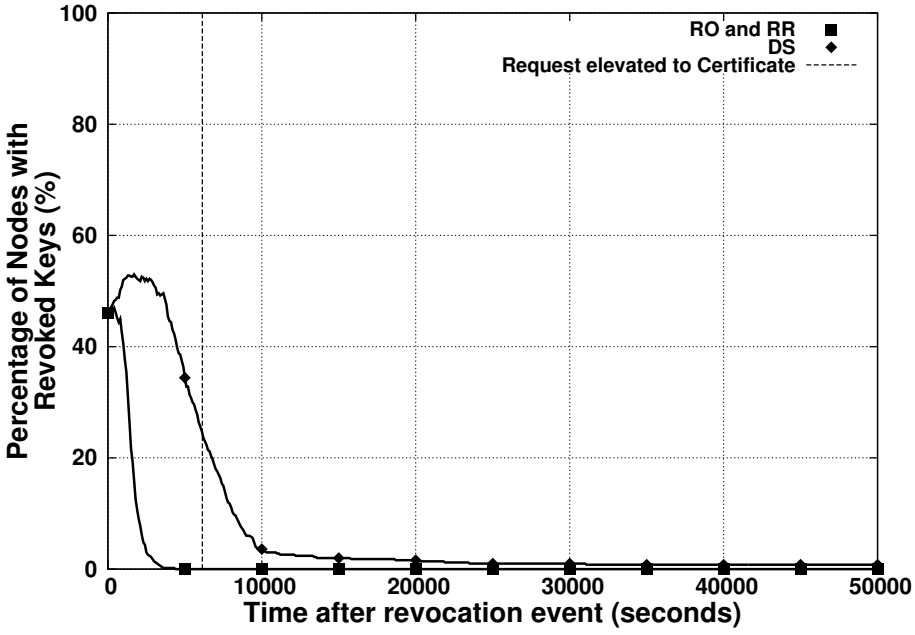


(b) Multiplying Attack

Figure G.7: Revoked Key Distribution over time with 20% Black Hat Nodes.

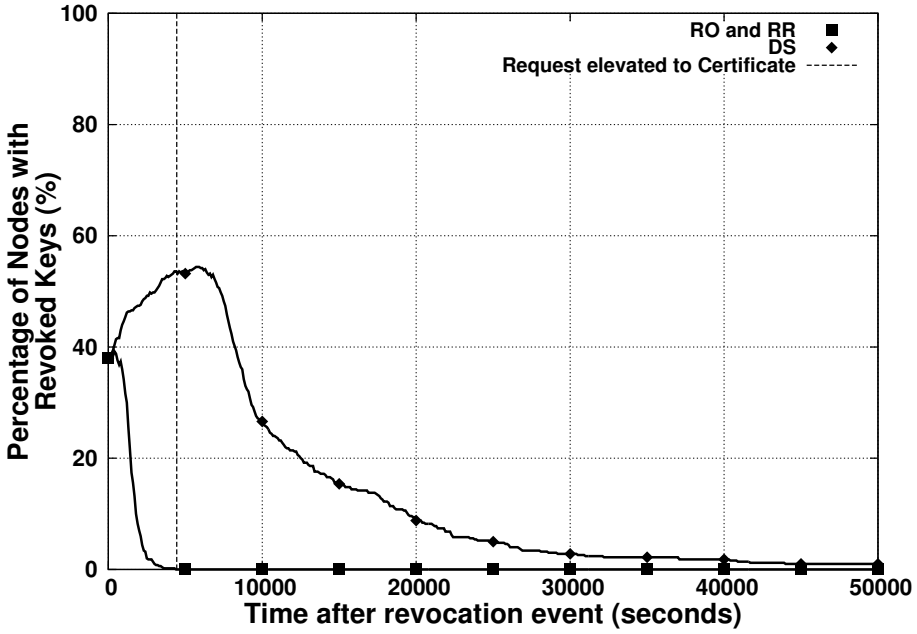


(a) Single Attack

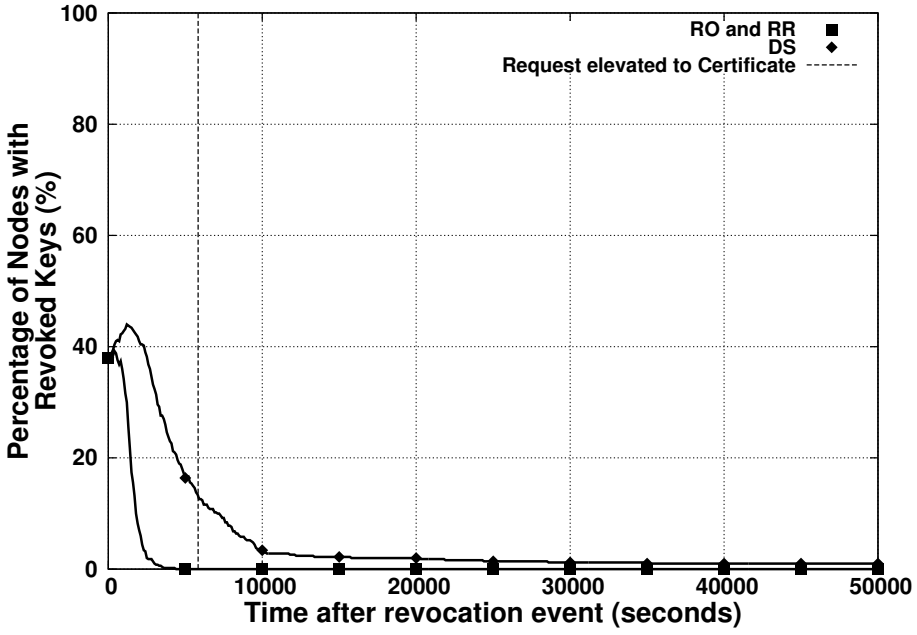


(b) Multiplying Attack

Figure G.8: Revoked Key Distribution over time with 25% Black Hat Nodes.



(a) Single Attack



(b) Multiplying Attack

Figure G.9: Revoked Key Distribution over time with 30% Black Hat Nodes.

Appendix H

New Public Key Results

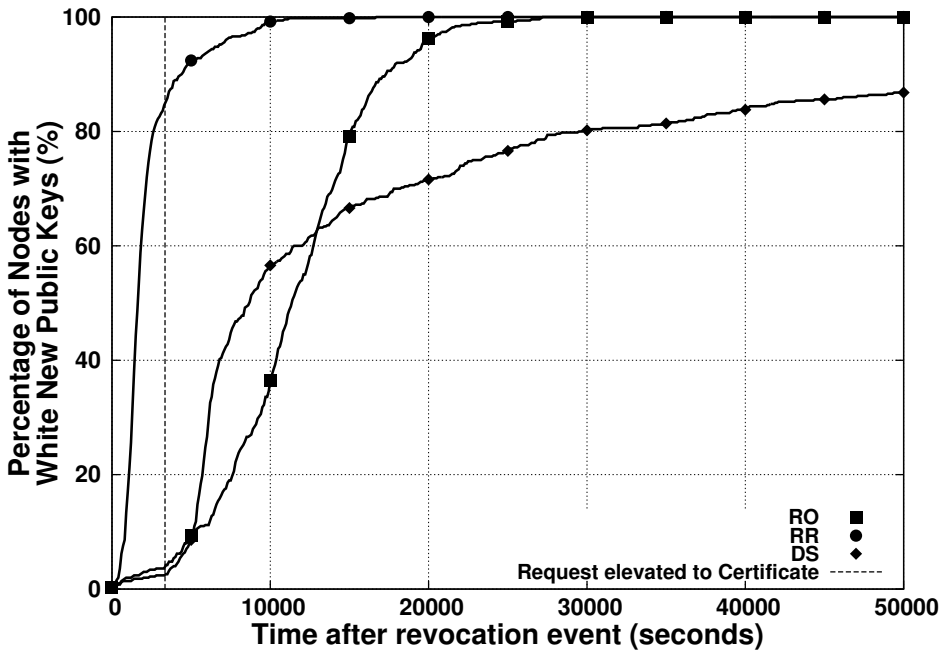


Figure H.1: New Public Key Distribution over time with no Black Hat Nodes.

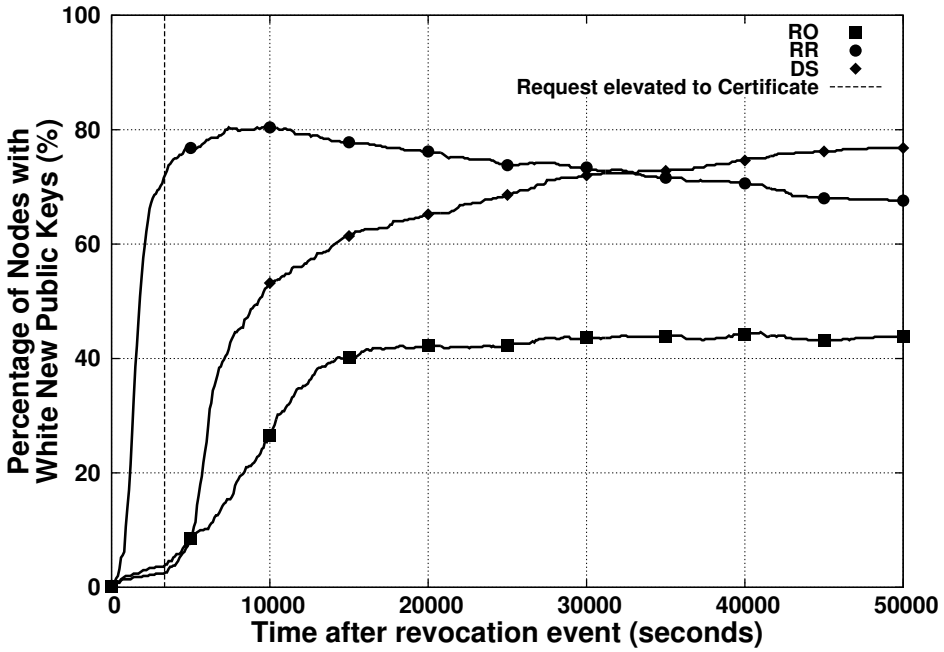
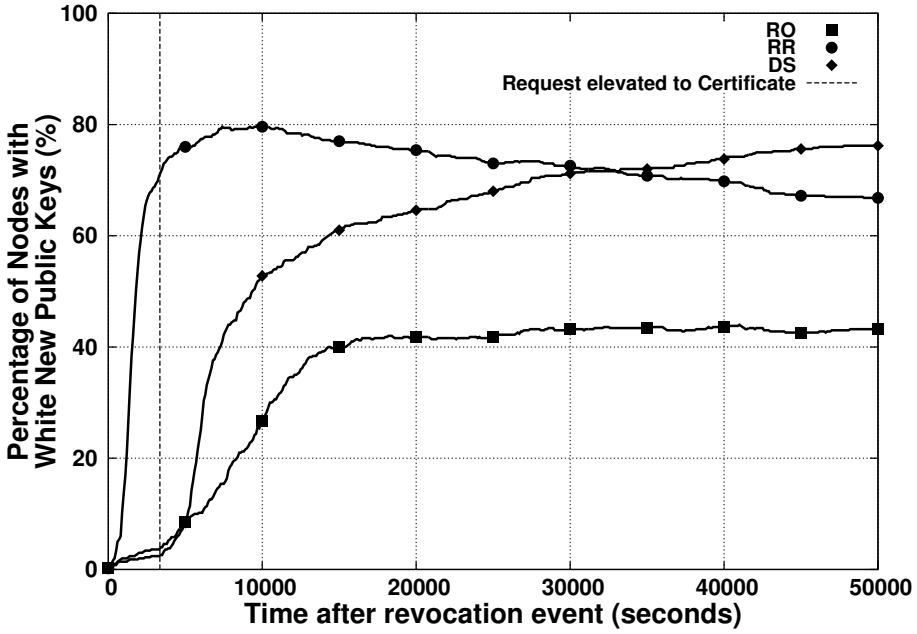
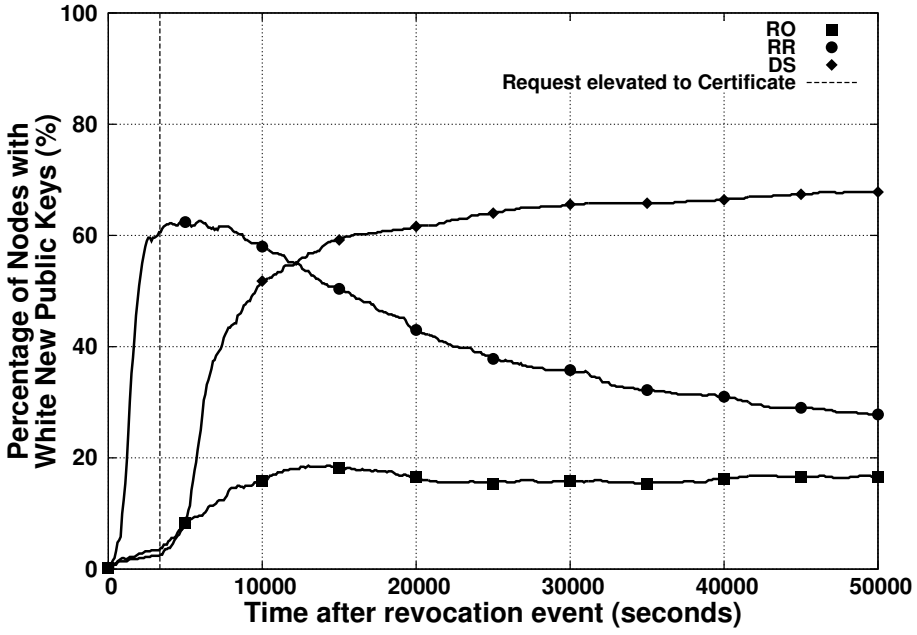


Figure H.2: New Public Key Distribution over time with 1 Black Hat Node.

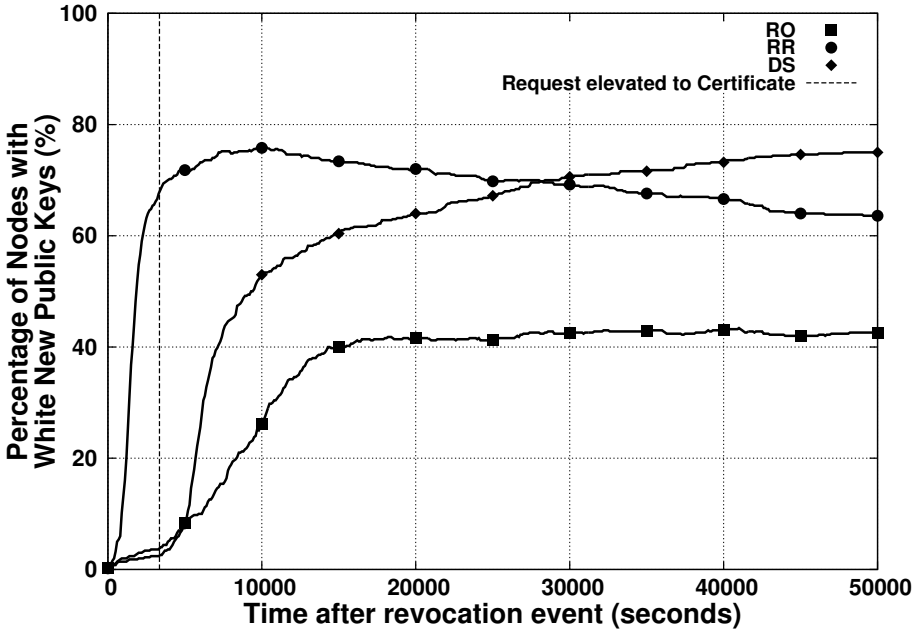


(a) Single Attack

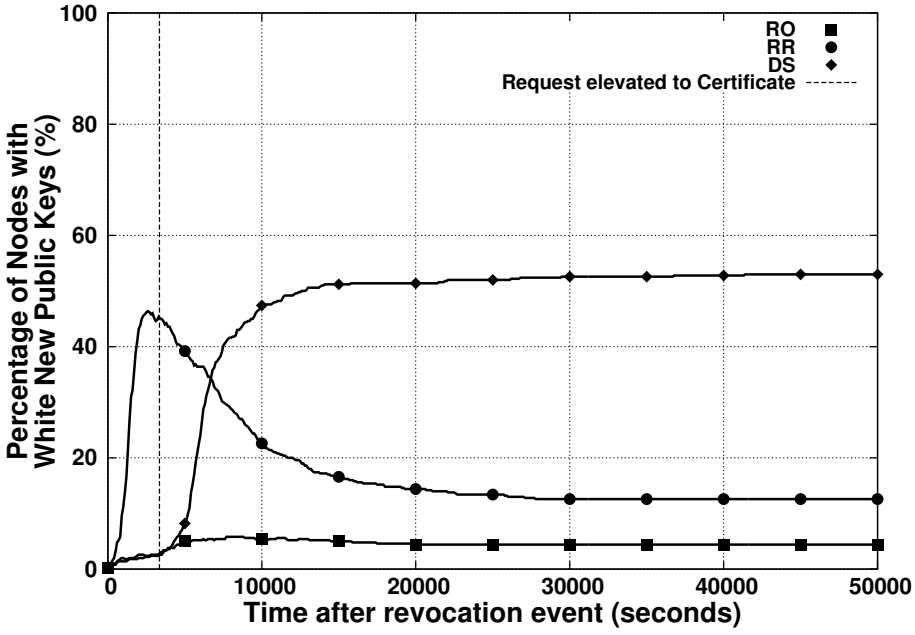


(b) Multiplying Attack

Figure H.3: New Public Key Distribution over time with 1% Black Hat Nodes.

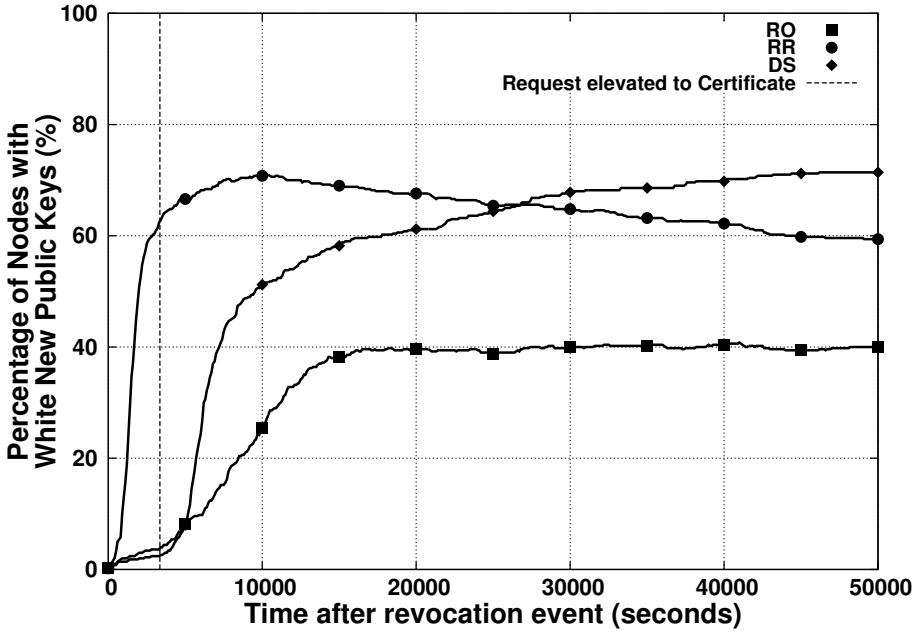


(a) Single Attack

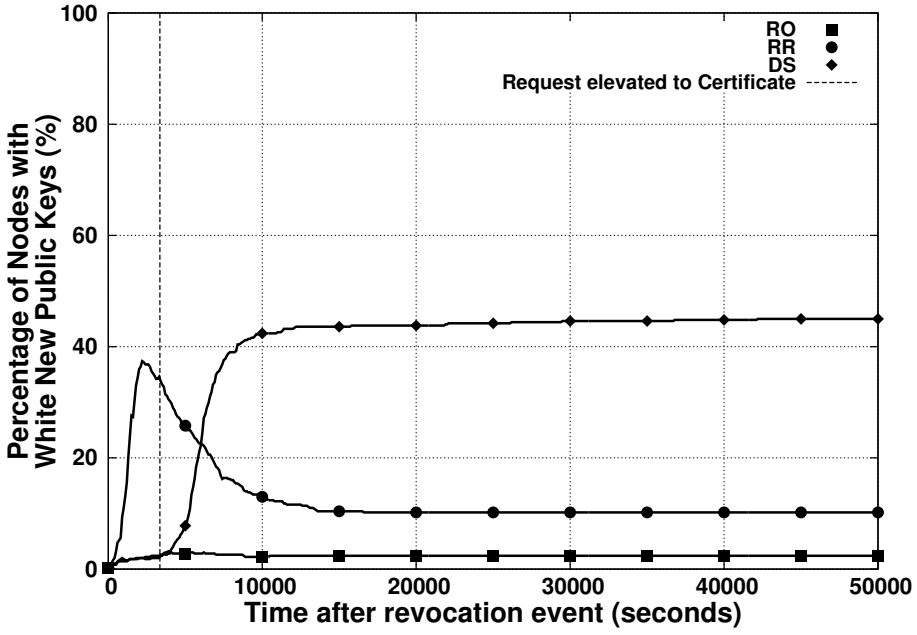


(b) Multiplying Attack

Figure H.4: New Public Key Distribution over time with 5% Black Hat Nodes.

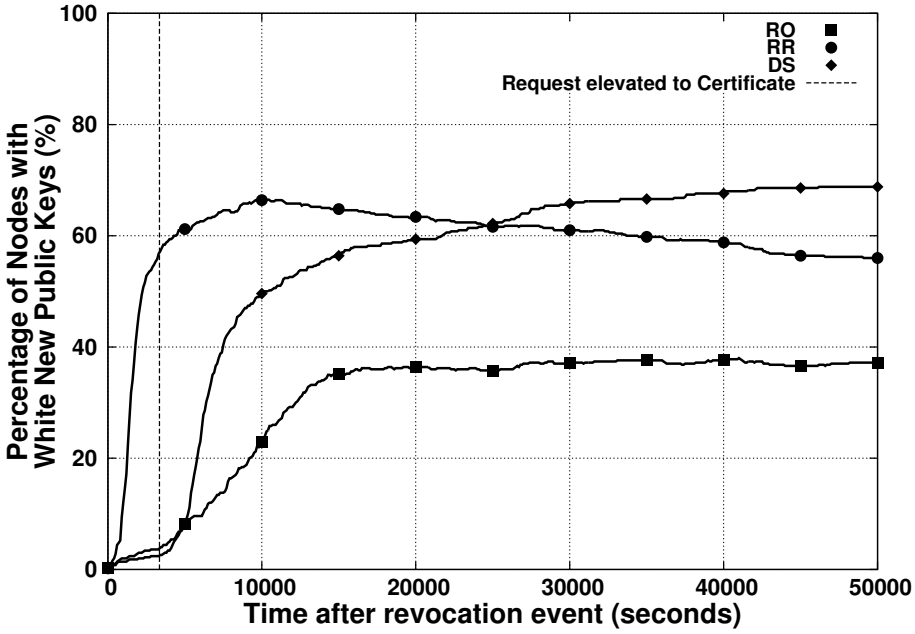


(a) Single Attack

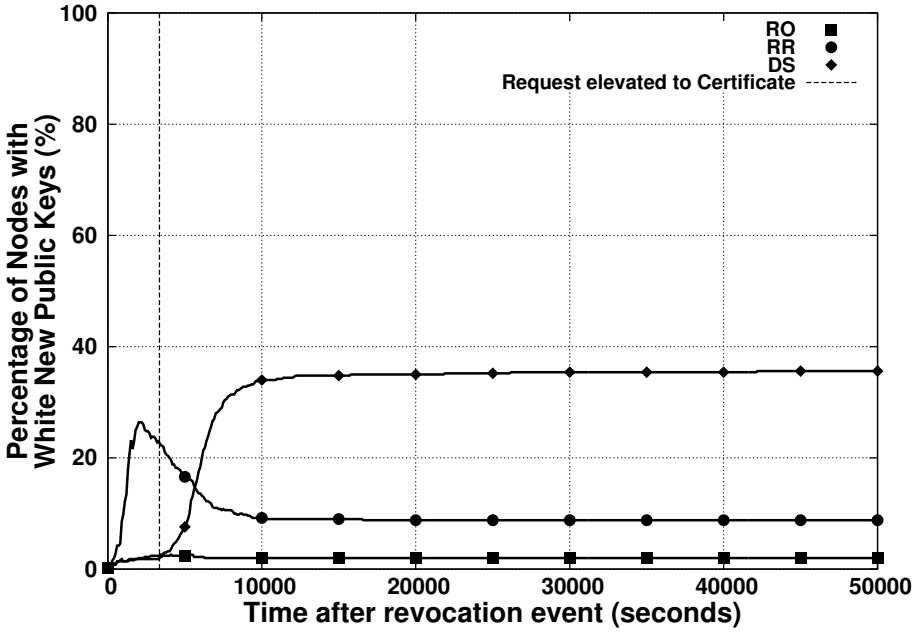


(b) Multiplying Attack

Figure H.5: New Public Key Distribution over time with 10% Black Hat Nodes.

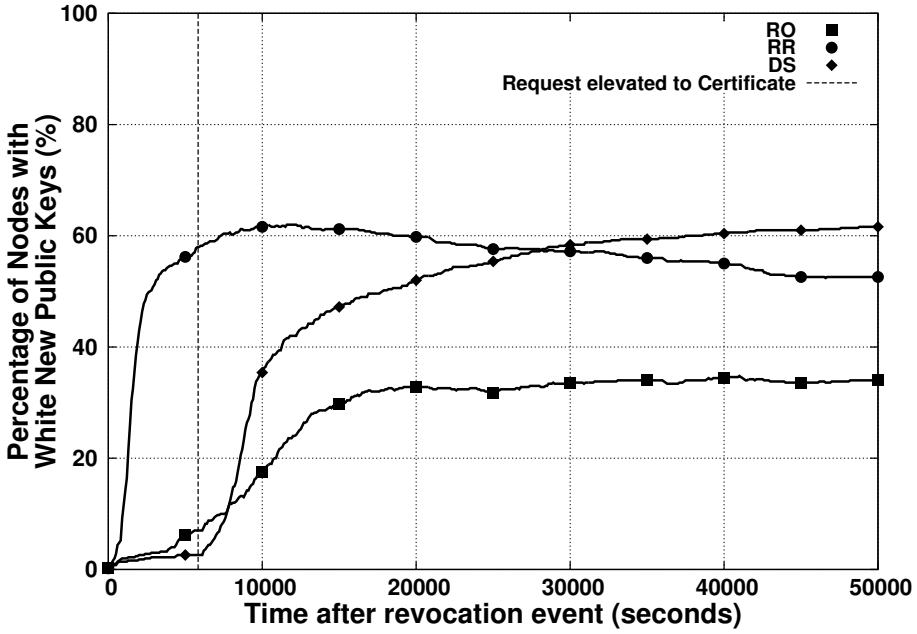


(a) Single Attack

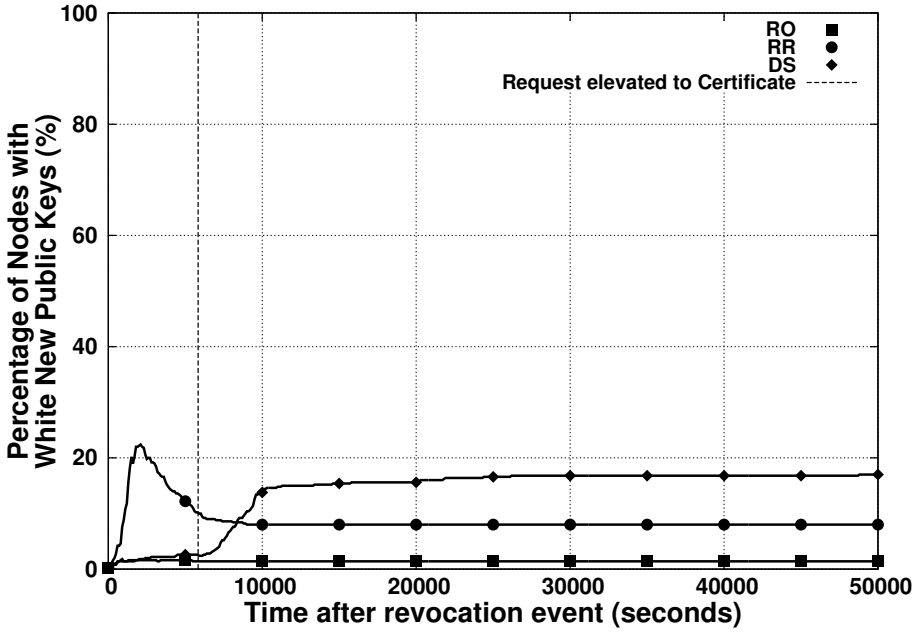


(b) Multiplying Attack

Figure H.6: New Public Key Distribution over time with 15% Black Hat Nodes.

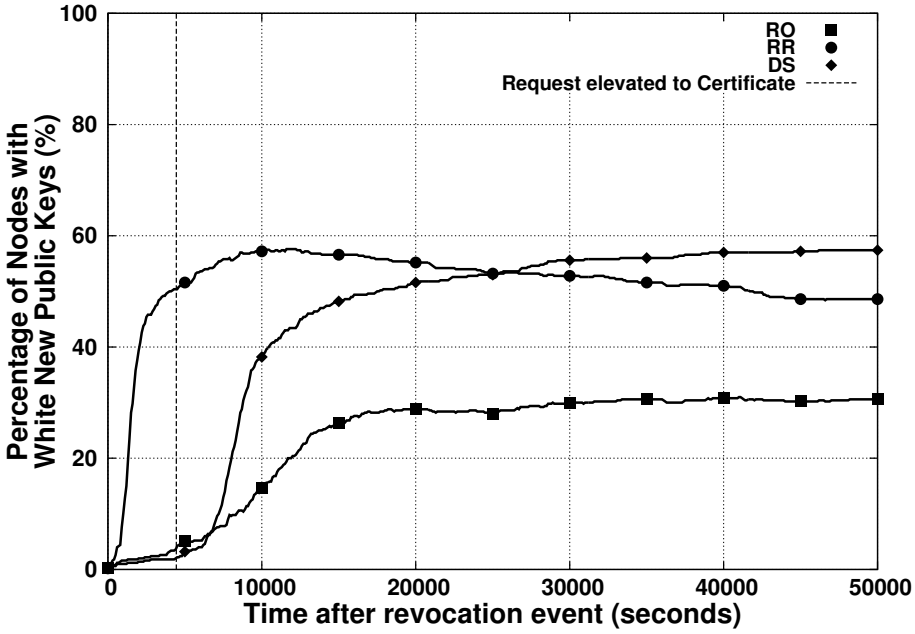


(a) Single Attack

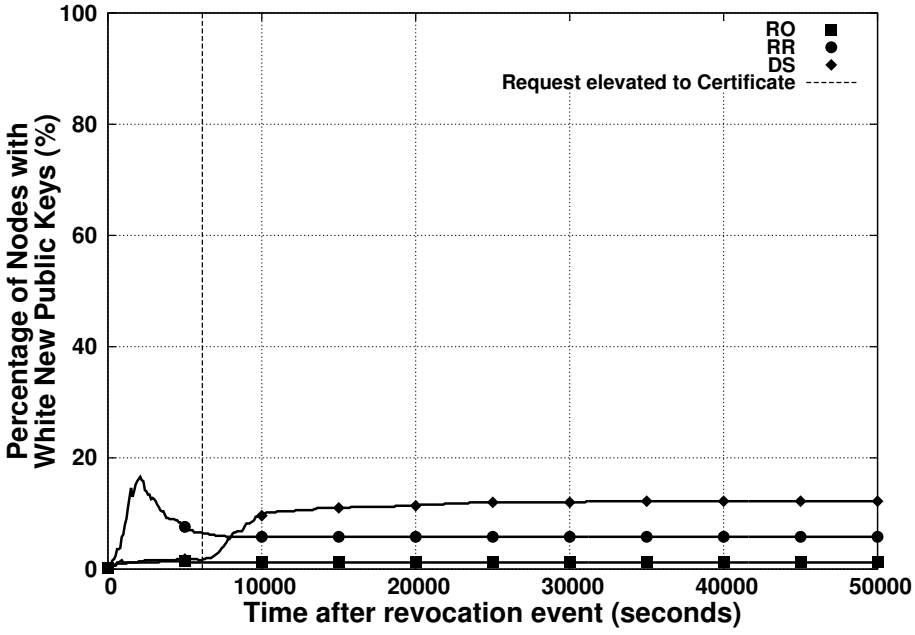


(b) Multiplying Attack

Figure H.7: New Public Key Distribution over time with 20% Black Hat Nodes.

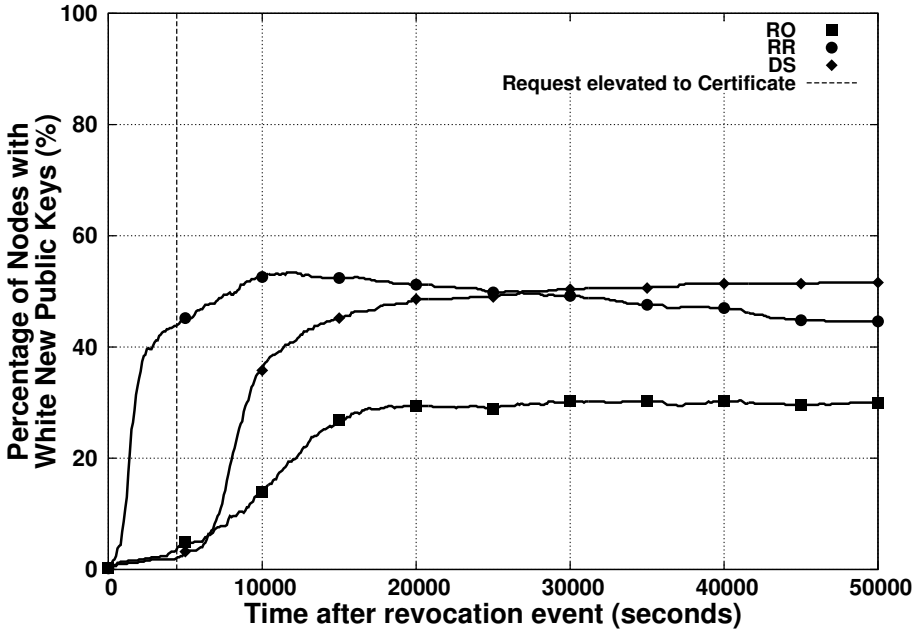


(a) Single Attack

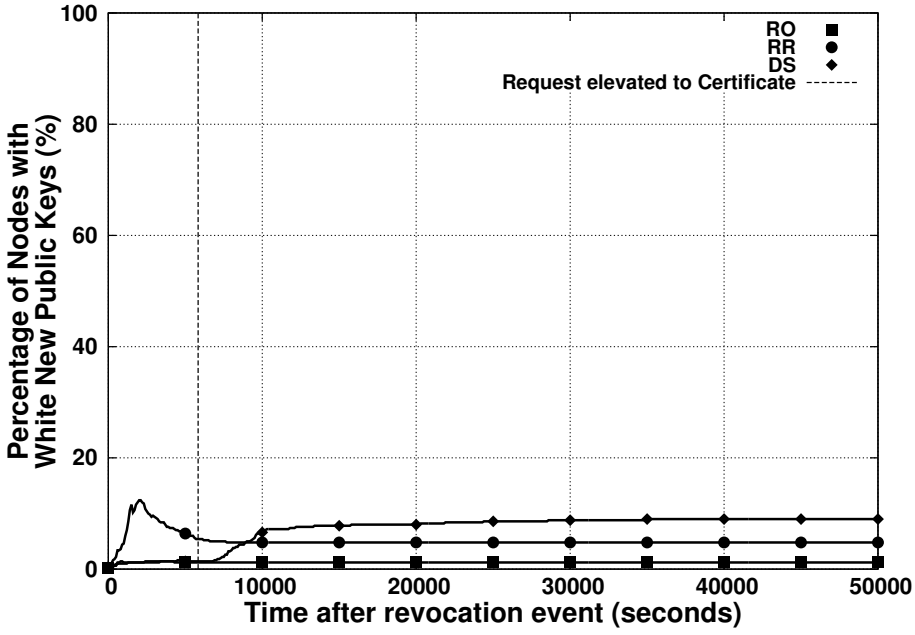


(b) Multiplying Attack

Figure H.8: New Public Key Distribution over time with 25% Black Hat Nodes.



(a) Single Attack



(b) Multiplying Attack

Figure H.9: New Public Key Distribution over time with 30% Black Hat Nodes.

Appendix I

Spoofed Revocation Key Results

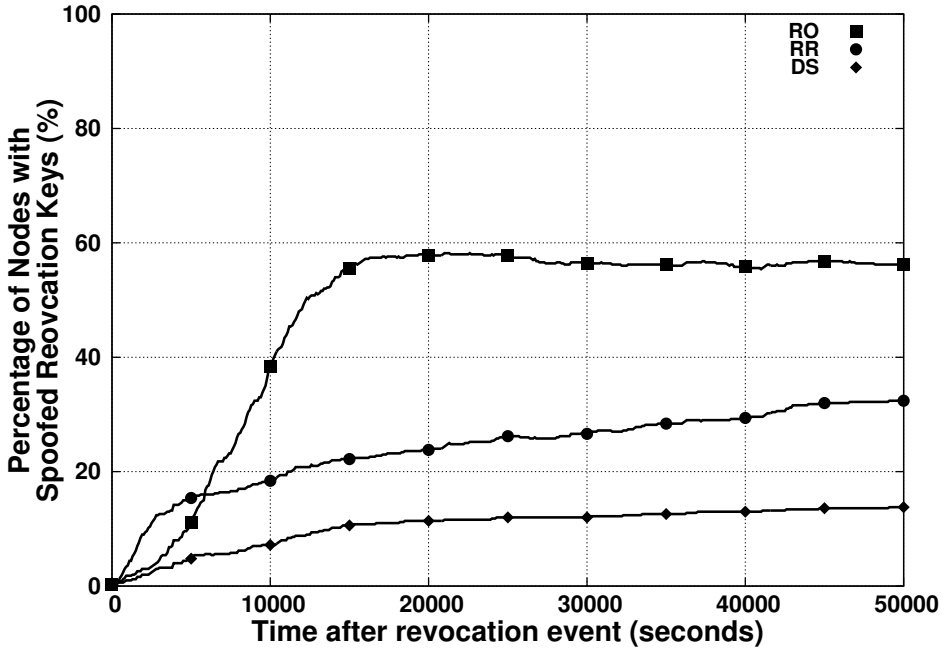
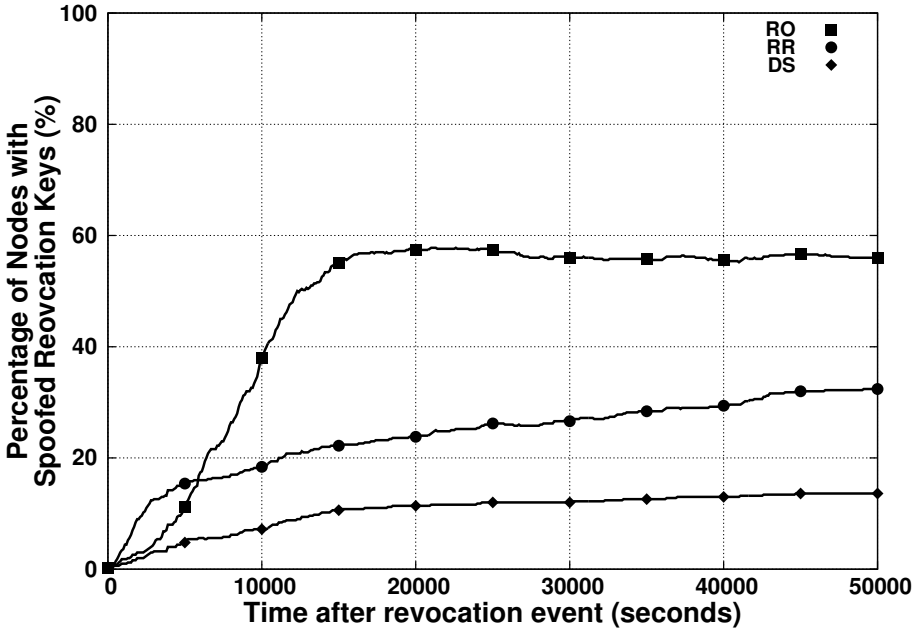
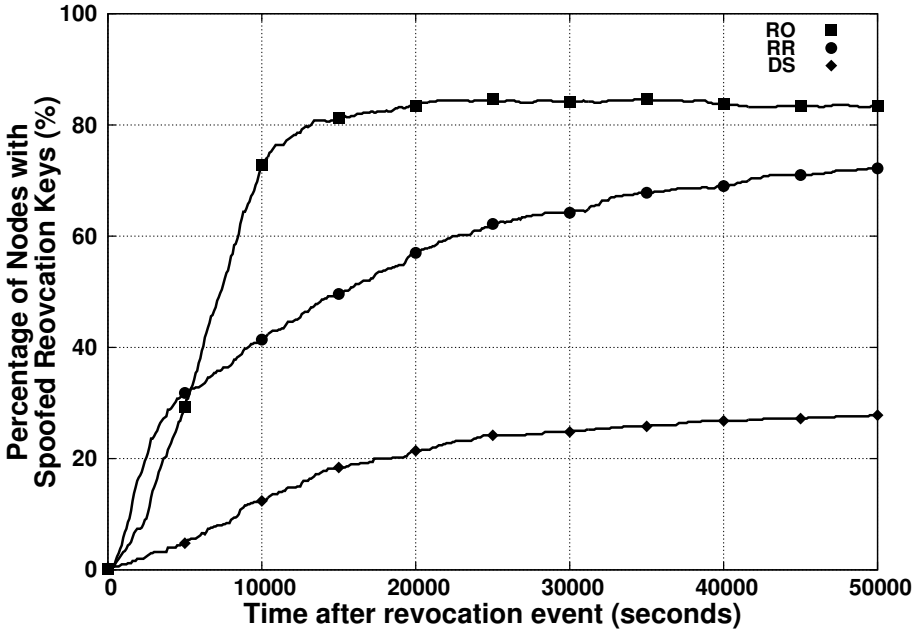


Figure I.1: Spoofed Revocation Key Distribution over time with 1 Black Hat Node.

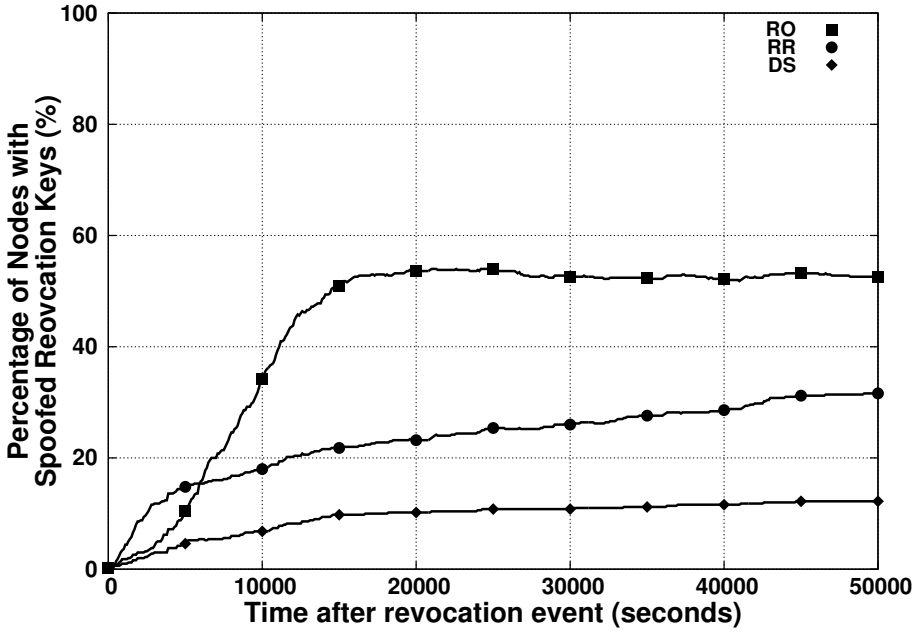


(a) Single Attack

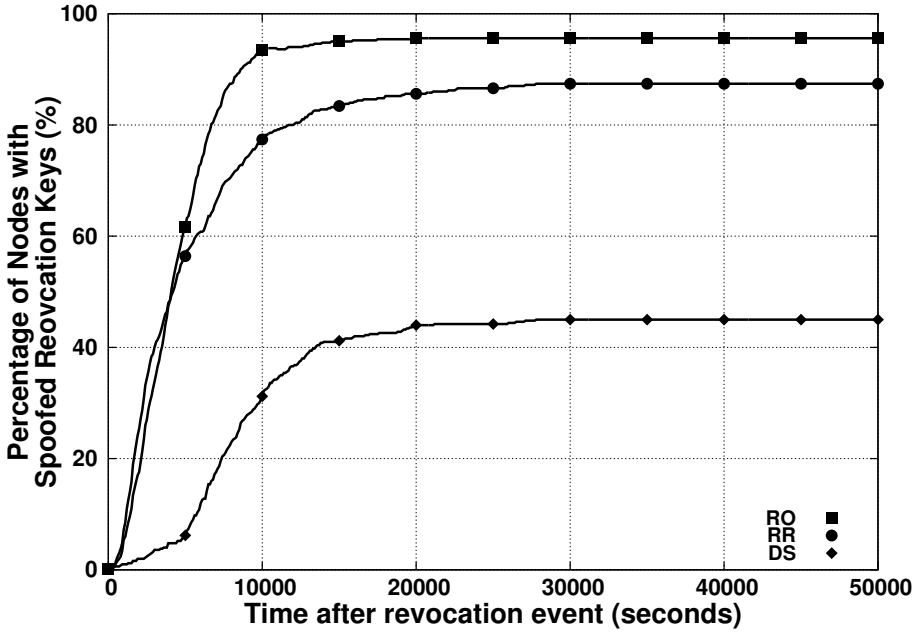


(b) Multiplying Attack

Figure I.2: Spoofed Revocation Key Distribution over time with 1% Black Hat Nodes.

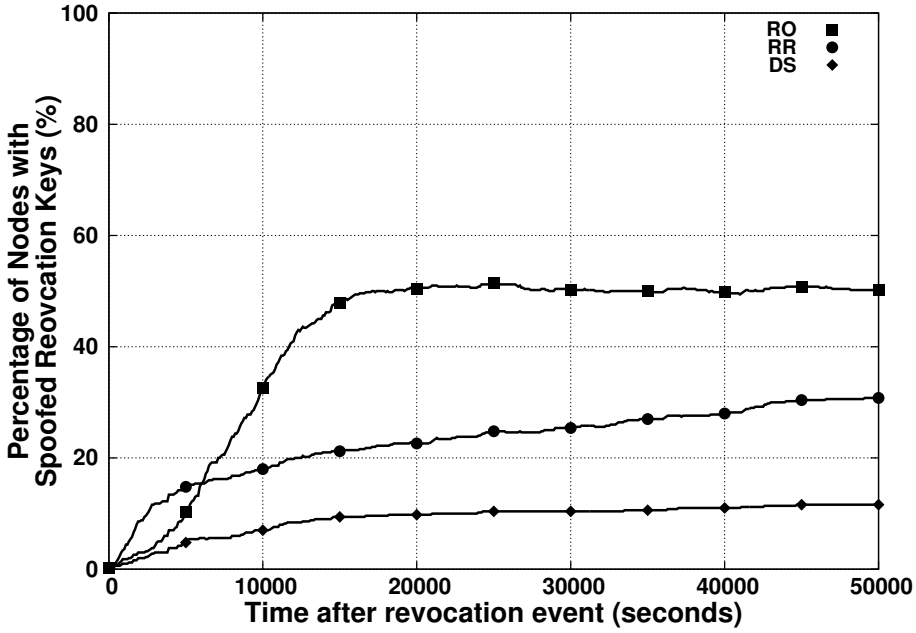


(a) Single Attack

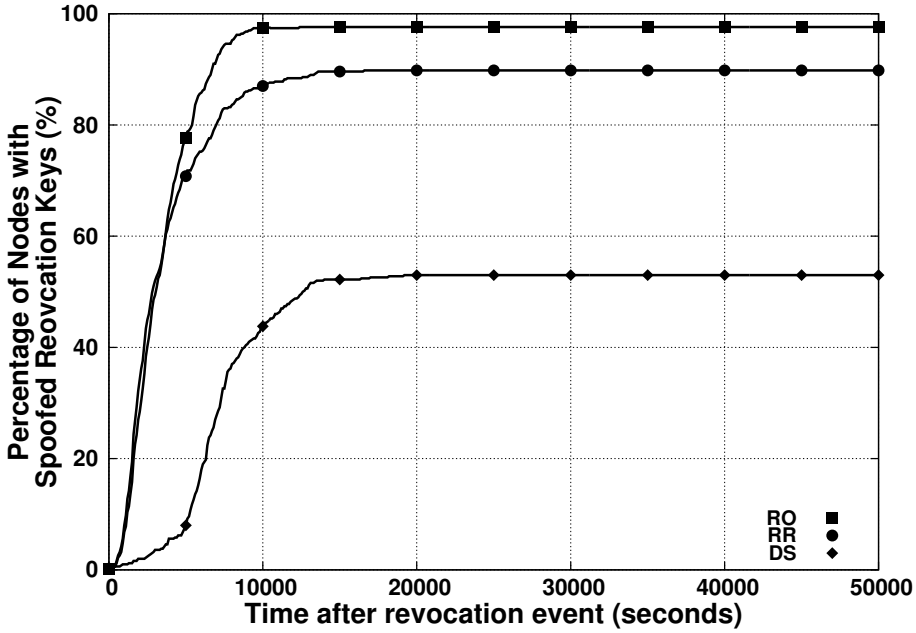


(b) Multiplying Attack

Figure I.3: Spoofed Revocation Key Distribution over time with 5% Black Hat Nodes.

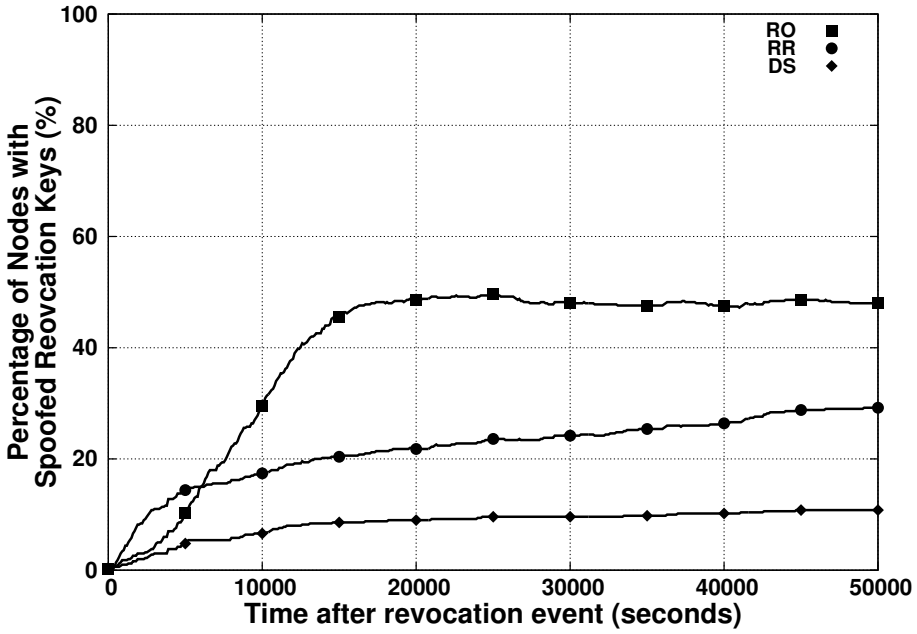


(a) Single Attack

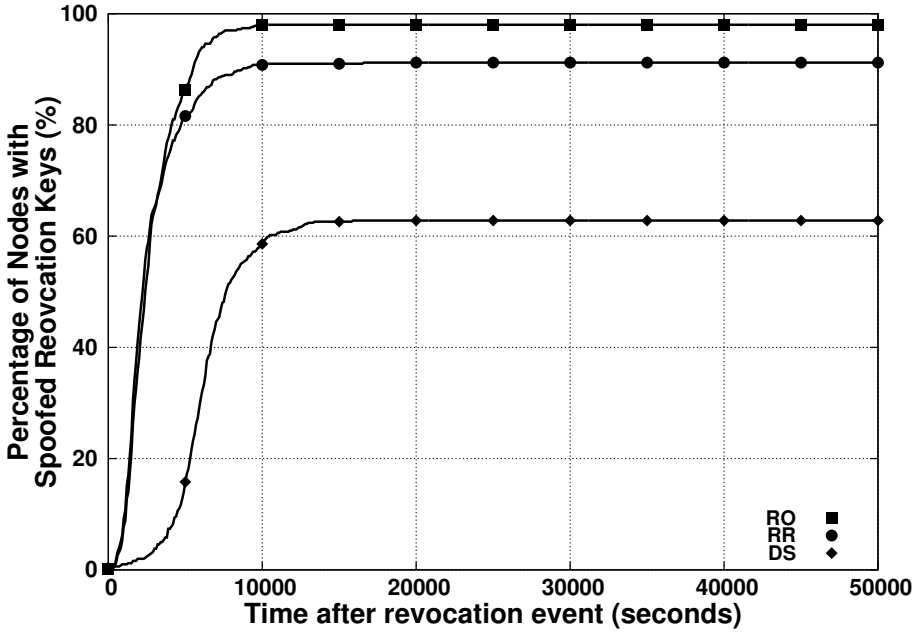


(b) Multiplying Attack

Figure I.4: Spoofed Revocation Key Distribution over time with 10% Black Hat Nodes.

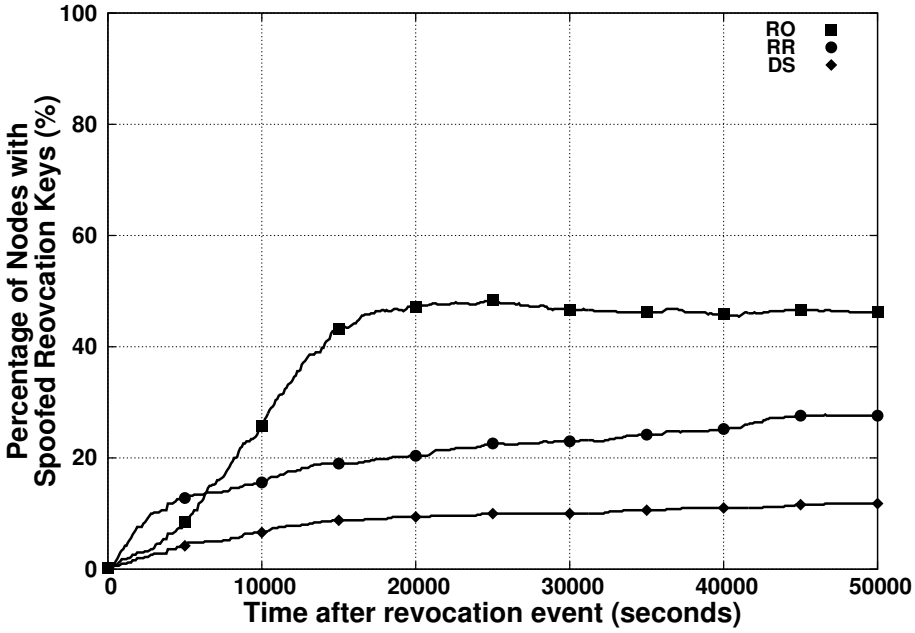


(a) Single Attack

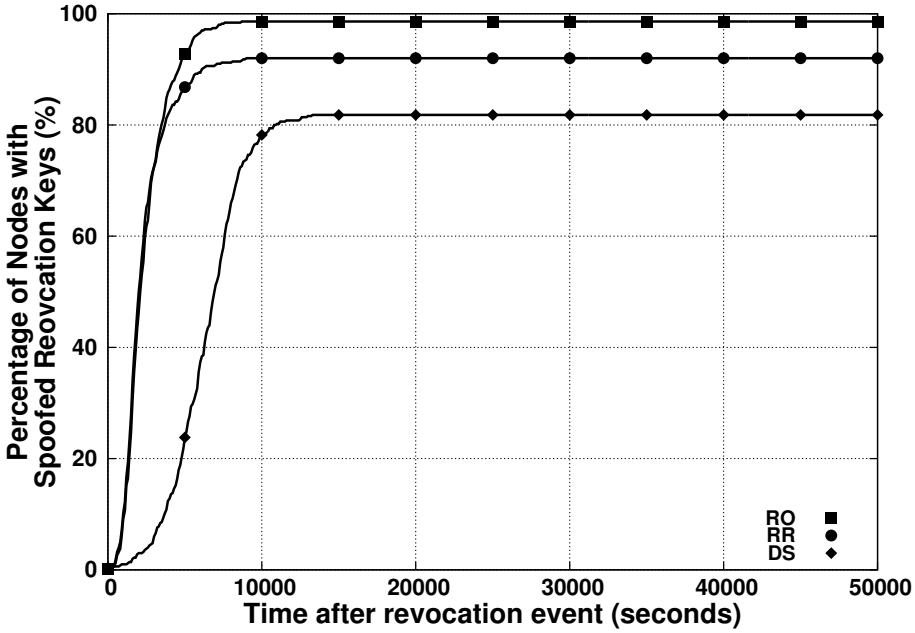


(b) Multiplying Attack

Figure I.5: Spoofed Revocation Key Distribution over time with 15% Black Hat Nodes.

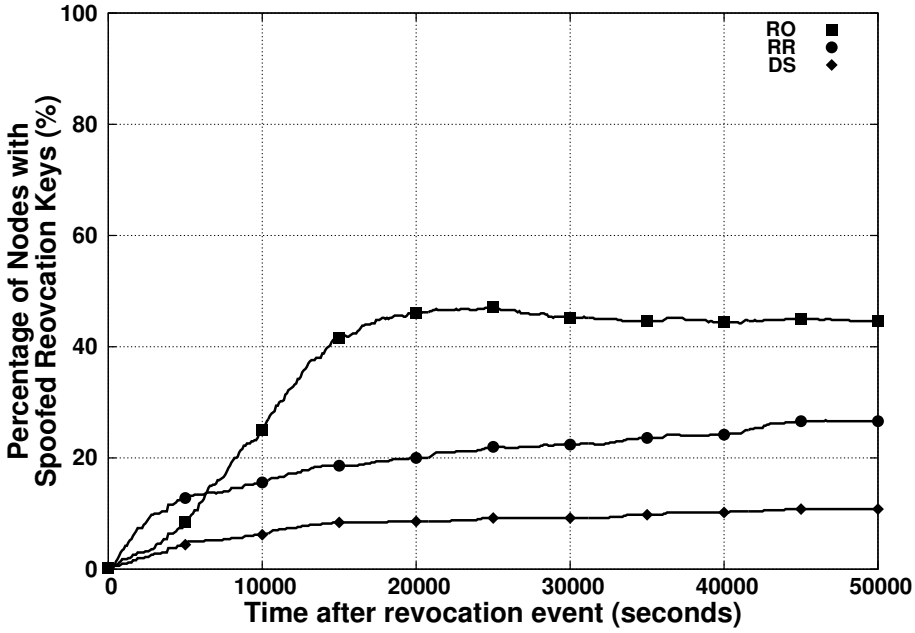


(a) Single Attack

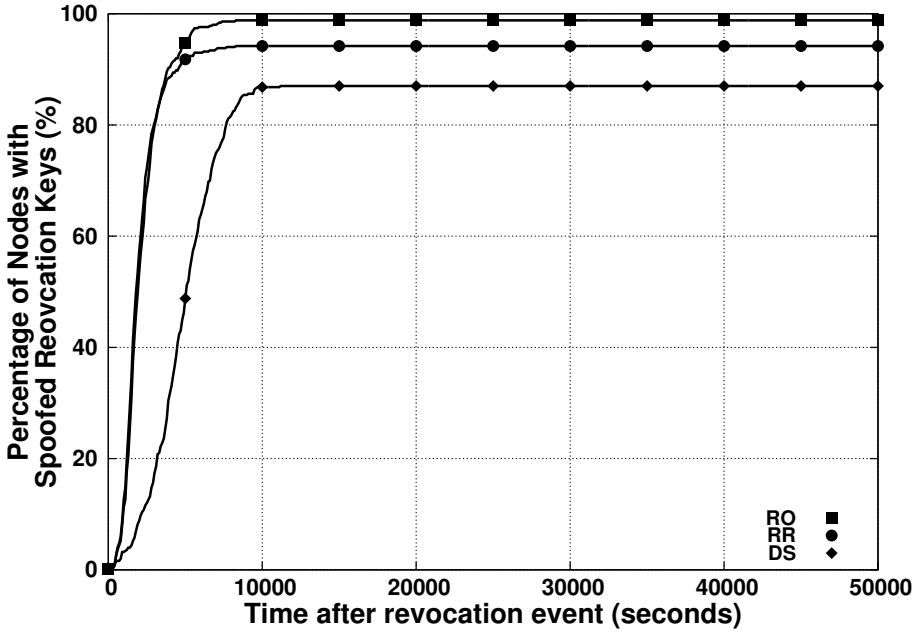


(b) Multiplying Attack

Figure I.6: Spoofed Revocation Key Distribution over time with 20% Black Hat Nodes.

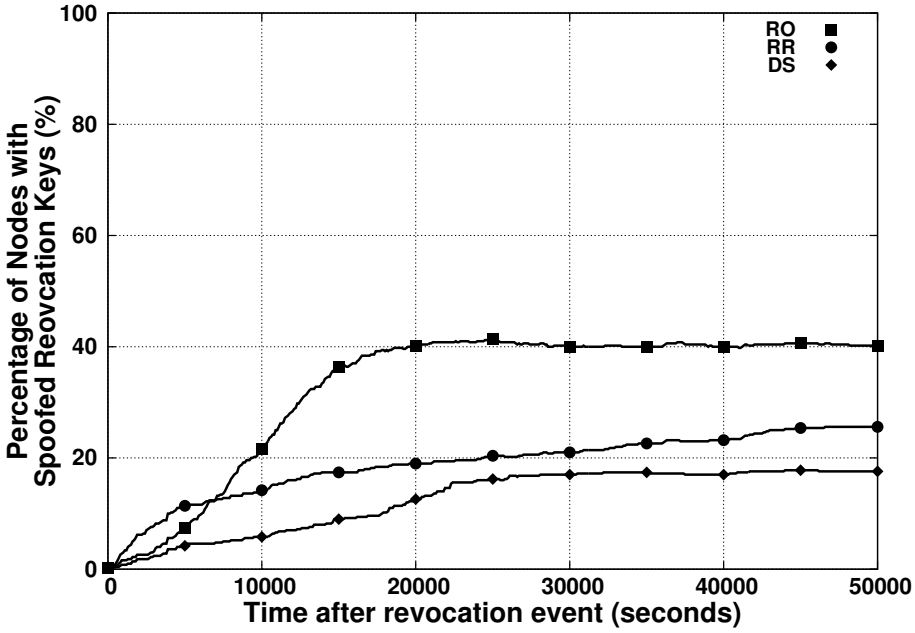


(a) Single Attack

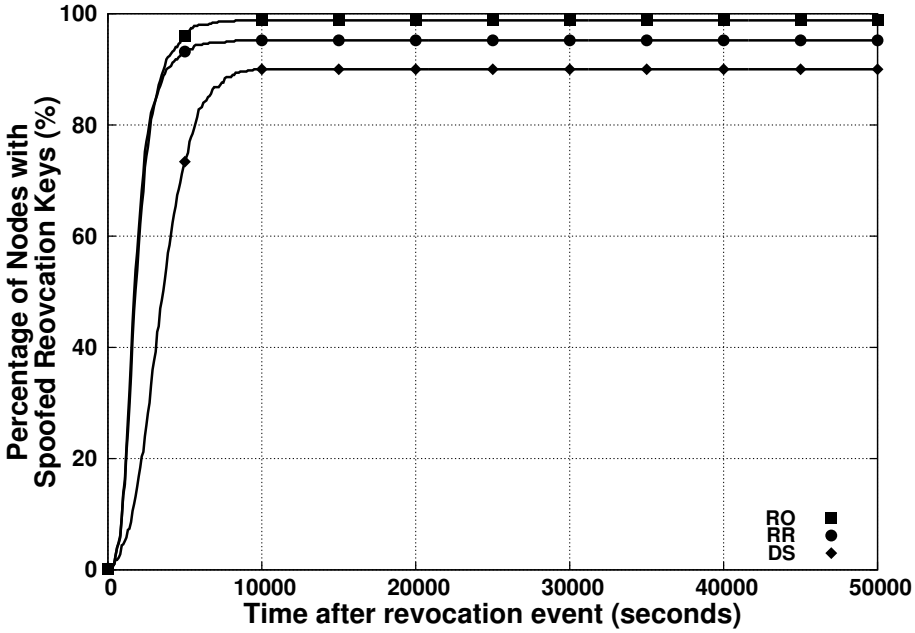


(b) Multiplying Attack

Figure I.7: Spoofed Revocation Key Distribution over time with 25% Black Hat Nodes.



(a) Single Attack



(b) Multiplying Attack

Figure I.8: Spoofed Revocation Key Distribution over time with 30% Black Hat Nodes.

Appendix J

White Hat Certificates Results

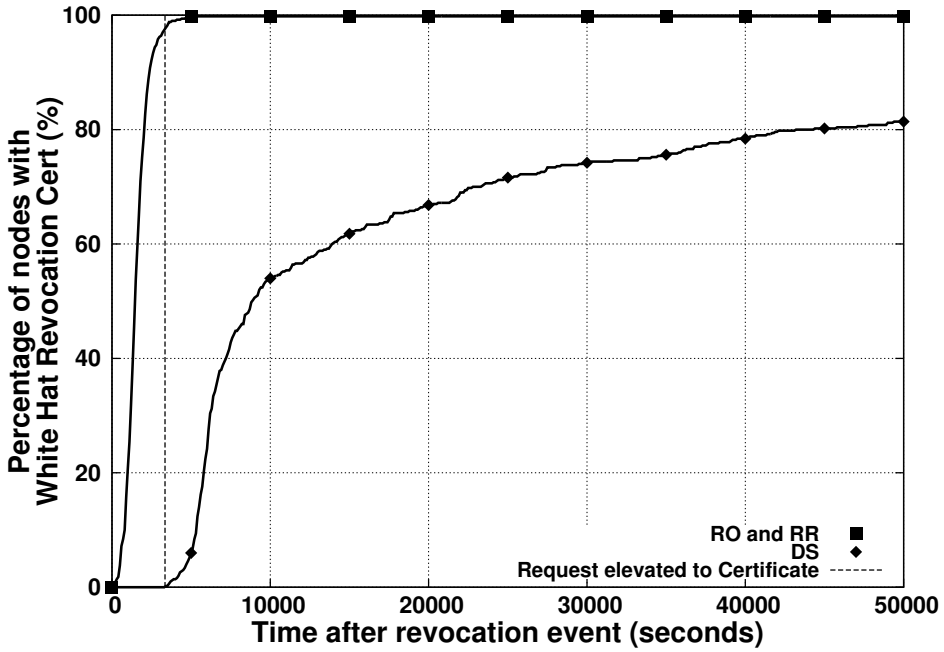


Figure J.1: Percentage of nodes with White Hat Revocation Certificates over time with no Black Hat Nodes.

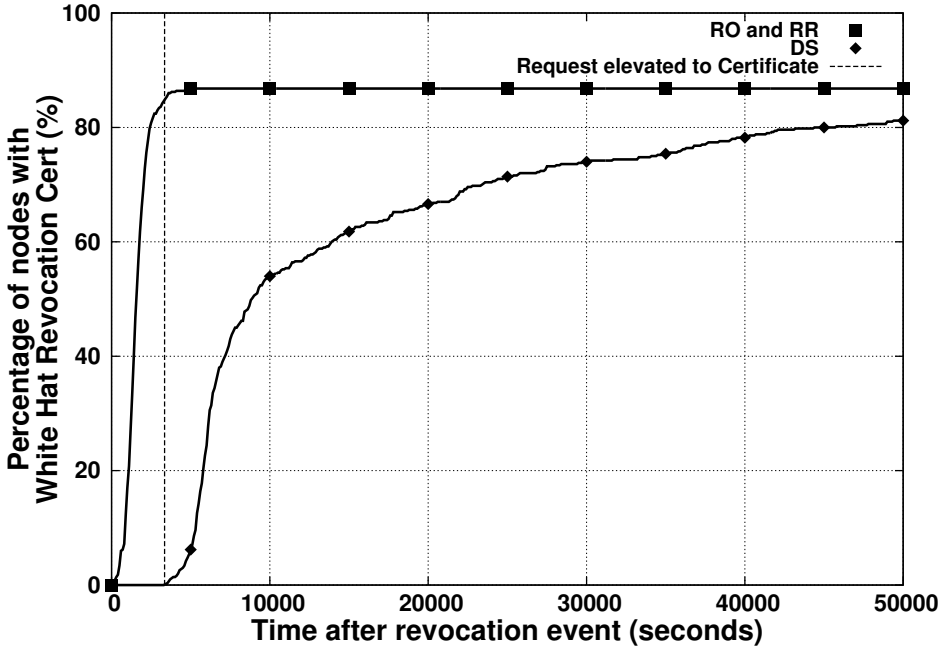
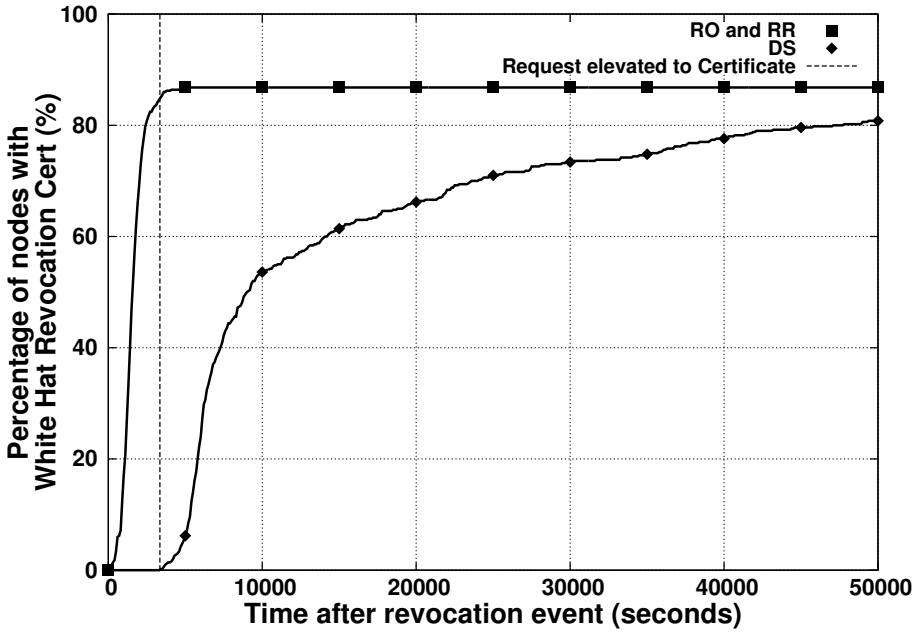
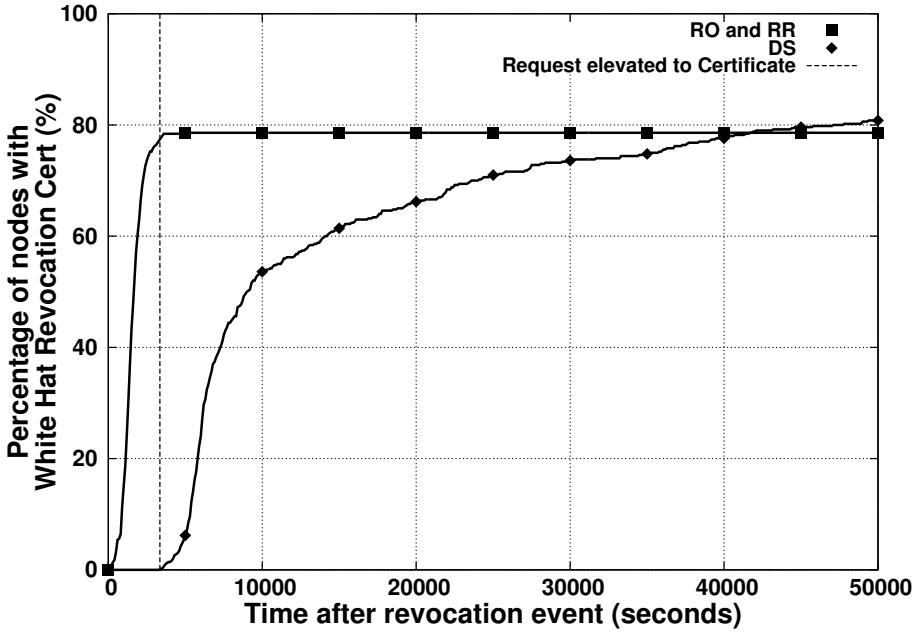


Figure J.2: Percentage of nodes with White Hat Revocation Certificates over time with 1 Black Hat Node.

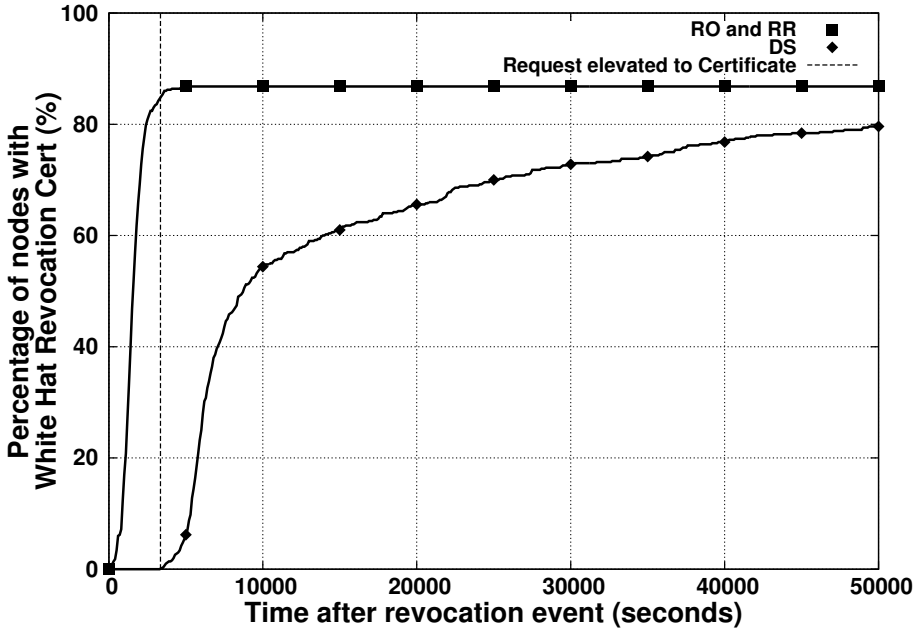


(a) Single Attack

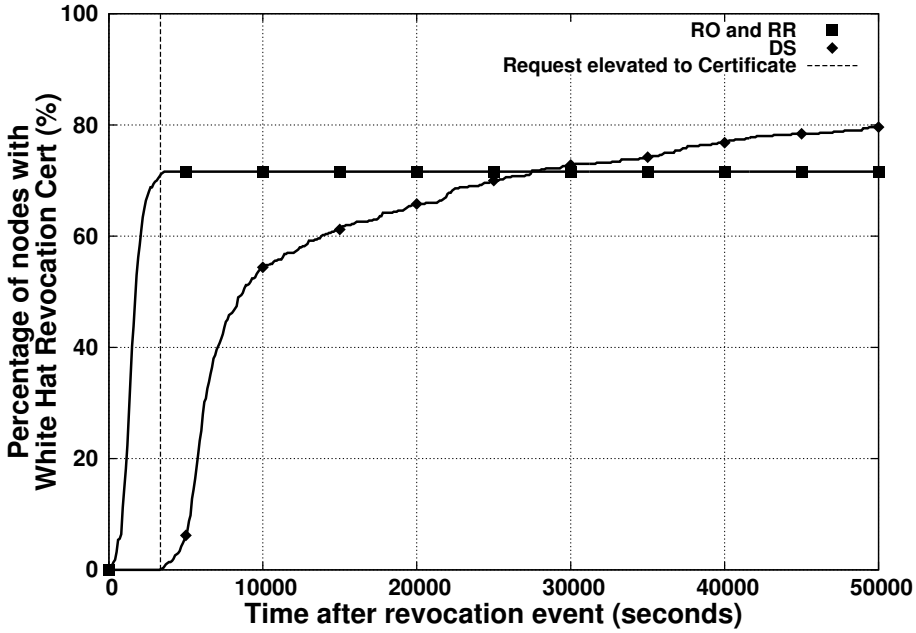


(b) Multiplying Attack

Figure J.3: Percentage of nodes with White Hat Revocation Certificates over time with 1% Black Hat Nodes.

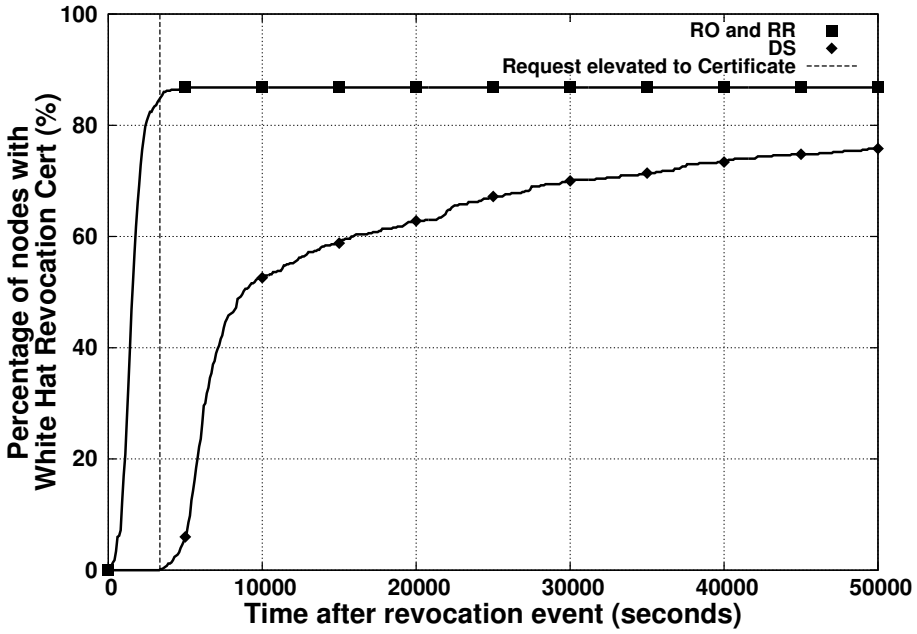


(a) Single Attack

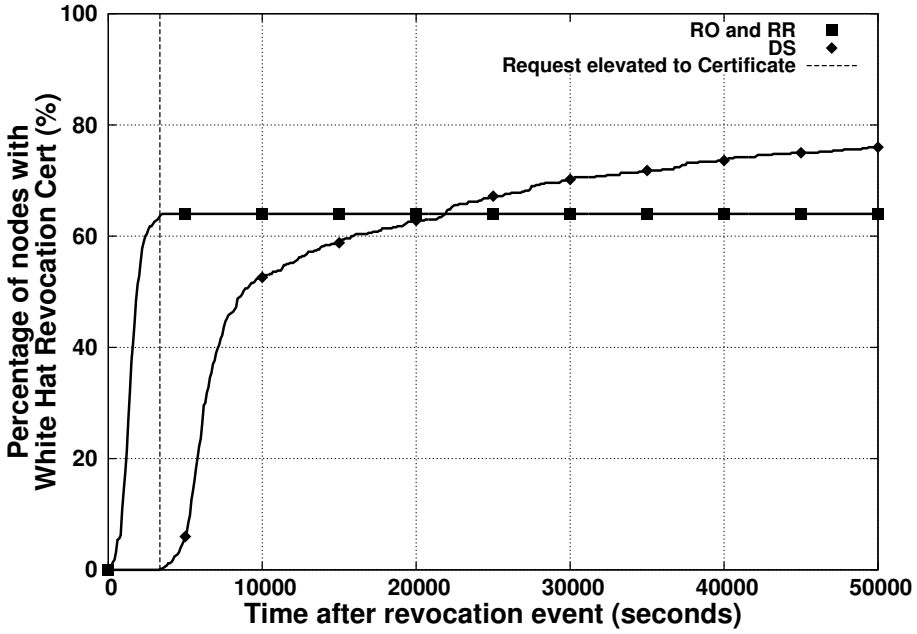


(b) Multiplying Attack

Figure J.4: Percentage of nodes with White Hat Revocation Certificates over time with 5% Black Hat Nodes.

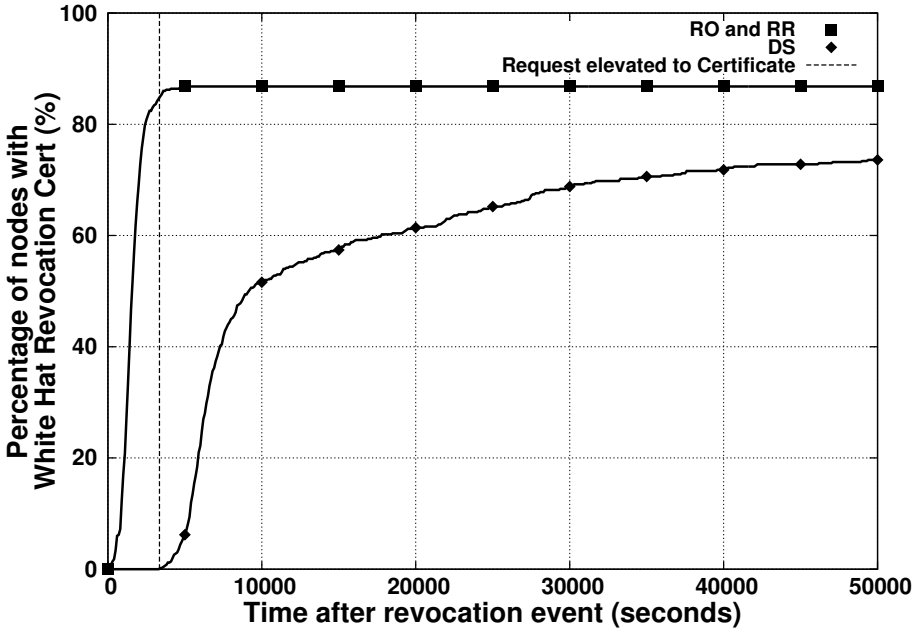


(a) Single Attack

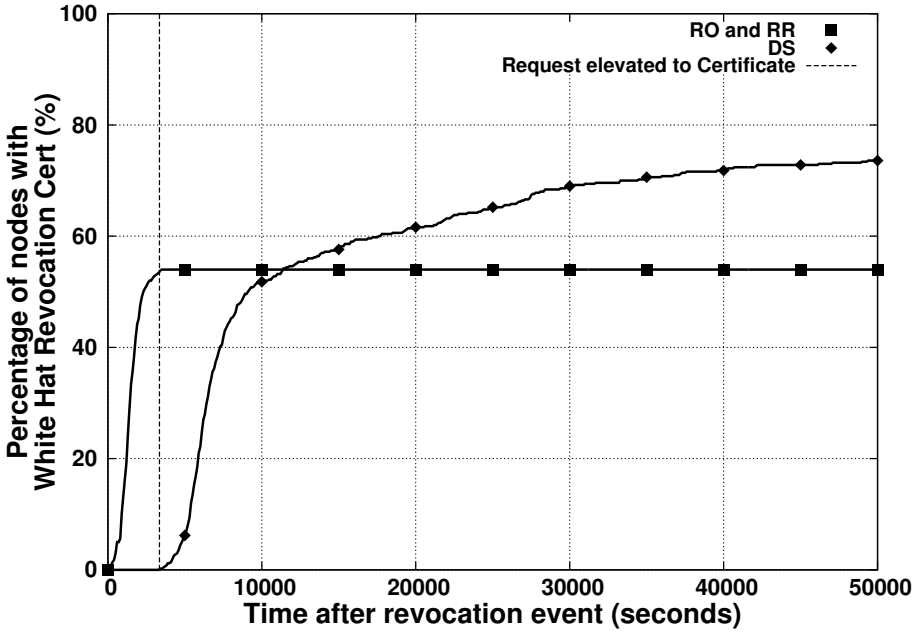


(b) Multiplying Attack

Figure J.5: Percentage of nodes with White Hat Revocation Certificates over time with 10% Black Hat Nodes.

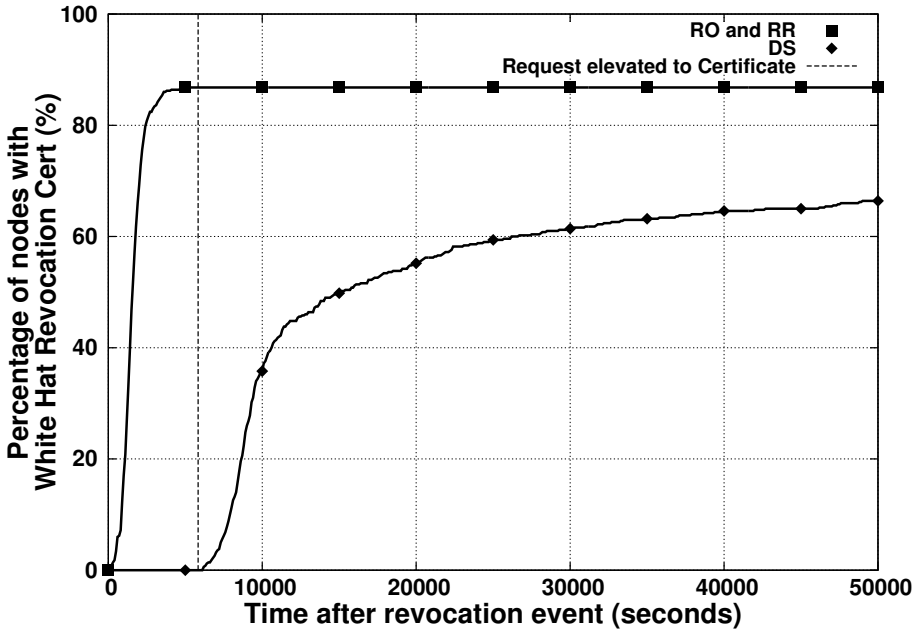


(a) Single Attack

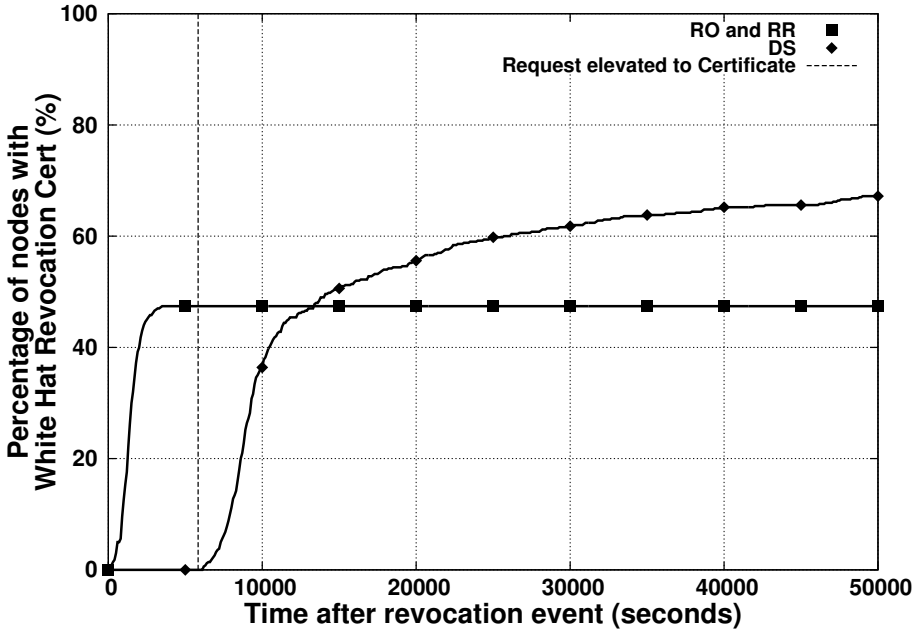


(b) Multiplying Attack

Figure J.6: Percentage of nodes with White Hat Revocation Certificates over time with 15% Black Hat Nodes.

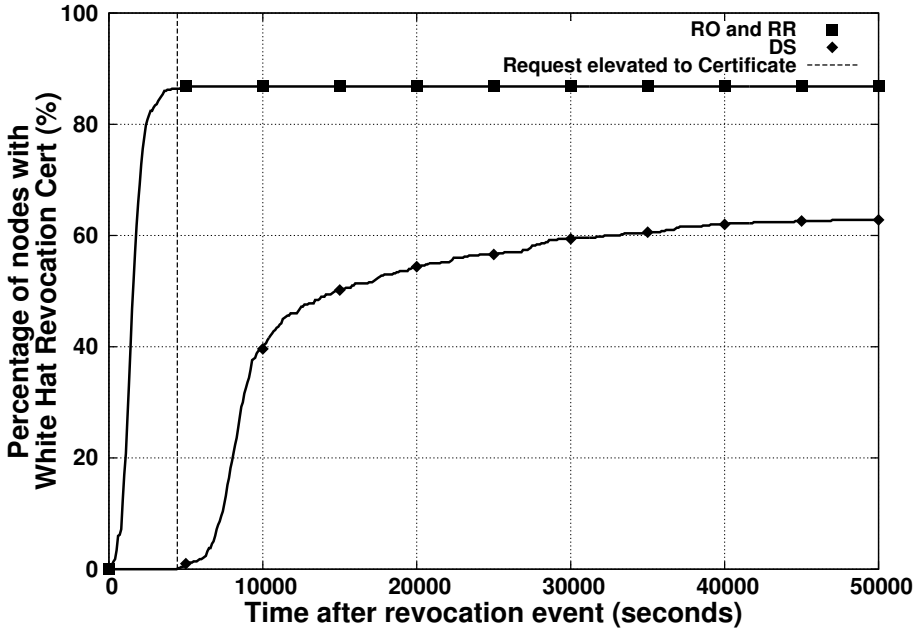


(a) Single Attack

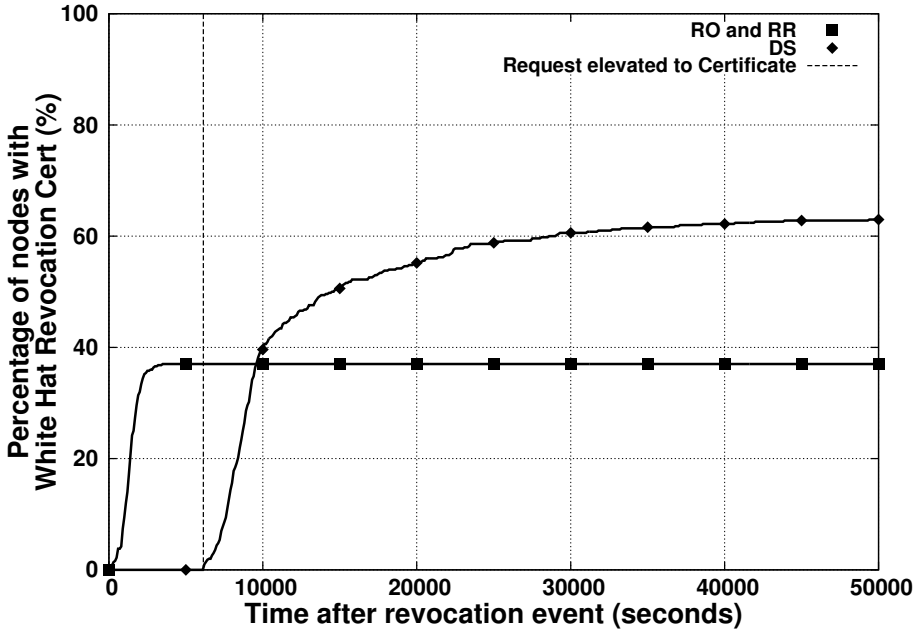


(b) Multiplying Attack

Figure J.7: Percentage of nodes with White Hat Revocation Certificates over time with 20% Black Hat Nodes.

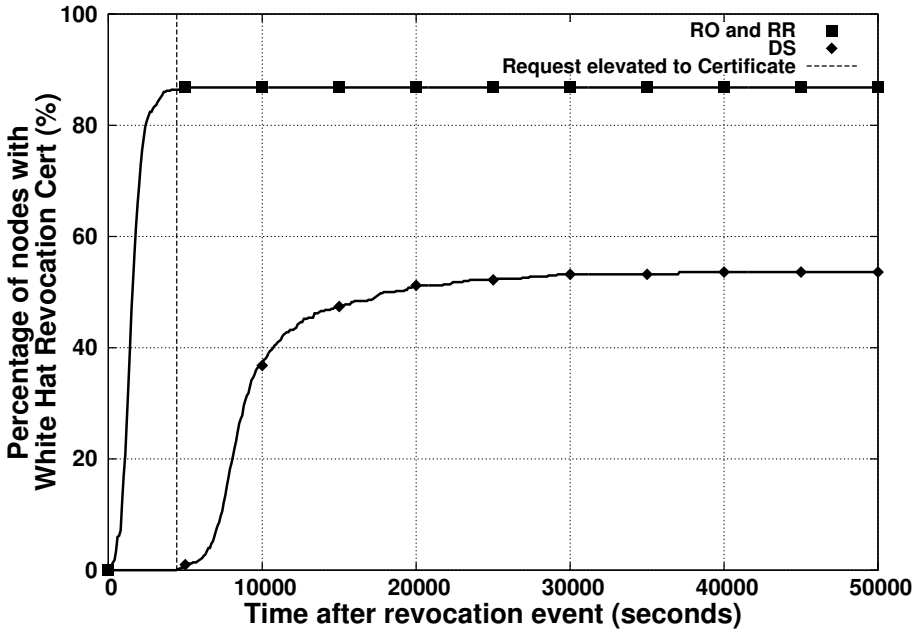


(a) Single Attack

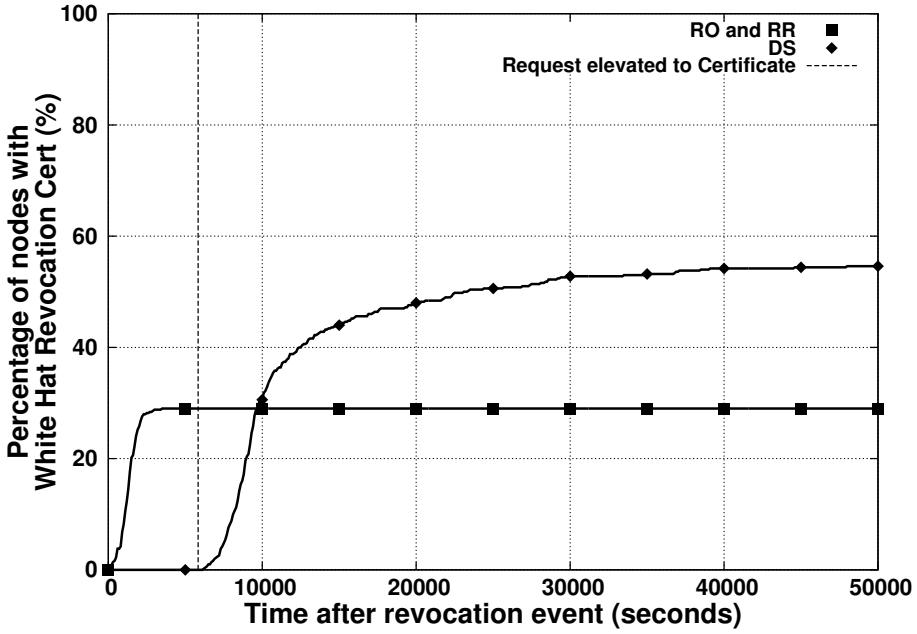


(b) Multiplying Attack

Figure J.8: Percentage of nodes with White Hat Revocation Certificates over time with 25% Black Hat Nodes.



(a) Single Attack



(b) Multiplying Attack

Figure J.9: Percentage of nodes with White Hat Revocation Certificates over time with 30% Black Hat Nodes.

Appendix K

Spoofed Revocation Certificates Results

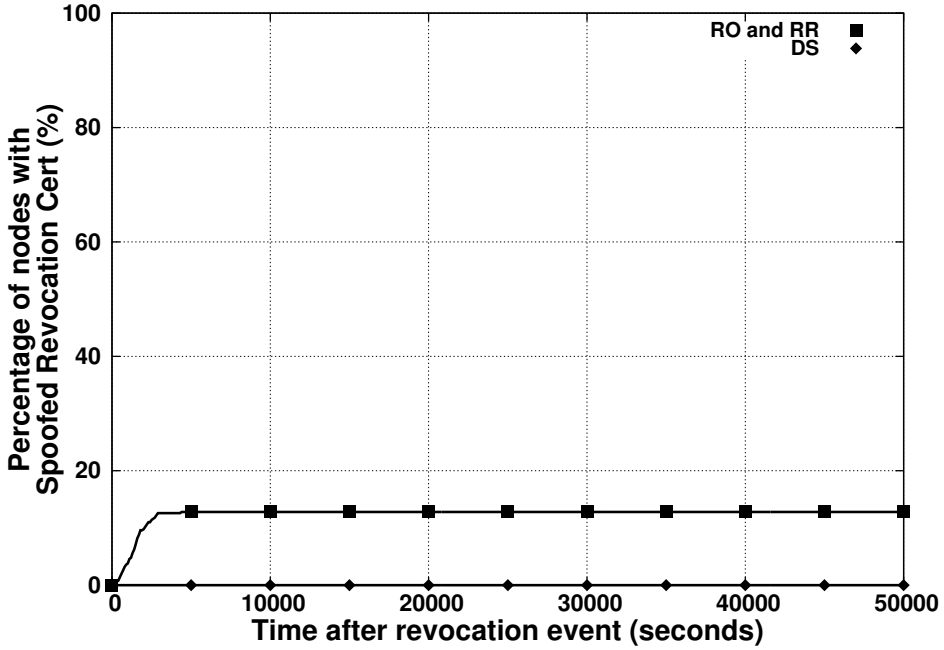
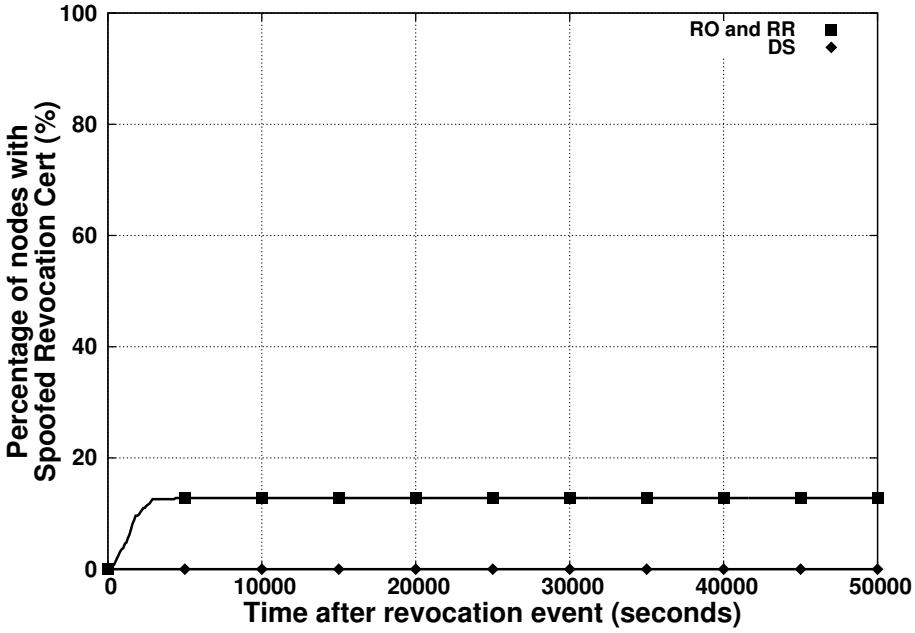
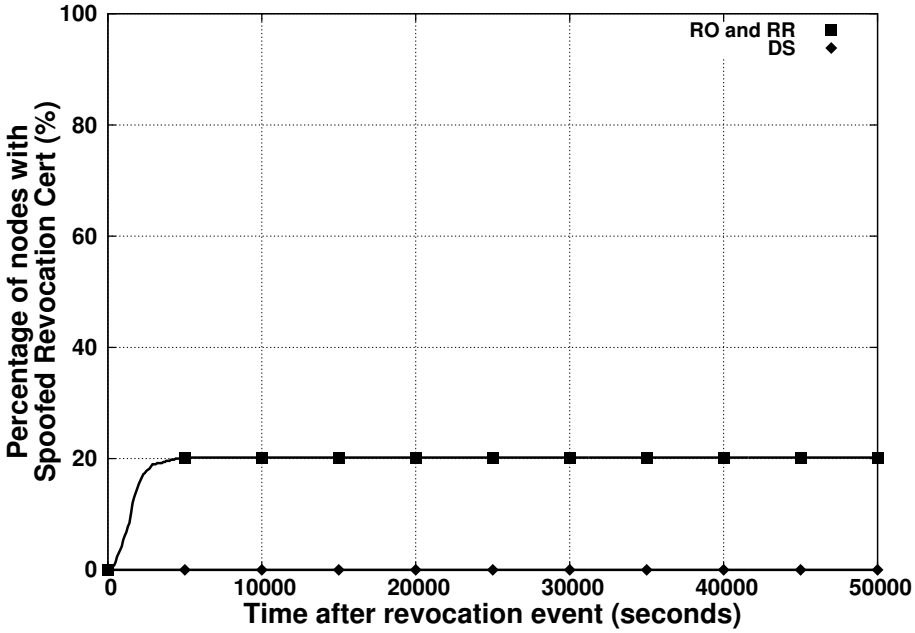


Figure K.1: Percentage of nodes with Spoofed Revocation Certificates over time with 1 Black Hat Node.

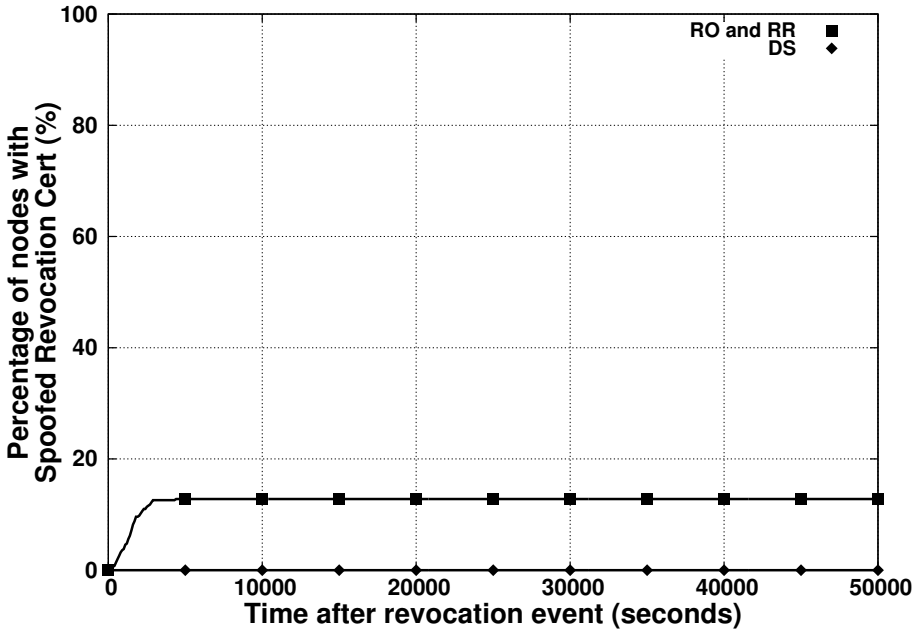


(a) Single Attack

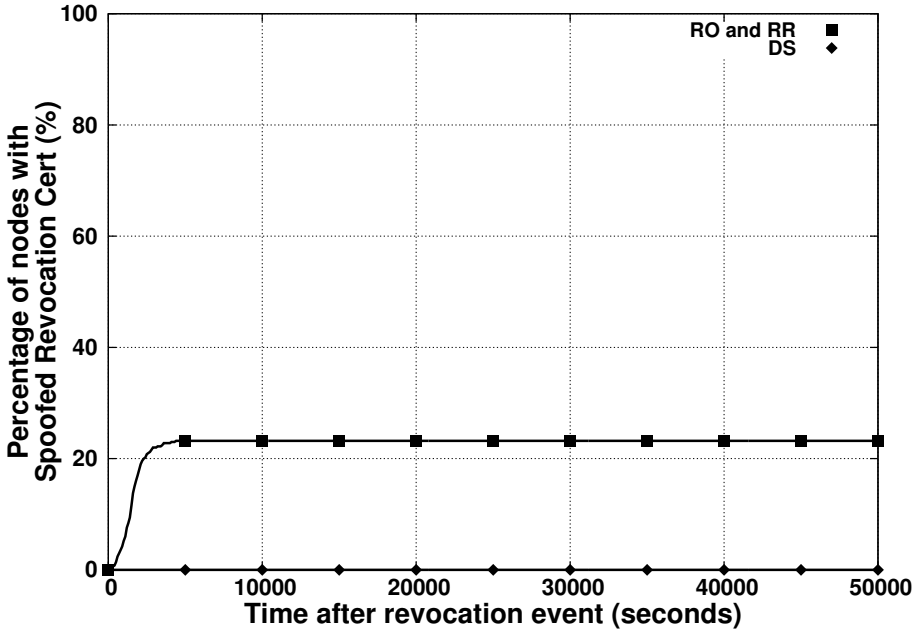


(b) Multiplying Attack

Figure K.2: Percentage of nodes with Spoofed Revocation Certificates over time with 1% Black Hat Nodes.

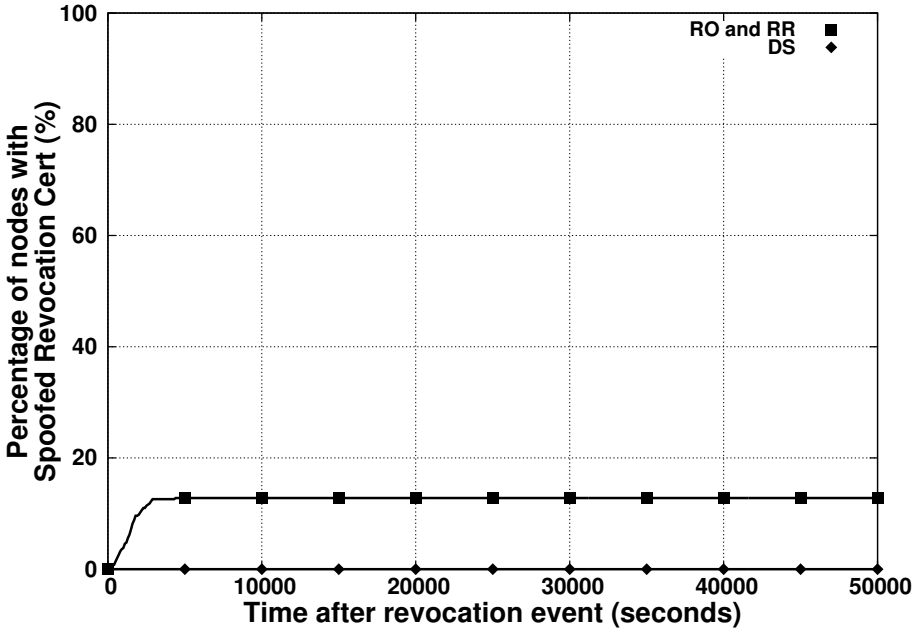


(a) Single Attack

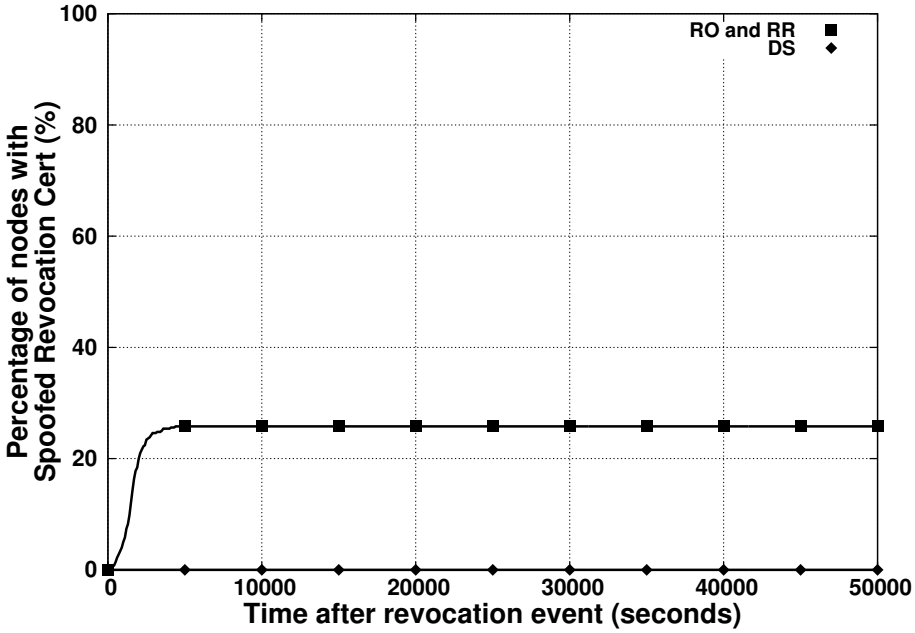


(b) Multiplying Attack

Figure K.3: Percentage of nodes with Spoofed Revocation Certificates over time with 5% Black Hat Nodes.

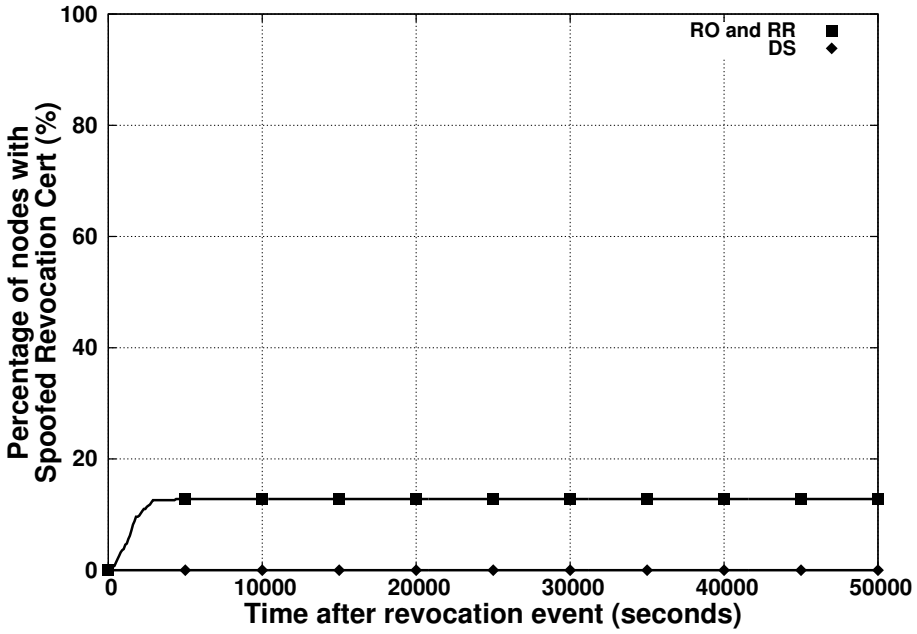


(a) Single Attack

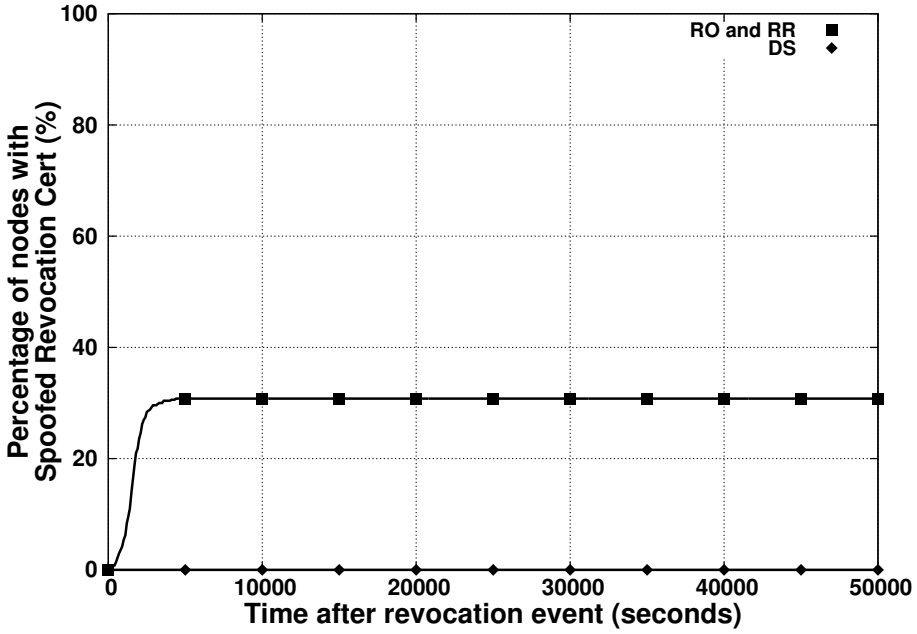


(b) Multiplying Attack

Figure K.4: Percentage of nodes with Spoofed Revocation Certificates over time with 10% Black Hat Nodes.

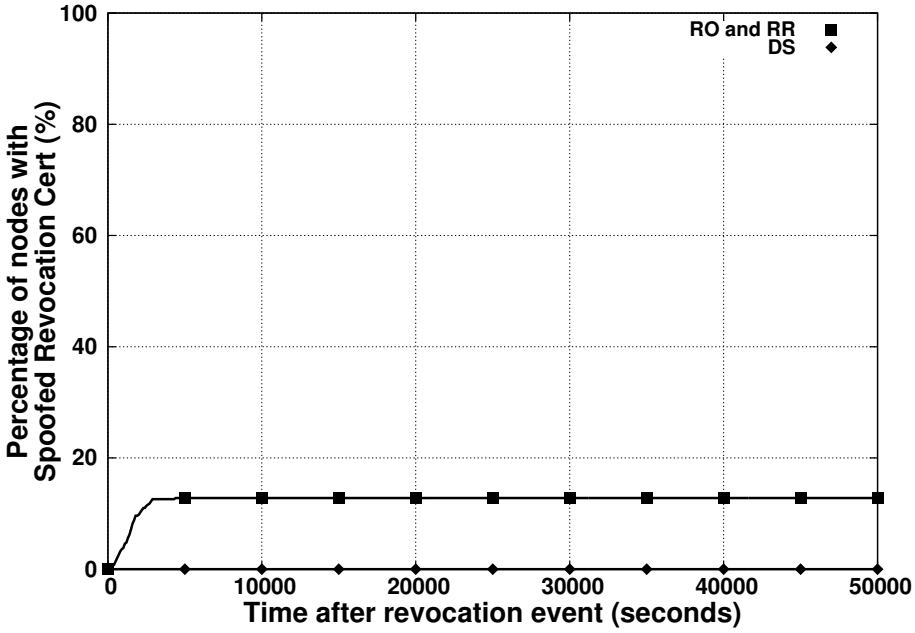


(a) Single Attack

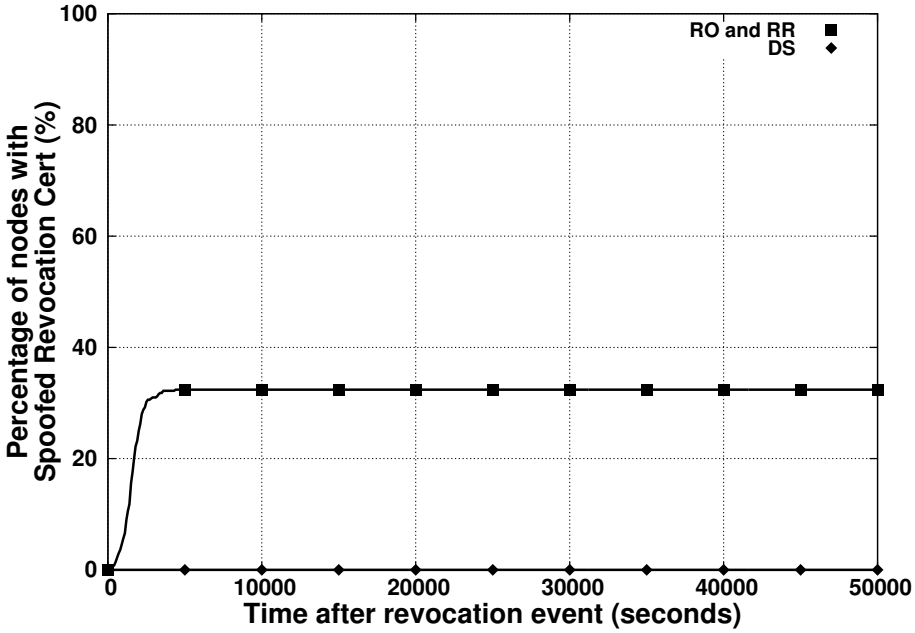


(b) Multiplying Attack

Figure K.5: Percentage of nodes with Spoofed Revocation Certificates over time with 15% Black Hat Nodes.

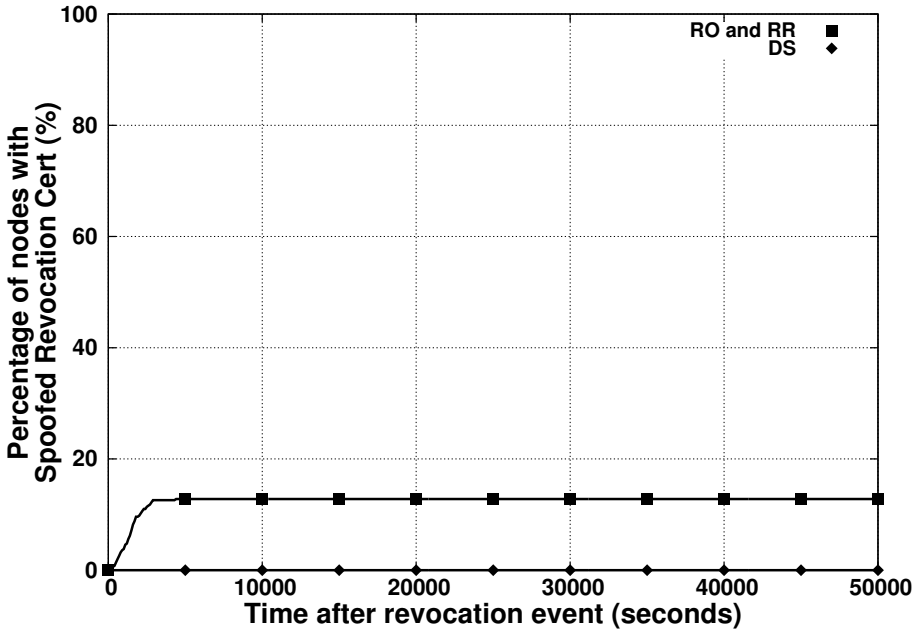


(a) Single Attack

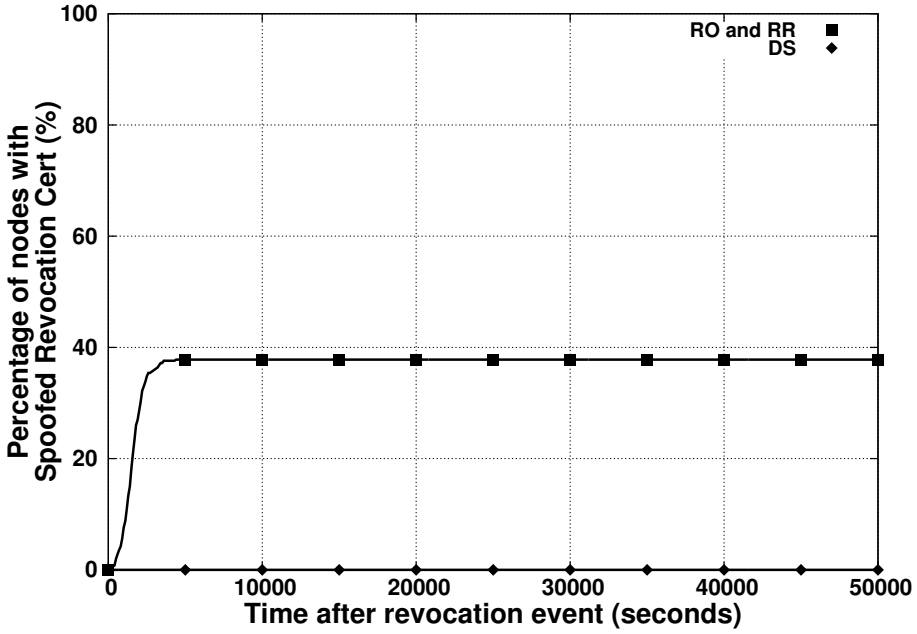


(b) Multiplying Attack

Figure K.6: Percentage of nodes with Spoofed Revocation Certificates over time with 20% Black Hat Nodes.

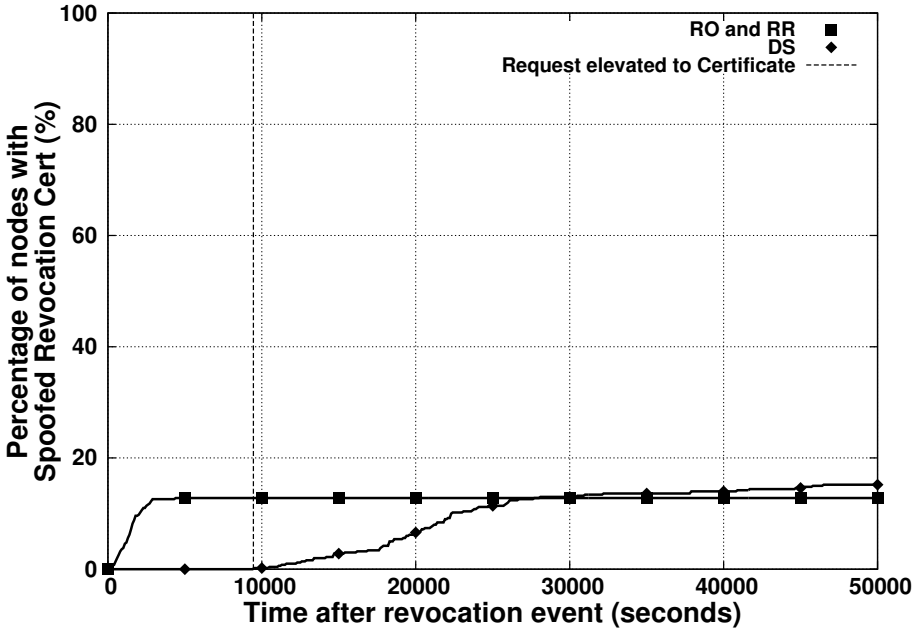


(a) Single Attack

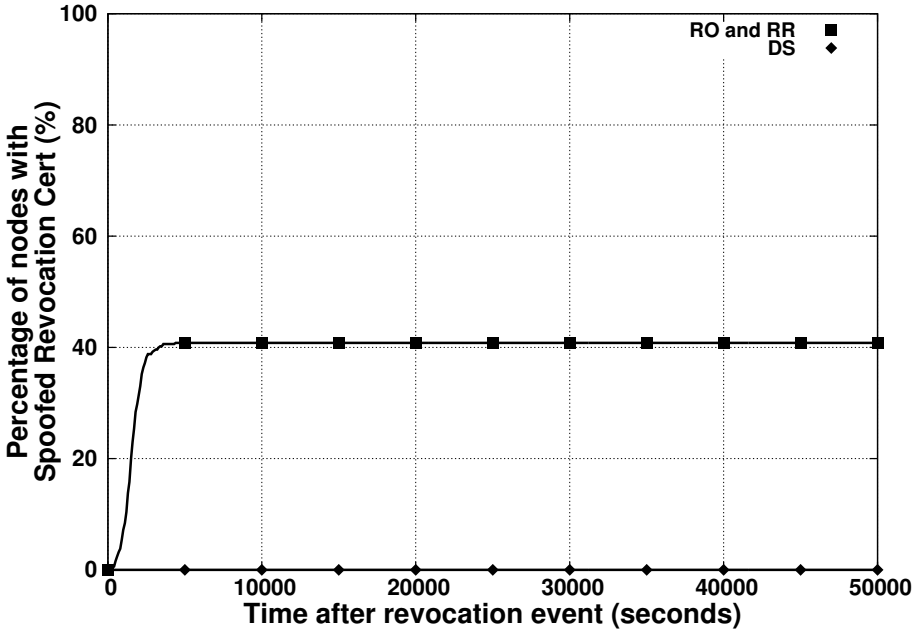


(b) Multiplying Attack

Figure K.7: Percentage of nodes with Spoofed Revocation Certificates over time with 25% Black Hat Nodes.



(a) Single Attack



(b) Multiplying Attack

Figure K.8: Percentage of nodes with Spoofed Revocation Certificates over time with 30% Black Hat Nodes.

Bibliography

[1] A. Abdul-Rahman and S. Hailes, "Supporting trust in virtual communities," in *System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on*, Jan. 2000, p. 9 pp. vol.1.

[2] A. Abdul-Rahman and S. Hailes, "A distributed trust model," in *Proceedings of the 1997 workshop on New security paradigms*, ser. NSPW '97. New York, NY, USA: ACM, 1997, pp. 48–60.

[3] A. F. A. Abidin and M. Kolberg, "Establishing trust in VANETs using information from social networks," in *PGNET 2013 Proceedings of the 14th Annual Postgraduate Symposium on the Convergence of Telecommunications*, M. Merasti and O. Arsuelma'atti, Eds., June 2013.

[4] G. Arboit, C. Crépeau, C. R. Davis, and M. Maheswaran, "A localized certificate revocation scheme for mobile ad hoc networks," *Ad Hoc Networks*, vol. 6, no. 1, pp. 17 – 31, 2008.

[5] E. Ayday, H. Lee, and F. Fekri, "Trust management and adversary detection for delay tolerant networks," in *MILITARY COMMUNICATIONS CONFERENCE, 2010 - MILCOM 2010*, Oct. 2010, pp. 1788–1793.

[6] E. Ayday and F. Fekri, "An iterative algorithm for trust management and adversary detection for delay-tolerant networks," *Mobile Computing, IEEE Transactions on*, vol. 11, no. 9, pp. 1514–1531, Sept. 2012.

[7] S. Basagni, K. Herrin, D. Bruschi, and E. Rosti, "Secure pebblenets," in *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, ser. MobiHoc '01. New York, NY, USA: ACM, 2001, pp. 156–163.

[8] N. Bhutta, G. Ansa, E. Johnson, N. Ahmad, M. Alsiyabi, and H. Cruickshank, "Security analysis for delay/disruption tolerant satellite and sensor networks," in *Satellite and Space Communications, 2009. IWSSC 2009. International Workshop on*, Sept. 2009, pp. 385 –389.

[9] N. Bissmeyer, H. Stubing, E. Schoch, S. Gotz, J. P. Stotz, and B. Lonc, "A generic public key infrastructure for securing car-to-x communication," in *18th ITS World Congress*, 2011.

[10] Bluetooth Special Interest Group, "Bluetooth Specifications Core Version 2.1 + EDR," Internet: https://www.bluetooth.org, 2007.

[11] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Advances in Cryptology - CRYPTO 2001*, ser. Lecture Notes in Computer Science, J. Kilian, Ed. Springer Berlin Heidelberg, 2001, vol. 2139, pp. 213–229.

[12] J. Burgess, G. D. Bissias, M. D. Corner, and B. N. Levine, "Surviving attacks on disruption-tolerant networks without authentication," in *Proceedings of the 8th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, ser. MobiHoc '07. New York, NY, USA: ACM, 2007, pp. 61–70.

[13] P. Caballero-Gil and C. Hernández-Goya, "Efficient public key certificate management for mobile ad hoc networks," *EURASIP J. Wirel. Commun. Netw.*, vol. 2011, pp. 18:1–18:11, Jan. 2011.

[14] J. Callas, L. Donnerhacke, H. Finney, D. Shaw, and R. Thayer, "OpenPGP message format," RFC Editor, RFC 4880, Nov. 2007.

[15] S. A. Camtepe and B. Yener, "Combinatorial design of key distribution mechanisms for wireless sensor networks," in *Computer Security - ESORICS 2004*, ser. Lecture Notes in Computer Science, P. Samarati, P. Ryan, D. Gollmann, and R. Molva, Eds. Springer Berlin Heidelberg, 2004, vol. 3193, pp. 293–308.

[16] S. A. Camtepe and B. Yener, "Key distribution mechanisms for wireless sensor networks: a survey," *Rensselaer Polytechnic Institute, Troy, New York, Technical Report*, pp. 05–07, 2005.

[17] Canonical, "Ubuntu Wiki - GPG Migration," Internet: https://wiki.ubuntu.com/SecurityTeam/GPGMigration, 2015.

[18] S. Capkun, L. Buttyan, and J.-P. Hubaux, "Self-organized public-key management for mobile ad hoc networks," *Mobile Computing, IEEE Transactions on*, vol. 2, no. 1, pp. 52 – 64, Jan.-Mar. 2003.

[19] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *Security and Privacy, 2003. Proceedings. 2003 Symposium on*, May 2003, pp. 197 – 213.

[20] C. Chen, J. Zhang, R. Cohen, and P.-H. Ho, "A trust modeling framework for message propagation and evaluation in VANETs," in *Information Technology Convergence and Services (ITCS), 2010 2nd International Conference on*, Aug. 2010, pp. 1–8.

[21] I.-R. Chen, F. Bao, M. Chang, and J.-H. Cho, "Trust management for encounter-based routing in delay tolerant networks," in *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, Dec. 2010, pp. 1–6.

[22] I.-R. Chen, F. Bao, M. Chang, and J.-H. Cho, "Dynamic trust management for delay tolerant networks and its application to secure routing," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 25, no. 5, pp. 1200–1210, May 2014.

[23] M. C. Chuah and P. Yang, "A message ferrying scheme with differentiated services," in *Military Communications Conference, 2005. MILCOM 2005. IEEE*, Oct. 2005, pp. 1521–1527 Vol. 3.

[24] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "Internet X. 509 public key infrastructure certificate and CRL profile," RFC Editor, RFC 5280, May 2008.

[25] M. Copeland, J. Grahn, and D. A. Wheeler, "The GNU Privacy Handbook," Internet: https://www.gnupg.org/gph/en/manual.html, 1999.

[26] C. Crépeau and C. R. Davis, "A certificate revocation scheme for wireless ad hoc networks," in *Proceedings of the 1st ACM Workshop on Security*

of Ad Hoc and Sensor Networks, ser. SASN '03. New York, NY, USA: ACM, 2003, pp. 54–61.

[27] Debian Project, "Creating a new GPG key," Internet: http://keyring.debian.org/creating-key.html, 2012.

[28] K. E. Defrawy, J. Solis, and G. Tsudik, "Leveraging social contacts for message confidentiality in delay tolerant networks," in *Proceedings of the 2009 33rd Annual IEEE International Computer Software and Applications Conference - Volume 01*. Washington, DC, USA: IEEE Computer Society, 2009, pp. 271–279.

[29] R. Di Pietro, L. V. Mancini, and A. Mei, "Random key-assignment for secure wireless sensor networks," in *Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Sensor Networks*, ser. SASN '03. New York, NY, USA: ACM, 2003, pp. 62–71.

[30] C. I. Djamaludin, "Traffic Djam simulator," Internet: https://github.com/djamaludin/traffic-djam-simulator, 2014.

[31] C. Djamaludin, E. Foo, and P. Corke, "Establishing initial trust in autonomous delay tolerant networks without centralised PKI," *Computers & Security*, vol. 39, Part B, pp. 299 – 314, 2013.

[32] D. Dolev and A. Yao, "On the security of public key protocols," *Information Theory, IEEE Transactions on*, vol. 29, no. 2, pp. 198 – 208, Mar. 1983.

[33] F. Dotzer, L. Fischer, and P. Magiera, "VARS: a vehicle ad-hoc network reputation system," in *World of Wireless Mobile and Multimedia Networks, 2005. WoWMoM 2005. Sixth IEEE International Symposium on a*, June 2005, pp. 454–456.

[34] H. Dubois-Ferriere, M. Grossglauser, and M. Vetterli, "Age matters: Efficient route discovery in mobile ad hoc networks using encounter ages," in *Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking &Amp; Computing*, ser. MobiHoc '03. New York, NY, USA: ACM, 2003, pp. 257–266.

[35] R. Dunbar, "Neocortex size as a constraint on group size in primates," *Journal of Human Evolution*, vol. 22, no. 6, pp. 469 – 493, 1992.

[36] P. Eckersley and J. Burns, "The EFF SSL Observatory," Internet: https://www.eff.org/observatory, 2010.

[37] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proceedings of the 9th ACM conference on Computer and communications security*, ser. CCS '02. New York, NY, USA: ACM, 2002, pp. 41–47.

[38] Exploratorium, "Cabspotting Project," Internet: http://cabspotting.org/, 2006.

[39] R. Falcone and C. Castelfranchi, "Social trust: A cognitive approach," in *Trust and Deception in Virtual Societies*, C. Castelfranchi and Y.-H. Tan, Eds. Springer Netherlands, 2001, pp. 55–90.

[40] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, ser. SIGCOMM '03. New York, NY, USA: ACM, 2003, pp. 27–34.

[41] D. Fisher, "Final report on DigiNotar hack shows total compromise of CA servers," Internet: https://threatpost.com/final-report-diginotar-hack-shows-total-compromise-ca-servers-103112/77170/, Oct. 2012.

[42] K. Fiskerstrand, "SKS Keyservers," Internet: https://sks-keyservers.net/i/, 2015.

[43] D. Gambetta *et al.*, "Can we trust trust," *Trust: Making and breaking cooperative relations*, vol. 2000, pp. 213–237, 2000.

[44] C. Ganan, J. Mata-Diaz, J. Munoz, O. Esparza, and J. Alins, "A model for revocation forecasting in public-key infrastructures," *Knowledge and Information Systems*, pp. 1–21, 2014.

[45] W. Gao, Q. Li, B. Zhao, and G. Cao, "Multicasting in delay tolerant networks: A social network perspective," in *Proceedings of the Tenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, ser. MobiHoc '09. New York, NY, USA: ACM, 2009, pp. 299–308.

[46] M. Ge, K.-Y. Lam, J. Li, and S.-L. Chung, "Ubiquitous and secure certificate service for mobile ad hoc network," in *Embedded and Ubiquitous Computing, 2008. EUC '08. IEEE/IFIP International Conference on*, vol. 2, Dec. 2008, pp. 312–317.

[47] S. Geetha and S. Sujatha, "Increase the performance and enhancing secure authenticated multipath encrypted protocol in MANET," in *Computational Intelligence and Computing Research (ICCIC), 2010 IEEE International Conference on*, Dec. 2010, pp. 1–7.

[48] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Robust threshold DSS signatures," in *Advances in Cryptology - EUROCRYPT '96*, ser. Lecture Notes in Computer Science, U. Maurer, Ed. Springer Berlin Heidelberg, 1996, vol. 1070, pp. 354–371.

[49] GlobalSign, "Certificate Revocation List," Internet: http://crl.globalsign.com/gs/gsorganizationvalg2.crl, July 2015.

[50] P. Golle, D. Greene, and J. Staddon, "Detecting and correcting malicious data in VANETs," in *Proceedings of the 1st ACM International Workshop on Vehicular Ad Hoc Networks*, ser. VANET '04. New York, NY, USA: ACM, 2004, pp. 29–37.

[51] B. Goncalves, N. Perra, and A. Vespignani, "Modeling users' activity on Twitter networks: Validation of Dunbar's number," *PLoS ONE*, vol. 6, no. 8, p. e22656, 08 2011.

[52] M. Grossglauser and M. Vetterli, "Locating nodes with EASE: last encounter routing in ad hoc networks through mobility diffusion," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, vol. 3, Mar. 2003, pp. 1954–1964 vol.3.

[53] H. Guo, J. Li, and Y. Qian, "HoP-DTN: Modeling and evaluation of homing-pigeon-based delay-tolerant networks," *Vehicular Technology, IEEE Transactions on*, vol. 59, no. 2, pp. 857–868, Feb. 2010.

[54] P. Gutmann, "PKI: it's not dead, just resting," *Computer*, vol. 35, no. 8, pp. 41–49, Aug. 2002.

[55] J. J. Haas, Y.-C. Hu, and K. P. Laberteaux, "Design and analysis of a lightweight certificate revocation mechanism for VANET," in *Proceedings of the Sixth ACM International Workshop on VehiculAr InterNETworking*, ser. VANET '09. New York, NY, USA: ACM, 2009, pp. 89–98.

[56] J. Haas, Y.-C. Hu, and K. Laberteaux, "Efficient certificate revocation list organization and distribution," *Selected Areas in Communications, IEEE Journal on*, vol. 29, no. 3, pp. 595–604, Mar. 2011.

[57] K. Hamouid and K. Adi, "Efficient certificateless web-of-trust model for public-key authentication in MANET," *Computer Communications*, vol. 63, pp. 24 – 39, 2015.

[58] A. A. Hanbali, P. Nain, and E. Altman, "Performance of ad hoc networks with two-hop relay routing and limited packet lifetime," in *Proceedings of the 1st International Conference on Performance Evaluation Methodolgies and Tools*, ser. valuetools '06. New York, NY, USA: ACM, 2006.

[59] Y. Hao, Y. Cheng, and K. Ren, "Distributed key management with protection against rsu compromise in group signature based VANETs," in *Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE*, Nov. 2008, pp. 1–5.

[60] Y. Hao, Y. Cheng, C. Zhou, and W. Song, "A distributed key management framework with cooperative message authentication in VANETs," *Selected Areas in Communications, IEEE Journal on*, vol. 29, no. 3, pp. 616–629, Mar. 2011.

[61] J. Harding, G. Powell, R. Yoon, J. Fikentscher, C. Doyle, D. Sade, M. Lukuc, J. Simons, and J. Wang, "Vehicle-to-vehicle communications: Readiness of V2V technology for application," United States Department of Transportation, National Highway Traffic Safety Administration, Washington, DC, Tech. Rep. DOT HS 812 014, Aug. 2014.

[62] K. Harras, K. Almeroth, and E. Belding-Royer, "Delay tolerant mobile networks (DTMNs): Controlled flooding in sparse mobile networks," in *NETWORKING 2005. Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications Systems*, ser. Lecture Notes in Computer

Science, R. Boutaba, K. Almeroth, R. Puigjaner, S. Shen, and J. Black, Eds. Springer Berlin Heidelberg, 2005, vol. 3462, pp. 1180–1192.

[63] A. Hernando, D. Villuendas, C. Vesperinas, M. Abad, and A. Plastino, "Unravelling the size distribution of social groups with information theory in complex networks," *The European Physical Journal B*, vol. 76, no. 1, pp. 87–97, 2010.

[64] K. Hoeper and G. Gong, "Bootstrapping security in mobile ad hoc networks using identity-based schemes with key revocation," Centre for Applied Cryptographic Research (CACR) at the University of Waterloo, Tech. Rep. 4, 2006.

[65] K. Hoeper and G. Gong, "Monitoring-based key revocation schemes for mobile ad hoc networks: Design and security analysis," Centre for Applied Cryptographic Research (CACR) at the University of Waterloo, Tech. Rep. 9 2009-15, Mar. 2009.

[66] W. Hu, P. Corke, W. Shih, and L. Overs, "secFleck: A public key technology platform for wireless sensor networks," in *Wireless Sensor Networks*, ser. Lecture Notes in Computer Science, U. Roedig and C. Sreenan, Eds. Springer Berlin / Heidelberg, 2009, vol. 5432, pp. 296–311, 10.1007/978-3-642-00224-3-19.

[67] D. Huang, M. Mehta, D. Medhi, and L. Harn, "Location-aware key management scheme for wireless sensor networks," in *Proceedings of the 2Nd ACM Workshop on Security of Ad Hoc and Sensor Networks*, ser. SASN '04. New York, NY, USA: ACM, 2004, pp. 29–42.

[68] Z. Huang, S. Ruj, M. Cavenaghi, M. Stojmenovic, and A. Nayak, "A social network approach to trust management in VANETs," *Peer-to-Peer Networking and Applications*, vol. 7, no. 3, pp. 229–242, 2014.

[69] P. Hui, J. Crowcroft, and E. Yoneki, "BUBBLE rap: Social-based forwarding in delay-tolerant networks," *Mobile Computing, IEEE Transactions on*, vol. 10, no. 11, pp. 1576 –1589, Nov. 2011.

[70] I. Ibrahim, M. Ibrahim, and A. Allam, "A method for fast revocation of certificateless public key cryptography," in *Computer Engineering and Systems, The 2006 International Conference on*, Nov. 2006, pp. 250–253.

[71] IEEE, "IEEE standard for information technology–telecommunications and information exchange between systems local and metropolitan area networks–specific requirements part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications," *IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007)*, pp. 1–2793, Mar. 2012.

[72] Z. Jia, X. Lin, S.-H. Tan, L. Li, and Y. Yang, "Public key distribution scheme for delay tolerant networks based on two-channel cryptography," *Journal of Network and Computer Applications*, vol. 35, no. 3, pp. 905 – 913, 2012, special Issue on Trusted Computing and Communications.

[73] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," *Forbes*, vol. 353, no. 2, pp. 153–181, 1996.

[74] A. Josang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," *Decision Support Systems*, vol. 43, no. 2, pp. 618 – 644, 2007, emerging Issues in Collaborative Commerce.

[75] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The eigentrust algorithm for reputation management in P2P networks," in *Proceedings of the 12th international conference on World Wide Web*, ser. WWW '03. New York, NY, USA: ACM, 2003, pp. 640–651.

[76] A. Kate, G. M. Zaverucha, and U. Hengartner, "Anonymity and security in delay tolerant networks," in *Security and Privacy in Communications Networks and the Workshops, 2007. SecureComm 2007. Third International Conference on*, Sept. 2007, pp. 504 –513.

[77] Y. Kim, I. Kim, and C. Shim, "Towards a trust management for VANETs," in *Information Networking (ICOIN), 2014 International Conference on*, Feb. 2014, pp. 583–587.

[78] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of computation*, vol. 48, no. 177, pp. 203–209, 1987.

[79] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang, "Providing robust and ubiquitous security support for mobile ad hoc networks," in *2012 20th IEEE International Conference on Network Protocols (ICNP)*. IEEE Computer Society, 2001, pp. 0251–0251.

[80] N. Kumar, R. Iqbal, S. Misra, and J. J. Rodrigues, "An intelligent approach for building a secure decentralized public key infrastructure in VANET," *Journal of Computer and System Sciences*, vol. 81, no. 6, pp. 1042 – 1058, 2015.

[81] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Trans. Program. Lang. Syst.*, vol. 4, no. 3, pp. 382–401, July 1982.

[82] J. Leguay, T. Friedman, and V. Conan, "Evaluating mobility pattern space routing for DTNs," in *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, Apr. 2006, pp. 1–10.

[83] Q. Li, S. Zhu, and G. Cao, "Routing in socially selfish delay tolerant networks," in *INFOCOM, 2010 Proceedings IEEE*, Mar. 2010, pp. 1–9.

[84] R. Li, J. Li, P. Liu, and H.-H. Chen, "On-demand public-key management for mobile ad hoc networks," *Wireless Communications and Mobile Computing*, vol. 6, no. 3, pp. 295–306, 2006.

[85] Y. Li, "An overview of the DSRC/WAVE technology," in *Quality, Reliability, Security and Robustness in Heterogeneous Networks*, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, X. Zhang and D. Qiao, Eds. Springer Berlin Heidelberg, 2012, vol. 74, pp. 544–558.

[86] C. Liao, J. Chang, I. Lee, and K. Venkatasubramanian, "A trust model for vehicular network-based incident reports," in *Wireless Vehicular Communications (WiVeC), 2013 IEEE 5th International Symposium on*, June 2013, pp. 1–5.

[87] X. Lin, R. Lu, C. Zhang, H. Zhu, P.-H. Ho, and X. Shen, "Security in vehicular ad hoc networks," *Communications Magazine, IEEE*, vol. 46, no. 4, pp. 88 –95, Apr. 2008.

[88] A. Lindgren, A. Doria, and O. Schelén, "Probabilistic routing in intermittently connected networks," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 7, no. 3, pp. 19–20, July 2003.

[89] D. Liu and P. Ning, "Location-based pairwise key establishments for static sensor networks," in *Proceedings of the 1st ACM Workshop on Security*

of Ad Hoc and Sensor Networks, ser. SASN '03. New York, NY, USA: ACM, 2003, pp. 72–82.

[90] H. Luo, J. Kong, P. Zerfos, S. Lu, and L. Zhang, "URSA: Ubiquitous and robust access control for mobile ad hoc networks," *IEEE/ACM Trans. Netw.*, vol. 12, no. 6, pp. 1049–1063, Dec. 2004.

[91] H. Luo, P. Zerfos, J. Kong, S. Lu, and L. Zhang, "Self-securing ad hoc wireless networks," in *2012 IEEE Symposium on Computers and Communications (ISCC)*. IEEE Computer Society, 2002, pp. 567–567.

[92] S. Maity and R. Hansdah, "Self-organized public key management in MANETs with enhanced security and without certificate-chains," *Computer Networks*, vol. 65, pp. 183 – 211, 2014.

[93] D. Mall, K. Konate, and A.-S. Pathan, "SECRET: A secure and efficient certificate revocation scheme for mobile ad hoc networks," in *Biometrics and Security Technologies (ISBAST), 2014 International Symposium on*, Aug. 2014, pp. 137–143.

[94] D. Manchala, "Trust metrics, models and protocols for electronic commerce transactions," in *Distributed Computing Systems, 1998. Proceedings. 18th International Conference on*, May 1998, pp. 312 –321.

[95] G. F. Marias, K. Papapanagiotou, and P. Georgiadis, "ADOPT. a distributed OCSP for trust establishment in MANETs," in *Wireless Conference 2005 - Next Generation Wireless and Mobile Communications and Services (European Wireless), 11th European*, Apr. 2005, pp. 1–7.

[96] Massachusetts Institute of Technology, "MIT PGP Public Key Server," Internet: https://pgp.mit.edu/, 2015.

[97] D. H. McKnight and N. L. Chervany, "The meanings of trust," MISRC Working Paper Series 96-04, University of Minnesota, Management Information Systems Research Center, Tech. Rep., 1996.

[98] V. Miller, "Use of elliptic curves in cryptography," in *Advances in Cryptology - CRYPTO '85 Proceedings*, ser. Lecture Notes in Computer Science, H. Williams, Ed. Springer Berlin Heidelberg, 1986, vol. 218, pp. 417–426.

[99] S. Misra, S. Goswami, C. Taneja, and A. Mukherjee, "Design and implementation analysis of a public key infrastructure-enabled security framework for ZigBee sensor networks," *International Journal of Communication Systems*, 2014.

[100] T. Moore, J. Clulow, S. Nagaraja, and R. Anderson, "New strategies for revocation in ad-hoc networks," in *Security and Privacy in Ad-hoc and Sensor Networks*, ser. Lecture Notes in Computer Science, F. Stajano, C. Meadows, S. Capkun, and T. Moore, Eds. Springer Berlin Heidelberg, 2007, vol. 4572, pp. 232–246.

[101] M. Naor and K. Nissim, "Certificate revocation and certificate update," *Selected Areas in Communications, IEEE Journal on*, vol. 18, no. 4, pp. 561–570, Apr. 2000.

[102] M. Narasimha, G. Tsudik, and J. H. Yi, "On the utility of distributed cryptography in P2P and MANETs: the case of membership control," in *Network Protocols, 2003. Proceedings. 11th IEEE International Conference on*, Nov. 2003, pp. 336–345.

[103] E.-H. Ngai and M. Lyu, "Trust- and clustering-based authentication services in mobile ad hoc networks," in *Distributed Computing Systems Workshops, 2004. Proceedings. 24th International Conference on*, Mar. 2004, pp. 582–587.

[104] M. Omar, Y. Challal, and A. Bouabdallah, "Reliable and fully distributed trust model for mobile ad hoc networks," *Computers & Security*, vol. 28, no. 3-4, pp. 199 – 214, 2009.

[105] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: Bringing order to the web," Internet: http://ilpubs.stanford.edu:8090/422, 1999.

[106] B. Parno, A. Perrig, and V. Gligor, "Distributed detection of node replication attacks in sensor networks," in *Security and Privacy, 2005 IEEE Symposium on*, May 2005, pp. 49–63.

[107] A. Pathan, H.-W. Lee, and C. S. Hong, "Security in wireless sensor networks: issues and challenges," in *Advanced Communication Technology,*

2006. ICACT 2006. The 8th International Conference, vol. 2, Feb. 2006, pp. 6 pp.–1048.

[108] A. Patwardhan, A. Joshi, T. Finin, and Y. Yesha, "A data intensive reputation management scheme for vehicular ad hoc networks," in *Mobile and Ubiquitous Systems: Networking Services, 2006 Third Annual International Conference on*, July 2006, pp. 1–8.

[109] R. Perlman, "An overview of PKI trust models," *Network, IEEE*, vol. 13, no. 6, pp. 38 –43, Nov./Dec. 1999.

[110] M. Piorkowski, N. Sarafijanovic-Djukic, and M. Grossglauser, "CRAWDAD data set epfl/mobility (v. 2009-02-24)," Internet: http://crawdad.org/epfl/mobility/, Feb. 2009.

[111] M. Piorkowski, N. Sarafijanovoc-Djukic, and M. Grossglauser, "A parsimonious model of mobile partitioned networks with clustering," in *The First International Conference on COMmunication Systems and NETworkS (COMSNETS)*, Jan. 2009.

[112] D. Purves, E. M. Brannon, R. Cabeza, S. A. Huettel, K. S. LaBar, M. L. Platt, and M. G. Woldorff, *Principles of cognitive neuroscience*. Sinauer Associates Sunderland, MA, 2008, vol. 83, no. 3.

[113] A. Rachedi and A. Benslimane, "Trust and mobility-based clustering algorithm for secure mobile ad hoc networks," in *Systems and Networks Communications, 2006. ICSNC '06. International Conference on*, Oct. 2006, pp. 72–72.

[114] L. Rasmusson, A. Rasmusson, and S. Janson, "Reactive security and social control," in *Proceedings, 19' National Information System Security Conference*, Swedish Institute of Computer Science; Untrusted Code, 1996.

[115] M. Raya, P. Papadimitratos, I. Aad, D. Jungels, and J.-P. Hubaux, "Eviction of misbehaving and faulty nodes in vehicular networks," *Selected Areas in Communications, IEEE Journal on*, vol. 25, no. 8, pp. 1557–1568, Oct. 2007.

[116] M. Raya, P. Papadimitratos, V. Gligor, and J.-P. Hubaux, "On data-centric trust establishment in ephemeral ad hoc networks," in *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, Apr. 2008.

[117] M. Raya, D. Jungels, P. Papadimitratos, I. Aad, and J.-P. Hubaux, "Certificate revocation in vehicular networks," No. LCA-REPORT-2006-006, Tech. Rep., 2006.

[118] P. Resnick and R. Zeckhauser, *Trust Among Strangers in Internet Transactions: Empirical Analysis of eBay's Reputation System.* Elsevier Science, Nov. 2002.

[119] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, pp. 120–126, Feb. 1978.

[120] R.J.D'Souza and J. Jose, "Routing approaches in delay tolerant networks: A survey," *International Journal of Computer Applications*, vol. 1, no. 17, pp. 8–14, Oct. 2010.

[121] S. Santesson, M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams, "X. 509 internet public key infrastructure online certificate status protocol-OCSP," RFC Editor, RFC 6960, June 2013.

[122] L. Sassaman and P. Zimmermann, "Efficient Group Key Signing Method," Internet: http://www.cryptnet.net/mirrors/docs/zimmermann-sassaman.txt, 2005.

[123] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, pp. 612–613, Nov. 1979.

[124] D. Solo, R. Housley, and W. Ford, "Internet X. 509 public key infrastructure certificate and certificate revocation list (CRL) profile," *Internet Engineering Task Force*, 2002.

[125] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and wait: An efficient routing scheme for intermittently connected mobile networks," in *Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-tolerant Networking*, ser. WDTN '05. New York, NY, USA: ACM, 2005, pp. 252–259.

[126] F. Stajano and R. Anderson, "The resurrecting duckling: security issues for ubiquitous computing," *Computer*, vol. 35, no. 4, pp. 22 –26, Apr. 2002.

[127] A. Vahdat, D. Becker *et al.*, "Epidemic routing for partially connected ad hoc networks," Technical Report CS-200006, Duke University, Tech. Rep., 2000.

[128] R. Viswanathan, J. Li, and M. C. Chuah, "Message ferrying for constrained scenarios," in *World of Wireless Mobile and Multimedia Networks, 2005. WoWMoM 2005. Sixth IEEE International Symposium on a*, June 2005, pp. 487–489.

[129] P. Wex, J. Breuer, A. Held, T. Leinmuller, and L. Delgrossi, "Trust issues for vehicular ad hoc networks," in *Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE*, May 2008, pp. 2800–2804.

[130] W. Whyte, A. Weimerskirch, V. Kumar, and T. Hehn, "A security credential management system for V2V communications," in *Vehicular Networking Conference (VNC), 2013 IEEE*, Dec. 2013, pp. 1–8.

[131] U. Wilensky, "NetLogo," Internet: https://ccl.northwestern.edu/netlogo/, Center for Connected Learning and Computer-based Modeling, Northwestern University, Evanston, IL, 1999.

[132] L. Wood, W. Eddy, and P. Holliday, "A bundle of problems," in *Aerospace conference, 2009 IEEE*, Mar. 2009, pp. 1 –17.

[133] W. Zhao, M. Ammar, and E. Zegura, "Controlling the mobility of multiple data transport ferries in a delay-tolerant network," in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 2, Mar. 2005, pp. 1407–1418 vol. 2.

[134] L. Zhou and Z. Haas, "Securing ad hoc networks," *Network, IEEE*, vol. 13, no. 6, pp. 24–30, Nov. 1999.

[135] H. Zhu, S. Du, Z. Gao, M. Dong, and Z. Cao, "A probabilistic misbehavior detection scheme toward efficient trust establishment in delay-tolerant networks," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 25, no. 1, pp. 22–32, Jan. 2014.

[136] H. Zhu, X. Lin, R. Lu, Y. Fan, and X. Shen, "SMART: A secure multi-layer credit-based incentive scheme for delay-tolerant networks," *Vehicular Technology, IEEE Transactions on*, vol. 58, no. 8, pp. 4628 –4639, Oct. 2009.

[137] P. R. Zimmermann, "PGP 2.6.3i User's Guide," Internet: http://www.pgpi.org/doc/guide/2.6.3i/en/, 1994.

[138] P. R. Zimmermann, *The official PGP user's guide*. Cambridge, MA, USA: MIT Press, 1995.