University of Strathclyde

SNAP: A Distributed Authentication Protocol

Aied Fayez Al Shahri

Thesis for the degree of PhD

Department of Electronic and Electrical Engineering

2003

The copyright of this thesis belongs to the author under the terms of the United Kingdom Copyright Acts as qualified by the University of Strathclyde Regulation 3.49. Due acknowledgement must always be made of the use of any material contained in, or derived from, this thesis.

I, Aied Fayez Al Shahri, hereby declare that this work has not been submitted for any other degree at this University or any other institution and that, except where reference is made to the work of other authors, the material presented is original.

Abstract

The explosive growth in network based applications and distributed systems allows the deployment of critical applications such as e-commerce, tele-banking, electronic government, etc. On the other hand, attacking networks and distributed systems becomes easier with the support of public cracking tools and the information sharing between hackers. Consequently, security is a crucial topic. Authentication is considered as one of the major components in security and, in fact, represents the front door for any secure system. It is an important requirement to ensure that the network resources are accessed only by authorized users, meanings that strong access control mechanisms are needed. Most existing authentication protocols are centralized such that a single authentication entity controls the authentication process. These traditional authentication protocols suffer from certain drawbacks pertaining to security, availability and trust. This thesis proposes and evaluates a novel authentication protocol: Secure Network Access Protocol (SNAP) that attempts to overcome some of the drawbacks of centralized authentication protocols. SNAP is a distributed authentication protocol and is based on secret sharing schemes which have a quorum access structure. Another significant contribution of this thesis has been to consider not only the security analysis of SNAP but also the implementation issues. SNAP is studied and found to be robust in term of its security, availability, overhead and performance. The implementation of SNAP is considered and the application of SNAP to a wireless networks undertaken. The research conducted has shown the need for distributed authentication protocols and the importance of studying the network implementation issues for any novel security protocol.

Acknowledgements

First, I thank God for giving me the strength and ability to complete my PhD.

I would like to thank Professor D. G. Smith, my initial supervisor for his support and guidance from the first day I met him until he passed away. I feel very sad for loosing a wise and respectful man. His concern about my study and my family was greatly appreciated.

I would also like to express my gratitude and appreciation to my supervisor, Dr. James Irvine, for his assistance and his valuable discussions. The knowledge and experience gained working with him is of great value.

I would like to thank my examiners Prof. Laurie Cuthbert, Head of the Department of Electronic Engineering at Queen Mary, University of London and Dr. Bill Nailon for their valuable discussion and useful comments.

I would like to thank the academic staff of the Communication Division of the EEE department, Dr. David Harle, Prof. J. Dunlop and Dr. D. Girma for their help and encouragement.

I would like to thank Olufemi Komolafe for his reading comments through some parts of my work. I would also like to thank all the members of the Communications Division for their help and support. In particular, I would like to thank my officemates, Wim, Andy, Florence, Kwong, Jalil, Omar, and Marwan.

I would like to express my appreciation to the Saudi Arabian government for their sponsorship and giving me the opportunity to continue my study.

Finally, I am grateful to my parents and my family for their patience and support throughout my PhD period.

Contents

Abbreviations and Acronyms

2G	2nd Generation
3G	3rd Generation
AES	Advanced Encryption Standard
AUC	Authentication Centre
CN	Core Network
COCA	Cornell On-line Certification Authority
CWlog	Crumbling Wall Quorum System
DES	Data Encryption Standard
DEA	Data Encryption Algorithm
DSA	Digital Signature Algorithm
EFT	Electronic Funds Transfer
FPP	Finite Projective Plane
FTP	File Transfer Protocol
GSM	Global System for Mobile Communications
G-SNAP	GSM Secure Network Access Protocol
HMAC	Hash-based MAC
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IDEA	International Data Encryption Algorithm
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
IDEA	International Data Encryption Algorithm
ISO	International Organisation for Standardization
ITU	International Telecommunications Union
KDC	Key Distribution Centre
LAN	Local Area Network
MAC	Medium Authentication Code
MAN	Metropolitan Area Network
MD5	Message Digest version 5
MSC	Mobile Switching Centre
MS	Mobile Station

NS-2	Network Simulator version 2
PEM	Privacy Enhanced Mail (PEM)
PGP	Pretty Good Privacy
PKI	Public Key Infrastructure
QoS	Quality-of-Service
QS	Quorum System
RAND	Random Number
RSA	Rivest, Shamir, and Adleman Algorithm
RFC	Read For Comments
SHA-1	Secure Hash Algorithm
SMTP	Simple Mail Transfer Protocol
SNAP	Secure Network Access Protocol
SSH	Secure Shell
SSS	Secret Sharing Scheme
TCP	Transport Control Protocol
TTP	Trusted Third Party
UDP	User Datagram Protocol

List of Publications

To date the work reported in this thesis has produced the following publications:

1. A. F. Al Shahri, D. G. Smith and J. M. Irvine, Performance Evaluation of Secure Network Access Protocol, in proceedings of the First International Working Conference on Performance Modelling And Evaluation of Heterogeneous Networks (HET-NETs '03), 21st - 23rd July, 2003, Ilkley, West Yorkshire, U.K.

2. A. F. Al Shahri, D. G. Smith and J. M. Irvine, A Secure Network Access Protocol (SNAP), in proceedings of the 8th IEEE Symposium on Computers and Communications (ISCC 2003) Kemer, Antalya, Turkey 30 June - 3 July 2003

3. A. F. Al Shahri, D. G. Smith and J. M. Irvine, Mobile distributed authentication protocol, The International Symposium on Wireless Systems and Networks Conference (ISWSN'03) Dhahran, Saudi Arabia, March 24-26, 2003.

4. A. F. Al Shahri, D. G. Smith and J. M. Irvine, Implementation of quorum systems to increase network security, Postgraduate Networking Conference (PGNET 2002), June 2002, Liverpool University, UK, Pages 186-191.

5. A. F. Al Shahri, D. G. Smith and J. M. Irvine, Implementation of Secret Sharing to Increase Network Security and Reliability, ESPRC Postgraduate Research in Electronics and Photonics (PREP), April 2002, Nottingham University, UK.

6. A. F. Al Shahri, D. G. Smith and J. M. Irvine, Secret Sharing and Quorum Systems: Concepts and Applications, submitted to IEE Electronics & Communication Engineering Journal.

Chapter 1

Introduction

1.1 Introduction

Computer networks are becoming virtually ubiquitous, as the number of users and the importance of the services they offer increases. Today individuals, commercial organizations and government agencies depend on networks for communication and information sharing amongst other things and so have much to lose if their sites are attacked. The Internet is an example of network growing in terms of the number of users and importance. The number of Internet users as of September 2002 reached 605.6 million [1]. The deployment of new sophisticated applications increases the need to migrate from traditional manually configured services to more automated and easy to implement services. The traditional networks provided relatively straightforward data services. Nowadays, however, networks provide a variety of services including data, voice, video and multimedia applications. In addition, the deployment of critical application is becoming attractive. These high security applications include e-commerce, tele-banking and military applications.

However, as networks become increasingly used to support critical applications, they become more prone to attack. Attackers are motivated by two main aims: firstly, to gain access to services they are not authorized to access and, secondly, to prevent authorized users from accessing their resources. Such attacks can be potentially very damaging, especially for critical applications. Internet related

crimes cost some of the top companies in the United States (who participated in the survey) at least $377.8 million in 2000 [2]. The growth of networking means that more information is available to and accessible by users. However, users can be genuine or malicious. It is obvious that as networks and applications increase, the method of attacking those networks and applications increases as well [3]. Therefore, security is a big issue and there is a demand to take the required counter measurements to make the network more safe and secure.

The importance of security differs from user to another. Some individuals and organizations take steps to protect their networks and resources from attackers, using certain protocols and a firewall. However, not all users are able or willing to go to such lengths to protect themselves. Figure 1.1 depicts an example of today's networks. Security is a wide subject and the main security services include: confidentiality, authentication, digital signature, and non-repudiation. Authentication represent the front gate for any secure system. Therefore, the theme of this thesis is authentication. Most, if not, all of the main security services requires authentication. This chapter introduces the thesis. Section 1.2 defines the thesis problem and Section 1.3 presents the aim of the thesis. The thesis objectives are outlined in Section 1.4. In section 1.5 the contribution is illustrated. Finally Section 1.6 describes the thesis outline.

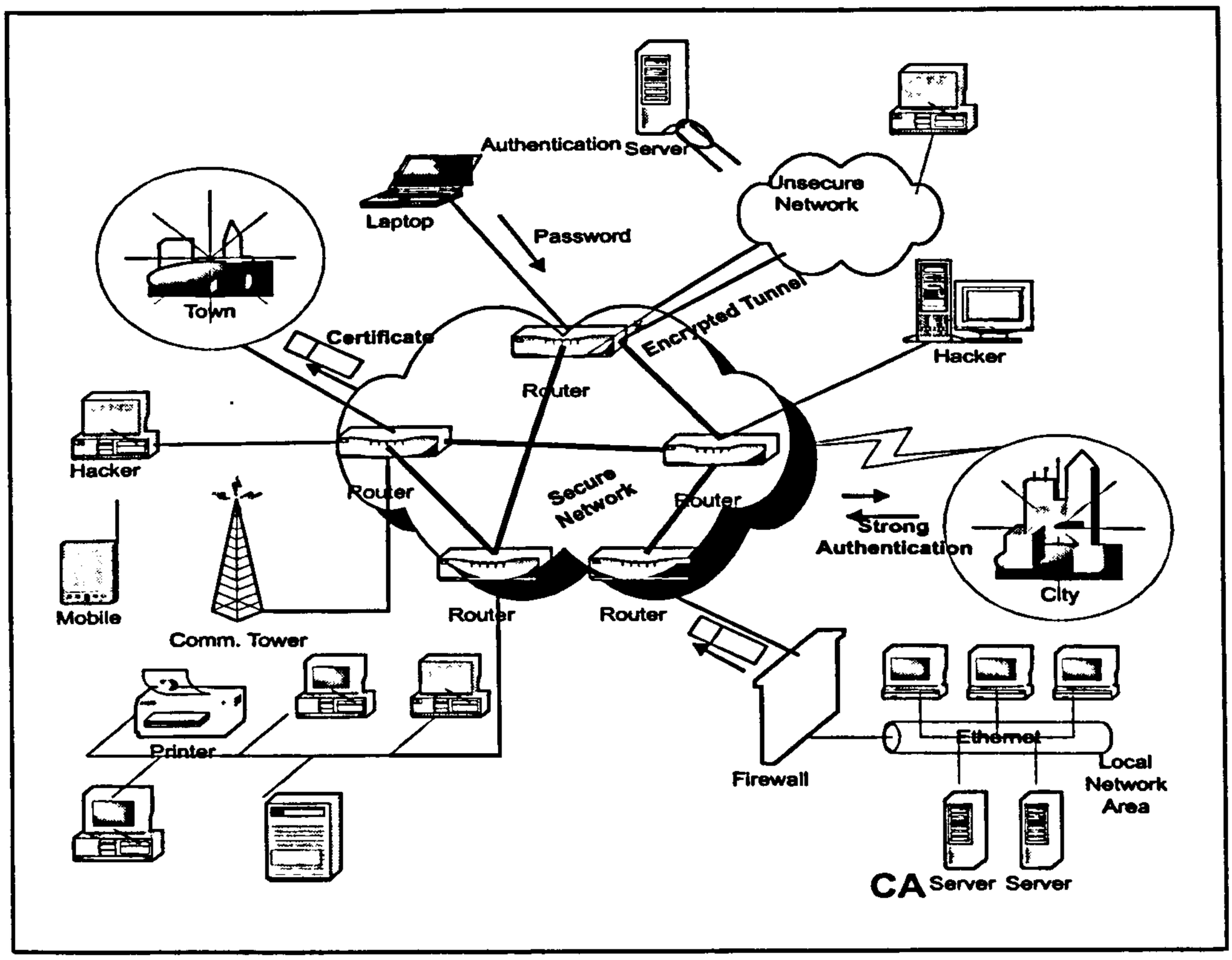


Figure 1.1: Example of today networks

1.2 Problem Definition

Authentication is crucial topic. Any authentication protocol must verify the claim of the claimant to be who they say they are. It must ensure that the access of the resources is given only to authorized users. Most existing authentication protocols are centralized in that they rely on a single authentication center or certificate authority [4, 5, 6]. In such protocols the single centralized entity is unconditionally trusted. These centralized authentication protocols suffer certain drawbacks: if the single entity is compromised then the entire system is no longer secure (security issue). If that entity breaks down and becomes unavailable then authorized users will not be able to access the required recourses (availability issue). Furthermore, the fact that the trust is focussed in a single entity increases the risk and affects the security (trust issue). The single entity will be highly loaded

since all the system users need to be authenticated through that entity. Hence, authentication protocols that can overcome these drawbacks are very attractive.

1.3 Aim

Centralized authentication protocols are ill-suited for critical applications. The aim of this work was to develop novel authentication protocols that overcome the previously mentioned drawbacks. These novel protocols, ideally, should be distributed authentication protocols such that more than one authentication server (node) participates in the authentication process. In other words, each authentication server is partially responsible for the authentication process. Therefore, none of the authentication servers handle the authentication process on its own as in the centralized protocols. The proposed authentication protocols must be secure such that compromising any authentication server does not compromise the entire system. Additionally, it is desirable that the user can be authenticated and can access the required resources despite the unavailability of a certain number of servers. Lastly, trust and the traffic load must be distributed between the authentication servers.

1.4 Objectives

In this thesis, a novel distributed authentication protocol, meeting the following objectives, is proposed.

- The protocol must be distributed such that more than one authentication server participates in authentication process.
- The protocol must ensure that the access is granted only to authorized users.
- The protocol should offer high availability.
- Trust should not be focussed on any single authentication server.
- The load is distributed.

- The protocol is scalable and its implementation is straightforward.

1.5 Thesis Contribution

The main contribution of this thesis is the development of a novel distributed authentication protocol. In summary the contribution includes the following:

- A survey of existing security and authentication mechanisms and protocols, focusing on the drawbacks on such schemes.

- Given the fact that the proposed protocol is based on secret sharing schemes that have a quorum based access structure, an overview of secret sharing and quorum systems concepts and application is presented.

- Secure Network Access Protocol (SNAP), a novel distributed authentication protocol is proposed.

- The performance of SNAP is evaluated, with emphasis being laid on security, protocol overhead and other such parameters.

- A mobile distributed authentication protocol; GSM Secure Network Access Protocol (G-SNAP) is proposed.

- In most of these contributions a comparison between the proposed protocols and centralized authentication approaches is undertaken.

1.6 Thesis Outline

The outline of the thesis is as follows:

Chapter 1 introduces the importance of security and particularly the authentication. It provides the problem definition and the aim to solve the problem. Objective and thesis contribution are outlined. Finally the thesis outline is summarized.

Chapter 2 reviews generic aspects of security and authentication. The chapter starts by presenting the general security requirements, as well as the general network requirements. Since the main contribution is related to authentication, a thorough authentication survey, examining different types of authentication protocols and examples of existing authentication techniques in real applications, is conducted. The survey starts with main authentication concepts (classes): peer entity authentication and message origin authentication and the different methods used. Next, the major types of authentication protocols are described. Some implementations of authentication protocols in the existing networks are investigated. Those protocols are Public key infrastructure (PKI) systems, Secure Shell (SSH) and another example from wireless networks, Global System for Mobile Communications (GSM) authentication. The investigation includes analysis of these protocols based on certain metrics. These metrics include security, availability and trust. The objective is to highlight the drawbacks and the problems that need more study and investigation.

In order to develop the distributed authentication protocol (SNAP), two tools were used secret sharing and quorum systems. Chapter 3 overviews their concepts and applications. These tools have certain characteristics that support the protocol design. The first part of the chapter is about secret sharing schemes. The definition and construction of secret sharing schemes are presented. Thereafter, the applications of secret sharing schemes are reviewed. The second part describes quorum systems. The definition and construction methods are summarized, and the application of quorum systems are then presented. After providing an overview about secret sharing and quorum systems another section is added. Since SNAP uses a quorum-based secret sharing scheme, after the overview of secret sharing and quorum systems, secret sharing schemes that have a quorum based access structure are addressed.

Chapter 4 proposes the main contribution of the thesis, a novel distributed authentication protocol (SNAP). The chapter starts with a background on authentication models as an input to SNAP. Next, the main components and the description of SNAP are discussed. The signalling between the protocol components is presented. In addition, the required messages and their contents are described. Thereafter, the security analysis of SNAP is discussed. This analysis shows how SNAP can perform against main attacks. Next the required management issues

are investigated such as message size, share revocation and update mechanisms. Since quorum systems are a main part of the development of SNAP and affect the protocol implementation, these have been studied to investigate quorum design issues. In this chapter the expected applications of SNAP are outlined. Finally, the objective goals of SNAP are discussed.

In order to implement a new protocol it is important to analyze the protocol before deployment. Chapter 5 analyzes SNAP and shows simulation results. This analysis highlights advantages and disadvantages, as well as overhead and implementation requirements. Using a simulation model, a comparison between SNAP and centralised authentication protocols, in terms of certain defined metrics, is undertaken. These metrics include security, availability, trust, overhead and delay. The overhead investigation includes the signalling overhead, the message size affect, and the load distribution. The investigation is further extended to analyze the performance of SNAP. The investigation is measured by the delay experienced by users. Different network implementation parameters are considered such as the degree of connectivity, the network size, quorum design, packet size, and link bandwidth.

Chapter 6 proposes a novel mobile distributed authentication protocol. The existing GSM authentication protocol is centralized, so a distributed authentication protocol for mobile networks is attractive. This protocol GSM Secure Network Access Protocol (G-SANP) is an extension of SNAP. It is one of the applications of SNAP in the wireless networks. In this chapter a brief introduction on mobile networks is presented along with a discussion of GSM authentication. Thereafter G-SANP is described. The performance of G-SNAP, evaluated through simulation, is presented and discussed.

Chapter 7 concludes the thesis and Chapter 8 presents the future work. In this chapter different extensions of the proposed work in this thesis are outlined.

Chapter 2

Security and Authentication

2.1 Introduction

The rapid growth of computer network sizes, and the increase in interconnectivity, are two major factors that increase concerns about network security. A key prerequisite to a secure network is the reliable authentication of communicating parties in the network. Authentication and access control are therefore essential tools for network security. Society's increasing dependency on pervasive computer networking infrastructures has created an urgent requirement for security structures to reflect those that are trusted in the real world. While there is a tendency to migrate existing services into the network, the network infrastructure must be secured sufficiently in order for traditional public trust to also migrate to using those services online. For example, in a bank, an individual is unable to open the bank vault alone, and to do so the concurrent action of more than one person is required. This means that the trust is distributed between an assigned number of employees.

Cryptosystems are used to facilitate a secure communication between parties. Usually, users can obtain the keys either from an authentication server or from a certification authority [4, 5]. In both cases, users place complete trust in these centralized authentication entities. In general, if trust is maximized then the risk increases and results in decreased security and availability. Hence, if the centralized authentication entity is compromised, the entire system becomes insecure.

If the authentication server breaks down, the authorized user will be affected. Most existing authentication protocols are centralized, which may affect the security and availability in spite of the use of strong authentication mechanisms [6]. It is clear that there is a need to have authentication protocols that are not centralized. Distributed authentication protocols therefore have attractions.

In this chapter security and authentication are discussed. More details about authentication techniques, architectures and protocols are provided. An overview of some authentication protocols and applications is presented. In addition, analyse of these protocols and exploration their drawbacks is conducted.

2.2 General Security Requirements

In general, secure communication between two parties involves certain requirements. Each party needs to know to whom he is communicating. It is important that all messages are enciphered so that others are unable to know their contents. In addition, it is important that resources are accessed only by authorized users. The security requirements can be classified into the following services. Some of these services are independent and some have overlapping functionality. This section describes each one of them briefly:

- **Confidentiality:**

  preventing the disclosure of transmitted data to unauthorized parties. Confidentiality comes in different forms [7].

  - **Selective field confidentiality** is to provide confidentiality of only certain parts of a message
  - **Whole message confidentiality** is to protect the privacy of the entire message.

  The security mechanism used is the encryption.

- **Integrity:**

  protecting the message or transmitted data from modification, insertion, deletion or replay by unauthorized users. An integrity service comes in the following forms [8]:

  - **Selective field integrity** occurs when only certain or critical parts in a large message require integrity protection.
  - **Whole message integrity** protects the whole message from any alterations being made to the message. This allows the receiver to detect any alteration to the received message.
  - **Session integrity** an intruder who cannot modify messages without being detected can still cause considerable mischief by manipulating valid messages. An intruder can delay messages, delete messages, replay and reroute of a valid message. Session integrity service allows the detection of some or all of these attacks.

  The main security mechanisms used to provide integrity are encryption, authentication and digital signatures.

- **Authentication:**

  The process of verifying the identity (or/and) the origin of the claimer [9]. In any communication between two parties it is crucial that the message is sent to the correct party. In addition, knowing the origin of the message is important. The authentication is paramount in critical applications such as e-commerce banking applications etc.

- **Non-repudiation:**

  Prevents either the sender or receiver in a communication from falsely denying their participation [10]. Non-repudiation of origin provides the receiver of a message with irrefutable proof that the message was produced by the purported sender, in other words prevents the sender denying that a message was sent. Non-repudiation of receipt prevents the receipt denying that the message was received. Digital signature mechanism with time stamping is used for providing non-repudiation.

- **Access control:**

  Ensures that only authorized parties can access the resources they are legible to access and limited to their privileges. Access control guards against unauthorized use of resources [10].

- **Availability**

The probability that a network or a system can perform its required function at any instant of time. Availability is an important issue and considered one of the required security services. A variety of attacks can result in the loss of or reduction in availability [11]. Authentication and encryption can be used as a countermeasures to attacks that may affect the network or system availability. Sometimes a physical action is required to prevent such attacks.

2.3 Security Mechanisms

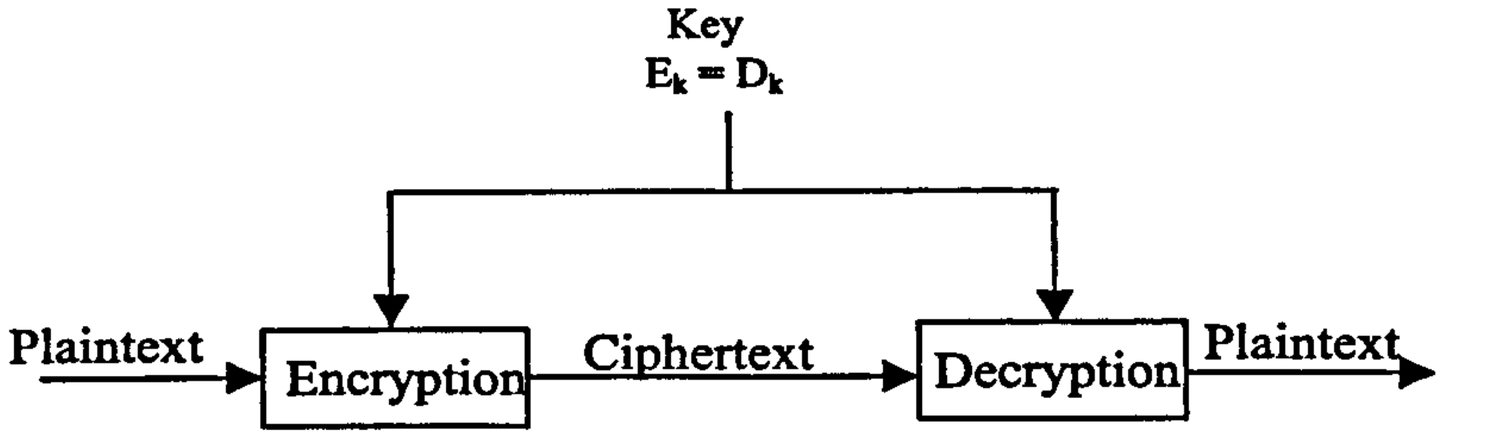
Security mechanisms consist of various algorithms and protocols that support security services. There is no single mechanism that will address all security services. Some of these mechanisms can address more than one security service (for example hash functions are commonly used for authentication, data integrity and non-repudiation) [10].

- **Encryption:**

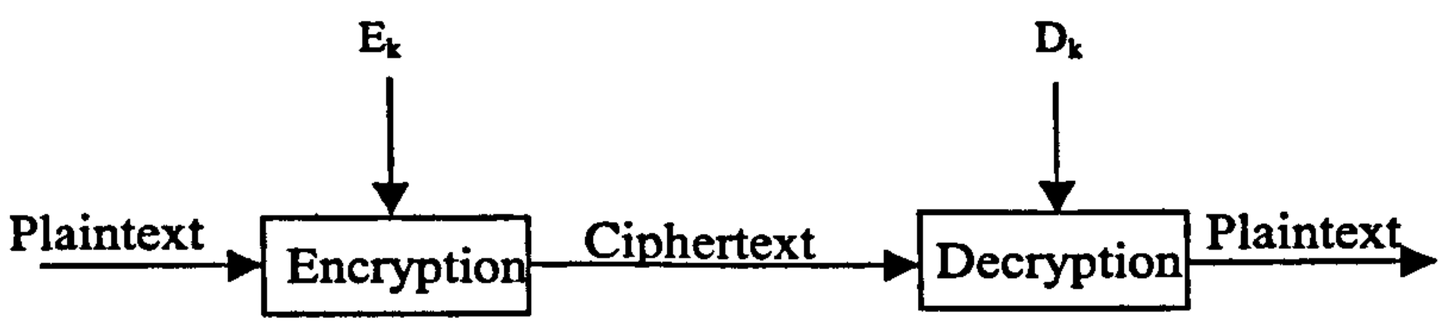
  Encryption is the process of converting the original plaintext to an encoded information or ciphertext using an encryption key [9]. It is the most well known security mechanism that deploys cryptographic techniques and has a long history [12] . Encryption algorithms either symmetric or asymmetric as follows (see figure 2.1).

  - *Symmetric encryption:* Is the oldest and most commonly used type of encryption. In this form of encryption, two communicating entities share a secret key used to encrypt and decrypt messages, it is described as a secret-key cryptography [9]. Data Encryption Standard (DES) 64 bit key, Advanced Encryption Standard (AES), triple DES and International Data Encryption Algorithm (IDEA) are examples of symmetric encryption algorithms [7].
  - *Asymmetric encryption:* Is called public key cryptography, in these algorithms there is a pair of keys, one called the public key and the other the private key which remains secret. In these algorithms the message is encrypted using the public key and decrypted using the

private key. The public key is made known to all parties interested in sending messages to the holder of the private key. The main popular public key algorithms are: Rivest, Shamir, and Adleman Algorithm (RSA), El Gamal, Elliptic curve cryptosystems and Rabin [7].



a. Symmetric encryption



b. Asymmetric encryption

Figure 2.1: Encryption Schemes, E_k: Encryption key, D_k: Decryption key

RSA Algorithm

The RSA public key cryptosystem was introduced in 1977 [13]. It is the most widely used public key cryptosystem [7]. The RSA security is based on the difficulty of factoring a large number n which is a product of two large prime numbers p and q. To generate the public and private keys the following steps are used:

1. Calculate $n = p \times q$
2. Randomly choose the public key e relatively prime to $(p-1) \times (q-1)$
3. Compute the private key $d = e^{-1} \pmod{(p-1 \times q-1)}$

In order to encrypt a message m it is divided into numerical blocks of fixed length and each block m_i is encrypted individually.

The encrypted block $c_i = (m_i)^e \pmod{n}$.

To decrypt c_i compute $m_i = (c_i)^d \pmod{n}$.

- **Hash functions:**

  A hash function takes as input a plain text message M and produces a fixed-length, short message digest m [7]. The hash function is considered a secure, one-way function if it satisfies the following requirements:

  - Any change to the initial bit string changes the message digest completely;
  - It is computationally infeasible to find any pair of messages that have the same hash value;
  - It is practically impossible to derive the original message from its digest even if the hashing function is known (one way property). The most popular hashing algorithms are Message Digest version 5 (MD5) and Secure Hash Algorithm (SHA-1) [7].

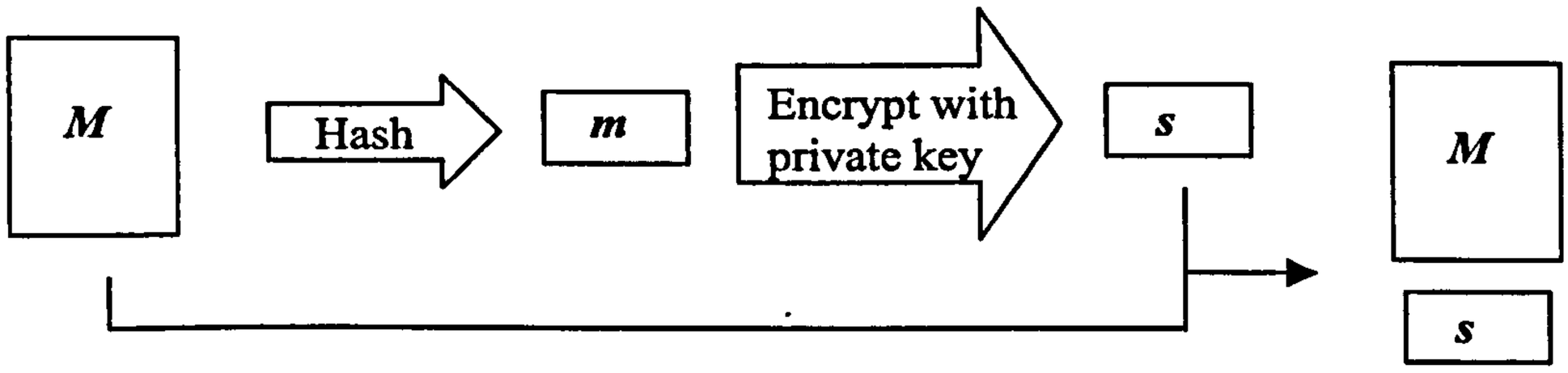
- **Digital Signature:**

  In general handwritten signatures have been used to prove the authorship. A digital signature is the counterpart of a handwritten one and should comprise some data to prove that the signer is the originator. It can be thought of as the reverse of public key encryption. The encoding key (signature key) is now the key that is kept secret, while the decoding (verification) key is made public. A digital signature of a message is typically a fixed length string of bits that can be derived from the original message and a secret private key [9]. Anyone who knows the corresponding public key can check that a claimed signature is indeed the correct signature. The following steps illustrates the digital signature process (see Figure 2.2):

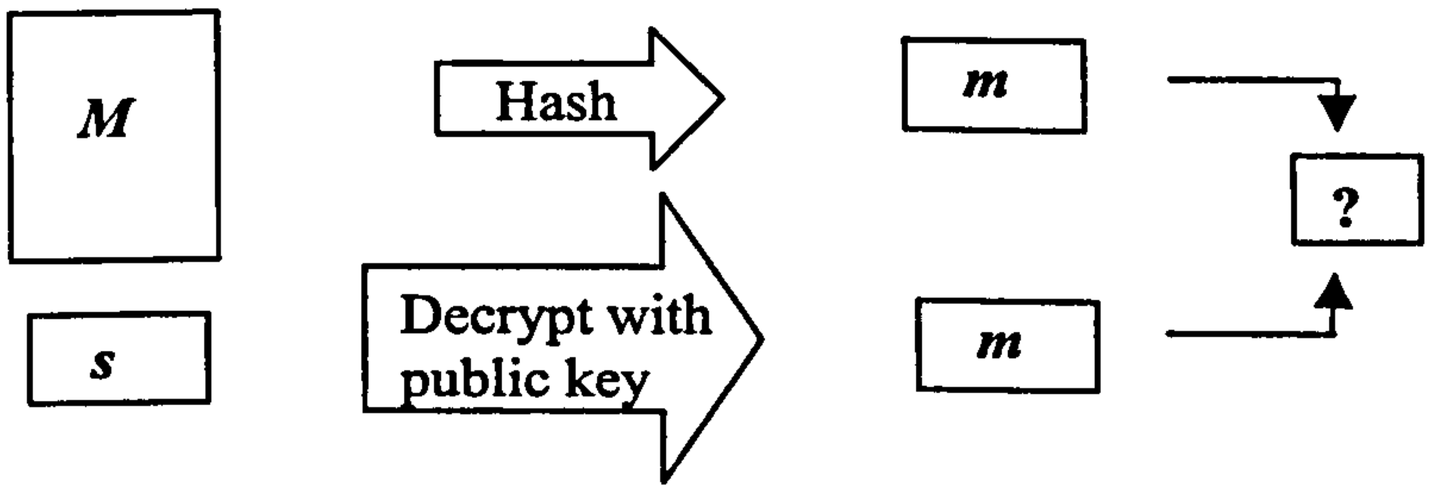
  1. The plain text message M to be signed is being transformed using a hashing function into a message digest m, the sender will encrypt the message digest m using his own private key. The result is the signature of message M.
  2. Upon reception of the signed message (M and s), the receiver computes the digest of M as in step 1 above and decrypts the signature s using the sender's public key. If the result of this operation is identical to m obtained before, then the receiver knows that the message was indeed

originated by the sender and has not been altered in any way. Also, since the sender is the only one that can sign M with his private key, he can not deny having sent the message.

Digital signature schemes can support integrity, authentication and non-repudiation security services.



1. Signing a message



2. Signature verification

Figure 2.2: Digital Signature

- **Certificates:**

  In public key encryption or techniques only the public keys need to be distributed. Such distribution can be done by broadcasting or via a directory, but must be done securely to be sure that customer's public key has not been changed by an intruder and belong to the correct owner. This assurance can be provided with a public key certificate. A public key certificate is a record that binds the user's public key to the user's unique identifier. The record is signed by a trusted Certification Authority (CA). The security

of the user communications depends on the authenticity of the CA's public key that the user uses. The International Organisation for Standardization (ISO) authentication framework X.509 defines all the fields for a certificate [10, 14].

- **Authentication:**

  A good authentication protocol is the one which allows two parties to correctly know each other's identity while communication over an untrusted network. As mentioned in the last section authentication can be accomplished directly between two parties using hash functions and encryption techniques, or it can be accomplished through a trusted third party [7].

- **Access Control:**

  Access control mechanisms usually described in terms of subjects, objects and access rights [8]. A subject is an entity that can access objects and it can be a host, a user or an application. An object is a resource to which access should be controlled. It can range from a single data field in a file to a large program. The access rights specify the level of authority for a subject to access an object. In other words, access control is the process of determining if a subject (either user or program) is authorized to perform a given operation on an object or a resource. Usually access control lists (ACLs) which are associated with objects, and capability lists which are associated with subjects are used to implement access control [10].

Figure 2.3 lists the various security services discussed in Section 2.2 and identifies which of the security mechanisms they support.

Security Service	Security Mechanisms: Access control	Hashing	Encryption	Digital signature	Digital Certificates
Access Control	✓				
Entity Authentication		✓	✓	✓	✓
Origin Authentication		✓		✓	
Integrity		✓		✓	
Confidentiality			✓	✓	
Non-repudiation				✓	

Figure 2.3: Security mechanisms/Security services (source [10])

2.4 Security support mechanisms

The security mechanisms outlined above are used to prevent security intrusion and to have a secure network. It is impossible to guarantee that no intrusion would succeed. The Internet and the availability of attacking tools help attackers to exploit the security breaches in most of the existing software to make a successful attack. Therefore it is necessary to be able to detect such intrusions and recover from them. There are two methods or support mechanisms that can help [10]:

- **Security audit log** This is used to detect insidious intrusions that may not have an immediate or obvious impact. It records security related events, especially the security alarms.

- **Security alarm** This is a basic security requirement for each system which should be able to issue a security alarm whenever an attempted security breach is detected.

2.5 Security Threats

In general an attack is an exploitation of vulnerability and there are two types of attacks: passive and active attacks. In passive attack the attacker can't alter the data while an active attacker can do so.

A passive attack threatens the confidentiality of data being transmitted. In this attack the transmitted data can be intercepted or analyzed to obtain the real data without the notice of the communicating parties [8].

Active attacks threatens the integrity or availability of data being transmitted. In this attack the intruder can modify, delete and replay data units. In addition the intruder can flood a receiver and cause a denial of service attack.

The most possible security threats are [10]:

- **Unauthorized access:** This means an attacker gaining unauthorised access to a resource. The access to a restricted resource can occur if the intruder can breach the security of the resource or by having valid credentials to fool the resource. The first case means that the security of the resource is not sufficient. In the second case the authorized user's credentials are not being securely stored.

- **Eavesdropping :** In this case the intruder can extract the information from the communication channel sometimes without the knowledge of the communicating parties.

- **Masquerade:** This occurs when the attacker is able to mimic an authorized user such that the resource believes that the attacker is the authorized user. This could happen using impersonation.

- **Modification of information:** Modification of information occurs when unauthorized information is entered into a resource. This can lead to a large security breach by changing the security practices so that the attacker can enter the resource at any time he wants.

- **Repudiation:** This refers to a user having requested or received a services and then denying having either requested them or received them. This kind

of threat can lead to disputes between the sender and the receiver especially if there are bills involved in the operation.

- **Replay, reroute, and delete messages:** In a replay attack the attacker copies a valid message and reuses it again in order to access or receive a services that he is not entitled. In a reroute attack the attacker can change the original route of the message to be able to obtain the message as it moves between source and destination. The misroute attack makes the message reach the wrong destination. In a delete attack the attacker prevents the message reaching any destination.

- **Network flooding:** This type of attack occurs when the attacker floods the network with false messages. This attack makes the targeted machine or system waste time and resources by processing these false messages. This can lead to the prevention of authorized users from accessing the system and their requests will be rejected.

2.5.1 Internet Security Threats

The security threats can be countered using different combinations of the previous security services. When connected to the Internet there are five methods of attack that are commonly used to compromise the integrity of networks [15]:

- **Network packet sniffers** A packet sniffer is a software application that uses a network adapter card in promiscuous mode. In this mode the network node sends all packets received on the physical network wire to an application for processing. The objective is to capture all network packets that are sent across a network.

  Because several network applications distribute network packets in clear text, a packet sniffer can provide its user with meaningful and often sensitive information, such as user account names and passwords.

- **IP spoofing** Internet protocol (IP) spoofing means that one host is claiming to have the IP of another host. Usually routers pass the packets based on the sender's IP address. The attacker can exploit this by sending packets to a host to take some sort of action.

- **IP session hijacking** This is a dangerous attack where the user's session is taken over and controlled by the attacker. In addition the attacker can execute any commands on behalf of the user. The attack occurs when a user carrying out a session with another host and a man in the middle attacker runs a tool to impersonate the user without the host noticing. The availability of certain toolkits allows unskilled persons to perpetrate this attack [16].

- **Passwords attacks** Password attacks can be implemented using several different methods, including brute-force attacks, Trojan horse programs, IP spoofing, and packet sniffers. Although packet sniffers and IP spoofing can yield user accounts and passwords, password attacks usually refer to repeated attempts to identify a user account and/or password; these repeated attempts are called brute-force attacks. Password attacks can provide access to accounts that can be used to modify critical network files and services. A Trojan horse is a program containing hidden code that when invoked perform harmful function. For example, it can capture sensitive information such as user account and password information and distribute it back to the attacker.

- **Denial-of-service attacks (DoS)** The concept of DoS attacks is to make the target machine too busy by sending many more requests than it can handle. Denial of service attacks are the most difficult to address. They are very easy to launch but difficult to track.

- **Application layer attack** In this case the attackers can gain access to a computer with the permissions of the account running the application, which is often a privileged system-level account. This occurs by exploiting the weaknesses in software commonly found on servers, such as sendmail, PostScript, and File Transfer Protocol (FTP). The Trojan horse attack is an example. In addition, the openness of several technologies such as Hypertext Markup Language (HTML) and Hypertext Transfer Protocol (HTTP) can help to launch an application layer attack.

2.6 Network requirements

It is important to meet the security requirement in developing new protocols or algorithms. Since networks are the real platform for most of applications it is crucial to know and consider what are the network requirements. In security protocols it is possible to find a protocol that has no cryptographic pitfalls but may introduce network management problems. This means that the protocol meets the security requirements but does not meet the network requirements. This section will therefore discuss the main network requirements and the related issues, in other words, what the security protocol has to do from the network perspective. These requirements must be considered in the design of security protocols to ensure their practicality.

- **Scalability**
  A scalable protocol continues to function effectively as the network size, bandwidth and complexity increases.

- **Ease of implementation**

  An easily implementable protocol is one that works well and conveniently with existing protocols, introduces low overhead (message quantities, sizes, etc) and low latencies associated with it.

- **Management**

  Management complexity is a crucial metric for evaluating a proposed system or protocol. Ideally, the management should not be excessively burdensome and affect the protocol.

- **Reliability**

  Reliability is the ability of a system or a network to perform its required functions under stated conditions for a specified period of time [17]. Resilience is one of the network reliability requirements. The security protocol should maintain its function in spite of the failure of some network nodes. In centralized systems, if the central node becomes unavailable the system users are affected. Ideally, the protocol should be designed to be immune to the loss of some messages and certain nodes.

- **Network security**

  Network security is defined as the protection of data during its transmission from one computer to another from attack. It is often considered distinct from computer security which deals with the protection of data stored on a computer [11]. In order to secure the flow of data from a source node to a destination node, the network administrator must focus on trying to prevent :

  - The interruption of the data flow. This is an attack on the data's availability.
  - The interception of messages sent across the network. This is an attack on the data's confidentiality.
  - The modification of messages sent by legitimate users of the network. This is an attack on the data's integrity.
  - The transmission of fabricated messages. This is an attack on the data's authenticity.

  The two major threats for networks are [16]:

  - **Denial-of-service attacks**
  - **Unauthorized access**

  In order to provide a kind of protection for networks connected to the IP network (Internet), firewalls have been employed. Most organizations today have to resort to using firewalls to protect themselves from hostile attackers on the Internet. The firewall must be the gateway for all communications between trusted networks and untrusted networks [18]. Trusted networks are the networks inside your network security perimeter and you are trying to protect [15]. The firewall is the first line of defence but can not protect against attacks that do not go through the firewall (see Figure 2.4). In general there are three basic types of firewalls as follows [8, 19]:

  Packet filters: Selectively blocks traffic by examining packets one by one and filtering out those that are disallowed based on their source or destination addresses or ports.

  Circuit gateways: They act as an intermediately relay of Transport Control Protocol (TCP) connections. The caller connects to a TCP port on

the gateway, which connects to some destination on the other side of the gateway. The gateway logs information about each connection such as the destination address and the number of bytes. The gateway acts as a wire.

Application gateways: Using the application gateway, a special-purpose code is used for each desired application that it supports (e.g. Simple Mail Transfer Protocol (SMTP), FTP). In application gateways, it is easy to log and control incoming and outgoing traffic. It is more secure than other alternatives. The gateway can open packets and examine their contents and builds new ones with the same contents. Hence, only packet types for which there is construction code can be sent out from the gateway. It is impossible to send unauthorized packet types because there is no code to generate them, which prevents "back door" which defined as a designed "hidden" entry into a system to bypass password entry [19].

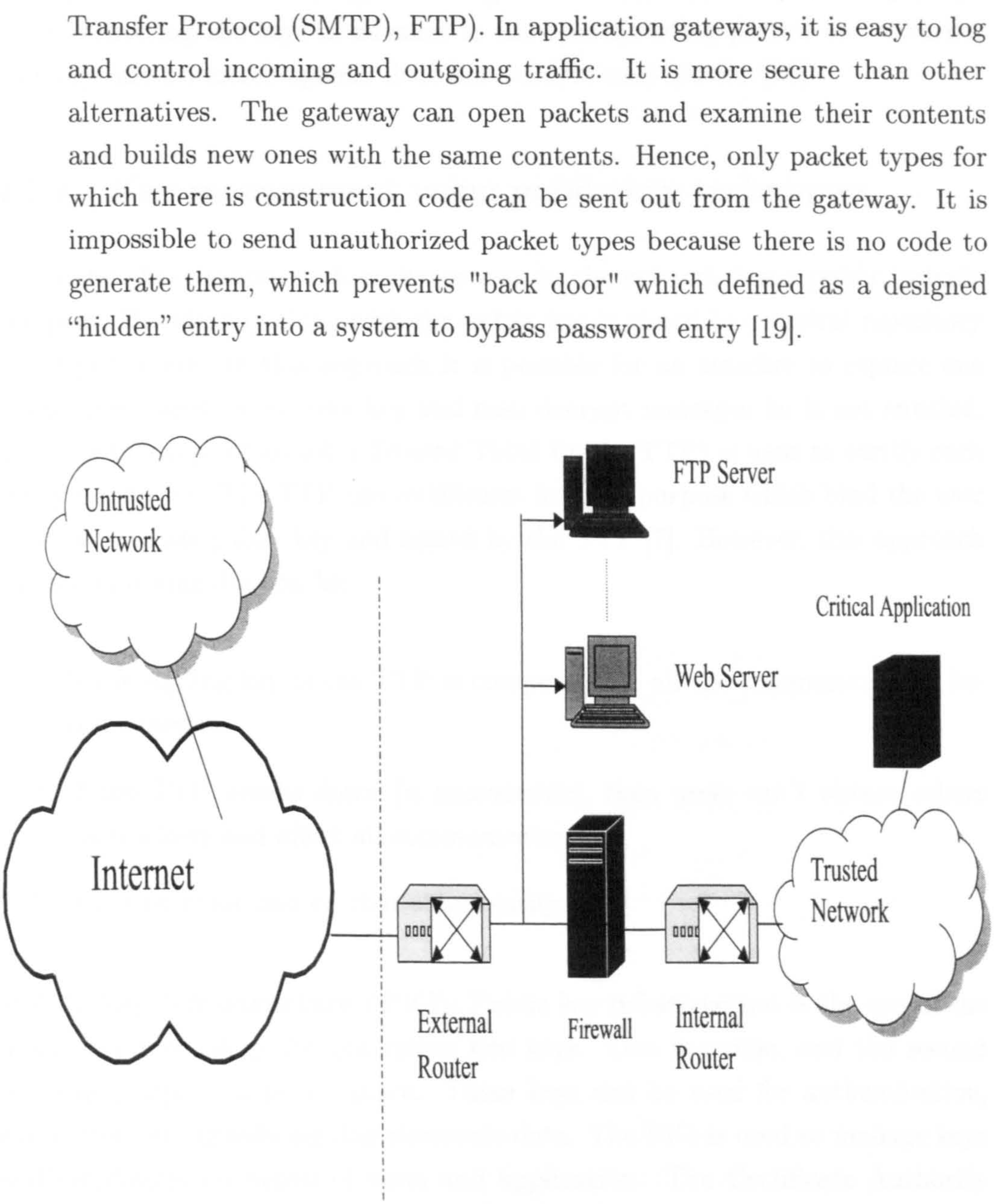


Figure 2.4: Firewall implementation

2.7 Key Management

Cryptographic techniques depend upon cryptographic keys. The security mechanisms described previously require the secure distribution of keys either symmetric or public. Management of keys is a crucial aspect of providing security. Key management includes ensuring that the generated keys have the necessary properties, delivering the keys in advance to the corresponding parties and ensuring that keys are protected against disclosure and/or substitution [20].

2.7.1 Key management using public-key techniques

In a public key system, each entity or user in the network has a public/private key pair. The identity along with the public key is stored in a central repository called public file. In this approach it is possible for an attacker to replace one of the user's keys by his own key and easy decrypt messages he is not entitled. To avoid this type of attack a Trusted Third Party (TTP) is used to certify each entity public key. The TTP use certificates for that purpose which bind the user identity with his public key and signed by the TTP [7]. However, this approach has the following drawbacks:

1. If the signing key of the TTP is compromised, all the communications become insecure.

2. If the TTP breaks down (is unavailable), then users can't obtain others parties keys and affect all communications.

3. All trust is focused on that single entity TTP.

Public key infrastructure (PKI): Public key infrastructure is the use of the public key technology for generating two keys. One is public, and the second is private, kept in a secure place. These keys can be used for authentication, encryption, or digitally signing electronic data. The PKI is used to manage keys and certificates on behalf of users and application. The Certificate Authority (CA) is required by PKI to securely issuing the X.509v3 certificates. Certificates are typically stored in a directory system, with applications using the Lightweight

Directory Access Protocol (LDAP) to retrieve them as needed. More information about PKI and key management issues can be found on [21, 22, 23, 24, 25].

2.8 Authentication

The increase in the development and use of networked and distributed systems provides more efficient use of resources. To obtain the benefits of such systems, users cooperate by exchanging messages over networks. When a user receives a message, the user wants to be sure that it has been created recently and in a good faith by the user who claims to have send it. It is important to be able to detect when a message has been created or modified. An authentication protocol is a sequence of message exchanges between users that either distribute secrets or allow the use of some secret [26]. In general, authentication protocols use encryption techniques to accomplish authentication between two parties.

2.8.1 Definition

Authentication is defined as the process of verifying that an identity is as claimed. It is used in computer networks and distributed systems to create a mutual identification between pairs of users (people, computers, services) [27].

Generally, there are two types of authentication: peer entity authentication and data origin authentication; peer entity authentication is verifying the identity of the principal while data origin authentication is verifying the source of data.

2.8.2 Peer entity authentication

This provides the assurance that the identity of the user or the system is as claimed in the received message. It is performed during the association set-up time between the communicating parties. There are three methods for providing peer entity authentication [7, 28].

2.8.2.1 Fixed Passwords

Theses are single unidirectional messages from the source to the destination, and form the most common method of user authentication. It is considered to be weak authentication since it is vulnerable to theft by an eavesdropper. One-time passwords where the password is used only once are more secure, but are more complex to manage.

2.8.2.2 Challenge-response authentication

This method is based on an exchange of messages containing time-variant parameters called nonces (random numbers, sequence numbers or time stamps). The idea is that for the claimant to prove their identity to the verifier, the verifier sends a time-variant challenge to the claimant. The claimant computes a response from the challenge using some encryption algorithm and sends the response to the verifier. The verifier compares the response with the result and makes a decision. Challenge-response can be based on secret-key or public-key techniques.

2.8.2.3 Trusted third party

There are some authentication protocols that involve a third party. The trusted third party could be an authentication server to provide a secret shared key to both communications parties to be used in the authentication process (e.g. Kerberos algorithm) or a CA which issues certificates [7]. A public-key certificate is a record that binds the user's public key to the user's unique identifier. The CA signs the record. Figure 2.5 depicts the three peer authentication methods which have been described above. Alice needs to obtain the certificate from the CA either using a secure link or using off line procedure.

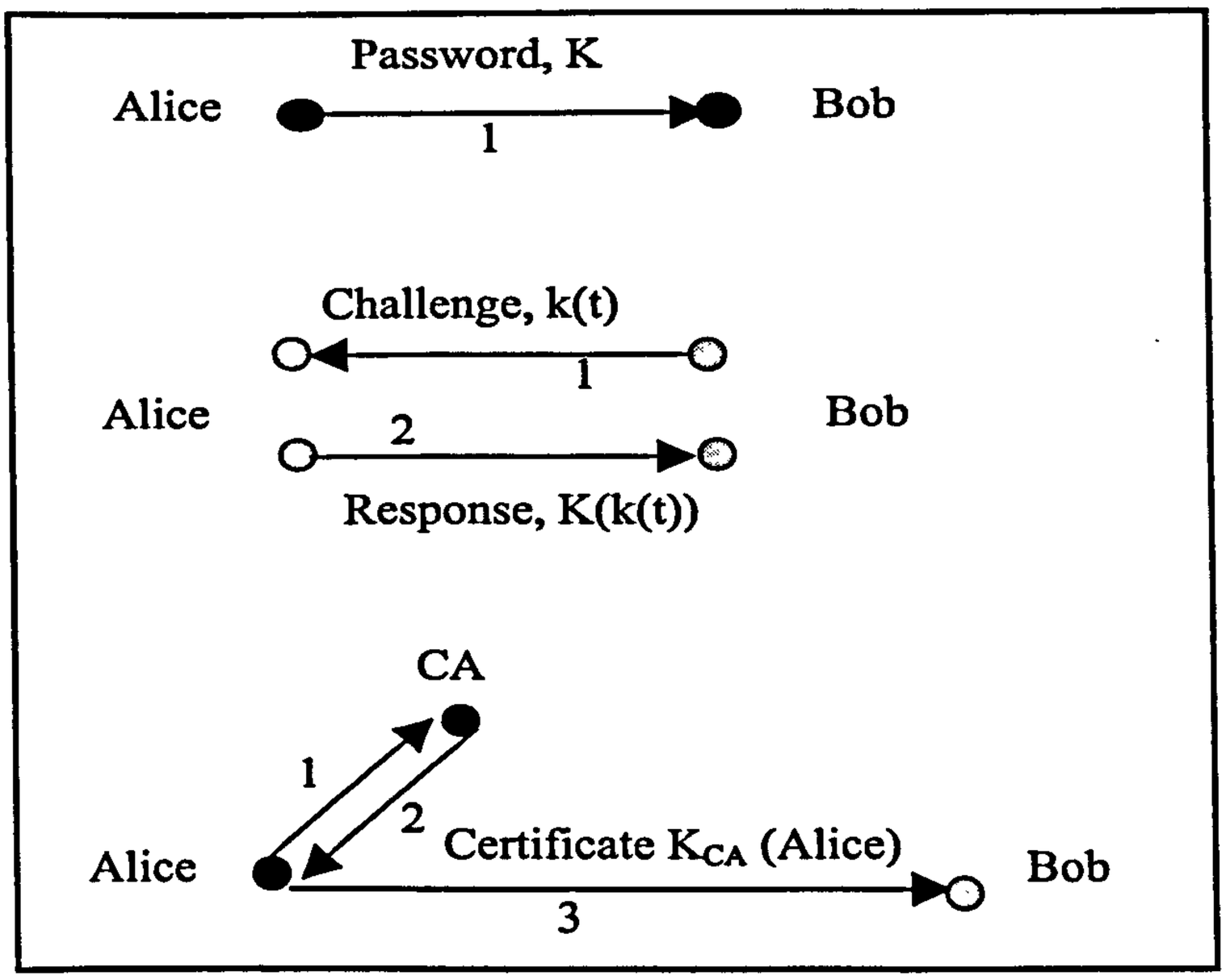


Figure 2.5: Peer entity authentication methods

2.8.3 Data origin authentication

Data origin authentication provides the assurance that the source (origin) of a message or transmitted data is as claimed. There are three methods for providing data origin authentication. All methods use hash functions. As described in section 2.3 a hash function takes as input a plain text message M and produces a fixed-length, short message digest m. These data origin authentication methods are :

- **Message authentication code (MAC)**

  It is a key-dependent one-way hash function. Only someone with the identical key can verify the hash and know the source of the message (see figure 2.6).

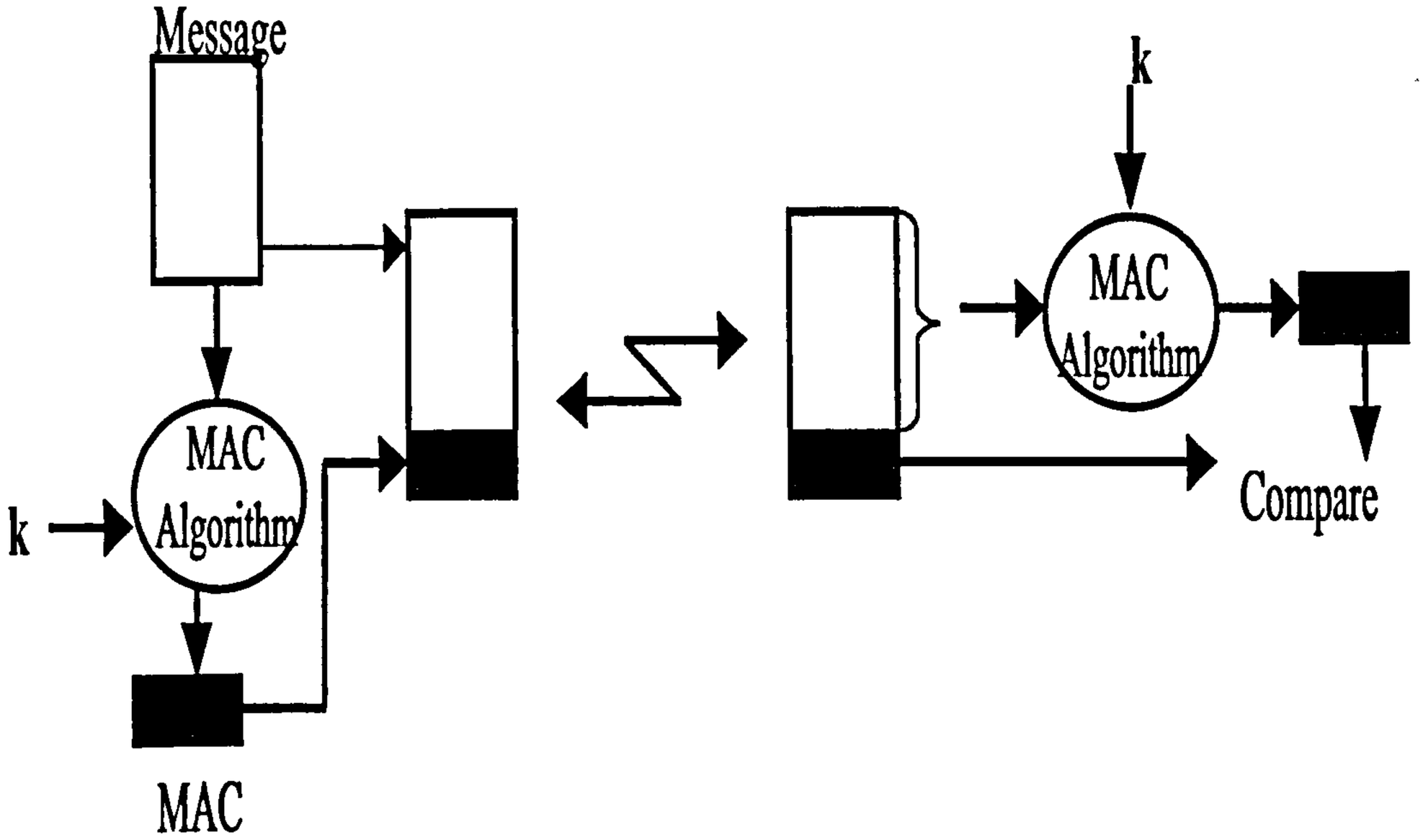


Figure 2.6: MAC scheme

- **Digital signature**

  A digital signature is based on public key techniques. The sender forms a hash of the message; the hash function being chosen to produce values from the same set as the ciphertext. The sender then decrypts the hash with his private key, and sends it with the message. The recipient then 'encodes' the signature, and if it matches the hash, which he independently calculates, the message is verified. In a public key system, only the holder of the private key could generate a signature which when encoded would yield the hash.

- **Manipulation detection code (MDC):**

  This is also known as modification detection code. It is mainly used for data integrity, which is considered as a message authentication process. The MDC is a hash of the message. Normally un-encrypted hash functions are used.

2.9 Authentication Protocol model and requirements

In this section an authentication model is introduced. In addition, the required properties that an authentication protocol should satisfy is outlined. In this model an assumption of having two communicating parties A and B is used. The protocol is assumed to have a finite sequence of messages as shown in Figure 2.7.

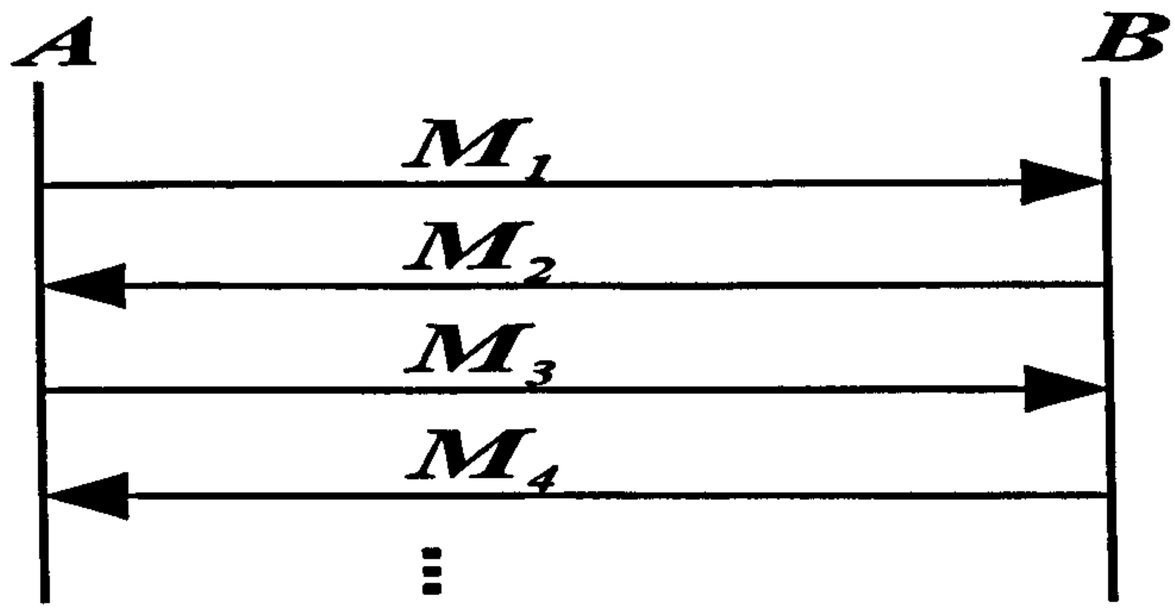


Figure 2.7: Authentication model

As shown in the Figure 2.7A starts the protocol and sends message M_1 to B. B then sends message M_2 to A. After receiving M_2 A sends M_3 and so on until the protocol is complete. It has been assumed that each of the communicating parties will not send message M_i until M_{i-1} has been received.

2.9.1 Requirements

It is important to know what the protocol should satisfy for both parties. In summery both parties want to be sure that the following requirements are achieved:

1. The messages assumed to be sent by the second party were indeed genuinely sent by him. For example, B wants to be sure that M_1, M_3, .. were all sent by A. This may require the identity of the sender.

2. The received messages are fresh, meaning that they were not replays of old messages. This requires some freshness mechanisms such as synchronized clocks and nonces.

3. The received messages were intended for the receiver not for any other party. This could imply the including of the receiver identity in the sent message.
4. The received messages were only generated by the sender after a correct reception of previous messages by the sender. For example M_2 and M_4 were generated by B after M_1 and M_3 were received correctly by B. This requires a sequence tracing mechanism such as including random numbers incremented each time they have been used.

2.10 Authentication protocol types

In this section some forms of authentication protocols in use today are overviewed. They have been categorized according to the cryptographic approach taken i.e symmetric or public key. A distinction can be made between those protocols that use Trusted Third Party (TTP) and those that do not. Authentication protocols can be classified based on the number of messages involved in the protocol such as one pass, two pass and three pass.

2.10.1 Symmetric key without trusted third party

There are some authentication protocols that are based on symmetric key and without using TTP. An example of such protocols are the simple challenge-response protocol (see Figure 2.8). As shown in the figure user A send a nonce N_a to B. B then carries out some transformation and send the results to A to verify that an appropriate transformation has occurred.

Figure 2.8: Challenge-response model

2.10.2 Symmetric key with trusted third party

In these type of protocols a trusted third party is involved in the authentication process. The Needham Schroeder protocol [5] is an example and described below:

In this protocol as described below A wants to communicate with B. A therefore needs to have the shared key from the authentication server S which is trusted by both users. A includes a nonce in the first message for freshness purposes. S creates a key K_{ab} and message 2. A decrypts the message successfully since he possesses the key K_{as}. A obtains the required share key K_{ab} and check that the message contain the nonce N_a. A passes on to B the other encrypted part of message 2. B then obtains K_{ab} in the same way and send a new nonce N_b encrypted by the obtained key K_{ab}. A decrypts this message using K_{ab} , forms $N_b - 1$, encrypts it and sends it to B. At the end of the protocol both users A and B possess the shared secret key K_{ab} generated by the server and believe that the other party has the key.

(1) $A \rightarrow S : A, B, N_a$; A sending to S a message containing A, B, N_a

(2) $S \rightarrow A : E(K_{as} : N_a, B, K_{ab}, E(K_{bs} : K_{ab}, A))$

(3) $A \rightarrow B : E(K_{bs} : K_{ab}, A)$

(4) $B \rightarrow A : E(K_{ab} : N_b)$

(5) $A \rightarrow B : E(K_{ab} : N_b - 1)$

Some authentication protocols establish shared encryption keys (session keys) that the users can use later. These protocols use an authentication server, which shares a key with each user and generates a new session key each time the users need to communicate. The Kerberos protocol [4] shown in Figure 2.9 is an example and works as follow. In the first message A sends a message to the authentication server S to obtain the session key between A and B. The server replies with a message containing the server time stamp, life time parameter, the session key K_{ab}, the B identity and a ticket all encrypted by the shared key between S and A. The ticket contains the time stamp, the life time parameter, A identity and the session key all encrypted by the shared key between B and S. A will forward the ticket and his ID and his time stamp encrypted by the obtained session key. B decrypts the ticket and obtains the session key then increments A's time stamp and sends the message encrypted by the new session key K_{ab}.

1. $A \rightarrow S : A, B$

2. $S \rightarrow A : (T_s, L, K_{ab}, B, (T_s, L, K_{ab}, A)K_{bs})K_{as}$

3. $A \rightarrow B : (T_s, L, K_{ab}, A)K_{bs}, (A, T_a)K_{ab}$

4. $B \rightarrow A : (T_a + 1)K_{ab}$

T_s and T_a are time stamps, and L is a life time. K_{as} is the shared key between user A and the server S and K_{ab} is the session key between A and B.

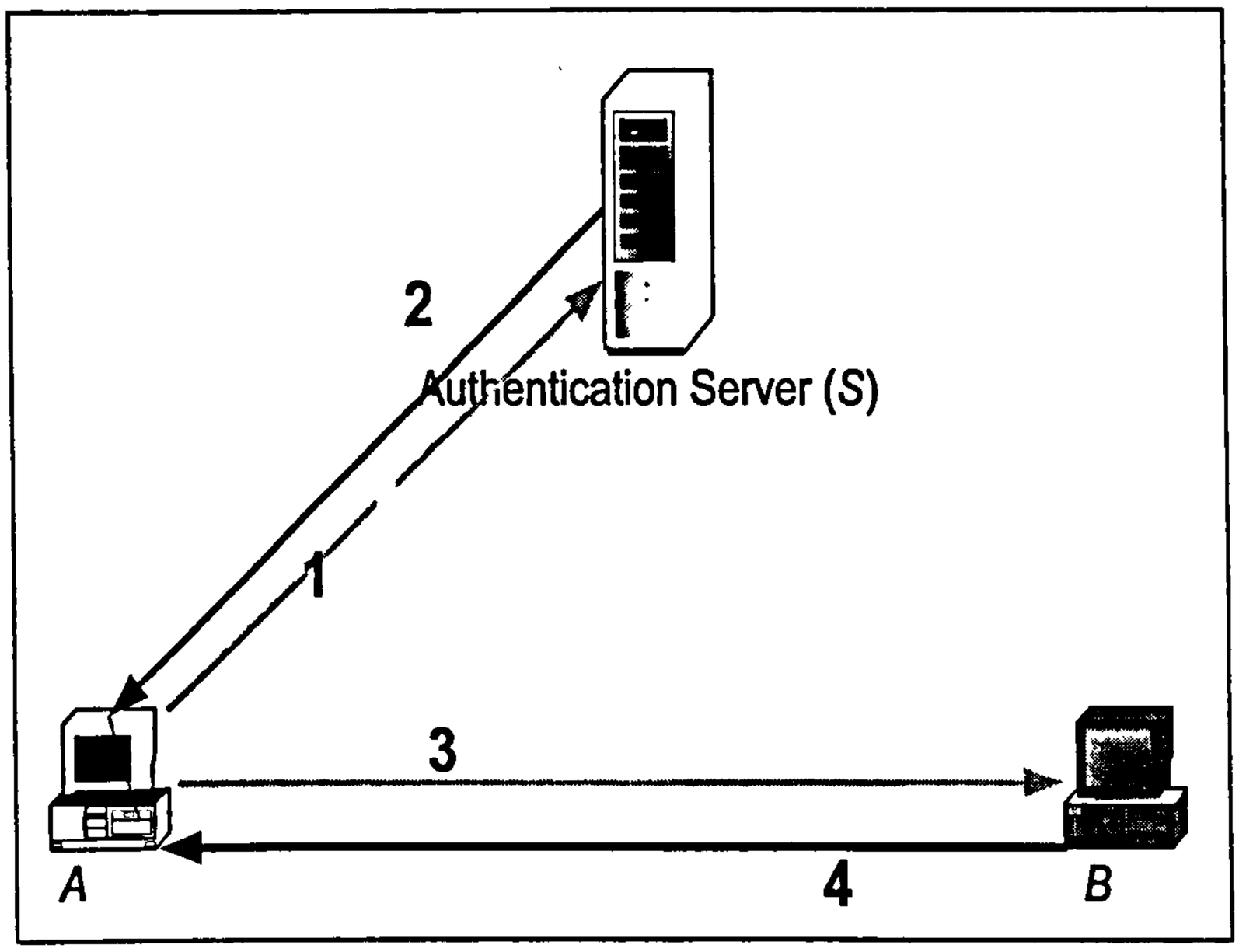


Figure 2.9: Kerbros protocol

2.10.3 Public key authentication protocols

There are some authentication protocols that use public key cryptography and find several applications. The processing speed of encryption and decryption in public key algorithms restrict the widespread use in communications. Needham and Schroder [5] proposed the following protocol:

The authentication server S some times called a certification authority stores the public keys of users and distributes them on request encrypted by its own private key Ks^{-1}. The protocol consists of 7 messages which are described as follows. In the first message user A sends a message to S requesting B's public key K_b. A decrypts message 2 to get B's public key and using that key A sends an encrypted message to B contains his ID and a nonce N_a as a challenge. In this case B needs to request A's Public key from the server. Thereafter B sends a message to A contains two nonces, N_a and N_b. A decrypts the message and getting N_a means that the previous message 3 was received by B and that he is communicating with B. A then sends B his nonce N_b. B then decrypts the message and checks that it contains his challenge and concludes that A is operational.

(1) $A \rightarrow S : A, B$

(2) $S \rightarrow A : E(Ks^{-1} : K_b, B)$ S encrypts the message contents K_b, B using his private key Ks^{-1}

(3) $A \rightarrow B : E(K_b : N_a, A)$

(4) $B \rightarrow S : B, A$

(5) $S \rightarrow B : E(Ks^{-1} : K_a, A)$

(6) $B \rightarrow A : E(K_a : N_a, N_b)$

(7) $A \rightarrow B : E(K_b : N_b)$

2.10.4 Hybrid Protocols

There are some protocols that use both symmetric and public key cryptography. An example of such protocols is the Encrypted Key Exchange (EKE) designed by Steve Bellovin and Michael Merritt [29]. EKE provides security and authentication on computer networks using both symmetric and public key cryptography. In this protocol a shared secret key is used to encrypt a public key. In general public keys are used to distribute and to encrypt shared keys. The EKE works as follow:

In this protocol two users share a common password (secret key P). The objective is to use the EKE protocol to generate a session key K. In the first message A sends his ID and public key K' encrypted by P. B decrypts the message using P and obtains K'. B then generates the required session key K encrypted twice with the public key received from A and the secret key P. The steps from 3 to 5 are used as a challenge-response proof between the two users. Both A and B can communicate using the new generated session key K.

1. $A \rightarrow B : A, E_P(K')$

2. $B \rightarrow A : E_P(E_{K'}(K))$

3. $A \rightarrow B : E_K(R_A)$

4. $B \rightarrow A : E_K(R_A, R_B)$

5. $A \rightarrow B : E_K(R_B)$

2.11 Some existing authentication implementations

In this section some implementations of existing authentication protocols are discussed. These implementations have been selected arbitrarily to cover different applications with different architectures, which have been used either in fixed or wireless networks. The implementations investigated include authentication using PKI systems, secure shell protocol and GSM authentication. Firstly, a brief description of each example is given. Thereafter, each example is analyzed based on certain metrics. These metrics include security, availability and trust. In doing so, an overview of some authentication implementations and architectures are gained.

2.11.1 PKI systems

Public key infrastructure (PKI)

Public key cryptography becomes an essential requirement to establish a secure channel between users over an open network like the Internet. Public key infrastructure can be defined as the use of the public key technology. The public and private keys can be used for providing the most required security services such as authentication, encryption, and digitally signing electronic data. The goal of PKI is to enable secure, convenient and efficient discovery of public keys. PKI is used to manage keys and certificates on behalf of users and application. If Alice wants to communicate with Bob securely, Alice needs to know Bob's public key. The critical question is how Alice is sure that a certain key really does belong to Bob. The answer for this question yields the need for a trusted CA. The CA is required by PKI to securely issue the certificates. As described in Section 2.3 the certificate is a structured data that bind the subject name with its public key. The binding is asserted by having a trusted CA digitally sign each certificate. Certificates are typically stored in a directory system, with applications using

the Lightweight Directory Access Protocol (LDAP) to retrieve them as needed [14]. There are several aspects need to be considered in developing a PKI protocol include the following:

- **Certificate contents**

  The certificate content or fields should be enough to provide the required information, but very long certificates are not preferred due to the overhead and efficiency issues. The certificate format most adopted is X.509 version 3 certificates, which has been published by International Telecommunications Union (ITU-T) [30].

- **CA arrangement**

  It is impractical to have a single CA handling the authority for the entire world. Therefore most PKIs have several CAs dealing with each other according to a certain relation in such away that CAs certify other CAs. When one CA certifies another CA, this means that the users of the former can trust the second CA certificate. Different PKIs use different ways and hierarchies of CAs arrangement. The CAs arrangement is a basic PKI characteristic and governs the PKI scalability [23].

- **Trust model**

  Trust is an essential point in communication relations between individuals, entities, organizations, etc. Trust becomes more important in critical applications such as commercial applications. In PKI it is crucial that the receiver of a certificate is confident that the certified sender is the real holder of the private key that has been used to sign or encrypt the message. This implies the answer of the question: how is the receiver to trust the certificate authority CA that issues the certificate? There are various PKIs which have been proposed. Each PKI is differs from others in terms of their certificate contents and the trust model. An overview of PKI trust models is found in [31].

- **Certificate validation**

  Certificate information can change or update over time. A certificate user needs to verify that the received certificate is valid. Although the certificate may, have a validity period, for a certain reason it might be revoked. In

general, to validate a certificate could imply validating the entire chain of included certificates . Such chains called certification paths.

- **Certificate revocation**

  A certificate is considered revoked if the validity period is expired. However, various circumstances may cause a certificate to become invalid before the expiration of the validity period. Such circumstances may include change of user name or other information, a compromise or suspected compromise of the certificate information particularly the user private key, etc. The approaches used for certificate revocation include online revocation and the use of certificates revocation lists (CRLs). A CRL is a list of revoked certificates that is signed and periodically issued by a CA [24]. It is essential that the user checks the latest CRL to validate a certificate.

In addition there are other aspects can be considered as a further required elements for a PKI such as certificate repository, key backup and recovery, support for nonrepudiation of digital signature, support for cross-certification and automatic update of key pairs and certificates [32]. Cross-certification means that one CA in certain domain can certify another CA in another domain.

2.11.1.1 Examples of PKI protocols

- **Privacy enhanced mail (PEM)**

  The Privacy enhanced mail (PEM) published in 1993 includes specifications for a public key infrastructure based on X.509 v1 certificates [33]. The intention was to provide a secure internet email using public key cryptography. The infrastructure defined a rigid top-down structure of CAs as shown if Figure 2.10. The Internet Policy Registration Authority (IPRA) acts as the root of the PEM certification within the Internet. Beneath IPRA are Policy Certification Authorities (PCAs). Each PCA is certified by the IPRA. Below PCAs, there are Certification Authorities (CAs) to certify users and organizational entities. PEM PKI hierarchy recommended a single root key, which everyone would know a *priori*. This rigid top-down model, with predefined policies and procedures for operating CAs, prevented PEM from achieving significant deployment [34].

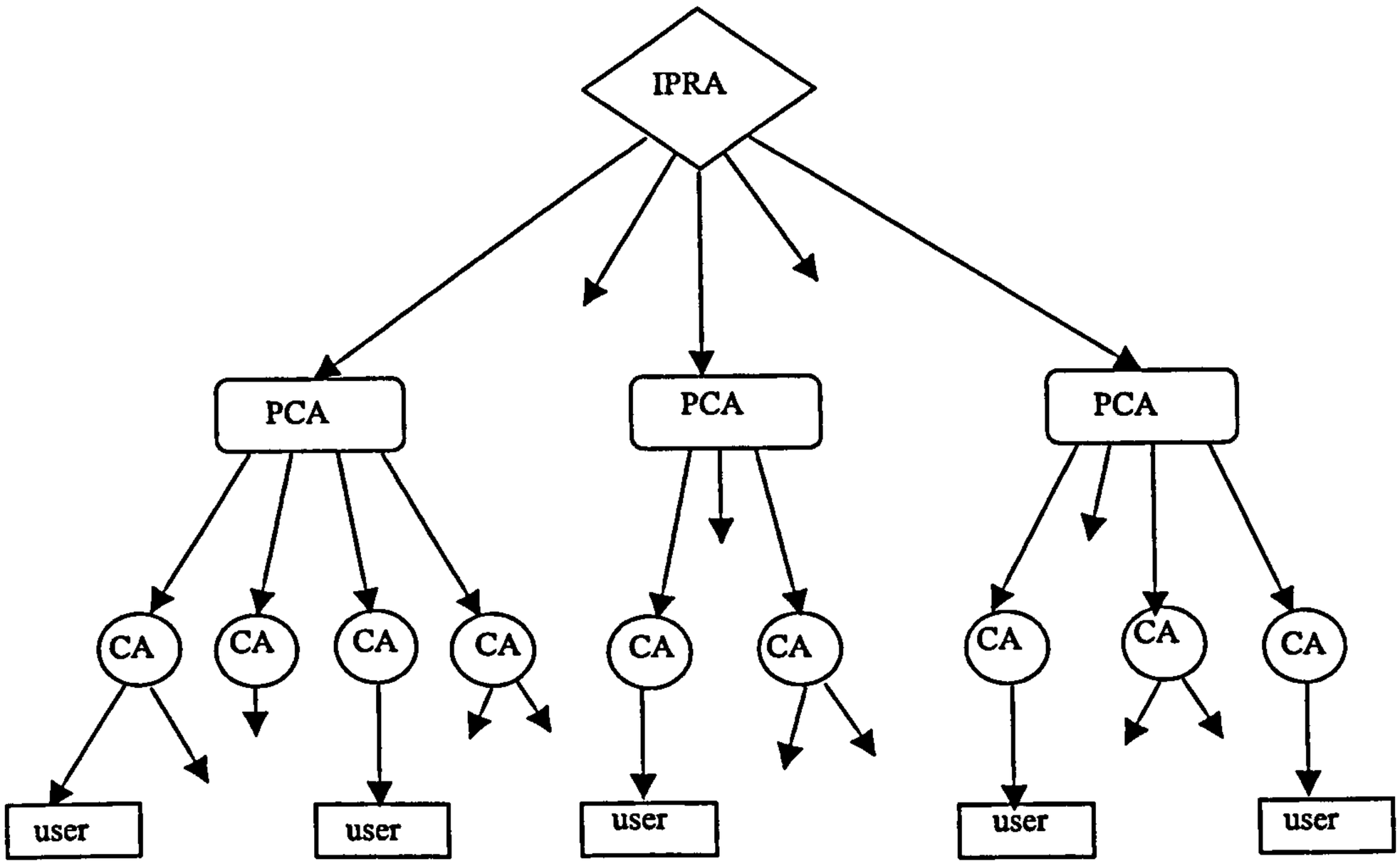


Figure 2.10: PEM trust model

- **Pretty Good Privacy (PGP)**

  PGP is a public key cryptography program originally designed for Internet email usage. PGP has wide deployment because the software is free and no certification infrastructure was needed before anyone could use PGP in a secure manner [34]. By downloading a copy of the software, two users can generate their own key pairs and exchange their public keys using off line process (meeting). Each user represents a root CA. A PGP certificate contains only an email address, a public key value and a degree of trust attribute. The PGP is a public trust model, since any member of the public may set their own trust chains to other PGP users. This lead to a web of trust chains being set up between PGP members. If one user trusts any person in the web, he must trust the entire web [23]. This is considered as a trust problem. Additionally PGP does not provide strong authentication of a person's identity since an email address is not enough. In PGP the certificate is not extendable and does not have a validity period. However, PGP is not suitable for critical applications in the absence of strong authentication and a good trust model.

- **Internet X.509 Public Key Infrastructure (PKIX)**

PKIX has been developed by Internet Engineering Task Force (IETF) as public key infrastructure for Internet applications. It is based on X.509 version 3. The main components used by PKIX are [21]:

- *End Entity (EE) :* User of PKI certificate and /or end user system that is the subject of a certificate.
- *CA* : Certificate Authority to issue and revoke certificates
- *RA :* Registration authority, an optional system to which a CA delegates certain management functions
- *Repository:* a system or collection of distributed systems that store certificate and CRLs and serves as a means of distributing these certificates and CRLs to end entities.

The PKIX working group has developed the following specifications:

Certificate and CRL profile: Profiles that describe in detail the X.509 v3 certificate format and the X.509 v2 CRL format.

Operational protocols: This specification describes the required operational protocols required to deliver certificates and CRLs including distribution procedures based on LDAP, HTTP, FTP and X.500 [35].

Management protocols: Management protocols are required to support on-line interactions between PKI user and management entities. The possible set of functions are: registration of entity, initialization, certification, key-pair recovery, key-pair update, revocation request and cross-certification [22].

Policy and certification practice statement: A certificate policy may be used by a certificate user to help in deciding whether a certificate, and the binding therein, is sufficiently trustworthy for a particular application. A certification practice statement (CPS) is a statement of the practices which a certification authority employs in issuing certificates.

In PKIX the certificate information is extensible which is considered as a feature. The CAs are arranged using general hierarchy with cross-certification. Each user is expected to fully trust at least one CA. CAs can constrain how their trust in subjects and other CAs is delegated. Certificates can be revoked using CRLs.

2.11.2 Secure shell (SSH)

SSH is a protocol for secure remote login and other secure network services over an insecure network [36]. SSH provides authentication, encryption and data integrity to secure network communications. It offers transport layer security. The client sends a service request once a secure transport layer connection has been established. The authentication process involves user authentication and host authentication. The secure shell working group of the IETF produced a new authentication method "keyboard-Interactive authentication" where the authentication data should be entered via a keyboard [37]. The user authentication methods are:

- password authentication.
- public-key authentication.
- server-based authentication.
- certificate authentication.
- kerberos authentication.
- pluggable authentication module (PAM).
- secureID

More details about these methods are found in [38, 37, 39]. In the host authentication, each server host should have a host key. A host key is used by a server during key exchange to prove its identity to a client and by a client to verify that it is really talking to the correct server. For this to be possible, the client must have a priori knowledge of the server's public host key. The host keys are changed infrequently. The authentication process includes the following steps: the client sends an authentication request and the server reply with its host key. The client will verify the server host key either against its own database or through a trusted certification authority [36]. Then both the client and the server will generate a secret key (session key) and both will agree on the required algorithms. Session keys are generated after a host authentication is successfully performed but before user authentication, so that username and passwords can be sent encrypted.

Both entities start using the session key and the connection is encrypted. In this stage, the server has been authenticated, and encryption and integrity protection are in use in the transport layer. In the next stage, the user authentication will start. The client starts the user authentication and sends requests to the server. The server replies always success or failure. In this stage one of the user authentication methods will be used. Figure 2.11 depicts the authentication using SSH protocol.

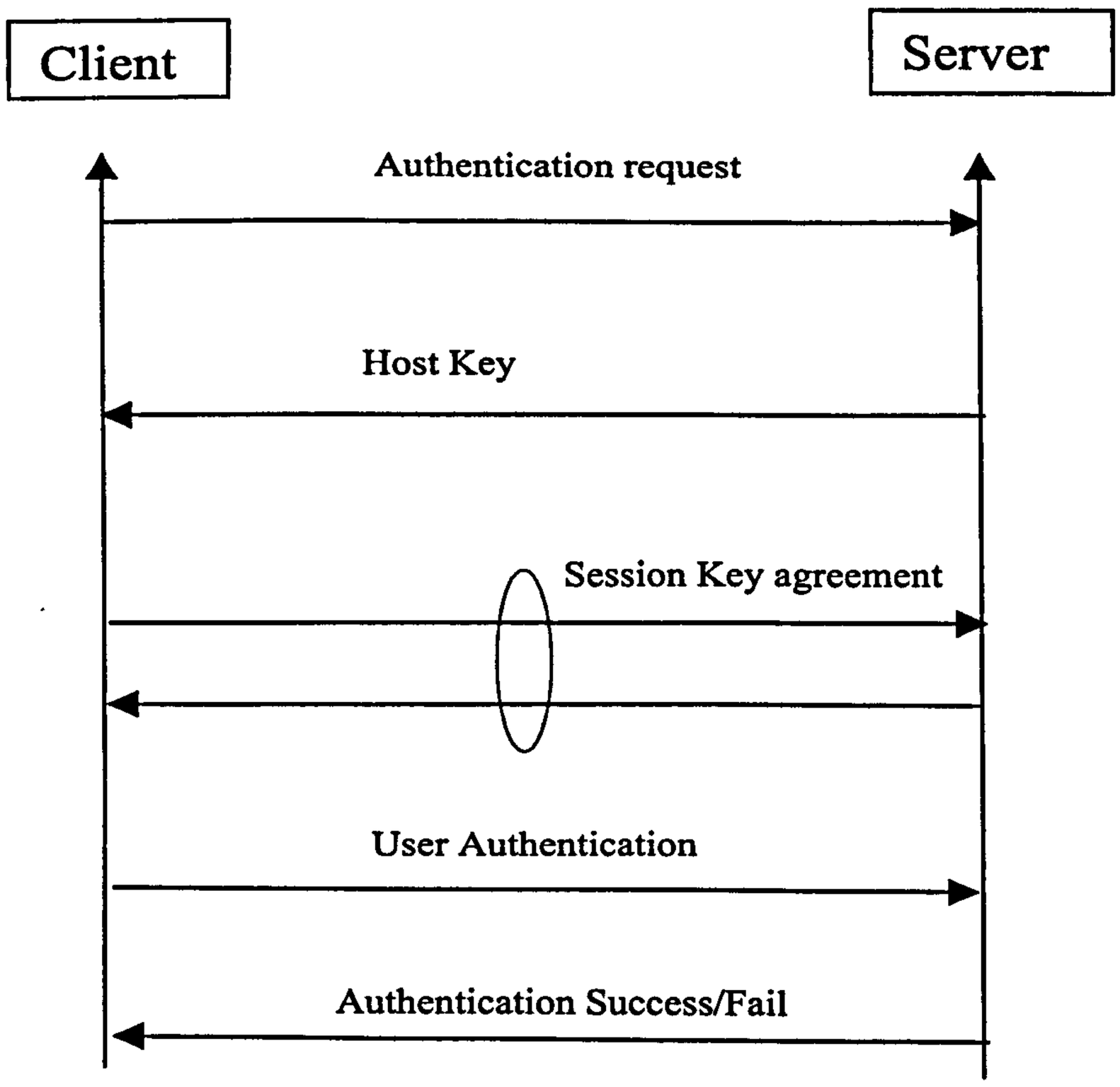


Figure 2.11: SSH authentication

2.11.3 Mobile Networks

To see how is the authentication process works on mobile networks, the authentication on the Global System for Mobile communication networks (GSM) will be taken as an example. In GSM, authentication of the user is necessary to prevent access by unregistered users which achieved by checking the validity of

subscriber's SIM card. The authentication is based on the authentication algorithm A3, which is stored on the SIM card and also in the authentication center (AC) on the network. It is a challenge- response authentication process. In GSM, the A3 algorithm uses two input parameters: one is the secret key, K_i, which is stored in the SIM card and in the network. This secret key is unknown to the user. The second value, a Random Number (RAND) is transmitted to the mobile station as a challenge. The mobile station passes the RAND to the SIM card. The A3 algorithm in the SIM card using the two inputs K_i and RAND will calculate the Signature Response (SRES). The mobile station will send back the SRES to the network as a response to the challenge. The network uses the same RAND, K_i, and A3 to produce a SRES, which is checked against the response from the mobile station [40]. Figure 2.12 depicts the GSM authentication procedure.

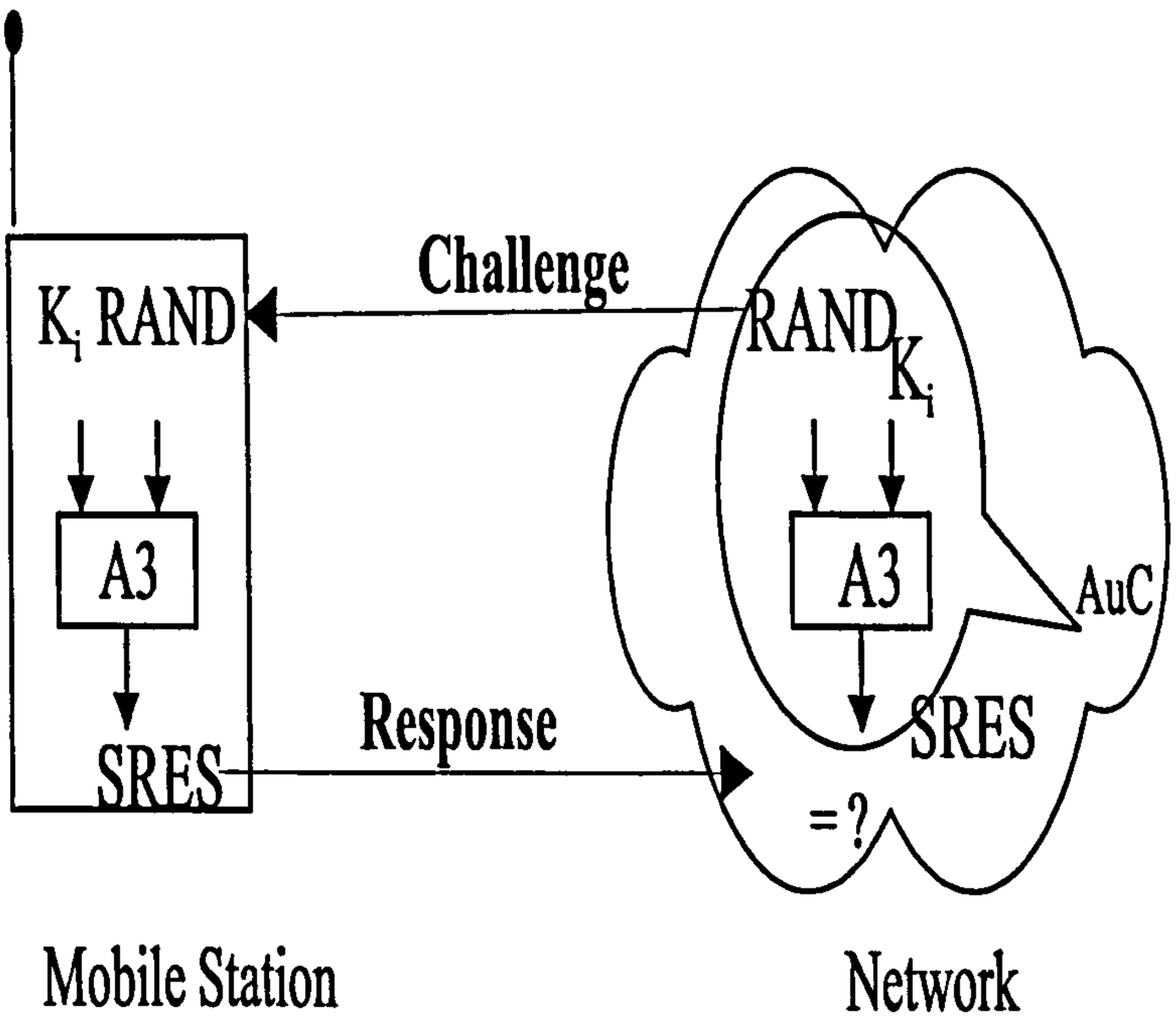


Figure 2.12: Authentication in GSM

2.12 Analysis and deficiencies

In this section, the major drawbacks of the above protocols will be analysed. Each protocol has its individual deficiencies. Certain metrics including security,

availability and trust model will be used to evaluate these different protocols.

2.12.1 PKI

The analysis of PKI systems; PGP, PEM and X.509 is conducted in the following paragraphs such that in each element of the given metrics the three systems are analyzed.

Security: The PGP uses email addresses to authenticate users, which is considered as weak authentication. PGP is suitable for casual applications like email but it is unsuitable for critical applications. The PEM PKI uses IPRA as a single root to certify all users and if the root private key has been compromised, then the whole system is compromised. The security is affected by the complete trust placed in the IPRA by all users. With respect to the PKIX, many attempts have been undertaken to make it the most adopted PKI for Internet applications. The developers tried to overcome the drawbacks and limitations of other PKI models such as PGP and PEM. In PKIX the root CA is the CA that is directly trusted by the end entity and could be any CA. Using X.509v3 adds more constraints which help provide more control, for example, the certification path constraints and certificate policies can restrict the growth of a certification path as well as the trust. Although this seems advantageous, the point is that in the end, a certain CA is controlling the next further steps since it is fully trusted. In other words, the trust in PKIX is a centralized trust, which increases risk and decreases security in case that the trusted CA is attacked or compromised.

Availability: In the case of PGP there is more freedom since a single user can obtain the public key of a certain user from several users hence a fault at one of them is not a problem. In the PEM PKI, if the required PCA is down then the user is affected. The same concept works in PKIX, if the trusted root CA is down the user is affected because at the end, one CA is controlling the action.

Trust model: The PGP trust model is called a public trust or a web of trust since each user is a root CA and he can trust another user without any prior knowledge or strong identification. This considered as trust problem and makes PGP unsuitable for critical applications. In the PEM model, all the users must place trust in the IPRA, as all certification paths start with the IPRA's key. A

user must also trust the PCAs and CAs in a certification path [23]. In PKIX, each user is expected to fully trust at least one CA. With the new extensions in X.509 v3, CAs can constrain the delegation of trust in subjects and other CAs. It is also considered as centralised trust because only a single CA is the decision-maker and if it is compromised, the user transactions is compromised. There is certificate validation, which comprises a chain of certification path to verify that the user is having the correct public key of trusted CA. Figure 2.13 depicts a PKI centralised-based trust model and the user certification path is shown in figure 2.14. It appears advantageous that there is a scheme or a mechanism to validate the certificate or to verify the root CA keys that issues the certificate. This can be done, but the essential point is that by the end there is only one entity or root CA that the user should trust. In other words, the trust is maximized on this entity and it is unconditionally trusted. If the trust is maximized on one entity, the risk will increase. Accordingly scenario or trust model will affect the system security. Due to centralised trust if the trusted entity is compromised then the further operations compromised. This centralised trust model may introduce a lot of risk and problems particularly in critical applications.

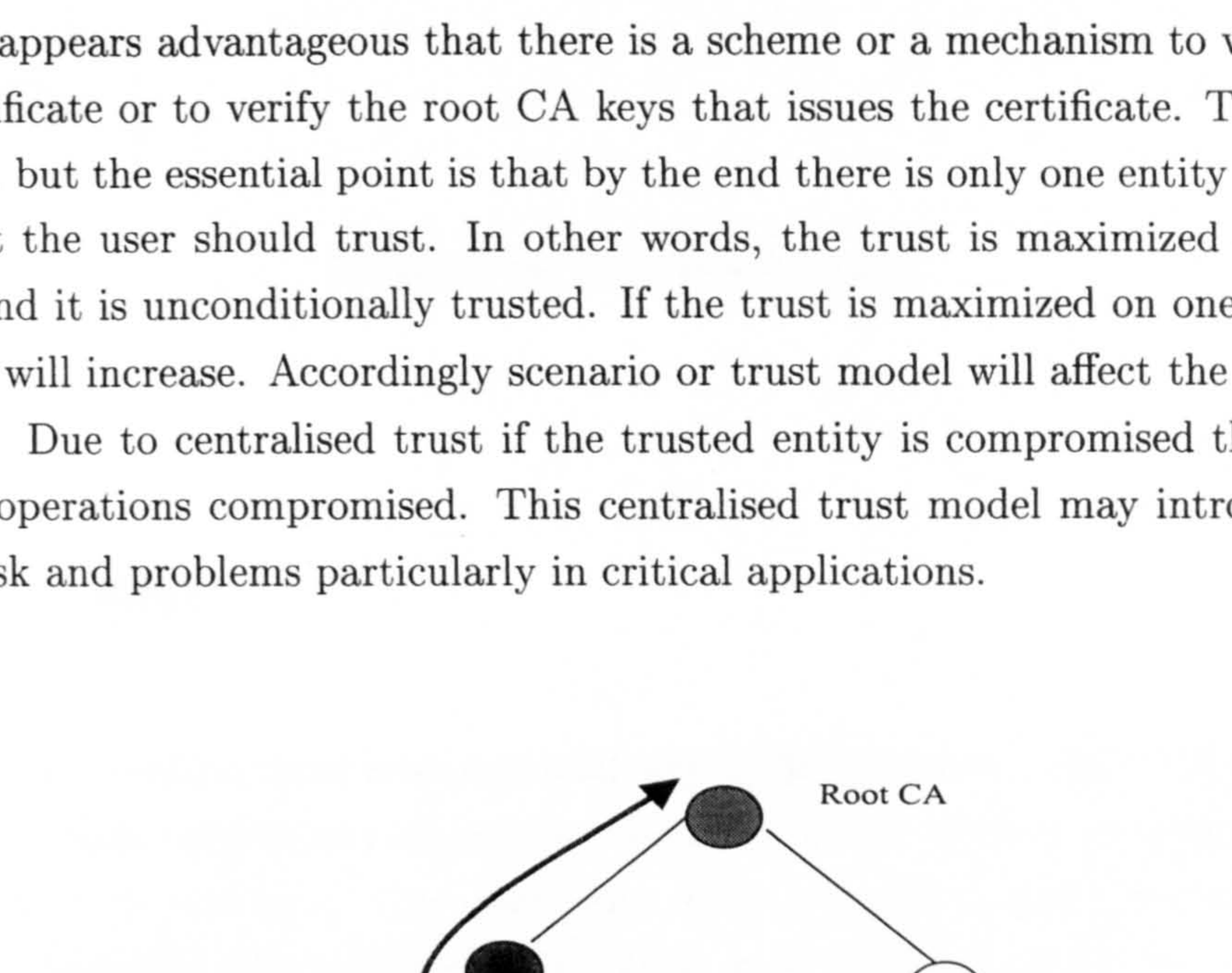


Figure 2.13: PKI centralized trust model

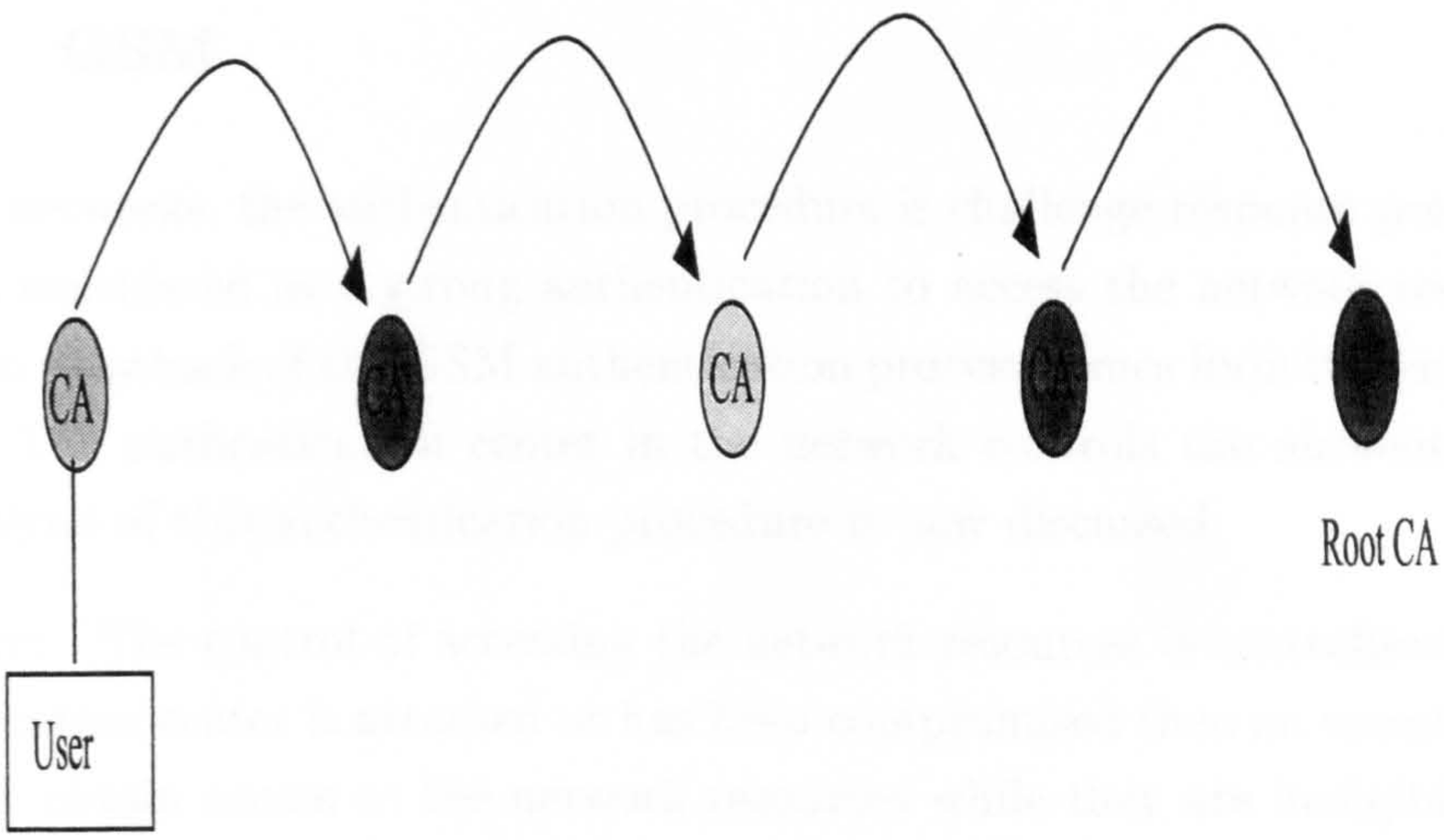


Figure 2.14: Certification path

2.12.2 SSH

The server authentication is an essential step in SSH protocol. The Client will be unable to make any secure connection with the server without prior knowledge of the server's host key. The client can store the server host key along with the corresponding server name in the client local database, which may become difficult to maintain. In other option is to obtain the server host key from a trusted third party. SSH may be formally assessed using the previously introduced metrics.

Security: As a security consideration there are two point; first, how to verify that the server host key really belong to the desired host. Second, if the server host key has been compromised or the third party is attacked then the whole operation is compromised.

Availability: Due to the existence of one centralised third party, if the third party breaks down then the authorized users will be unable to obtain the access.

Trust model: In SSH protocol a lot of trust is placed on the third party. It is a central infrastructure which increases the risk and results in a decreased security.

2.12.3 GSM

In GSM networks, the authentication procedure is challenge-response procedure, which is considered as a strong authentication to access the network resources. The main drawback of the GSM authentication process comes form its centralised nature. The authentication center in the network controls the authentication. The analysis of this authentication procedure is now discussed:

Security: The control of accessing the network resources is centralised. If the authentication center is attacked or has been compromised then an unauthorized user may obtain access to the network resources while they are ineligible. This may yield many financial costs. This is because one entity is taking the control and the permission grant, which can be considered as security weak point.

Availability: Since there is a single entity, which authenticates and controls the access of the mobile stations, the authorized users will be unable to obtain access if that entity breaks down. In this case the service will not be available to users.

Trust model: In GSM networks, the authentication center is unconditionally trusted. It is a centralised trust model, which is not recommended.

From the above analysis, all the described authentication protocols can be considered as a centralised infrastructure since the trust models used are centralised-trust models. The centralised trust model is a drawback in these models and a possible solution is to have authentication and access control protocols based on distributed-trust models. Figure 2.15 depicts a distributed-based trust model. The goal is to not have one central entity (server) that is trusted. This thesis introduces such a proposal.

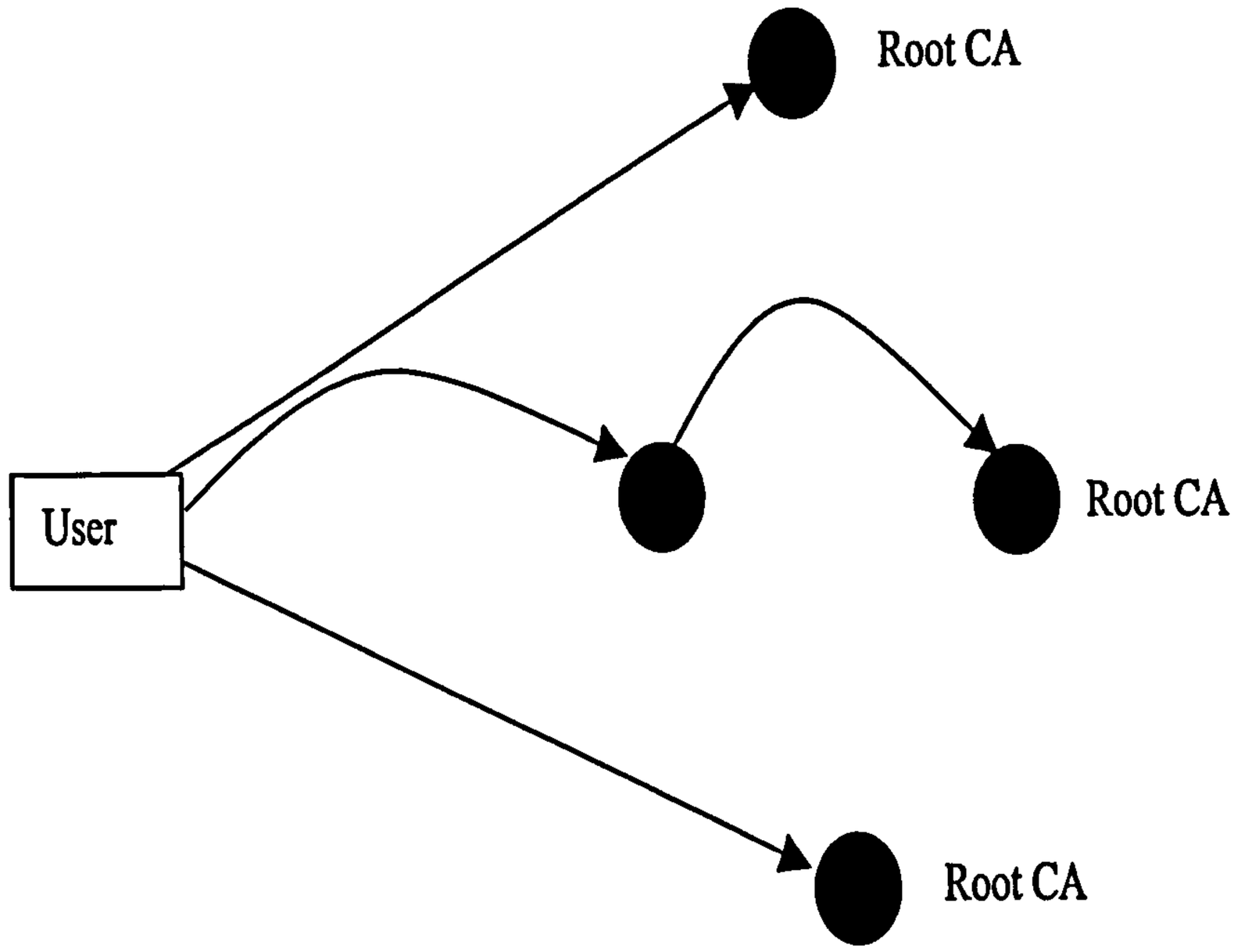


Figure 2.15: Distributed trust model

2.13 Discussion

From the above authentication background it is clear that the traditional methods for authentication are based on centralized authentication using a single authentication server or certificate authority that is unconditionally trusted. Trust is an essential point; in the traditional authentication and access methods trust is focussed on a single entity (third party). In summary, centralized authentication approaches have the following drawbacks:

- **Security bottleneck:** if the third party is compromised the whole system is compromised.

- **Availability:** the probability that a network or a system can perform its required function at any instant of time. In the traditional authentication protocols the probability that the third party breaks down is high which means that for a certain time the resources will not be available for eligible users.

- **Trust:** the third party has unconditional trust. In general, if trust is increased, risk will increase. There are some applications in which users do not trust each other or a third party and the above schemes do not satisfy their needs.

- **Efficiency:** in the above mechanisms, the load is concentrated on one entity, which may increase the processing time, communication overhead and capacity requirements.

2.13.1 Duplicating the centralized server

The traditional solution for centralized systems is to duplicate the centralized server. In other words, instead of having one centralized server is to have two servers each one is a copy (backup) for the other one. In this case users will contact both servers. The analysis of having parallel servers is follow. This model increases the amount of data that must be secured and therefore the risk of compromise is high compared to centralized system. This means that the security is decreased. The availability is increased in this approach since if one server becomes unavailable users can contact the other one. This approach introduces management overhead in order to maintain consistency between the servers data. The trust requirement is increased since the users must trust more entities. The efficiency is improved but there is management overhead. The only advantage is increasing the availability.

2.13.2 Distributed approach

It is clear that authentication is one of the major security services. However, this chapter shows the drawbacks of centralized authentication protocols where a centralized authentication server (CAS) is handling the authentication process. The centralized approaches has the advantage of decreased management overhead. However they are not suitable for the recent and future critical applications, which may be security dependent such as e-commerce and any critical application. Therefore, there is a demand of developing and constructing distributed authentication protocols, which must overcome the centralized authenti-

cation protocols's deficiencies. In distributed based authentication protocols the user needs to contact more than one authentication server (AS) to be authenticated. This means that each authentication server is partially participating in the authentication process by sending the corresponding share (S_i) such that the provided shares can reconstruct the required access code. Therefore, no single server can grant the authentication on its own. Figure 2.16 shows the general concept of distributed authentication protocols compared to centralized approaches where each authentication server partially participates on the authentication process.

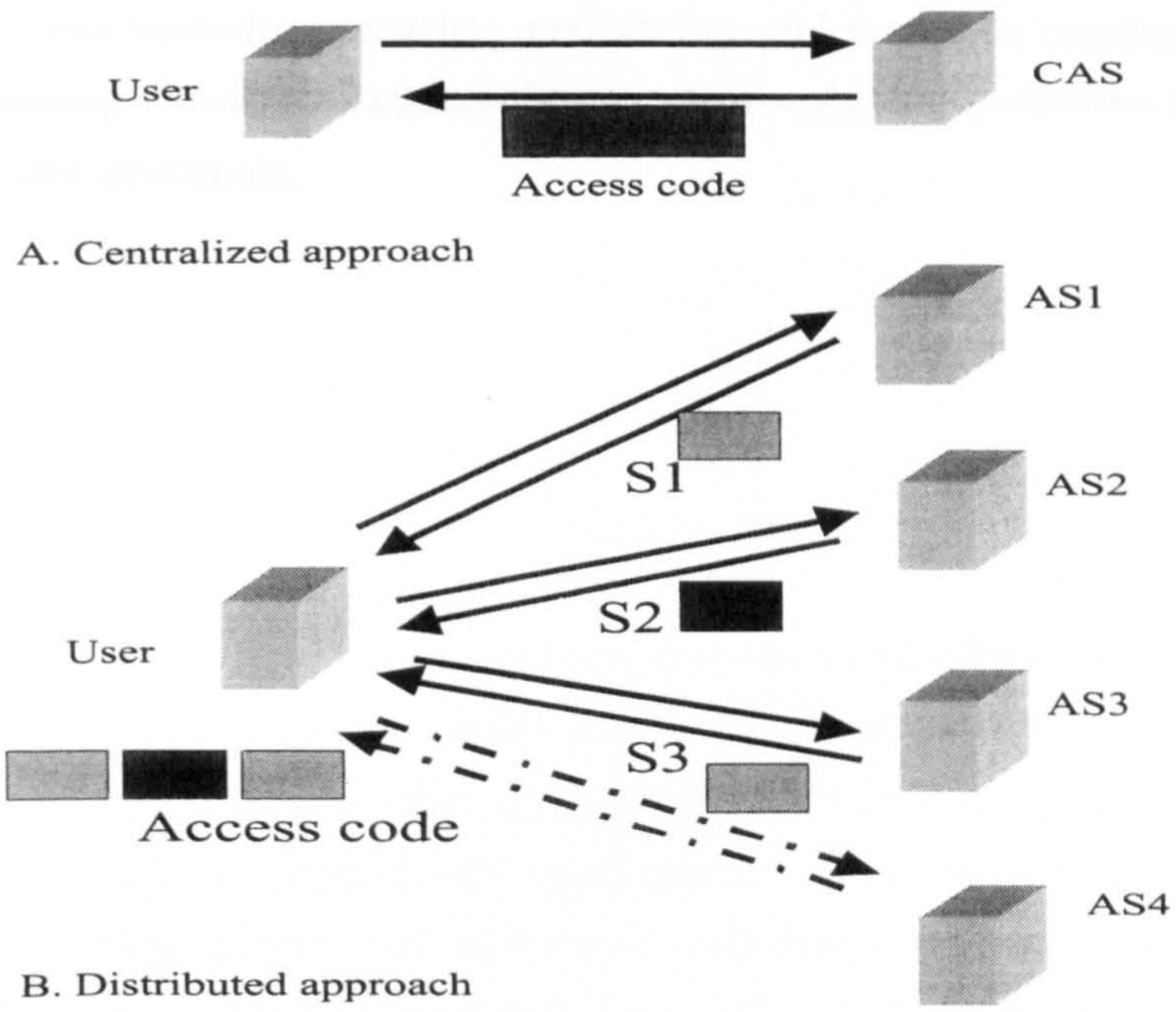


Figure 2.16: Centralized and distributed approaches

The use of a distributed directory to supply certificates can solve the availability problem but introduces management drawbacks. Obtaining multiple certificates from different CAs results in issues relating to revocation and trust, as well as increased management overhead. Secret sharing and quorum systems are tools that can overcome these drawbacks and have several applications that deal with the above issues.

2.14 Conclusion

Authentication and access control are crucial in network based and distributed systems. This chapter has provided a general background about security and authentication, starting with an overview about the general security requirements and mechanisms, before detailing network requirements. Since authentication is key to this thesis, a detailed background has been given and authentication methods described. In addition the main types of authentication protocols have been summarized. The chapter has investigated some real examples of certain authentication implementations. These examples have been analyzed based on certain metrics including security, availability and trust. In conclusion there is a need to develop distributed authentication protocols that overcome the drawbacks of centralized protocols.

Chapter 3

Secret sharing schemes and quorum systems

3.1 Introduction

Centralized authentication systems have certain drawbacks as described in Chapter 2. The solution is to have distributed authentication systems. Secret sharing and quorum systems shows promise in allowing distribution of security, reliability and availability in network-based applications. They have been used in several applications: distributed and replicated databases, mutual exclusion, and distributed access control and authentication. The implementation of both systems in networks is a crucial topic and requires more investigation and research.

As mentioned in Chapter 2 there is a need to have a distributed authentication protocols. Secret sharing and quorum systems are good tools to develop a distributed authentication and access control protocols. This chapter presents the concepts and applications of secret sharing and quorum systems. Section 3.2 contains the definitions of secret sharing schemes, secret sharing aspects and constructions. Section 3.3 describes secret sharing applications. Section 3.4 presents the definition of quorum systems, their measures, and constructions. Section 3.5 contains applications of quorum systems and Section 3.6 describes some quorum secret sharing protocols.

PAGE NUMBERS CUT OFF

IN

ORIGINAL

3.2 Secret Sharing

The basic idea of secret sharing is to divide a secret into pieces called shares and distribute these pieces amongst users such that the pooled shares of specific subsets of users allow reconstruction of the original secret, but any other subsets have absolutely no information on the secret. All subsets that are capable of reconstructing the original secret are called qualified subsets (see Figure 3.1). The collection of all subsets of participants (users) qualified to reconstruct the secret is usually called the *access structure* of the secret sharing scheme. The access structure is said to be monotone if any set which contains a subset that can recover the secret, can itself recover the secret [41]. As an example an (m, n) threshold secret sharing scheme divides a message (secret) into n pieces, called shares, such that any m shares can be used to reconstruct the message, but any $m-1$ of them cannot. The threshold scheme is a special case of a secret sharing scheme.

An *ideal secret sharing scheme* is a scheme in which the size of the share given to each participant is equal to the size of the secret [42]. A *perfect secret sharing scheme* is a scheme in which an unqualified subset of participants provides no information about the secret [43].

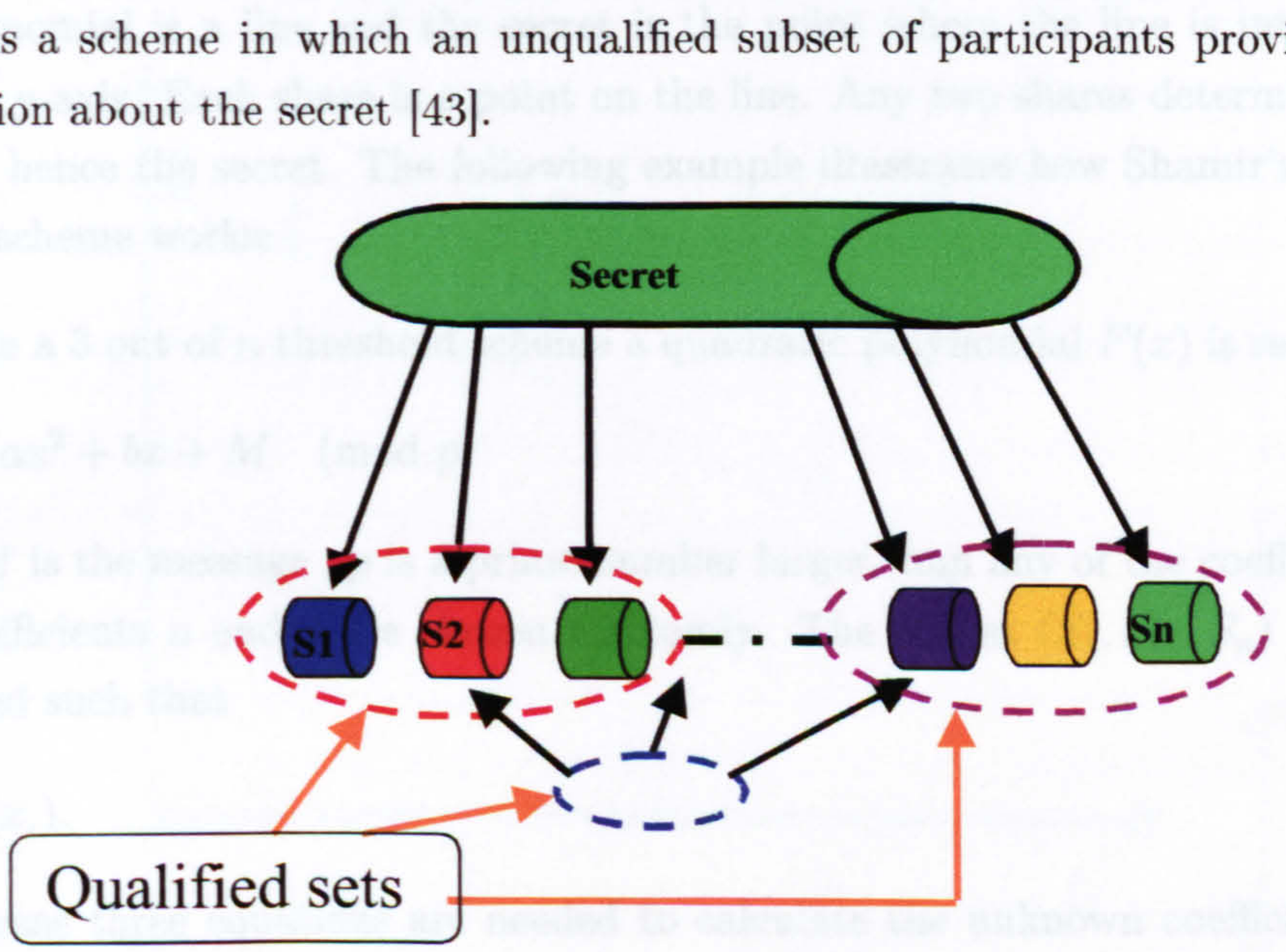


Figure 3.1: Secret sharing schemes

Secret sharing schemes were originally designed to be used for key management

in cryptosystems to avoid the following scenarios: Firstly, if there is one key and that key has been lost then the authorized users will not be able to access the system. Secondly, the compromising of the key will give unauthorized users the ability to use the system. Secret sharing schemes facilitate distributed trust and shared control for critical applications. A secret sharing scheme may serve as a shared control scheme if inputs (shares) from two or more users are required to enable a critical action. The well-known two-man control rule that the United States enforces for critical military applications (nuclear weapons) is an example of shared control schemes [42]. Shamir and Blakley discovered secret sharing schemes independently in 1979 [44, 45] and both schemes are m out of n secret sharing schemes. Shamir's secret sharing scheme [44] is based on polynomial interpolation. An $(m-1)$-degree polynomial over the finite field GF(q) where q is a prime number is constructed such that the coefficient is the secret and all other coefficients are random elements in the field. Each of the n shares is a point (x_i, y_i) on the curve defined by the polynomial, where x_i is not equal to 0. The coefficients are randomly chosen in the field GF(q). Given any m shares the secret can be computed. A special case where $m = 2$ is given in Figure 3.2, the polynomial is a line and the secret is the point where the line is intersects with the y-axis. Each share is a point on the line. Any two shares determine the line and hence the secret. The following example illustrates how Shamir's secret sharing scheme works:

To create a 3 out of n threshold scheme a quadratic polynomial $F(x)$ is required.

$$F(x) = ax^2 + bx + M \pmod{p}$$

where M is the message , p is a prime number larger than any of the coefficients. The coefficients a and b are chosen randomly. The shares $(S_1, S_i, ..S_n)$ can be generated such that

$$S_i = F(x_i).$$

In this case three equations are needed to calculate the unknown coefficients a, b, and M. For example, to reconstruct a 3 out of 5 threshold scheme let $M = 15$, p =17, $a = 5$ and b= 11. The quadratic equation is

$$F(x) = 5x^2 + 11x + 15 \pmod{17}$$

the five shares are:

$S_1 = F(1) = 5 + 11 + 15 = 14 \pmod{17}$

$S_2 = F(2) = 20 + 22 + +15 = 6 \pmod{17}$

$S_3 = F(3) = 45 + 33 + 15 = 8 \pmod{17}$

$S_4 = F(4) = 90 + 44 + 15 = 13 \pmod{17}$

$S_5 = F(5) = 125 + 55 + 15 = 8 \pmod{17}$

To reconstruct M from three shares for example $S2, S4$ and $S5$ we need to solve the following linear equations using interpolation:

$6 = a2^2 + b2 + M \pmod{17}$

$13 = a4^2 + b4 + M \pmod{17}$

$8 = a5^2 + b5 + M \pmod{17}$

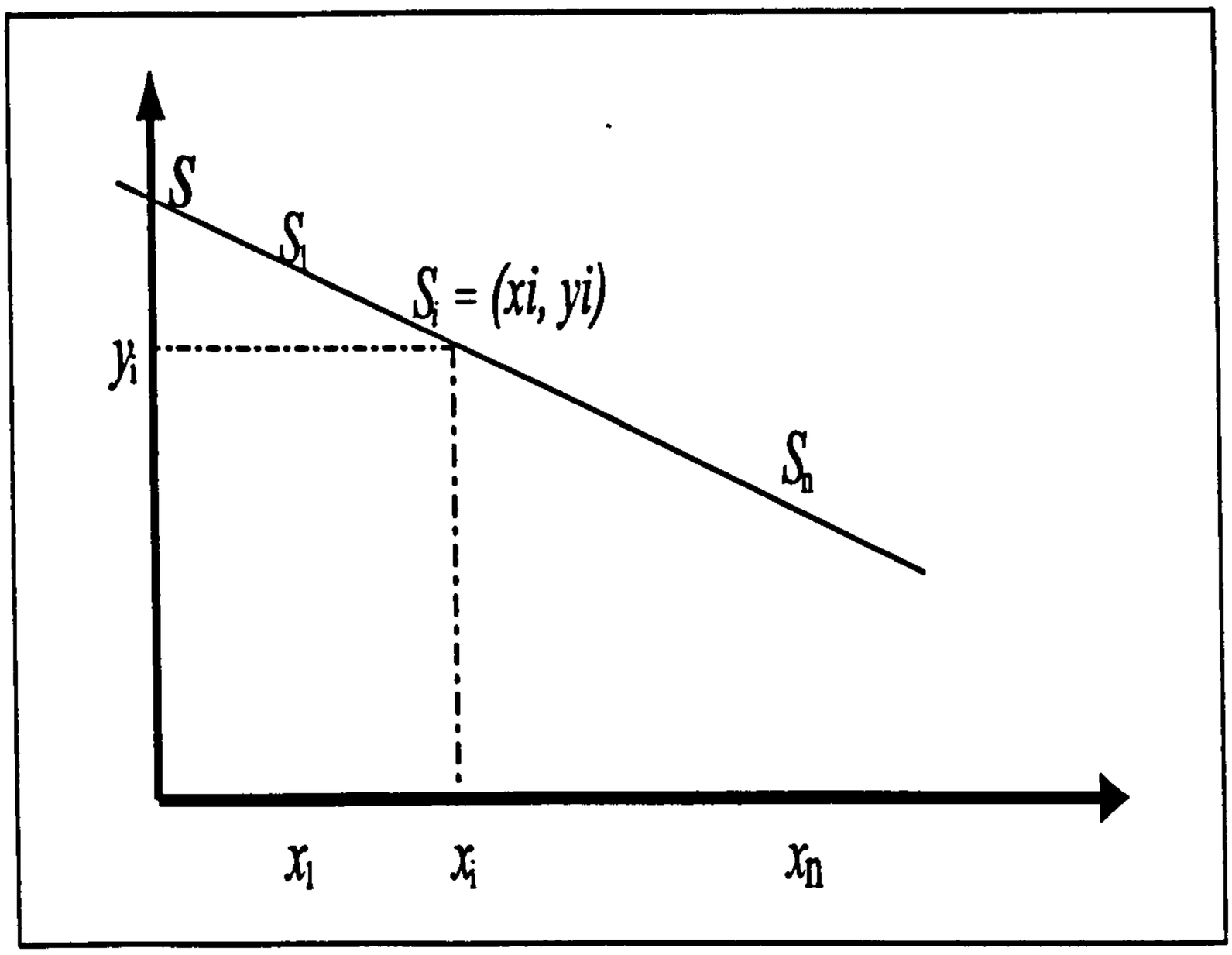


Figure 3.2: Shamir Secret Sharing Scheme

In Blakley's secret sharing scheme [45] the secret is a point in an m-dimensional

space. In this scheme each share defines a hyper plane in this space and the secret is the point of intersection of the hyper planes. By finding the intersection of any m planes the secret can be computed.

Secret sharing schemes facilitate distributed trust and shared control for critical applications. A secret sharing scheme may serve as a shared control scheme if inputs (shares) from two or more users are required to enable a critical action. Shamir and Blakley discovered secret sharing schemes independently in 1979 and their schemes are both m out of n threshold secret sharing schemes.

3.2.1 Secret Sharing Aspects

Ito, Saito, and Nishizeki [46] described a more general method of secret sharing that realizes secret sharing schemes for arbitrary monotone access structures. Benaloh and Leichter [47] gave an efficient way to realize schemes for structures represented by monotone functions. In secret sharing schemes, the size of the share is an important issue for several reasons: security, capacity requirements and efficiency [48, 49, 50]. In general, the shares of a secret sharing scheme must be at least as long as the secret itself. Krawczyk [51] proposed an m-threshold scheme, in which shares corresponding to a secret S are of size $|S|/m$ plus a short piece of information whose length does not depend on the secret size but on a security parameter. Simmons [42] discussed secret sharing schemes and secret shared control schemes. His article considers constructing concurrence schemes, the geometry of shared secret schemes, and setting up shared secret schemes in the absence of a trusted party or authority.

In anonymous secret sharing schemes [52], the secret can be reconstructed without knowledge of which participants hold which share. In such schemes the computation of the secret can be carried out by giving the shares to a trustworthy machine that does not know the identities of the participants holding those shares. The difference between a secret sharing scheme and an anonymous secret sharing scheme depends on the reconstruction function used by the trustworthy machine. Fully dynamic secret sharing schemes have been proposed by Carlo Blundo, *et. al.* [53]. They considered secret sharing schemes in which the dealer is able, after a pre-processing stage, to activate a particular access structure out of a given set

and/or to allow the participants to reconstruct different secrets in different time instants by sending them the same broadcast message. Kulesza and Kotulski [54] described secret sharing schemes with extended capabilities. Sun and Shieh [55] proposed the construction of perfect secret sharing schemes for graph-based access structures. Lee, Yeh, Chen and Ku [56] proposed a probability model for reconstructing secret sharing and algorithms to perform share assignment.

3.2.2 Secret Sharing Construction

A number of practical constructions for secret sharing scheme are available. These include threshold access structures, geometric access structures, graph-based accesses structures and Quorum-based accesses structures. Stinson [41] described the construction of perfect secret sharing schemes.

3.2.2.1 Threshold access structures

Threshold access structures are structures where the secret can be reconstructed if a certain threshold number of shares are available but any combination of fewer shares cannot reconstruct the secret. The Shamir [44] and Blakley [45] approaches described above were the first schemes to be constructed as (m, n) threshold access structures where m is the threshold number of shares required to reconstruct the secret and n is the total number of shares. An (n, n) threshold scheme is called a unanimous scheme in which all the shares are required to reconstruct the secret. This scheme can be used to construct a secret sharing scheme in the absence of a trusted third party [42]. In multi-level threshold schemes, some participants who are more important than others can have more than one share.

3.2.2.2 Geometric access structures

Geometry can be used to construct secret sharing schemes as the following example illustrates. Consider three senior tellers at a bank who could open the bank vault or authenticate electronic funds transfer (EFT). We need to construct a secret sharing scheme that realises this access structure using geometry. Let the secret be the co-ordinates of a point P on a line l in 3-dimensional space. Let g be

a plane which intersects the line only at the secret point P. Let C be any circle in g. Any three distinct points in C can be used to determine g and therefore the secret, but two or fewer points cannot. Figure 3.3 depicts this construction. This is a $(3, n)$ secret sharing scheme. In this scheme, we have used a circle to be able to determine g since if we said three points, these points could come on a line, which does not determine the plane. Blakeley's [45] scheme as described in Section 3.2 uses points in space; it is a geometric construction in nature. In this scheme if three shares are required to reconstruct the secret, then it is a point in three-dimensional space. Simmon [57] describes how to use geometry to construct secret sharing schemes.

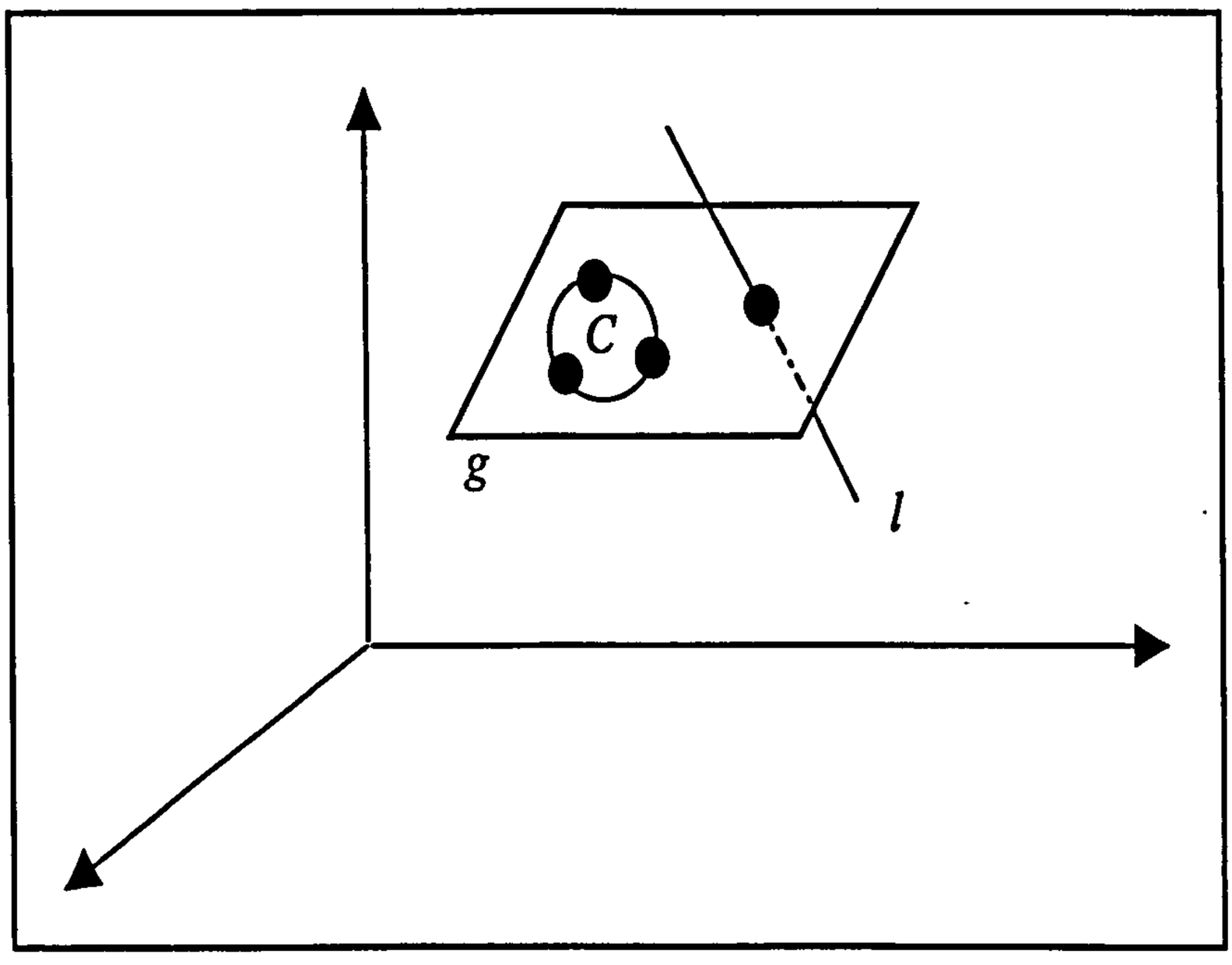


Figure 3.3: Geometric Construction

3.2.2.3 Graph-based access structures

Secret sharing schemes can be constructed using graph-based access structures [58]. Shieh and Sun [15] proposed a construction of perfect secret sharing scheme for the access structures consisting of the closure of a graph, where a vertex denotes a participant and an edge denotes a minimal qualified pair of participants. Figure 3.4 depicts a graph that defines the access structure between members of

a set of four participants p_1, p_2, p_3, p_4. Based on this construction, the qualified sets that can reconstruct the secret are $\{\{p_1, p_2\}, \{p_1, p_3\}, \{p_2, p_3\}, \{p_3, p_4\}, \{p_1, p_2, p_3\}, \{p_1, p_2, p_4\}, \{p_1, p_3, p_4\}, \{p_2, p_3, p_4\}, \{p_1, p_2, p_3, p_4\}\}$. In contrast the non-qualified sets includes: $\{\{p_1\}, \{p_2\}, \{p_3\}, \{p_4\}, \{p_1, p_4\}, \{p_2, p_4\}\}$. In [59] they proposed a scheme of constructing a secret sharing in graph-based prohibited structures. The prohibited structure is the collection of subsets of participants that cannot reconstruct the secret.

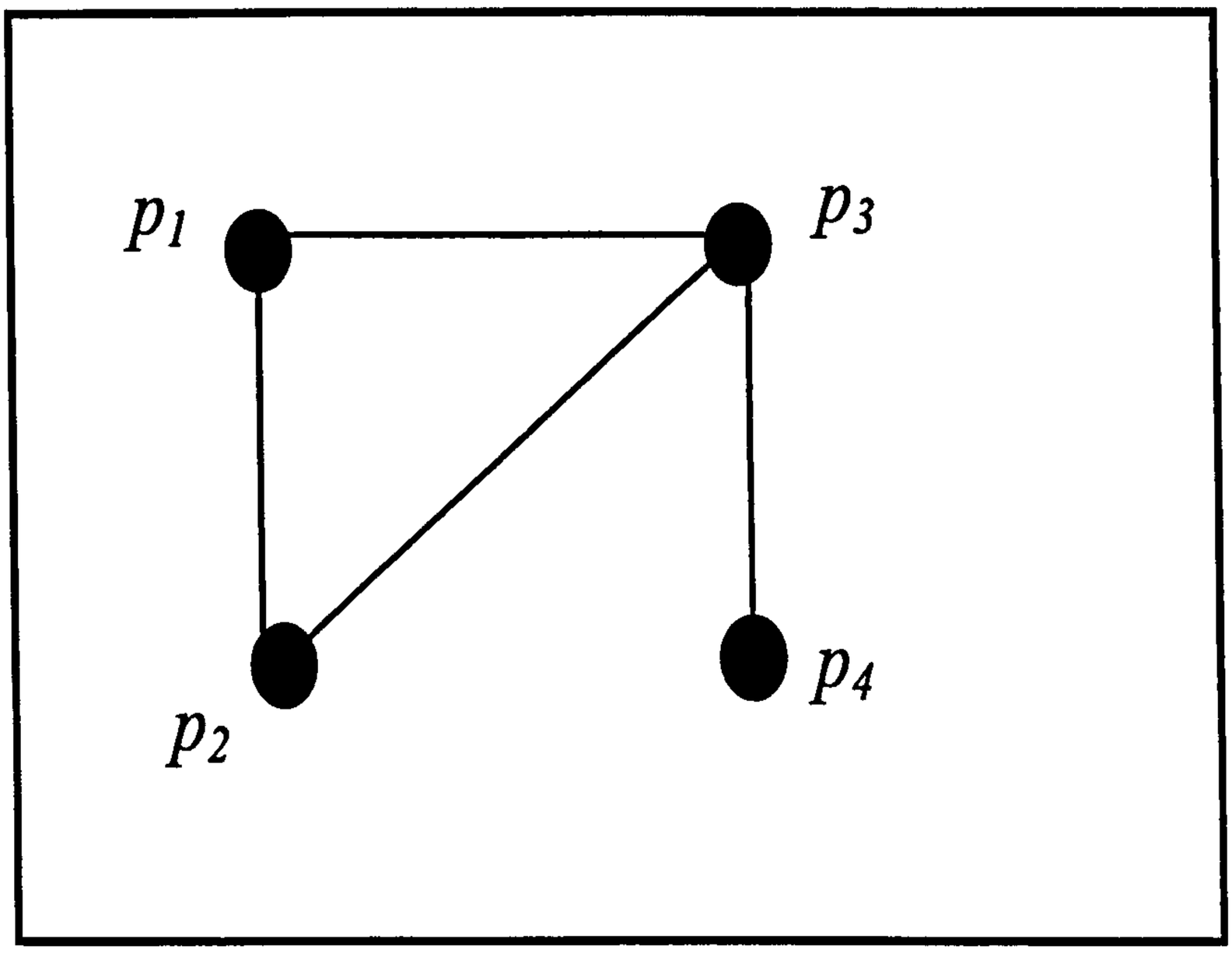


Figure 3.4: A graph representing four nodes

3.2.2.4 Quorum-based access structures

Organizing objects, nodes, and participants in this case, into sets called quorums form a quorum system, such that every two quorums intersect and no quorum contains another quorum. More detail about quorum systems and their applications is found in Section 3.4. Secret sharing schemes that realise the access structures of quorum systems can be constructed. Naor and Wool [60] provided several efficient constructions of secret sharing schemes that are based on quorum system access structures. Two examples of quorum secret sharing schemes are presented in Section 3.6.

3.2.2.5 Verifiable secret sharing

In secret sharing schemes if the dealer, or one of the participants, is dishonest by providing incorrect shares the result will be incorrect secret reconstruction. To avoid this type of cheating and resist malicious parties Chor, *et. al.* [61] proposed what is called *verifiable secret sharing* (VSS) where every party can verify that the piece of secret (share) received is indeed a correct piece to ensure the reconstruction of the original secret.

3.2.2.6 Publicly verifiable secret sharing scheme

A publicly verifiable secret sharing scheme (PVSS) is a verifiable secret sharing scheme in which any party (not limited to the perspective participants receiving the shares) can verify that the shares distributed by the dealer are correct. Stadler [62] introduced PVSS. In PVSS, the dealer using a public encryption function sends the encrypted shares (using the public encryption key) to all participants and publishes the encrypted shares. The participants can decrypt the shares and recover the secret. PVSS can be used in applications such as electronic payment systems, escrow cryptosystems (where the communication is encrypted with the encryption key escrowed to trusted agents) and electronic voting [63].

3.2.2.7 Threshold cryptography

In cryptography, information can be protected by highly sensitive operations such as decryption or signing. In threshold cryptography, these operations can be performed by a group of participants or servers by exploiting secret sharing. This introduces threshold decryption and threshold signature.

A threshold decryption scheme

This is a protocol that allows any qualified subset of m out of n participants to decrypt a ciphertext but $m-1$ participants are not allowed. In these schemes the decryption key is the secret which is divided into n parts such that each participant holds a part of the decryption key. Each participant computes a decryption share, which is a result of decrypting the ciphertext using the participant's shared

decryption key. A public algorithm combines all the results (decryption shares) and outputs the original plaintext [64].

A threshold signature scheme

A threshold signature scheme is a digital signature scheme provided by a group of participants such that each participant produces a partial signature on the given message. The combined partial signatures can result in a full signature on the message. In threshold signature schemes a qualified subset of m out of n participants can collaborate to produce a valid signature but $m-1$ cannot. Distributing the signature process increases security and availability, and reduces the probability of signature forgery [65].

3.3 Secret Sharing Schemes Applications

Some critical applications require more than one person or entity to make a decision or to initiate a critical action. Below are some examples of these applications that have been investigated in the literature.

Shared access to databases, Reiter and Birman [66] presented a method for constructing replicated services that retain availability and integrity despite several servers and clients being corrupted by an intruder. In this technique, the client need not be able to identify or authenticate even a single server. Instead, the client is required to possess only a single public key for the service and can treat the service as a single object for the purposes of authentication. In this scheme, it is the responsibility of the clients to verify that the data sent by the servers is genuine. The scheme is based on an (m, n) threshold secret sharing scheme.

Naor and Wool [67] proposed a method of controlling the access to a secure database via quorum systems. Their scheme is based on secret sharing schemes that realise the access structures of quorum systems. In their proposal the database items are encrypted using a secret key. To access a database item the user needs to contact the access servers to collect the shares. Using the reconstruction function the user can obtain the secret key from the collected shares and hence decrypt the encrypted item.

In shared decision and control applications, secret sharing schemes are useful in any important action that requires the concurrence of several designated people to be initiated. In shared control schemes only certain individuals acting in concert can only perform an action. The simplest shared control schemes are unanimous consent schemes [42] in which an action can only be performed by n designated individuals all acting in concert. Shared decisions and control are more obvious in banking systems and military applications.

One of the applications of secret sharing is the development of distributed authentication protocols. Gong [68] proposed a secure authentication service using an (m, n) threshold secret sharing scheme. The author proposed a distributed authentication protocol by replicating the authentication server in such a way that multiple servers share the responsibility of providing the authentication service such that each server provides only a fraction of the authentication. In the Gong protocol, a client chooses a candidate session key x and by using a threshold function generates n shadows (parts) of x and sends one shadow (x_i) to each server S_i. It is easy to recover x from any t shadows, but less than t shadows reveals no information about x. In contrast to the Kerberos protocol, this protocol involves more signalling but with increased availability and security.

Secure multi-party computation (MPC) is another application of secret sharing. Secure MPC can be defined as the problem of n players computing an agreed function (addition, multiplication) of their inputs in a secure way, where security means guaranteeing the correctness of the output as well as the privacy of the inputs. Cramer *et. al.* [69] studied the use of linear secret sharing schemes to provide a multi-party computation. In addition, secret sharing has been used in cryptographic keys management. Blakely and Shamir [45, 44] were the first people to use secret sharing schemes for key management purposes.

3.4 Quorum systems

A quorum system for a universe of nodes is a collection of subsets (quorums) of nodes, each pair of which has a non-empty intersection. Each quorum can act on behalf of the system. For example, let U be a set of nodes, $U = \{a, b, c, d\}$. Then $Q = \{\{a, b\}, \{b, c\}\}$ is a quorum set under U. Figure 3.5 depicts a quorum

system with three quorums, where each quorum has four nodes. A node could be a computer in a network or a copy of a data object in a replicated database. In distributed systems replicated data is employed to improve data availability and reliability. Replication protocols are used to distribute copies of a data item to different locations over a wide area network. The difficulty comes from keeping the data item copies consistent with each other despite system failures and network partitioning. Quorum systems overcome these difficulties. A mutual exclusion protocol is a control protocol that ensures that a certain process is unable to obtain a resource access while it is allocated to another process. Quorum systems achieve that control by restricting the resource access to be possible only after obtaining permission from all the members of some quorum. In both replication and mutual exclusion protocols consistency and coordination are required. Quorum systems are capable of achieving these requirements because of the intersection property that different quorums overlap. Generally quorum systems are important tools for increasing availability and reliability. A useful extension of quorum systems is to increase network security. Quorum systems and secret sharing can work together to obtain secure systems with distributed trust.

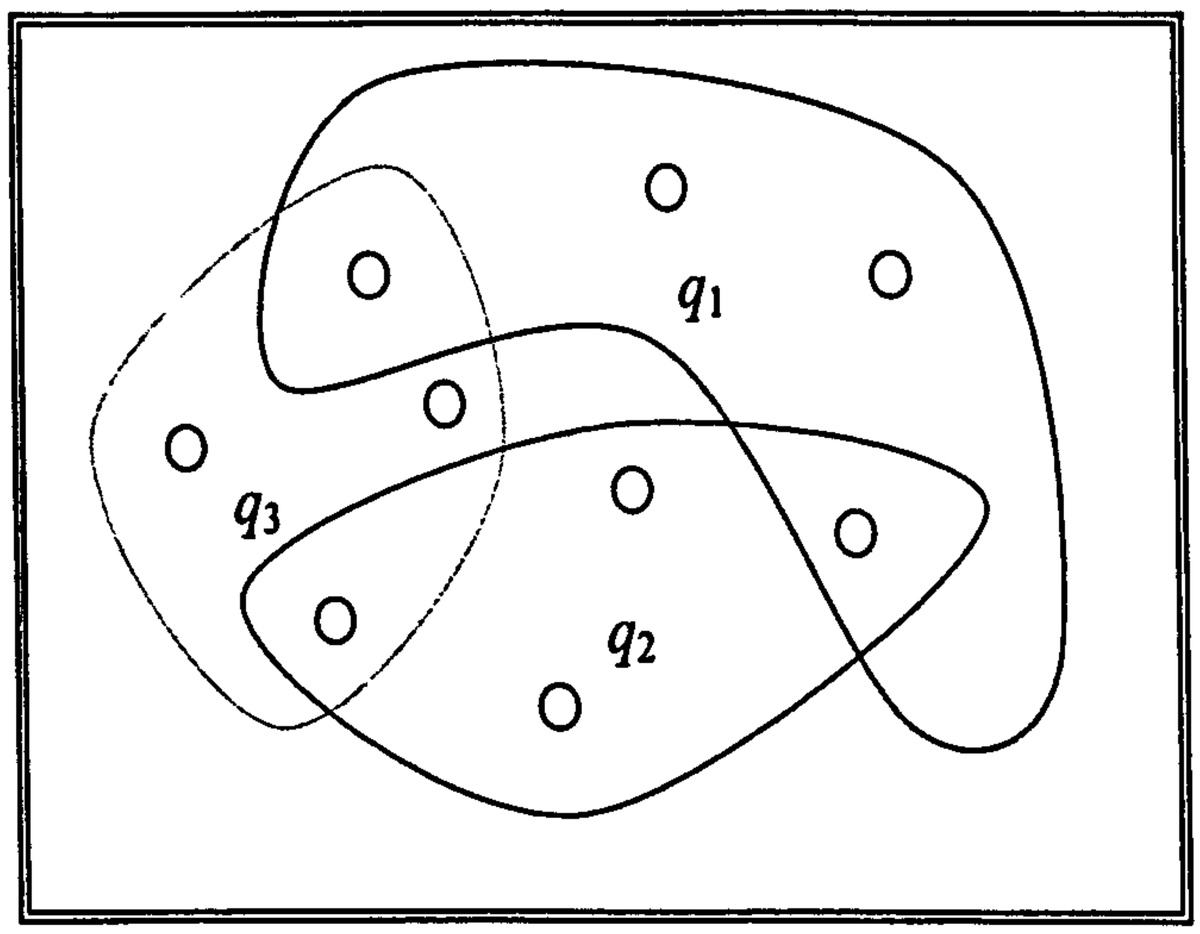


Figure 3.5: A quorum system $Q = \{q1, q2, q3\}$

3.4.1 Quorum System Measures

To know which quorum system works best for a given set of nodes, three criteria for measuring the quality of the quorum system have been considered: load, availability and quorum size [67].

- Load: Noar and Wool define in the load of a quorum system as the fraction of the time that a member of the quorum system (node) is accessed under the best possible strategy for choosing quorums [67]. If the load is low then the node is accessed rarely. This can be achieved if an optimal access strategy is implemented.

- Availability: The availability of a quorum system is the probability that at least one quorum survives. It is a measure of the quorum system's capability to survive even when individual nodes may fail. This ensures that authorized users may continue accessing the nodes in the case of node failure. The best quorum systems will have high availability.

- Quorum size: In quorum systems, the nodes in each quorum can co-ordinate with each other. If the quorum size is small, the communication overhead becomes small and the load is low. Therefore, quorum systems of small size are preferred [70].

3.4.2 Quorum Constructions

Below we review briefly the most well known quorum structures.

- **Singleton:**

  If Q consists of only a single quorum q, $Q = \{q\}$ is termed a singleton quorum system.

- **Grid:**

  If the total number of nodes in the system is n, the nodes are arranged in a $d \times d$ grid as shown in Figure 3.6 where $d^2 = n$. A quorum is the union of a full row and a representative node from all the other rows [71].

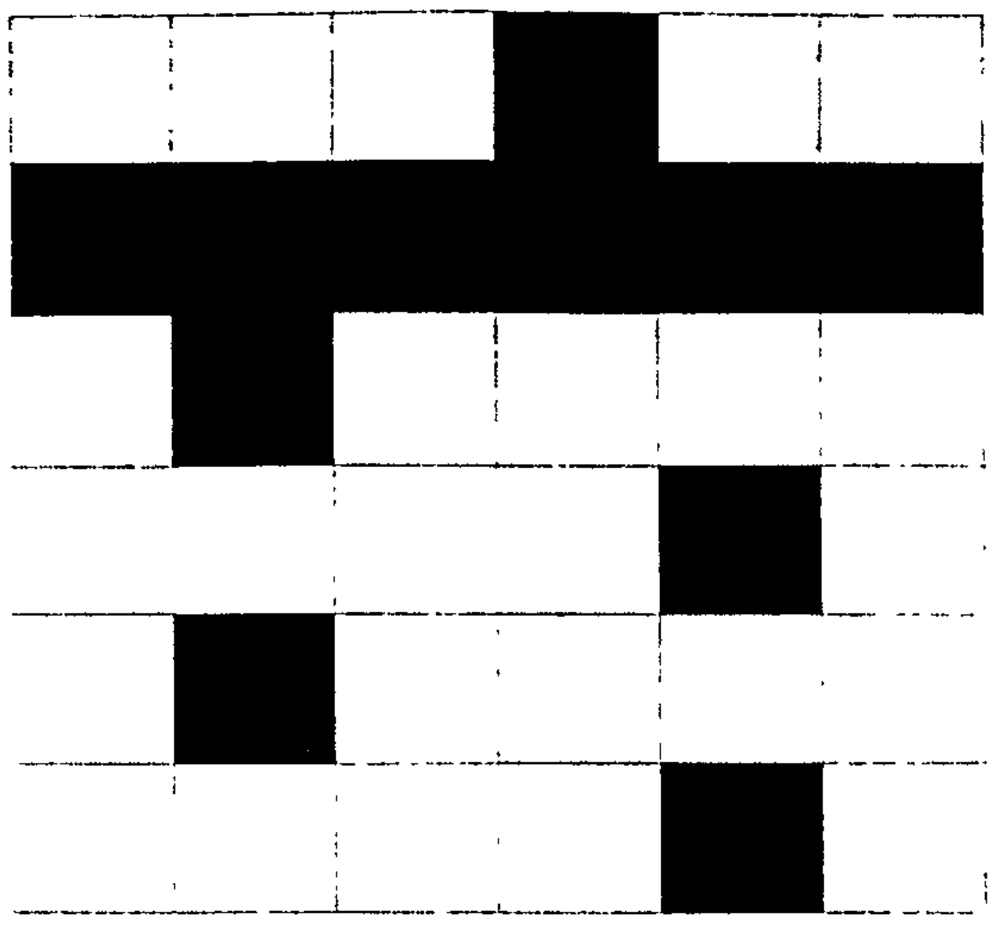


Figure 3.6: Grid construction

- **SuperGrid:**

  In this system the quorum is constructed by selecting a majority of rows and then a majority of nodes from each selected row. Figure 3.7 shows a 5×5 grid and illustrates an example where one quorum is shaded [72].

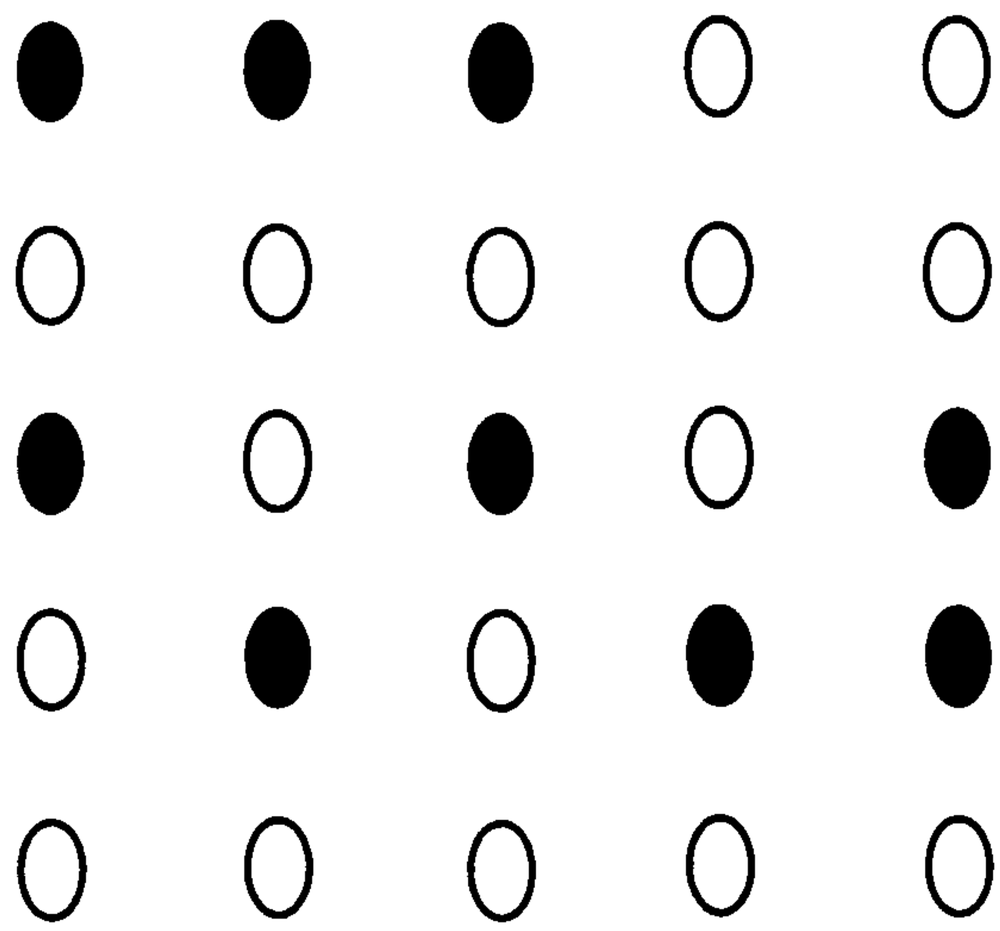


Figure 3.7: SuperGrid construction

- **Tree:**

  In this system the nodes are organized in a complete rooted binary tree. A quorum in the system is defined recursively to be either (i) the union of the

root and a quorum in one of the sub-trees, or (ii) the union of two quorums, one in each sub-tree. Figure 3.8 shows an example where one quorum is shaded [73].

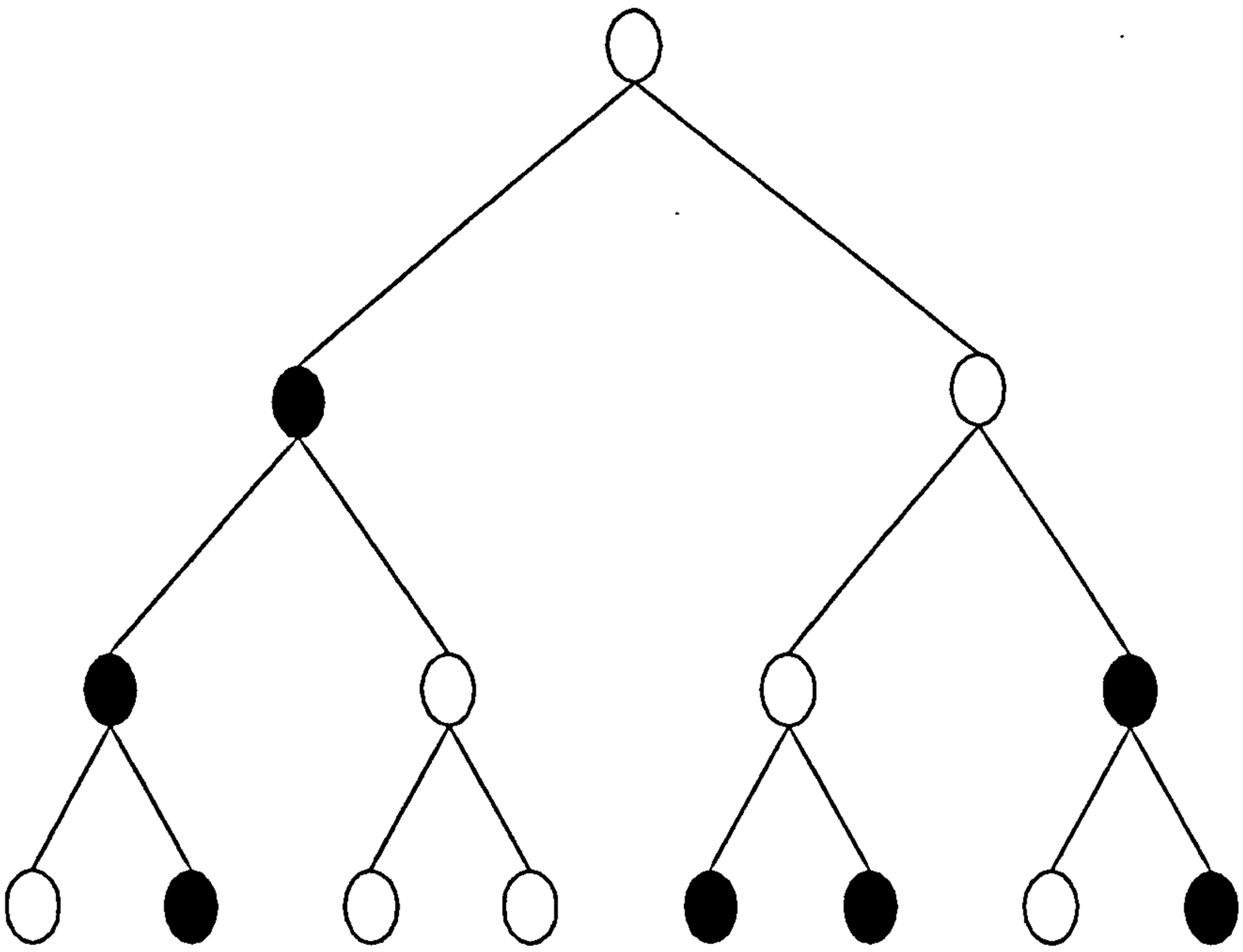


Figure 3.8: Tree construction

- **Weighted majorities:**

  In this system each node is assigned a specific number of votes. A quorum is formed by obtaining at least a threshold of votes [74, 75]

- **Finite Projective Plane (FPP)**

  For a prime r let $t = r^k$ for some integer k. Then the finite projective plane of order t is a quorum system with n nodes and m quorums. Each quorum is of size $t + 1$. In this system $n = m = t^2 + t + 1$. Each node is a member of $t + 1$ quorums and the intersection of every two quorums consists of a single node [76, 77]. For $t = 2$ there are 7 nodes and 7 quorums as shown in Table 3.1. Each node is a member of 3 quorums.

Quorum	Members
Q1	1,2,3
Q2	1,4,5
Q3	1,6,7
Q4	2,4,6
Q5	2, 5,7
Q6	3,4,7
Q7	3,5,6

Table 3.1: Finite projective plane with t = 2

- **Crumbling Wall (CW):**

  Peleg and Wool [78] introduced a general class of quorum systems called crumbling walls. The best crumbling wall is the Cwlog system as shown in Figure 3.9. The nodes in a wall are logically arranged in rows of varying widths. A quorum in a wall is the union of one full row and a representative from every row below the full row. The best crumbling walls are those in which the top row has width $n_1 = 1$ and every other row has width $n_i \geq 2$. In the Cwlog system, the width of row i is

  $n_i = \lfloor \log 2 \times i \rfloor$.

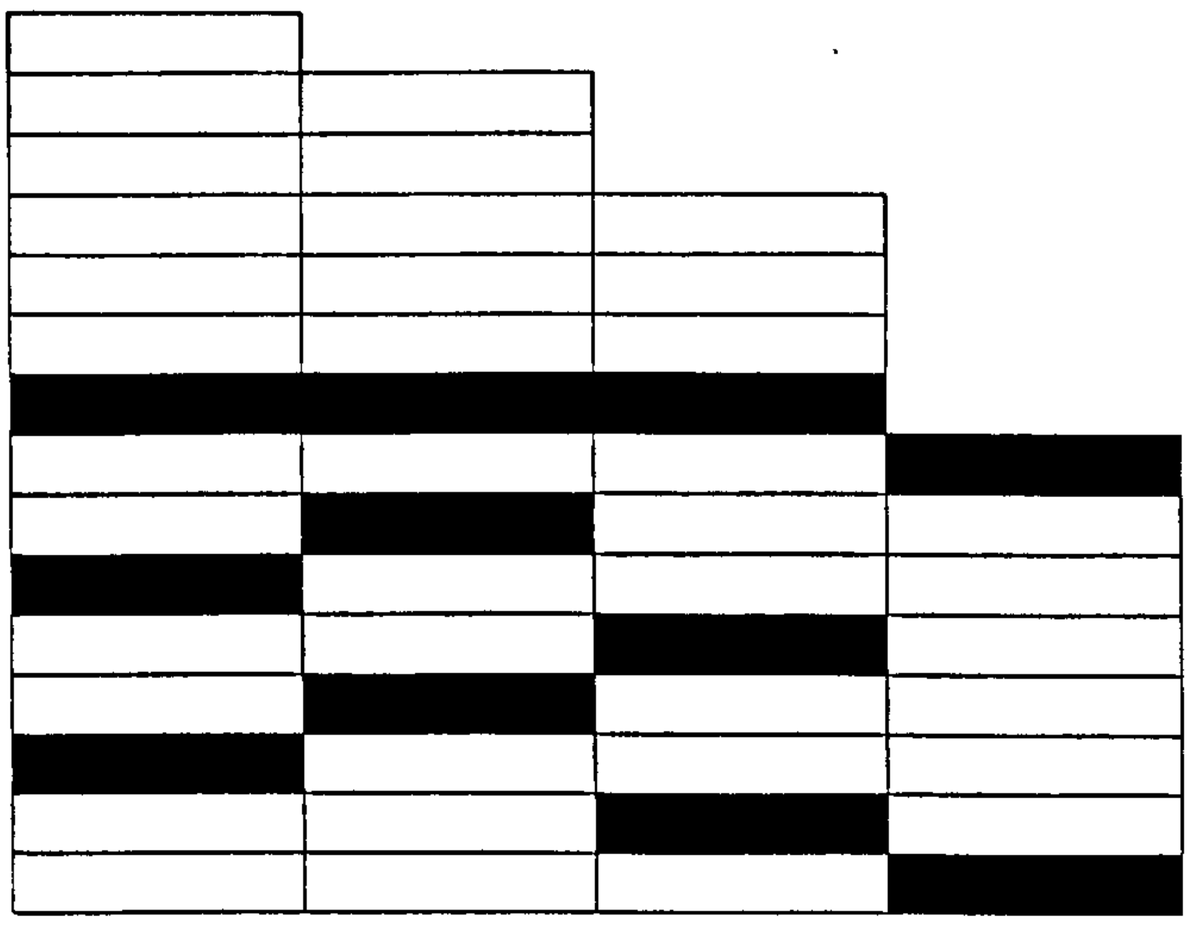


Figure 3.9: Cwlog construction

In general there are three types of protocols to construct quorum systems. First organizing group of nodes into logical structures such that any pair of quorums must intersect in at least one node. In the second way nodes assigned votes such that obtaining certain threshold reveal a quorum. The third type is uniform quorum system (UQS) where the quorum system is characterised by certain parameters such as the quorum size, number of quorums, the intersection set, etc. Tree, grid and crumpling systems are Examples of first type.

A uniform quorum system can be characterised by the following elements: The number of nodes n, the number of quorums q, the quorum size k, the node membership m which equal to the number of quorums the node is belong to and the size of intersection set between each two quorums i. These parameters are related using the following formulas [79].

1. $nm = qk$

2. $i(q-1) = k(m-1)$

3. $q \leq n$

4. If $k|n$ then $q \leq n - k + 1$

Finite projective plane is an example of a uniform quorum system where $i = 1, q = m = t + 1, n = q = t^2 + t + 1$.

Using the above metric a uniform quorum system can be constructed. Table 3.2 shows a uniform quorum system (UQS) where the parameters are 6, 4, 3, 2, 1 respectively. Hass and Liang [79] used uninform quorum system to solve the mobility management in ad hoc networks.

Quorum	Members
Q1	1,2,3
Q2	1,4,5
Q3	2,4,6
Q4	3,5,6

Table 3.2: A uniform quorum system

3.5 Quorum Applications

Quorum systems have been used in several application as a control tool to increase availability and reliability. Quorum system applications include the following. Quorum-based consensus protocols to exchange information between nodes with a minimum number of messages, which is achieved by the intersection property [80, 81]. In quorum-based mutual exclusion algorithms a node is authorized to take an action (for example accessing a printer in the network of nodes) after receiving permission from a quorum of nodes. The intersection property ensures that other nodes cannot take the same action until the first node exits. Maekawa [76] proposed an algorithm that uses only messages to create mutual exclusion in a computer network using quorums, where N is the number of nodes and c is a constant between 3 and 5.

Quorum systems have been used in replication control protocols such that with each data item several read and write quorums are formed. The intersection property of quorums ensures that each read item has access to the most recently written value of the data item. Integrating security issues in a replica control protocol is another application of quorum systems. Herlihy and Tyger [82] uses a quorum protocol to secure replicated databases. The protocol has two versions: in the first version each replica is encoded by using a single key, the same key is used for decryption too. Using (m, n) threshold secret sharing, the key is divide into n shares and at least m shares are required to recover the key. To read a data item , a read quorum of copies is read and decrypted using the key. To write a data item, the new value is encrypted using the key and then distributed to a write quorum of copies. In the second version, two keys are used, one for encrypting the data and another one for decrypting the data.

Naor and Wool [60] proposed a method of controlling the access to a secure database via quorum systems. Their scheme is based on secret sharing schemes that realise the access structures of quorum systems. The servers in a complete quorum can collectively grant access permission. In this proposal, the database items are encrypted using a secret key. To access a database item the user needs to contact the access servers to collect the shares. Using the reconstruction function, the user can obtain the secret key from the collected shares and then decrypt the encrypted item. Quorum systems can be used for fault-tolerant solutions in

networks and distributed systems [83, 73]. Additionally, they can be used for multi-party computation protocols [77].

3.6 Quorum-based secret sharing schemes

Secret sharing and quorum systems can be used together in certain applications to increase network security. Both systems can provide efficient security protocols. In (m, n) threshold secret sharing systems, m must be greater than $n/2$ to ensure greater consistency, and for large n this may degrade the efficiency. Quorum systems overcome this drawback with a small quorum size as a result of the intersection property. In summary these applications include the following:

- Network access control protocols.
- Distributed authentication protocols.
- Databases access control.
- Distributed signatures.
- Public key management protocols.

Secret sharing and quorum systems have some drawbacks. There is increased delay and they require more management. There are trade-off decisions to be made, but the advantages are significant in critical applications.

Naor and Wool [60] and Beaver and Wool [77] showed how to build quorum secret sharing schemes from several known quorum systems. As a basic building block in these schemes, consider a simple secret sharing scheme where a secret bit S is shared by forming a set of n random bits s_1 to s_n with the constraint that $S = \sum s_i$, where all operations are over the binary field (i.e. GF(2)). The secret can be considered to be the parity of the shares. For example, if $S = 0$, and $n = 4$, then we could choose 0, 1, 0, 1. It can be seen that this forms an *(n, n)* threshold scheme, because all the shares must be available before the secret is known. It is also an ideal scheme, since the size of each share (one bit) is equal to the size of the secret. Secrets with more than one bit can be formed by operating the scheme a number of times, once for each bit of the secret.

This simple scheme is nevertheless perfect, because possession of fewer than n shares gives no information about the secret. Say we have shares 1 to $n-1$. If the sum of these shares, called a partial parity, is 0, the secret will equal the remaining share s_n, but if the partial parity is 1, the secret will equal the inverse of the remaining share. Assuming the scheme is implemented properly, the individual shares will be effectively random, so no information on the secret can be obtained.

A Crumbling Wall secret sharing scheme can be implemented using this as follows. Consider a crumbling wall of d rows, with row 1 having width $n_1 = 1$ and $n_i > 2$ for all $i > 2$. This secret S is first randomly shared into d bits such that $a_1 + ... + a_d = S$. Using these a_i bits we can define their partial parities, $t_i = a_1 + .. + a_{i-1}$ and $t_1 = 0$. For a row i, the t_i is itself shared into n_i bits A_i^j bits such that $A_i^1 + . . + A_i^{n_i} = t_i$. The share s_i^j of the j'th element in row i contains two bits: a_i and A_i^j

A quorum Q which contains a full row i and a representative from each row $k > i$ can reconstruct the secret bit S from the shares $(s_i)^j$ generated by

$$S = ((\sum_{(j=1)}^{n_i} A_i^j) + \sum_{(k \geq i)} a_k)$$

From the above expression the first summation is equivalent to t_i of row i which is equal to the partial parity of all a_k above row i , and the whole expression is the parity of all a_k which therefore yields S.

The basic secret sharing scheme can be extended over larger finite fields, i.e. over GF(q) with arithmetic modulo q, but it is important that each share is selected 'randomly' over the full range [0, $q-1$] so that information does not leak out from a partial sum of shares. We can use this to implement a Finite Projective Plane (FPP) secret sharing scheme. In the FPP scheme discussed in Section 3.4.2, for a plane of order t, a node will appear in $t+1$ quorums. Since the intersection property means that each pair of quorums have only one node in common, this means that if we take the set of all elements in all quorums containing a node, that node will appear $t+1$ times, and all other nodes will appear once only. We use this to construct the scheme.

Take a FPP with order t and $n = t^2 + t + 1$ quorums. The secret S to be shared is split into a sum of n random elements x_1 to x_n as before, although

in this case we use arithmetic over a field with t elements (i.e. GF(t)). Each element is assigned to a quorum. The share of node u, a_u, is the sum of the parts corresponding to the quorums that contain u, i.e., for example if we take $t = 2$ as in Table 1, the number of nodes is 7. Also, there are 7 quorums, and so $S = x_1+x_2+x_3+x_4+x_5+x_6+x_7$. Node 1 is a member of q_1, q_4 and q_6, so the share of node 1, $a_1(x) = (x_1 + x_4 + x_6)$. The share of node 2 is $a_2(x) = (x_1 + x_2 + x_5)$, since node 2 appears in q_1, q_2 and q_5 (see Table 3.1). The share of node 3 is $a_3(x) = (x_1 + x_3 + x_7)$. The three nodes in q_1 can calculate the secret x since $a_1(x) + a_2(x) + a_3(x) = (3x_1 + x_4 + x_6 + x_2 + x_5 + x_3 + x_7) = x$.

It is clear that all members of each quorum can reconstruct the secret.

3.7 Conclusion

Secret sharing and quorum systems are of great interest in communication networks as they offer the possibility of distribution of trust, increasing reliability and availability. This chapter has introduced the concepts and applications of secret sharing and quorum systems and overviewed their concepts and applications.

Secret sharing and quorum systems have been studied extensively, however, most of the work on secret sharing has been conducted from an information theory and not a networking perspective. The next chapter proposes how secret sharing and quorum systems can be used to create trusted networks that can be depended upon to run critical applications.

Chapter 4

SNAP

4.1 Introduction

As described in Chapter 2 most of the existing protocols for authentication and network access are based on centralized authentication using one authentication server or certificate authority that is unconditionally trusted. Despite the use of strong authentication mechanisms, the process is between the user and a single authentication entity. In these protocols, the decision to authenticate or grant service to users is controlled by this single entity. Trust is an essential point and in these protocols the trust is maximized on a single entity. In general, if trust is increased risk will increase but availability and security will decrease. In this chapter, a new protocol called SNAP is proposed. The objective is to increase the network security and availability by distributing the trust between several authentication entities. SNAP is proposed to overcome certain attacks in the centralized authentication protocols. This protocol is based on secret sharing and quorum systems by exploiting their features to increase security and availability. This chapter describes SNAP's messages, management, vulnerability and applications.

4.2 Authentication models background

SNAP is a distributed authentication protocol which ensures that only authorized users may access network resources. This distributed approach means that more than one authentication server participates in the authentication process. In general, authentication models can be classified into three approaches as shown in Figure 4.1. In the centralized approach, all users contact one authentication server (AS) to obtain the required keys or to be authenticated. The centralized approach has the drawback that trust is focussed in a single AS. If the AS is compromised then the whole system is no longer secure. Also, the load on the AS is very high, and if the AS breaks down then authorized users will not be able to access the system. However, the management and updating overhead are low. A second approach is to parallel the AS, so that there is more that one AS that can handle the authentication process. Each AS is a duplication of the original one and users can access any one. The advantages are increased availability and distribution of the load between ASs. The drawbacks are that there are more points to attack, as each AS is fully trusted, and also the increased management requirements. We have designed our new SNAP protocol to overcome the drawbacks of these approaches. In SNAP the user needs to contact a quorum of servers in order to be authenticated. A distributed approach has the advantage that the trust is distributed between the quorum members AS and each one is partially trusted. There is also increased availability since if one quorum is not available, the user can contact another quorum. The load is distributed if users are managed to access different quorums. The disadvantages are increased management and signalling overhead since the users needs to contact a quorum of AS.

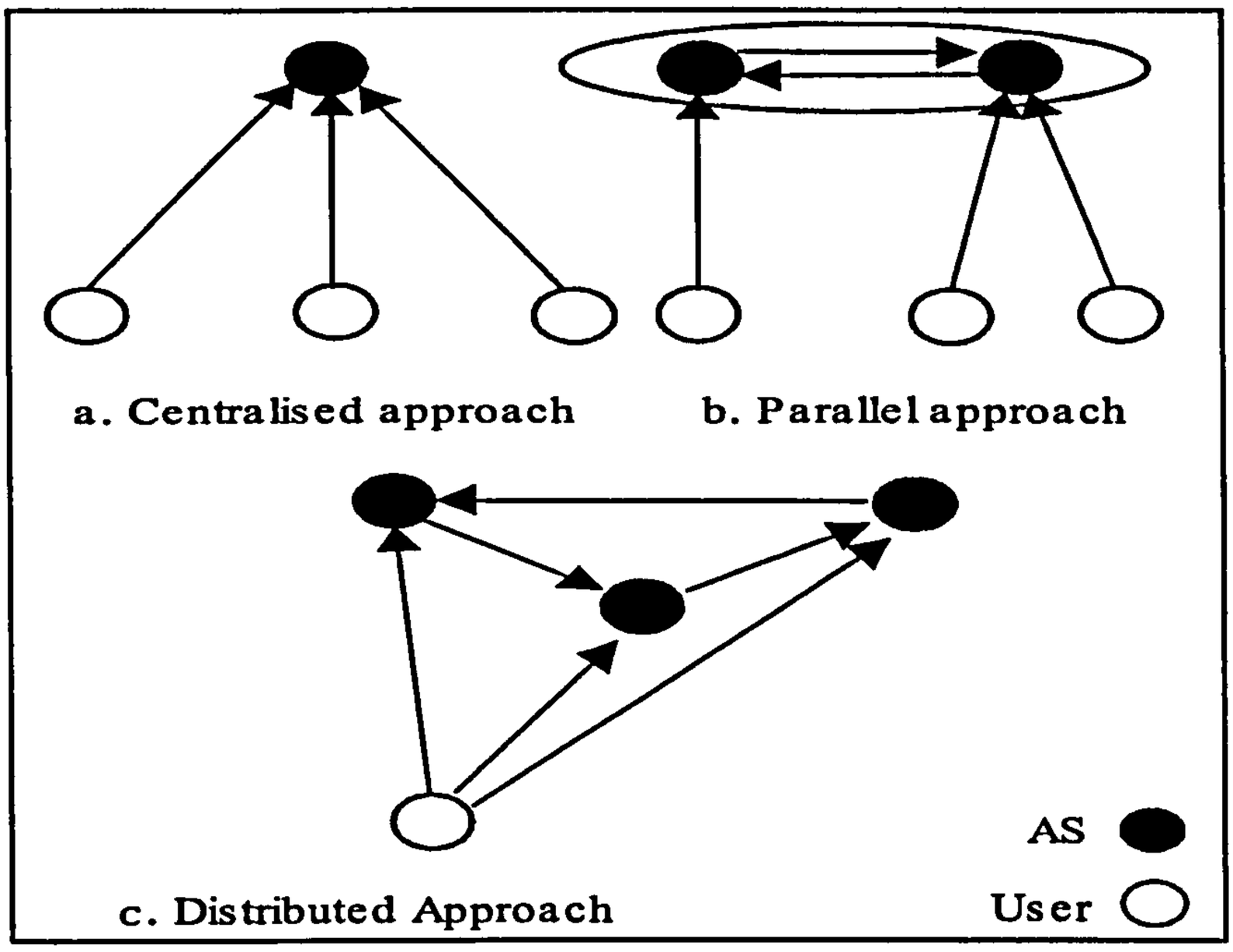


Figure 4.1: Authentication models

4.3 Related work

Reiter and Birman [66] presented a method for constructing replicated services that retain availability and integrity despite several servers and clients being corrupted by an intruder. In this technique, the client need not be able to identify or authenticate even a single server. Instead, the client is required to possess only a single public key for the service and can treat the service as a single object for the purposes of authentication. In this scheme, it is the responsibility of the clients to verify that the data sent by the servers is genuine. The scheme is based on a k-out of-n threshold secret sharing scheme and, to keep the consistency property, k must be greater than or equal to $n/2$.

Herlihy and Tyger [82] use a quorum protocol to secure replicated databases. The protocol has two versions. In the first version each replica is encoded by using a single key; the same key is also used for decryption. Using a k out of n threshold secret sharing, the key is divide into n shares and at least k shares are required to recover the key. To read a data item a read quorum of copies is accessed and

decrypted using the key. To write a data item, the new value is encrypted using the key and then distributed to a write quorum of copies. In the second version two keys are used, one for encrypting the data and another one for decrypting the data. The scheme is also based on a k out of n threshold secret sharing scheme, which requires a large communication overhead.

Naor and Wool [60] proposed a method for controlling the access to a secure database via quorum systems. Their scheme is based on secret sharing schemes that realize the access structures of quorum systems. In their proposal the database items are encrypted using a secret key. To access a database item the user needs to contact the access servers to collect the shares. Using the reconstruction function, the user can obtain the secret key from the collected shares and can then decrypt the encrypted item. In this protocol, the user needs to get the secret key by the co-operation of a quorum of servers. However, the protocol does not address the problem of a cheating user. In other words, if the user is malicious and gives this secret to other unauthorized users, then the secure database is no more secure. Additionally, as stated in the protocol, each access server knows the whole encryption key (the secret) of the database and if obtaining this key has compromised one server, the whole database is compromised. In many ways this is similar to the issue of replicating a centralized access server, so the protocol does not provide a solution to the trust problem.

Gong [68] proposed a secure authentication service using a k out of n threshold secret sharing scheme. The author proposed a distributed authentication protocol by replicating the authentication server in such a way that multiple servers share the responsibility of providing the authentication service such that each server provides only a fraction of the authentication. This protocol provides a distributed authentication between two parties to share a secret key and although it provides increased security and availability, it involves a lot of signalling which might affect the network overhead, and in addition, scalability is an issue to be investigated. Since this protocol is based on a k out of n secret sharing scheme, k should be greater than or equal to $n/2$ to keep the consistency between the server's keys. Most related work assumes that a secure channel exists between the network nodes (servers). It is a problem to have this assumption since not all the network channels are secure and one cannot assume such an authentic communication channel prior to actual authentication [6].

Zhou, Schneider and Renesse [84] proposed a secure distributed on-line certification authority (Cornell Online Certification Authority (COCA)). Using the on-line certificate authority the user can check the validity of certificates just before using them. COCA is based on Byzantine quorum systems and threshold cryptography. A Byzantine quorum system is the system that can tolerate arbitrary (Byzantine) failures such that each pair of quorums intersects in sufficiently many servers to mask out the behavior of faulty servers [85]. The authors achieve availability using replication and employ proactive recovery with threshold cryptography for digitally signing certificates [86]. In other words, the certificate must be signed by a threshold number of servers (a quorum of servers). The quorum system is used to manage the certification replication. In COCA there are two main protocols: query and update. Query retrieves certificates specifying a certain binding. Update is used to create, update and invalidate certificates. In COCA, the client contacts one of the servers which is considered as the delegate. On behalf of the client the delegate sends the client request to a quorum of servers which then send the reply to the client. Each server in the quorum replies to the delegate and partially signs its message using its share. For example, in the query operation, the delegate verifies the certificates in the received messages are correctly signed by a quorum of servers. As stated in the proposed approach to avoid the case where the delegate is compromised and might not follow the protocol, the client send his request to t+1 delegates instead of one. However, each delegate sends the client request to a quorum of servers and wait for the respondence. If each user in the network did that, the network would be flooded by the generated traffic and this may cause an implementation problem. The solution to this problem is to let the user send his request to another delegate only if he did not receive a response in a certain time, so that the client contacts delegates sequentially instead of in parallel.

4.4 Secure Network Access Protocol (SNAP)

In order, to address the deficiencies of previously published solutions SNAP has been developed as part of this work. SNAP ensures that only authorized users may access network resources. It is based on a distributed approach, so that more than one authentication server participates in the authentication process.

In the development of SNAP, certain objectives were considered. It must provide increased security, increased availability, distributed trust and protect against known attacks as described Section 4.7.

4.4.1 Design goals

The design goals of the protocol were driven by a number of requirements:

- **Security**

  SNAP is used to establish the following:

  - Strong authentication between two parties
  - Secure connection between two parties
  - Message integrity
  - Secure access control for network resources

- **Availability**

  SNAP provides increased availability by utilizing secret sharing and quorum systems features. Due to these features, a limited number of nodes can fail (based on the specified parameters) and the user may still obtain the required shares and access the network.

- **Distributed Trust Model**

  Instead of relying on one unconditional trust entity, as most of the existing networks protocols do, SNAP provides distributed trust which minimizes the risk and increases the security.

- **Simplicity**

  SNAP is flexible and simple to use. The network or security administrator can readily select the required algorithms for encryption, public key cryptography, hash function and the secret sharing scheme from the standards.

4.4.2 Description

The main components in SNAP are users and the set of authentication nodes. The *user* is the entity attempting to access the network resources, while nodes are routers, servers, or any terminal used for network access. The network security manager (NSM) supervises the required security services in the network. These include the secret sharing scheme, quorum system, quorum size, share generation and distribution, and the access code AC. Additionally the NSM is responsible for the creation and updating of the reconstruction functions as well as for periodically updating the shares. The share revocation mechanism is important if the NSM is required to update or inform the network nodes about share revocation by providing new shares to be used instead of old shares. In this protocol any authentication node (AN) can verify the AC based on the recent shares.

In SNAP, there is an AC to access resources, for example the user will be unable to access these resources until permission is obtained from a certain number of nodes, in other words a quorum of nodes. These nodes should provide the user with the corresponding parts (shares) of the AC, which means that each user should contact multiple nodes independently. The registration with nodes should be done off-line using an authentication proof and appropriate supporting documentation. The network security manager should construct the AC using a secret sharing scheme (SSS) that has a quorum access structure. Thereafter, the AC is divided into n pieces (shares) and distributed to n selected nodes among the network nodes such that a quorum of k nodes is qualified to reconstruct the secret share from their shares. Any k (where $k < n$) shares can reconstruct the AC but fewer than k cannot. The user can access the network if he obtains k shares from a quorum of k nodes in the network. A simple network example with $k = 3$ is shown on Figure 4.2. In this figure, the user needs to contact a quorum of three nodes to collect the required shares to access the network. The access to the network using SNAP involves the following steps:

1. Using any strong local authentication procedure, the user can authenticate himself to the local node (LN) and make a request to access the network.

2. If the user is known or authorized, LN will ask the user to provide the AC.

3. The user will start independently communicating with a quorum of nodes

with which the user has registered and ask for the corresponding shares, to be able to provide the AC.

4. Each node will check if the user is authorized. If so the user will be sent the corresponding share.

5. After receiving the required k shares the user will reconstruct the secret share or the AC using a reconstruction function and present it to LN.

6. LN will verify the AC and give the user the permission to access the network if he is eligible or alternatively output a rejection message.

The signalling between the protocol components is depicted in Figure 4.3.

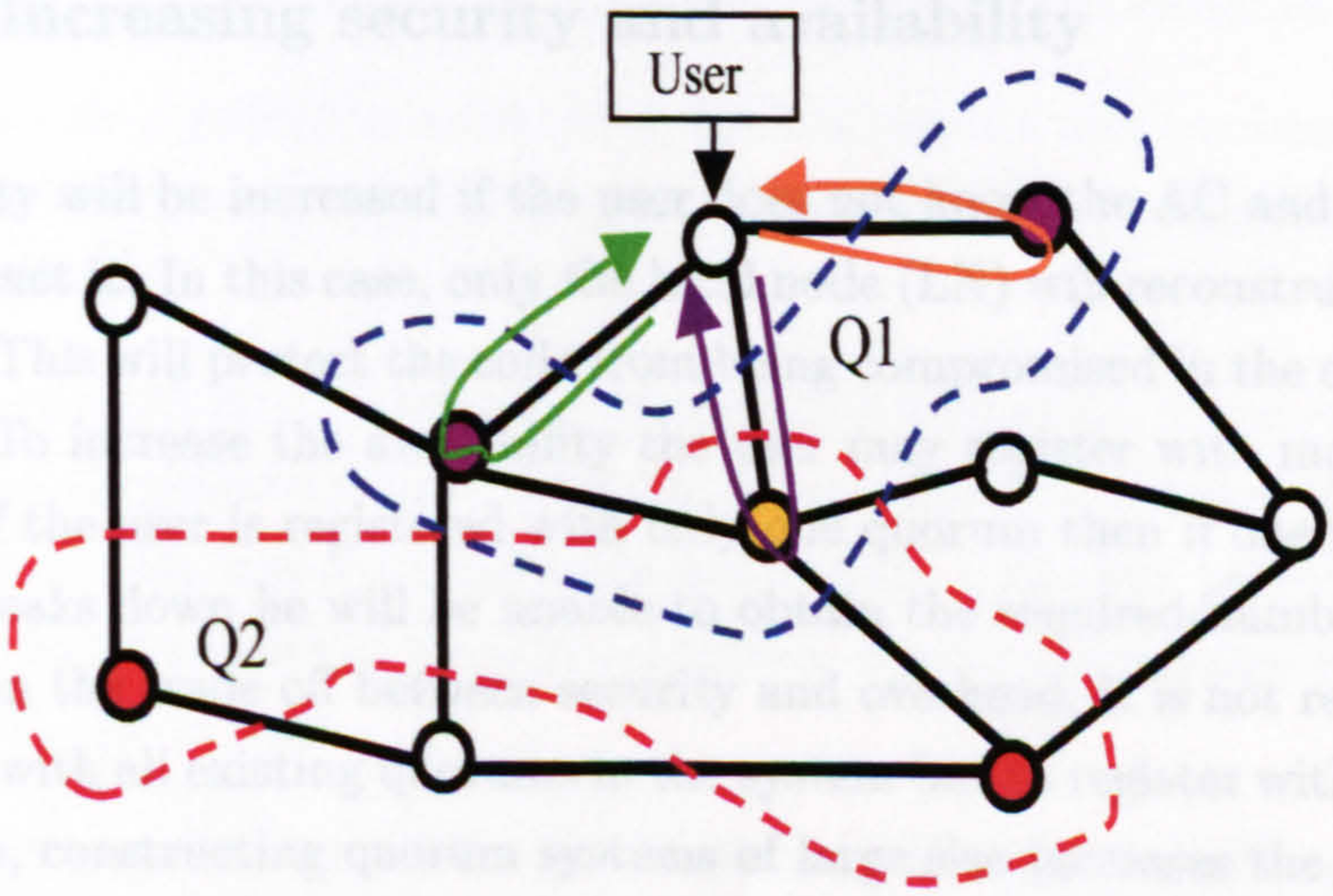


Figure 4.2: Network example

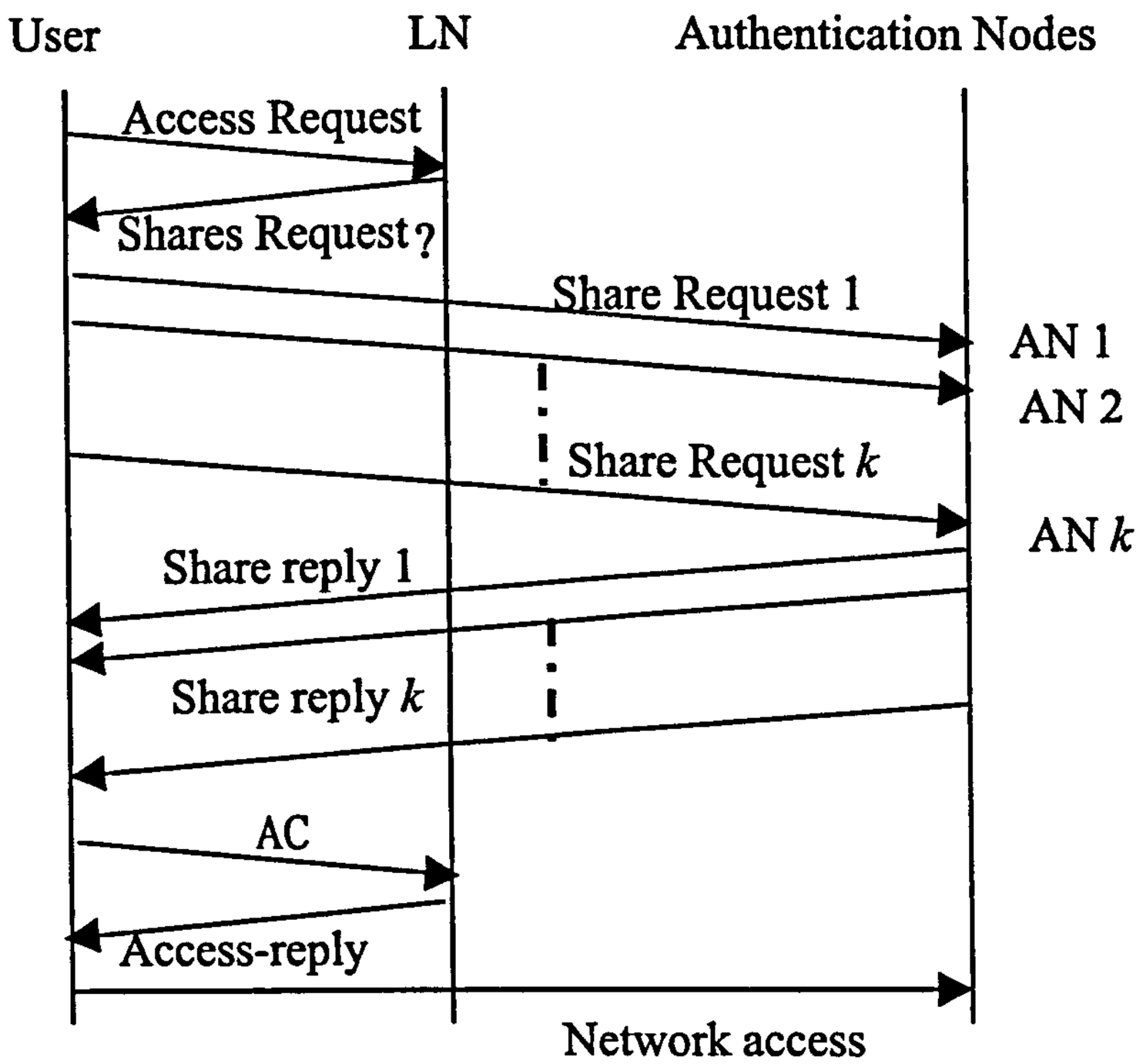


Figure 4.3: SNAP signalling

4.4.3 Increasing security and availability

The security will be increased if the user does not know the AC and so is unable to reconstruct it. In this case, only the local node (LN) will reconstruct the secret share AC. This will protect the code from being compromised in the case of a user cheating. To increase the availability the user may register with more than one quorum. If the user is registered with only one quorum then if one node in that quorum breaks down he will be unable to obtain the required number of shares. To maintain the trade off between security and overhead, it is not recommended to register with all existing quorums in the system but to register with l quorums. In addition, constructing quorum systems of large size increases the security but involves more signalling. The NSM must select the appropriate quorum size.

In SNAP, quorum based secret sharing schemes are used. SNAP achieves the

required objectives using threshold secret sharing schemes. *A resilient threshold secret sharing scheme (l, k, n)* can be used. In this scheme there are three parameters l, k and n; l is used to increase availability, k is used for security (to increase privacy) and represents the number of shares required to reconstruct the secret share, n is the total number of shares. The relation between the three parameters is $k < l < n$. The user is assumed to register with l nodes such that he needs only to obtain k shares and if one node is not available then he can contact another node, up to l nodes in total, and in this case the availability is increased. This scheme is recommended if the network security manager is using only secret sharing systems to run the protocols and the quorum system is not used. However, in threshold schemes, to maintain consistency between shares such that a user will not obtain revoked or old shares, which might prevent the user from accessing the network while he is eligible, k must be greater than or equal to $n/2$. In systems where n is large this will cause an increased overhead. Quorum systems are more flexible, and the intersection property between quorums maintains consistency and the condition that $k > n/2$ is not required. This will result in minimizing the overhead and the network security manager can select the appropriate quorum size k.

The new signalling is depicted in Figure 4.4; the modified SNAP is as follows:

- Using any strong local authentication procedure, the users can authenticate themselves to the local node (LN) and make a request to access the network.
- If the user is known or authorized, LN will ask the user to obtain permission or to be authenticated by a quorum of nodes by providing k shares from any quorum.
- The user will start independently communicating with the nodes, which the user has registered with and ask for the corresponding shares. If he does not receive a response from one node in a certain time then he will try another node up to l nodes in the case of scheme one or contact another quorum if scheme two is used
- Each node will check if the user is authorized and then will send the corresponding share if appropriate.
- After receiving the required k shares, the user will present them to LN

- LN will reconstruct the secret share or the AC using a reconstruction function. Access will be granted if and only if AC is correct, otherwise the user is rejected.

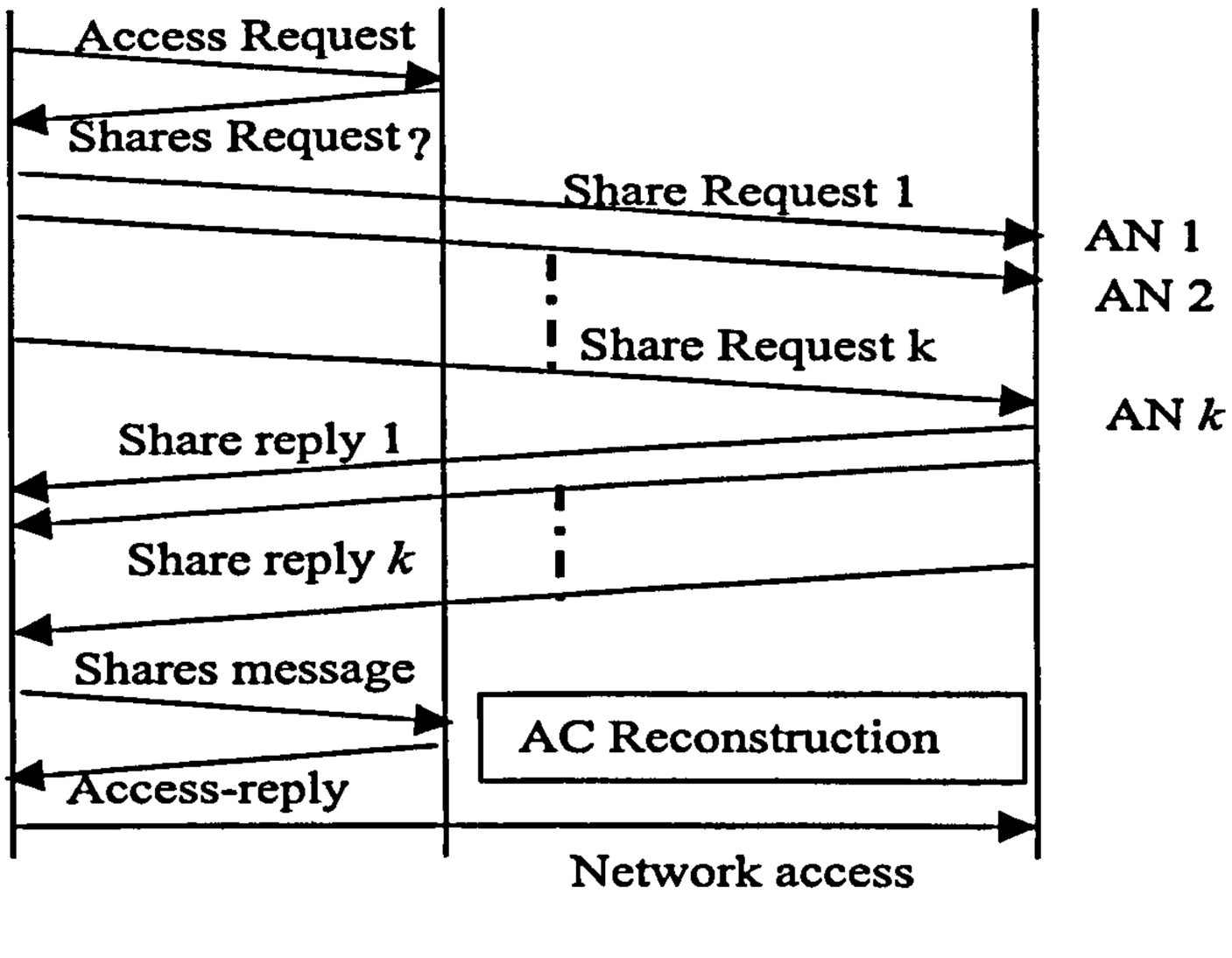


Figure 4.4: SNAP signalling

4.5 Notations

This section explains the required notations used by the protocol.

- *la* local authentication
- *LN* Local Authentication node
- AN_i Authentication Node number i
- K_{ANi} Authentication Node's public key
- t_i Time stamp from AN_i
- S_i share from AN_i

- t_a Alice's time stamp
- p_a Alice's password
- r_a Alice's random number
- K_{Alice} Alice's public key
- K_{LN} Local Authentication node public key

4.6 SNAP Messages

In this section, further details about the content of the messages between the main components of the protocol are discussed. The modified version of SNAP is considered. It is assumed that each component of SNAP has a *priori* knowledge of the other components public keys. In SNAP any public key infrastructure protocol can be used for key distribution. The required messages to access the network using SNAP are shown on Figure 4.5. The six different messages, which exist in SNAP, are now discussed.

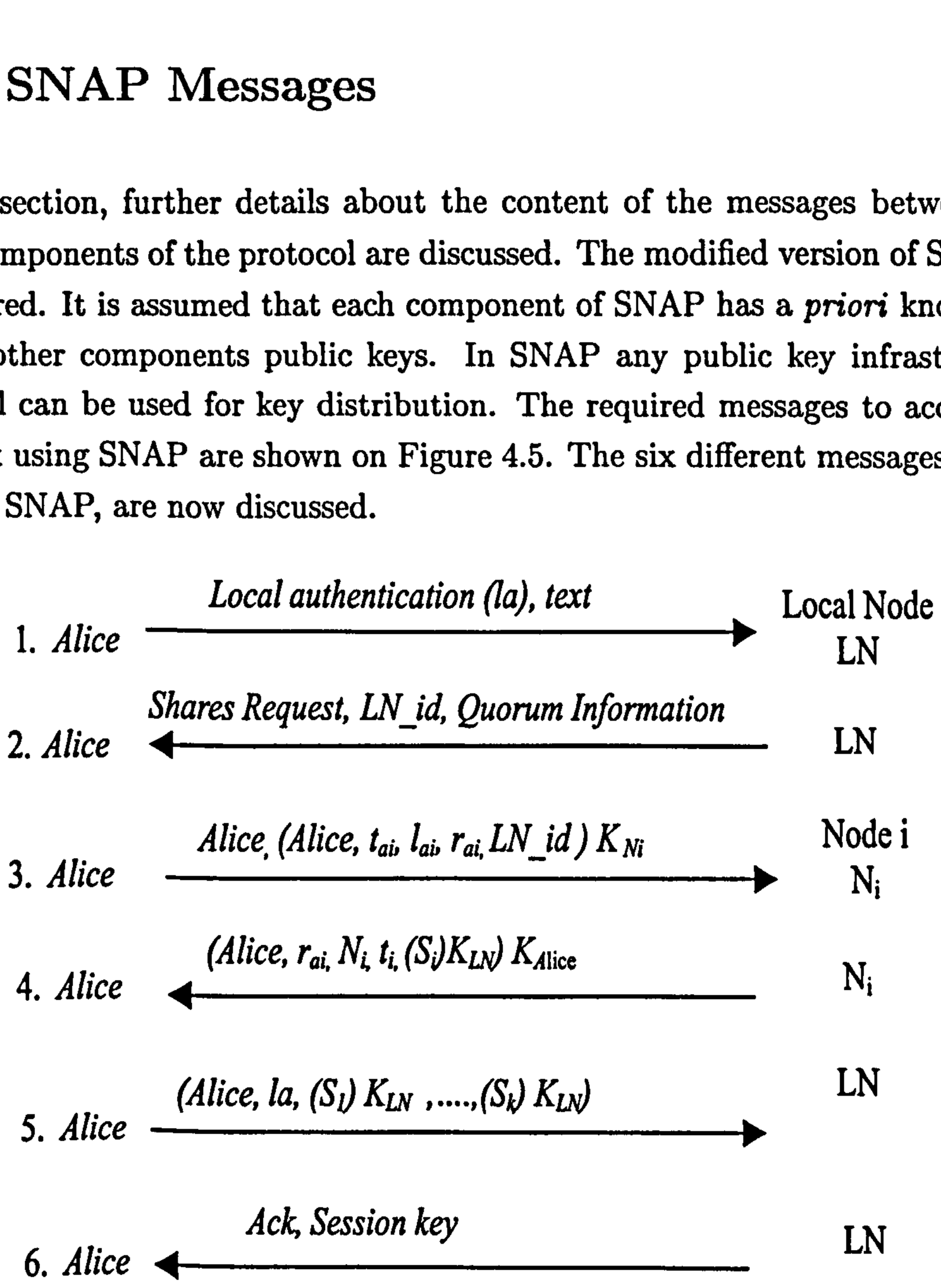


Figure 4.5: SNAP message sequence

1. *Access request*

In this message, the user (Alice) accesses the local node using local authentication techniques such as password, certificates, fingerprint, etc.

2. *Shares request*

The LN will verify its database to see if Alice is eligible, if so, the local node reply includes share request, the local node ID and, optionally information on available quorums (this information could also be provided by a directory service). If Alice is not eligible, LN will respond to Alice with a reject message

3. *Share Request i*

After receiving the shares request message Alice will contact k servers (nodes) to ask for the corresponding share from each node. Alice should provide the server with her identification, time stamp (t_a), random number (r_a) and local authentication to each node (la_i) and the local node ID (LN_id) to help the other nodes to use its public key. Alice should have a different (l_a) with each network node. Random numbers and time stamps are values used no more than once for the same purpose. They may be used to provide timeless and uniqueness guarantees, to detect message replay. The time stamp may also be used to detect forced delays caused by attackers. These parameters are necessary to avoid certain attacks, as described in Section 4.7. If the node receives the message at time T, then using $t = T - t_a$ the node can detect if the message takes more than the expected time t due to some attack and can decide to proceed or send a rejection message. The authentication between the user and network nodes is considered to be a strong authentication process. For more security, the message contains two parts; Alice's identification, sent unencrypted to make it easier for the node to recognize the sender and simplify the database search, the second part includes all parameters encrypted by the recipient's node public key to avoid impersonating attacks.

4. *Share reply i*

After receiving Alice's share request message each node will check Alice's status. If Alice is unauthorized, the node replies with a REJECT message. If Alice is authorized, the node will send Alice the node identification N_i, node time stamp t_i

and the required share (S_i) encrypted twice. First, the share (S_i) will be encrypted by the local node's public key. This is to prevent Alice from knowing the share. Second, the whole message will be encrypted by Alice's public key. Alice will use the time stamp t_i for the same reason mentioned above. The node and Alice's identification are used again to avoid impersonating attacks. For example, a malicious node might send Alice fault shares to deprive her from accessing the network while she is eligible. Using the network nodes' identification, Alice will be able to recognize the source of the message.

5. *Shares message*

Once Alice receives the share reply messages from the nodes, Alice will decrypt each message and check its identification and check to see if any message has been used by attackers using the time stamp in each message. If Alice does not receive one of the shares in an estimated time, Alice might think that either the node is down or the message has been hijacked. In this case Alice will contact another quorum and will not consider the message if it subsequently arrives. Alice knows this message by the node identification and the time stamp t_i. Alice will extract the encrypted shares, concatenate them in one message and send this, along with her identification to the LN using a local authentication procedure.

6. *Access reply*

After LN receives the encrypted shares, it will decrypt them and use the reconstruction function to reconstruct the secret (AC). If LN finds that it is correct it will send Alice an access message welcoming her to the network. The reply message may include an optional session key to be used by Alice for that session. Alice in this case can use the network resources as an eligible and authorized user who has been granted access permission by a quorum of nodes in the network. If the LN discovers that the AC is incorrect, it will respond with a REJECT message.

4.7 Security analysis

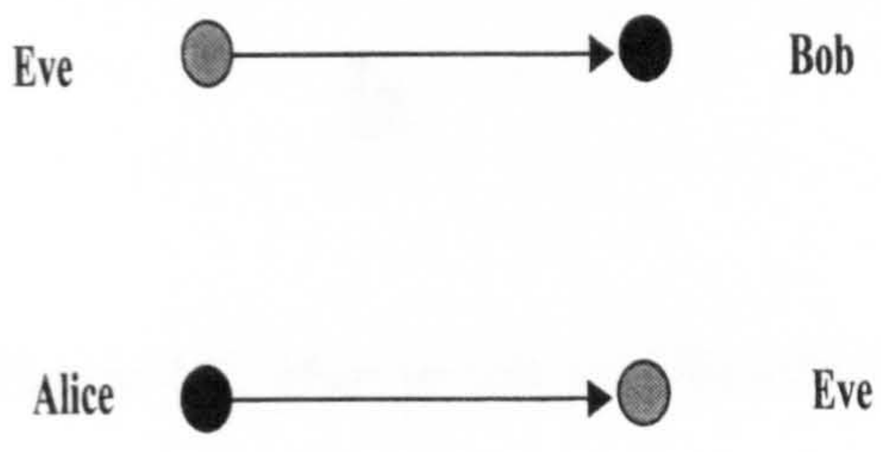
SNAP is designed to protect the connections between its components from attack. In the authentication process the use of public key algorithms (RSA, DSA), time

stamps and random numbers will prevent the following attacks:

4.7.1 Impersonation attacks

In the valid scenario as shown in Figure 4.6 the user (Alice) contacts the authentication node (Bob) and she knows that Bob is the correct destination. In the other direction, Bob contacts Alice knowing that the destination is Alice. The question is how can we keep the valid scenario between users and authentication nodes? In contrast, the impersonation attack occurs when there is an impersonator, Eve, who contacts Bob claiming that she is Alice as shown in the same Figure. So how can Bob know that Eve is not the correct user? The other impersonation scenario is that Alice contacts Eve thinking she is Bob. Therefore, how can Alice know that she is contacting the incorrect user or node? In order to avoid an impersonation attack, certain information is necessary in the message to prove the identity of both sender and receiver. In addition, the impersonator must be unable to change the message content to spoof the other party. SNAP provides a protection from impersonation attacks either in the communication between user and network nodes or in accessing the network resources. In the first case, the use of time stamps plus the user identification encrypted with the node's public key prevents the impersonator from decrypting the message. If the impersonator generates a new message deceiving the network node that she is Alice, she needs to include Alice's password in the message, which is difficult to know. In the second case, the impersonator needs to deceive all the required network nodes and to know Alice's private key to decrypt the shares.

A. Valid scenario



B. Impersonation

Figure 4.6: Impersonation attack

4.7.2 Man in the middle

A man in the middle can sit between two communication parties and receive and perhaps alter the transmitted information without detection from the two parties (see Figure 4.7) [87, 15]. Man in the middle attacks occur in two scenarios. In the first one the attacker tries to receive the messages between the two communicating parties without any alteration and without detection. In this case, the attacker does not want to contact any of the two parties. The attacker is able to achieve his objective if the transmitted data is not protected. In order to avoid this type of attack, the data must be protected and SNAP achieves that by encrypting the messages between users and nodes. In the second type, the attacker wants to contact either one or both of the communicating parties and to deceive them that he is the correct party. The attacker could receive one of the messages and change some parameters, especially the sender ID, and put his ID instead to fool the other party. SNAP prevents such an attack since the sender's ID is encrypted. Therefore, the attacker is unable to replace or change the message contents.

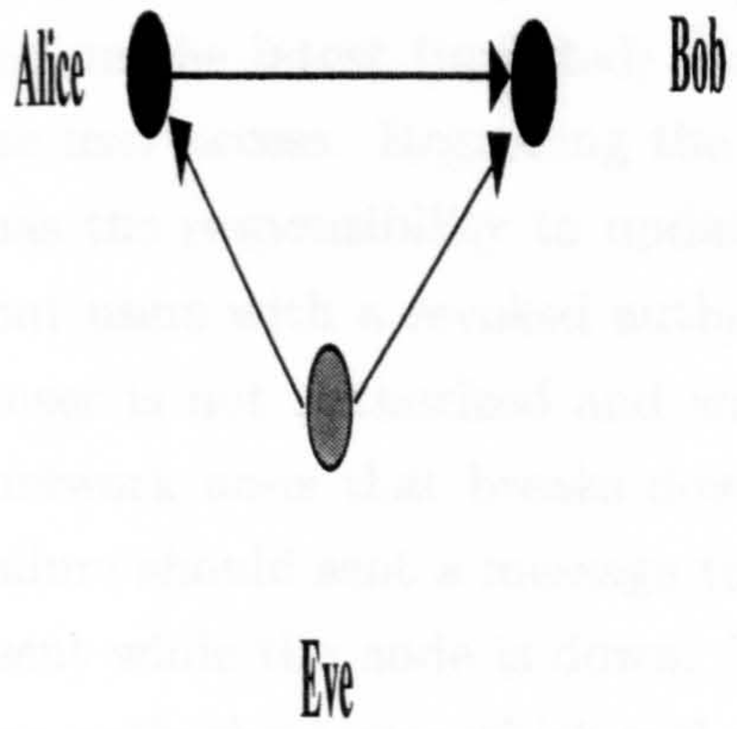


Figure 4.7: Man in the middle attack

4.7.3 Replay attacks

In a replay attack, an intruder copies a valid message and then reuses the message in order to access a resource. This can be done in order to receive services to which the intruder is not entitled. In SNAP, the use of time stamps and random numbers prevents replay attacks and forced delay attacks as well [7].

4.7.4 Unauthorized access

This kind of security threat is the main justification of SNAP. The objective of developing SNAP is to prevent unauthorized access to network resources. The authorized users are required to obtain a permission to access the network by a quorum of nodes in which strong authentication mechanisms are used. The possible threat is to use revoked shares to deceive the LN to obtain unauthorized access to network resources. To deal with this issue there are two cases:

Case 1 the user is authorized but the shares are already revoked;

Case 2 the user is no longer authorized but provides valid shares.

In the first case, the LN can detect that easily by verifying the share values.

The LN as stated in the protocol will compare the provided shares with the recent AC, which is based on the latest (updated) shares; it will find different results, and will reject the user access. Regarding the second case, the network security manager NSM has the responsibility to update and inform all network authentication nodes about users with a revoked authorization. In this case the LN will detect that the user is not authorized and will reject the access. It is recommended that any network node that breaks down for any reason because of a node crash or link failure should sent a message to NSM to ask for updated shares which have been sent while the node is down. This is to avoid providing old or revoked shares to an authorized user which will result in preventing access to the network.

4.7.5 Denial of service (network flooding)

It occurs when an intruder floods the network with false messages. This type of attack requires the targeted system to waste resources in processing such messages [15]. This may result in preventing the network from allocating resources to legitimate users. SNAP minimises the effect of this type of attack since the attack has to compromise all the quorums. Due to the distributed nature of the protocol, if one node has been attacked using this attack, the user can detect this attack since the nodes do not reply in the expected time and the user will contact another quorum to access the network resources. In the central-based authentication protocols, if the centralized entity is attacked, the legitimate user will not be able to access the resources.

4.7.6 Share theft

Additionally, SNAP prevents shares eavesdropping by encrypting all shares while transferring between SNAP components. In general, SNAP provides the following security services: Access control, entity authentication, origin authentication and confidentiality. Figure 4.8 shows a comparison between SNAP and other security mechanisms which have been described in Section 2.3.

SecuRity Service	Security Mechanisms: Access control	Hashing	Encryption	Digital signature	Digital Certificates	SNAP
Access Control	✓					✓
Entity Authentication		✓	✓	✓	✓	✓
Origin Authentication		✓		✓		✓
Integrity		✓		✓		
Confidentiality			✓	✓		✓
Non-repudiation				✓		

Figure 4.8: Security mechanisms/Security services

4.8 Management issues

This section discuss the crucial management issues that SNAP involves. These issues include the required message size based on the message contents and the required update and revocation mechanisms.

4.8.1 Message size

SNAP uses 6 messages between its components to authenticate and provide access to an authorized user. The knowledge of the message sizes will help to know the real impact of message size on the packet delay. This shows the impact of delay on the protocol's users. It is necessary to make some assumptions regarding the length and size of used variables in the message contents. These assumptions must be based on existing standards for compatibility and comparison purposes. Table 4.1 shows these variables. It is assumed that the ID (name of user or any component) is 16 bytes compatible with the IP6 standard [88]. The public keys

are assumed to be 128 bytes based on RSA public keys [7]. The share length must be equivalent to the encryption key length (128 bytes). The local authentication size depends on the mechanism used; if it is a password then the length is assumed to be 16 bytes. Using this information the size of each message can be calculated. Message 1 size is 16 byte (ID) + 16 byte (password) + 10 bytes (text) = 42 bytes. Message 2 sizes will be in the range (4-20) bytes based on the message contents. Message 3 size is equivalent to 128 bytes. The size of message 4 is 180 bytes but since the message will be encrypted by the other party's public key, this means that we need to divide the message into two blocks of length 128 bytes. In this case two blocks are used and the message size is 256 bytes. Message 5 size is based on the quorum size k. If a password is used in the local authentication then the total message size is $128 \times k + 32$. If $k = 3$ the message size = 416 bytes and if $k = 10$ the size = 1312 bytes. Message 6 is notification message and the size ranges from 130-140 bytes, 128 bytes for the session key.

Variable	Size/bytes
ID	16
Password	16
Time stamp	4
Random number	16
Public key	128
Share size	128

Table 4.1: Variable Size

As mentioned in Section 4.4.3 increasing the quorum size leads to an increase in security but in contrast it introduces more signalling and extra message size. From this discussion, it is clear that the main messages which affect the results (performance) are messages 3 and 4. Although message 5 size is the largest, its impact on the network load is negligible since it only occurs once locally between the user and the LN and it is not on the network. Messages 3 and 4 are the ones that should be considered in studying the protocol overhead on the network. However, in centralized approaches the same type of messages can be used to achieve a strong authentication process [4, 5]. In conclusion, the message size has almost the same affect on both approaches. The major overhead caused by SNAP

is the increased numbers of messages generated by users to get authenticated. The number of messages is increased by $(c+k)$ compared to centralized systems where c is number of messages needed to contact the LN; in this case $c = 4$ and k represents the quorum size. This cost is accepted if the gain is high. In critical applications security is the major issue and users can tolerate some increase in overhead as a price of a secure system.

4.8.2 Share update and revocation

Updating the shares and revocation the user authorization are main function to be addressed in SNAP implementation. In this section the update and revoke schemes and their two scenarios are discussed in more details.

4.8.2.1 Update schemes

In order to update the system parameters, there are two cases based on the implementation and the user relation to the system quorums:

Case 1 Global update: In this case the user is known to all system nodes and the user can access the system using any quorum in the system. Therefore in order to update the nodes shares or change them, the NSM needs to update all the nodes. There are two scenarios based on the relationship between quorums as follows. If there is no communication between nodes or quorums, the NSM has to send an update message to all nodes by broadcasting that message and waiting for the reply from each node.

If nodes can contact each other then the NSM can send an update message to only one or two quorums (to speed up the process). Thereafter, each node in the quorum will spread the message to its friends in the other quorums and so on until all nodes are updated and the NSM receives the acknowledgement.

Case 2 Restricted update: In this case the user is known or registered with certain quorums and not all the quorums. In this case the NSM needs to send an update message to the corresponding quorums.

In both cases the NSM uses the broadcast concept to update shares if there is no

communication existing between nodes. The advantages of this scheme is that it ensures that all nodes are updated in short time. The drawback is a lot of messages are generated which may increase the communication overhead. In the second type the overhead is low but the update process is slow. The update protocols is as follow:

1. $NSM \rightarrow N_i : (NSM, update, N_i, t_s, r_s, S_i)PK_{Ni}$

2. $N_i \rightarrow NSM : (NSM, ACK, Ni, r_s)PK_{NSM}$

The NSM update message contain the sender identity(NSM), update text,the authentication node identity (N_i), the NSM time stamp t_s for freshness purpose, a random number r_s for sequence trace and the new share S_i. the whole message is encrypted by the public key of the authentication node. Upon receiving the update message, the authentication node reply by an acknowledgement message.

Regarding the user addition to the system the user needs to register to the authentication node directly and the authentication node apply the registration policy given by the NSM.

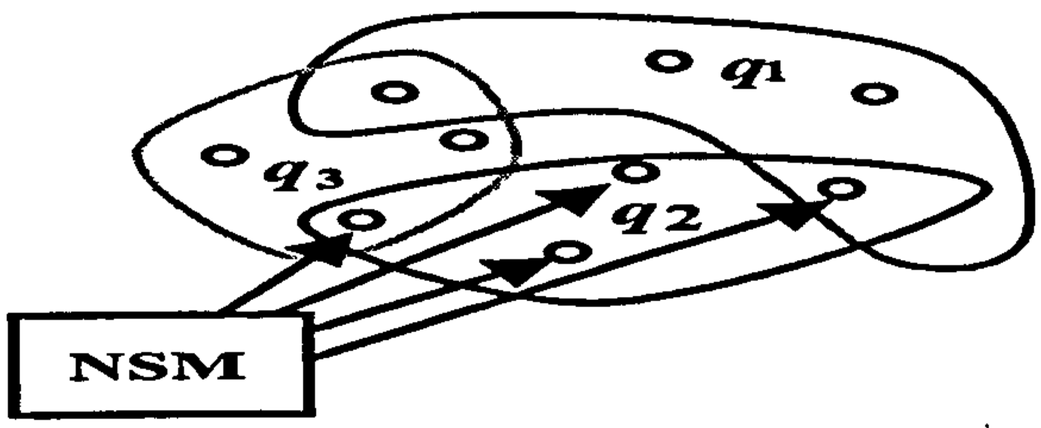
4.8.2.2 Revoke scheme

The revoke scheme has two cases. The first one is to revoke certain shares and the second is to revoke user authorization. In both cases the network security manager (NSM) contacts only one quorum in the network by sending a revoke message. The intersection property of the quorum system will guarantee that at least one member of each quorum is notified. If one revoked user is trying to access the system he needs to be authenticated by one quorum. In this case not all the quorum members will authorize him. This method is efficient since the communication overhead is low since only one quorum is contacted. The revoke protocol works as follow:

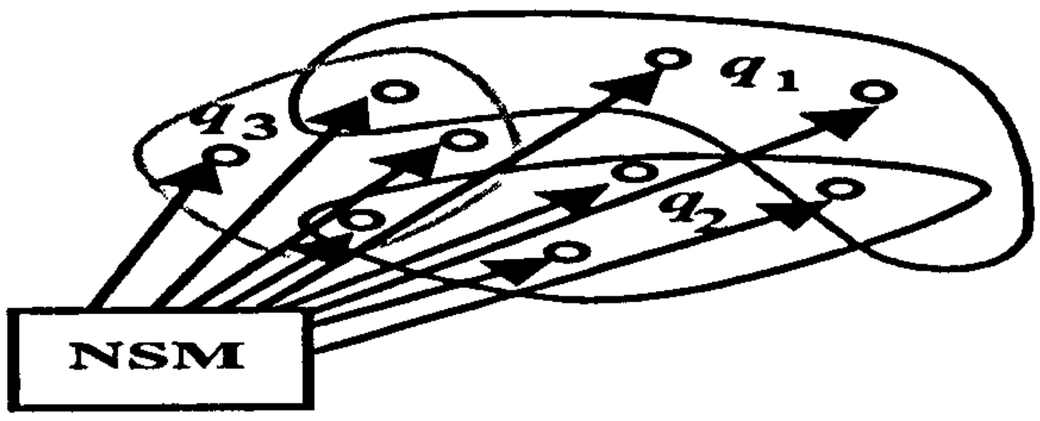
1. $NSM \rightarrow N_i : (NSM, revoke, N_i, t_s, r_s, S_i/User\_ID)PK_{Ni}$

2. $N_i \rightarrow NSM : (NSM, ACK, Ni, r_s)PK_{NSM}$

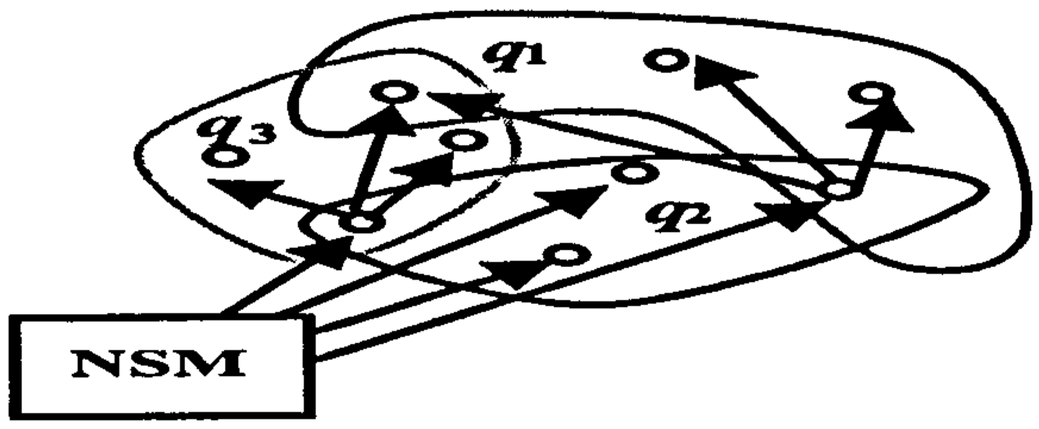
Figure 4.9 shows the update and revoke schemes



a. Updating one quorum



b. Updating all quorums (broadcasting)



c. A node contacting their members

Figure 4.9: Update and revoke schemes

4.9 Quorum design issues

In Chapter 3 an overview about quorum systems is presented. SNAP is based on secret sharing schemes that has a quorum access structure. Naor and Wool [60] and Beaver and Wool [77] proposed some efficient quorum based secret sharing systems. It is efficient from the load and availability metrics. However these quorum systems have some restrictions in the practical implementations. None of them can be constructed for any network size which is considered as a drawback of these systems. For example for a network of n nodes such that $1 < n < 100$ only 10 grid constructions can be formed. Table 4.2 shows the quorum system

and the number of constructions that can be formed for the network size range from $1 - 100$. In this case if the quorum system can't contain all the system elements then the system administrator must assign certain nodes that can fit with the used quorum system.

Quorum system	Number of constructions
Grid	10
Tree	6
Crumbling wall	25
Super grid	10
FPP	9

Table 4.2: Number of construction per quorum system

4.9.1 Quorum size

The quorum size is one of the main elements used to evaluate quorum systems and it is one of the quorum design parameters. The quorum size has a direct impact on the following issues:

- **Security:** As described in SNAP, the user needs to contact a quorum of size k to obtain the required shares that can reconstruct the original secret. The attacker needs to compromise at least one quorum to be able to compromise the system. Therefore increasing the quorum size makes the attacker's job more difficult and results in an increased security. Therefore it is advantageous to make the quorum size large from the security point of view.

- **Overhead:** Using large size quorum systems means that the number of messages and the required signalling is large. From the overhead point of view it is recommended to have small size quorums. In addition, if there are n elements in the system and all of them are required to participate in the quorum system, then the elements are divided to certain number of quorums based on the quorum construction. This means that a large quorum size provides a small number of quorums and visa versa. Sometimes it is good

to have small number of quorums based on the application and the required operations.

- **Update and revoke mechanisms:** Update and revoke mechanisms affected by the quorum size and small quorums are preferred.

- **Load:** There is a relation between load and the quorum size. For example if we have single quorum system and the user needs to contact all the elements in the quorum then for each access each member is accessed and the load is 1. If the same number of nodes in the system are divided into two quorums then the user can contact any quorum and the load can decrease to 0.5 except for the intersection nodes and so on. This means that increasing the number of quorums leads to decreased load. Therefore, decreasing the quorum size results in decreased load.

4.9.2 Variable quorum size

It is possible to have a quorum system that have a variable quorum size. The CWlog quorum system is one of these systems that has a variable quorum size [78]. In Cwlog the quorum in the bottom row is the smallest quorum in the system. The user can access the quorums randomly or can start from the bottom based on the provided access strategy. If users start from the bottom this means that the bottom quorum will be loaded more than the others. As mentioned before there is a trade off between increasing the security and minimizing the overhead and the quorum size affects both metrics. Therefore, in quorum systems which have variable quorum size it is possible to use the large size quorums for users or application that require high security and the small quorums in case of low security. In this case there are two classes high and low class. For example, in CWlog the first half of the quorums (in the bottom) can be used for low class and the upper half for the class high.

4.9.3 Example of poor design

In this section a quorum system is constructed in such a way that one node is a member of all quorums, so the quorums are intersected in a single node.

In this design, a quorum system is obtained and the intersection property is achieved. If we analyze this design we find the security is increased in a sense that a quorum of nodes are participating in the required operation. The trust is distributed between the quorum members. The main drawback is that the system has a poor availability since if the common node becomes unavailable then the entire system is unavailable. In this case the quorum system is equivalent to the centralized system from the availability point of view. Figure 4.10 shows the described example.

In general, the quorum selection and design is a crucial topic. It has a direct impact on the system operation and performance. It is recommended to design a quorum systems that provide an efficient system.

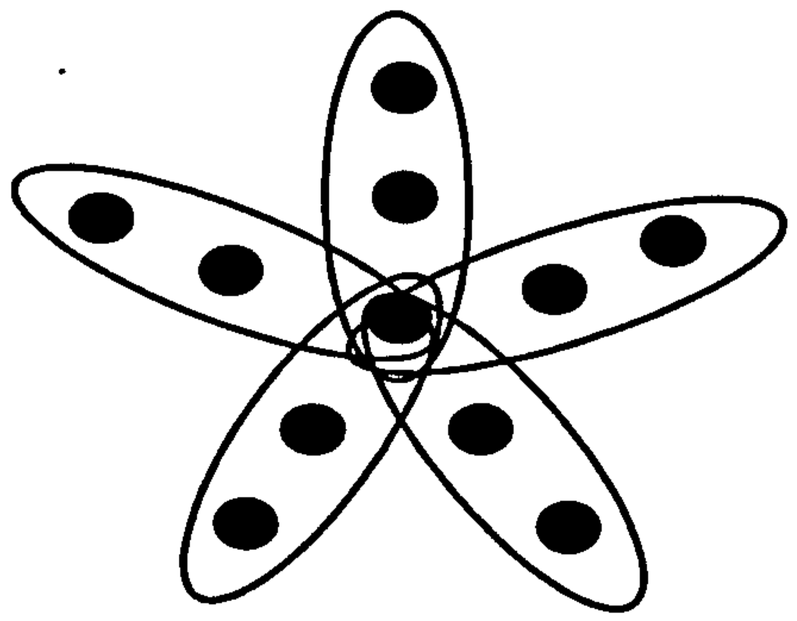


Figure 4.10: Example of a quorum system

4.9.4 Optimum design

A well-design quorum system has the following properties: high availability, optimal load and a small quorum size. In some quorum systems, a single node failure affect more than one quorum, one such quorum system is FPP. In contrast in the grid system the cost of one node failure is affect only one quorum. The quorum size in the best case for a tree system is $\log_2 n$ but in the worst case, where the root is down, the quorum size is $\lceil (n+1)/2 \rceil$. As an example, if we have a network of 49 nodes, the quorum size will range from 4 to 15 if CWlog quorum system is used. If grid system is used the quorum size is 13 in all cases. This shows the advantage of using CWlog system which can provide small and variable quorum size. Wool[67] and Komar [72] proposed CWlog and SuperGrid systems as an

efficient quorum systems.

4.10 Multi level access control

In SNAP, three access control levels are provided as shown in Figure 4.11. In the first level, the user can access the LN using the local authentication. After the user has been authenticated, LN will provide him with the chance of contacting the network nodes. In the second level, the user needs to access the network nodes. The user sends his identification and the required parameters to the nodes to get only the corresponding shares. In this level the user will not be able to access the network resources. In the third level, the user needs to access the network resources through providing the shares to the LN and if he is eligible then he has the permission to gain access to network databases and resources. It is a multilevel access control protocol. In these three levels three authentication methods are used: local authentication (passwords, etc) for the first level, public keys for the second level, and the AC for the third level.

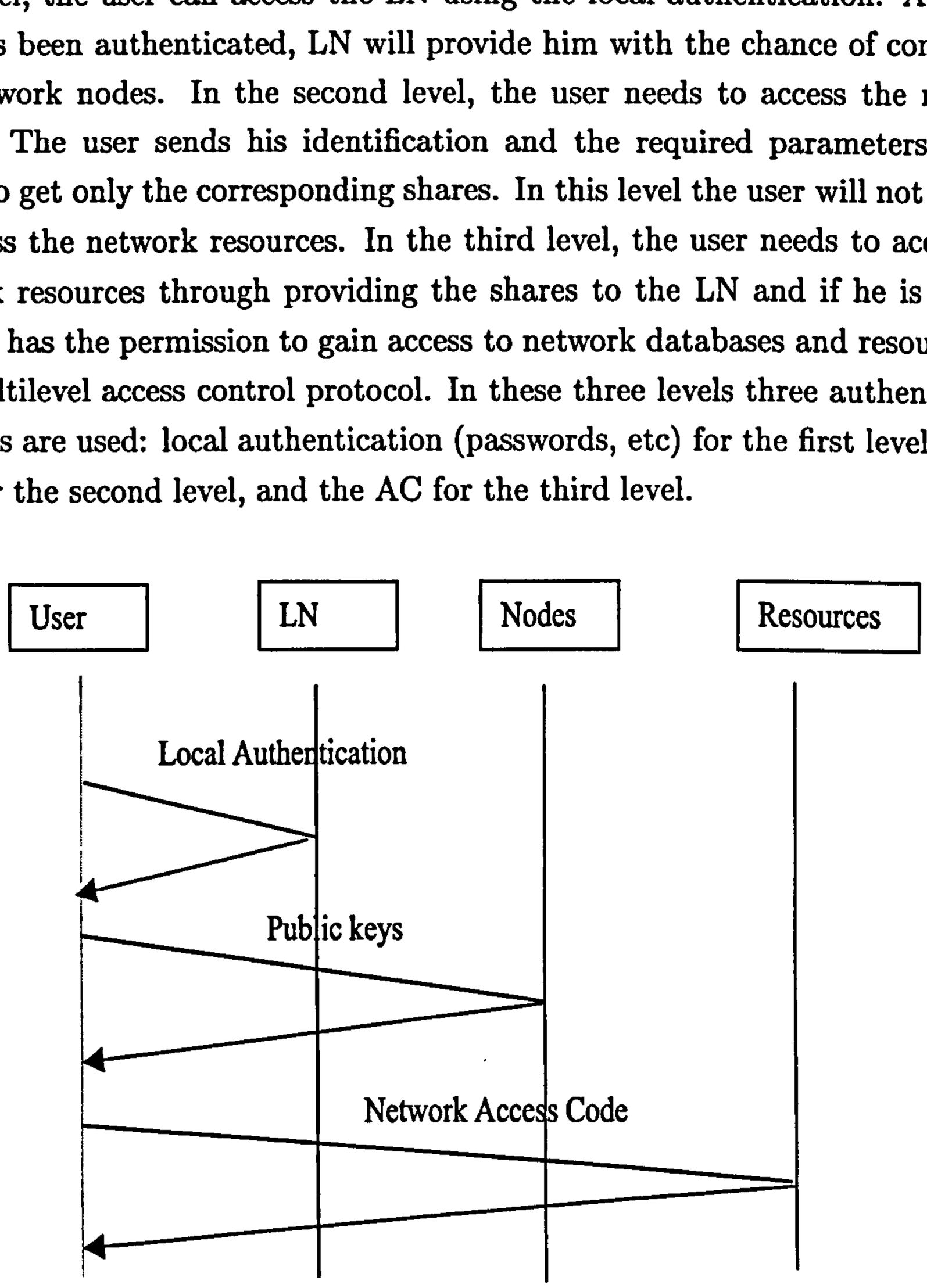


Figure 4.11: Multi level access

4.11 SNAP applications

SNAP may be considered as a method of increasing network security. It is applicable in network-based applications and distributed systems. SNAP can be used in most applications that requires secure communication. It is mostly important for critical applications. SNAP can support several implementations such as:

- Network access control
- Distributed authentication
- Database access control
- Key management
- Distributed digital signature

Some critical applications, where security is crucial and which require very secure protocols, such as commercial transactions, military applications, banking systems and so on will find SNAP well-suited to satisfy their security needs. SNAP is applicable to most network environments. The implementation of SNAP is not limited to fixed networks; it can be implemented in mobile networks as well. In GSM [40] instead of having one authentication center (AuC) that handles the user authentication and controls the network access, multiple AuCs can be used for authentication and access. In this case, the trust is distributed among a certain number of AuCs and multiple AuCs control the network access. Chapter 6 addresses the implementation of SNAP in GSM networks.

4.12 Achieved goals

The key issue of how well SNAP meets the required objectives is now discussed.

4.12.1 Security

SNAP achieves strong authentication and secure connection between the communication parties by applying time stamps and random numbers to avoid any

replay attacks. Additionally, the critical information (share) is encrypted. SNAP provides privacy and integrity by allowing the use of very secure encryption algorithms such as RSA and DSA [7]. Regarding the access control, SNAP ensures that the access of the network is given only to authorized users. This is achieved by the essence of SNAP that the permission of accessing the resources should be given by a certain number (quorum) of nodes after a prerequisite strong authentication process has been implemented.

4.12.2 Availability

Using SNAP, increased availability is achieved since in case any node breaks down the user has the ability to contact another quorum. This means that the user can still access the network despite the failure of a certain number of nodes. In a centralized system, failure of the authentication center makes the whole system inaccessible. If a centralized authentication center is duplicated for redundancy, this can reduce security since if any one is compromised, an attacker has access to the complete system.

4.12.3 Distributed trust

A very important feature of SNAP is implementation of a distributed trust model. This is achieved by distributing the network access permission among a certain number of nodes instead of using one central node. The trust is distributed and the risk is reduced resulting in increased security. If one node is compromised, the attacker will only receive one share, so the system as a whole remains secure. In the centralized case, a successful attack on the authentication center will compromise the entire system.

4.13 Meeting the Requirements

In this section, an investigation has been conducted to see if SNAP can meet the authentication protocol requirements as described in Section 2.9. The network requirements are discussed in the next chapter .

4.13.1 Authentication Requirements

The required properties that an authentication protocol must satisfy are:

- The sender identity must be one of the received message contents. If we check the main messages contents (message 3 and 4) in Section 4.6 we find the sender identity is contained. In message 3 Alice is sending her message to the authentication node and includes her identity in the message. From message 4 the authentication node N_i include its identity in the message. In addition, the same thing occurs in message 5. This means that SNAP satisfy the first property.

- The second property is to assure the freshness of the message. If we see message 3 and 4 we find the time stamp is included in both messages. As mentioned in the description of SNAP messages in Section 4.6 there is time window if the receiver did not receive the message within that window then it is considered invalid message. Therefore, SNAP satisfies the second condition.

- The third property is to include the receiver identity in the sent message. By studying SNAP messages 3 and 4 we find that the receiver identity is included.

- The fourth condition is to have a sequence tracing mechanism. In SNAP the random number is used for that purpose.

It is clear that SNAP satisfies all the required properties in the authentication model.

4.14 Conclusion

In this chapter a secure network access protocol (SNAP) has been proposed. SNAP management issues and applications are discussed, and SNAP is investigated from the security perspective against certain attacks. SNAP is a distributed authentication protocol, based on secret sharing schemes which have a quorum

access structure. SNAP exploits the features of secret sharing and quorum systems to provide an efficient, secure protocol. It ensures that users intending to connect to the network are authenticated and have valid access permission. In this protocol none of the quorum members can grant the authentication or the access on their own. Each authentication node participates partially in the authentication process. It has been suggested that SNAP is better than existing authentication protocols in terms of parameters such as availability, security and distributed trust.

SNAP does introduce additional signalling overhead. However, this is the cost of the additional security and availability features. The relative performance of SNAP in terms of metrics such as number of messages, load balancing and delay are investigated in the next chapter.

Chapter 5

Performance Evaluation of SNAP

5.1 Introduction

In order to implement the proposed distributed authentication protocol (SNAP), it is crucial to analyze it. The security analysis of SNAP was introduced in chapter 4. It is clear that SNAP has certain advantages such as increased security, increased availability and distributed trust. However, there are certain issues which must be discussed and analyzed in order to see the performance of SNAP compared to centralized approaches. In security protocols it is possible to find a protocol that has no cryptographic pitfalls but may introduce network management problems. Therefore, it is necessary to investigate the protocol performance and to study the implementation issues. This chapter investigates SNAP performance evaluation compared to centralised protocols.

5.2 Comparison metrics

In order to compare two protocols it is necessary to have the comparison based on certain metrics. The comparison metrics between centralized approaches and SNAP include the following:

- Security

- Availability
- Trust
- Overhead analysis
- Performance evaluation

5.3 Security

As previously described in Chapter 2, in centralized authentication protocols if the central node is compromised then the whole system in no longer secure. In this regards SNAP outweigh the centralized protocols since the attacker needs to compromise all the members of a quorum to compromise the system. In SNAP if the attacker succeeds in compromising one or a number of nodes less than the quorum size he still unable to compromise the entire system. From the security analysis in Chapter 4, SNAP increase the network or system security compared to centralized systems.

5.4 Availability

Regarding the availability issue, SNAP performs better than centralized protocols since if one authentication node or a whole quorum is not available the user can contact another quorum. In centralized protocols availability is a bottleneck.

5.5 Trust

SNAP distributes the trust between the quorum members since none of the members can authenticate or take the control on its own compared to centralized approaches where the trust is totally focussed on a single node. Increasing the trust leads to a risk increase and in this case SNAP improves on centralized approaches.

5.6 Overhead analysis

This section investigates SNAP overhead by studying the message size, number of messages, load balancing and management overhead.

5.6.1 Signalling

In general, increasing the quorum size leads to an increase in security but in contrast introduces more signalling. As described in Chapter 4, the main messages which affect the results (performance) are messages 3 and 4. Although message 5 size is the largest, its impact on the network load is negligible since it only occurs once locally between the user and the LN. Messages 3 and 4 are the ones that should be considered in studying the protocol overhead on the network. However, in centralised approaches the same type of messages can be used to achieve a strong authentication process. In conclusion, the message size has almost the same affect on both approaches. The major overhead caused by SNAP is the increased numbers of messages generated by users to get authenticated. The number of messages is increased by $(c+ k)$ compared to centralised systems where c is number of messages needed to contact the LN; in this case $c = 4$ and k represents the quorum size. The total number of messages for N users equal $N(c+k) = O(N)$. This cost is acceptable if the gain is high. In critical applications security is the major issue and users can tolerate some increase in overhead as a price of a secure system.

5.6.2 Load distribution

SNAP protocol is based on secret sharing and quorum systems. In centralised systems the load on the authentication server is high. One of the main advantages of SNAP is load distribution. SNAP can achieve a load balancing between the authentication servers. If users are distributed fairly to available quorums, the load in each quorum is

Quorum load $= load/l$

Node max. Load $= (m \times load)/l$

The constant m is caused by the intersection property in quorum systems since the node is a member of m quorums. l is the number of quorums in the system. Decreasing the quorum size and increasing the number of quorums in the system minimise the load on the whole network, quorums and nodes.

5.6.3 Management overhead

In SNAP, the network security manager (NSM) needs to take some management actions to facilitate SNAP implementation. These actions involve the following issues: *Selecting the secret sharing scheme.* There are many secret sharing schemes available. However, the NSM can select the required SSS to reconstruct the secret AC. SNAP is based on secret sharing schemes that have a quorum based access structure. The Cwlog quorum based secret sharing scheme and Finite Projective Plane FPP [78, 76] are such examples that can be used to implement SNAP. *Selecting the quorum system* involves selecting the quorum size, which has a direct impact on the protocol security and its efficiency. *Identifying the authentication nodes is another issue.* The NSM can choose a fixed number of nodes to handle the authentication process. The authentication nodes could change from time to time or be changed dynamically. The NSM needs to distribute the users to the available quorums. The user can be a member in more than one quorum. It is recommended to distribute users to local quorums so as to reduce the time required to contact the quorum members. *Share generation, distribution and updating* need to be considered. In addition, *share revocation and user membership creation and update* are necessary. The NSM is required to create the user membership to the corresponding quorum. This membership can change from time to time. The NSM must create the *user authorization list.* This list contains the users authorized to access the system. The authorization list must be distributed to the authentication nodes before starting the authorization process. In addition the NSM must inform the authentication nodes about any *revoked authorization.* Although SNAP involves an increased management overhead, end users of SNAP will not experience this. The entire management overhead is experienced by the NSM.

5.7 Performance evaluation

In order to investigate the performance of SNAP compared to centralised approaches, the delay is a major issue. It is important to see if SNAP will introduce high delay that may affect user convenience. The delay affect was investigated by simulation.

5.7.1 Simulation environment

The network simulator NS-2 [89] is used for the simulation. NS is a discrete event simulator targeted at networking research. NS provides substantial support for simulation of TCP, routing, and multicast protocols over wired and wireless (local and satellite) networks.

5.7.1.1 Simulation validation

In order to validate the simulation , two different methods are considered. In the first method an illustrative example is used to calculate the round trip (RT) delay that a packet will take between two nodes. Figure 5.1 depicts a 4 network nodes and the propagation delay is chosen to be 10 ms. The link capacity is 1 MB/s for all links. The queueing delay is assumed equal 0. The total delay is:

Total delay = Transmission delay + propagation delay + queueing delay.

RT delay = 2× Total delay.

For a packet of size 256 bytes, the transmission delay is 2ms. By inspection, the RT delay between node 0 and nodes 1, 2, 3 are 24 ms, 48 ms and 72 ms respectively. The simulation results for the same network are 24.099 ms, 48.19 ms and 72.30, hence the error between these sets of results is negligible.

In the second method, the NS-2 has a number of tutorial examples [90]. These examples have been simulated and the obtained results matched the given and expected results.

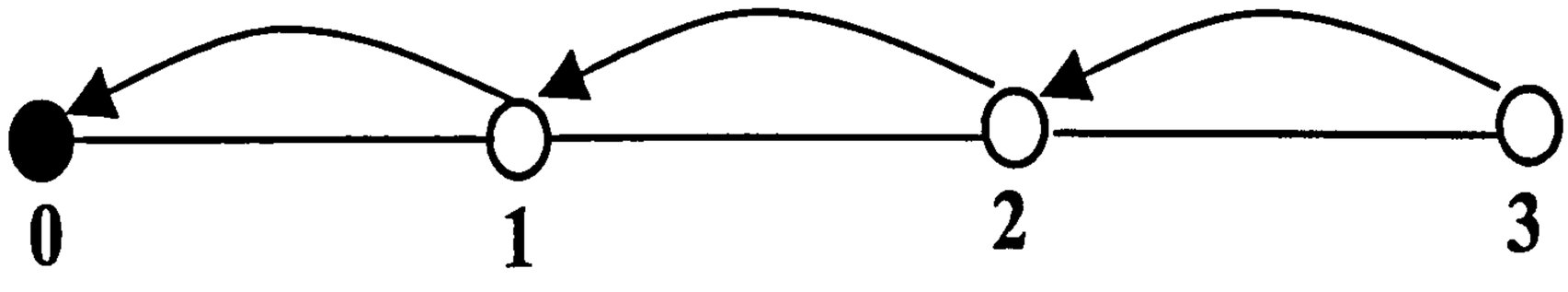


Figure 5.1: Validation example

5.7.1.2 Statistical evaluation

In order to ensure that the obtained results are a good representation, the statistical confidence of the simulation output is important. This statistical confidence is represented by the Confidence Interval which is calculated using Replication/Deletion approach [91]. This approach involves running n statistically independent simulation replications and considering the output of each run. These outputs are then used to calculate the confidence interval. The average delay and the 95% confidence interval of users from different nodes contacting a certain node in a ring network of 6 nodes are shown in Table 5.1. The relative error of the observed mean values is below 0.05 in all cases.

Average delay (ms)	26.653	53.49	81.09	53.94	27.99
95% Confidence interval	0.37	0.535	0.824	0.516	0.442

Table 5.1: The average delay (ms) and confidence interval

Different topologies and network sizes are considered in the study. Figure 5.2 shows different networks and network size of 6, 9, 12, and 16 nodes. To make a valid comparison, a consistent approach between all networks is required. Therefore we start using the topology in Figure 5.2 part (a) where the network size is 6 and start expanding the network size keeping the connectivity as similar as possible.

To improve the statistical significance of the results, the simulation was run for a large number of cycles. It is assumed that each node represents an authentication center. The user will send a packet that contains the required information for the user to identify themselves to the other nodes. In this case we assume that the authentication packet is large enough to handle all the needed information.

The packet size is 256 bytes based on the calculation conducted in Section 4.8.1. Using SNAP, the user will simultaneously send k packets to the authentication nodes. The selection of the routing path is based on the built-in routing protocol provided by NS-2. Since ping agents are used, packets will go back directly after reaching the destination towards the source. The conducted scenarios to simulate centralized systems and SNAP are follow. For the centralized case all users in the network will contact a certain node as the centralized authentication center then the average is taken. This scenario will continue until all the network nodes are checked. At the end, the average delay for all nodes will be considered as the average delay for the centralized approach. In the case of SNAP, each user will contact his quorum by sending ping packets to the quorum members. In case of SNAP, the user needs to wait for the last packet to arrive such that the waiting time does not exceed the predefined time limit. Therefore, the last packet time is the one which considered. The scenario will continue for all users then the average delay is calculated.

In order to have a real implementation, it is important to send SNAP packets while the network is loaded. This is because the network traffic have a direct impact on any results and it is crucial to consider the worst case not the apposite. Therefore, it is important to dump the network with what we call it background traffic and let the network reach a steady state then SNAP traffic starts. More details about the assumptions are listed below.

5.7.2 Assumptions

- To generate the background traffic, UDP agents are used. Both source and destination nodes use UDP agents. By simulating using UDP, this may in fact be the worst case because in a real network TCP will back-off in high loads.

- The background traffic is exponentially distributed with packet size 500 bytes.

- For SNAP traffic, ping agents are used in both the source and destination nodes. Using the ping agent we can calculate the round trip time delay between source and destination.

- The authentication packet size is assumed to be 250 bytes as has been calculated in Section 4.8.1.
- The link capacity between nodes is 1 Mb/s and the link delay is 10 ms.
- The authentication processing time is not considered in this study.

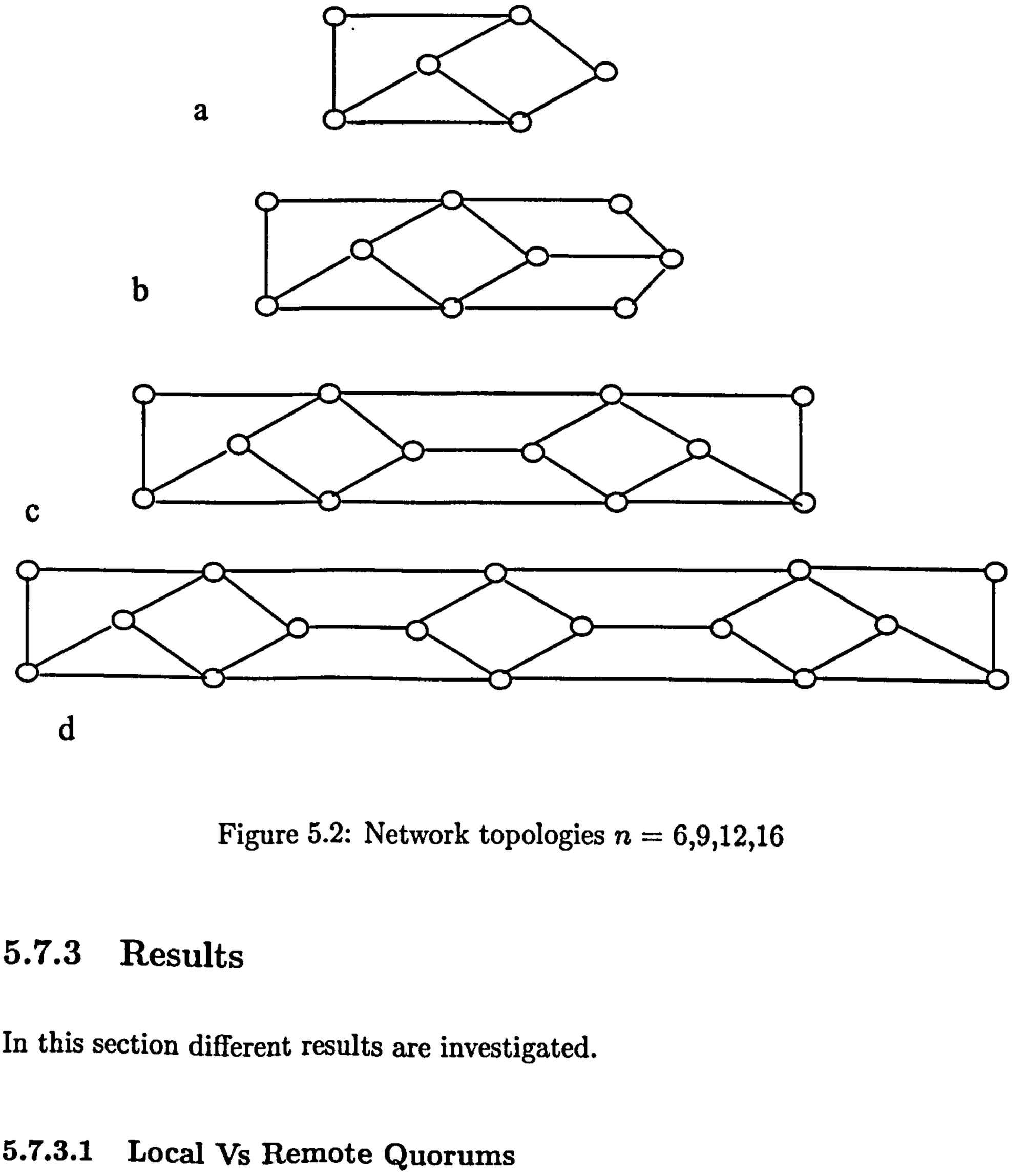


Figure 5.2: Network topologies n = 6,9,12,16

5.7.3 Results

In this section different results are investigated.

5.7.3.1 Local Vs Remote Quorums

In order to study the delay caused by SNAP relative to centralised approaches, a network of 7 nodes is simulated (see Figure 5.3). Table 5.4 shows the average

round trip delay resulting from a user sending authentication packets to all network nodes. The delay is the queuing delay and the transmission delay. In SNAP the user needs to wait for the last packet to arrive before being able to construct the AC. As shown in Figure 5.3, there are four quorums and each quorum has 3 members; the quorum size, k, is 3. Each node is a member of two quorums to have the intersection property as shown in Table 5.2. The user distribution on quorums can lead to different results and has a direct impact on the protocol efficiency. Figure 5.5 shows the average delay experienced by users accessing local and remote quorums. Table 5.3 shows the user distribution on quorums. It can be observed from Figure 5.5 that accessing local quorums lead to decreased delay. Figures 5.6 and 5.7 show a comparison between SNAP and centralised approaches. In Figure 5.6, the average delay experienced by users contacting a centralised authentication node (CAN) or contacting local quorums is shown. The CAN is node 3 and SNAP users accessing local quorums. From Figure 5.6 it is evident that SNAP will not result in increased delay for all users. There are some cases where SNAP may perform better than a centralised approach if it is used to provide a more localized distributed authentication. Figure 5.7 shows that users accessing a remote CAN (node 6) will experience increased delay compared to SNAP.

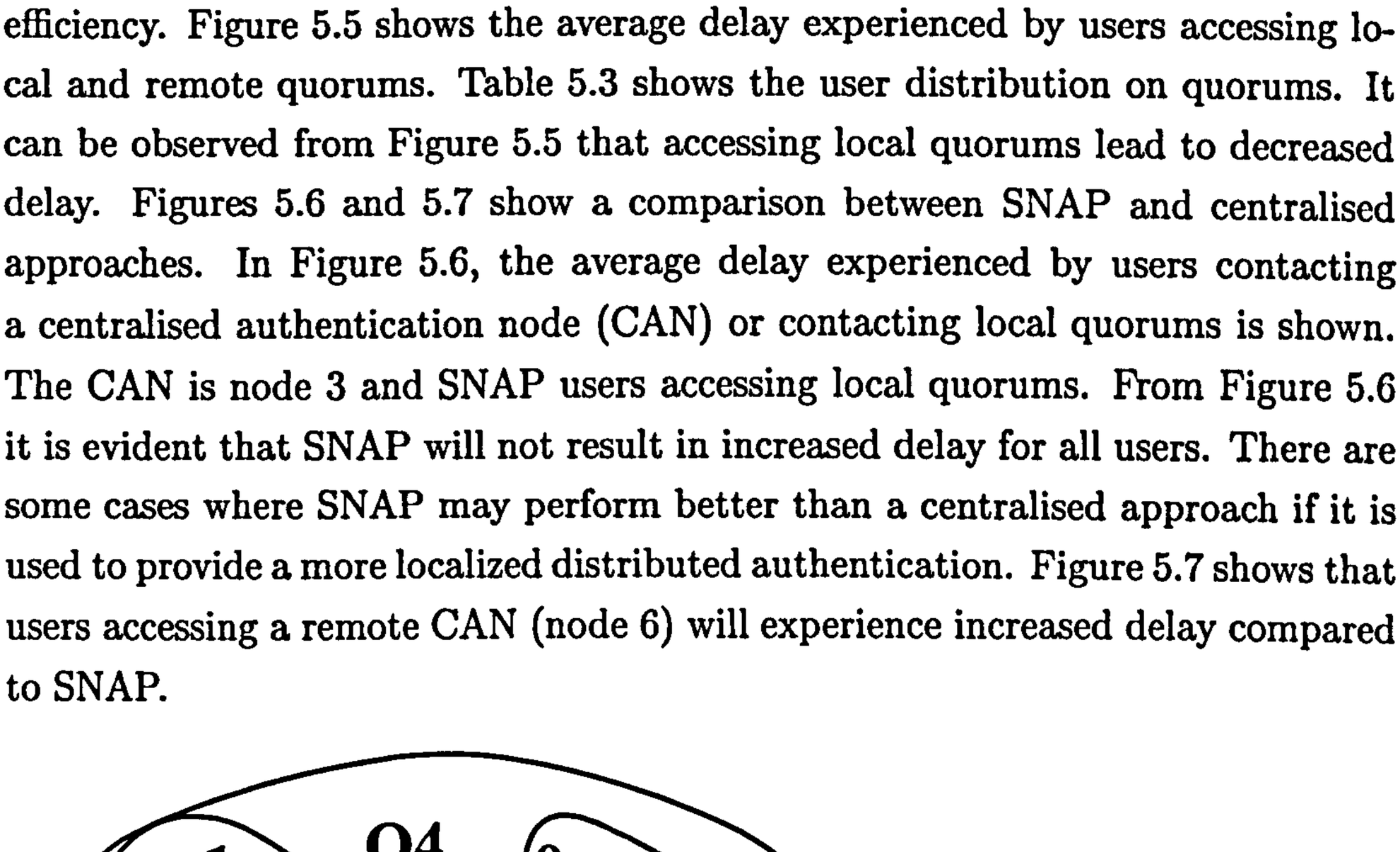


Figure 5.3: A quorum system where n = 7

user \ node	node 0	node 1	node 2	node 3	node 4	node 5	node 6
user 0	0.0	27.01	53.91	27.3	52.86	26.85	53.6
user 1	26.43	0.0	26.87	53.35	53.62	53.51	79.83
user 2	53.98	26.83	0.0	26.69	27.55	53.75	80.69
user 3	26.79	53.86	26.85	0.0	26.48	53.27	80.63
user 4	53.99	53.50	26.55	26.79	0.0	27.1	54.06
user 5	26.39	53.07	53.85	53.35	26.76	0.0	27.1
user 6	53.92	80.62	80.22	80.61	53.78	26.87	0.0

Figure 5.4: Average delay

Quorum	Members
Q1	1,2,5
Q2	2,3,4
Q3	0,4,5
Q4	0,1,3

Table 5.2: Quorum members

User	Local Quorum	Remote Quorum
U0	Q4	Q1
U1	Q1	Q2
U2	Q2	Q3
U3	Q2	Q3
U4	Q2	Q4
U5	Q3	Q4
U6	Q3	Q2

Table 5.3: User distribution

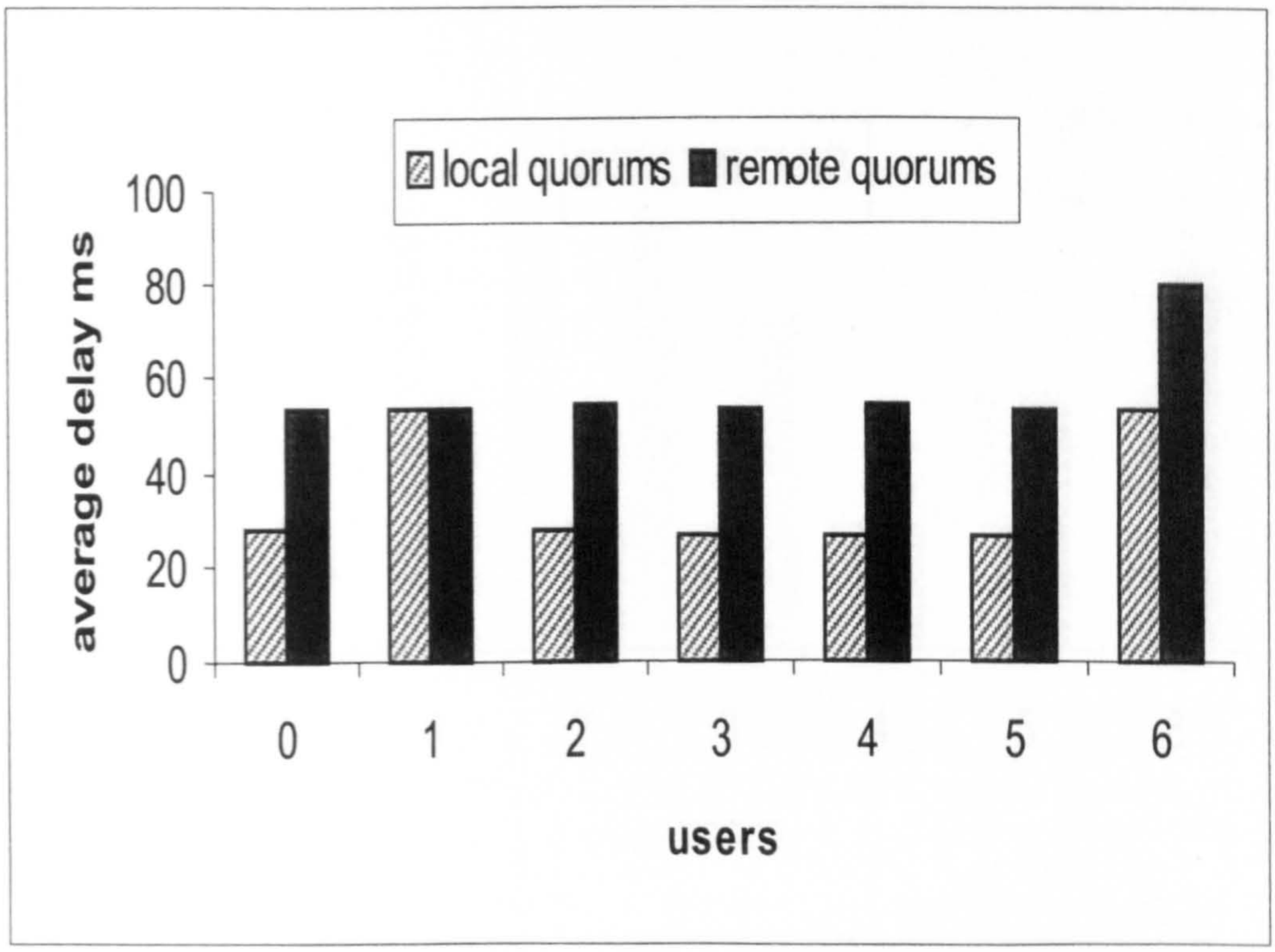


Figure 5.5: Local V remote quorum

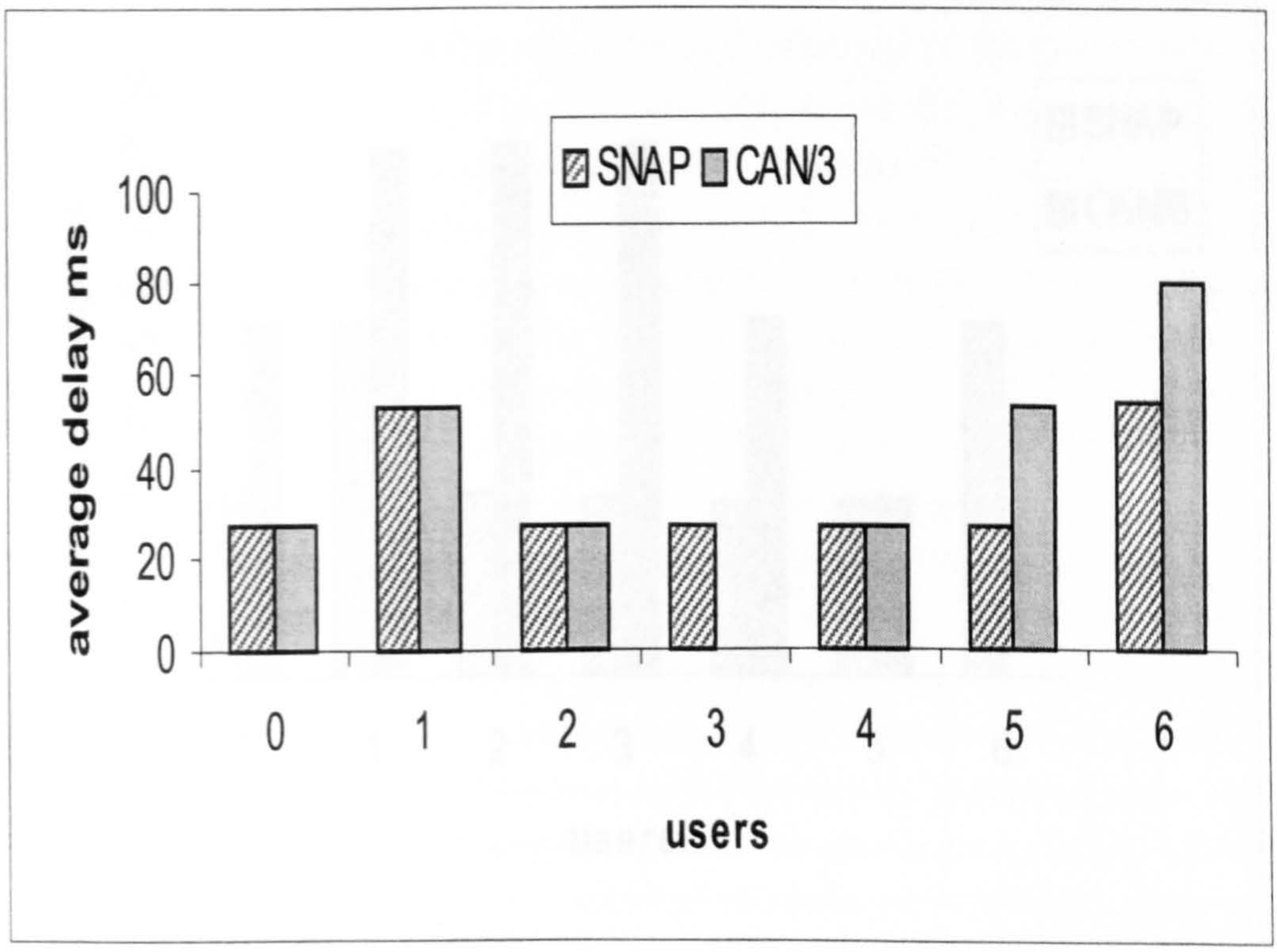


Figure 5.6: CAN is node 3 V local quorums

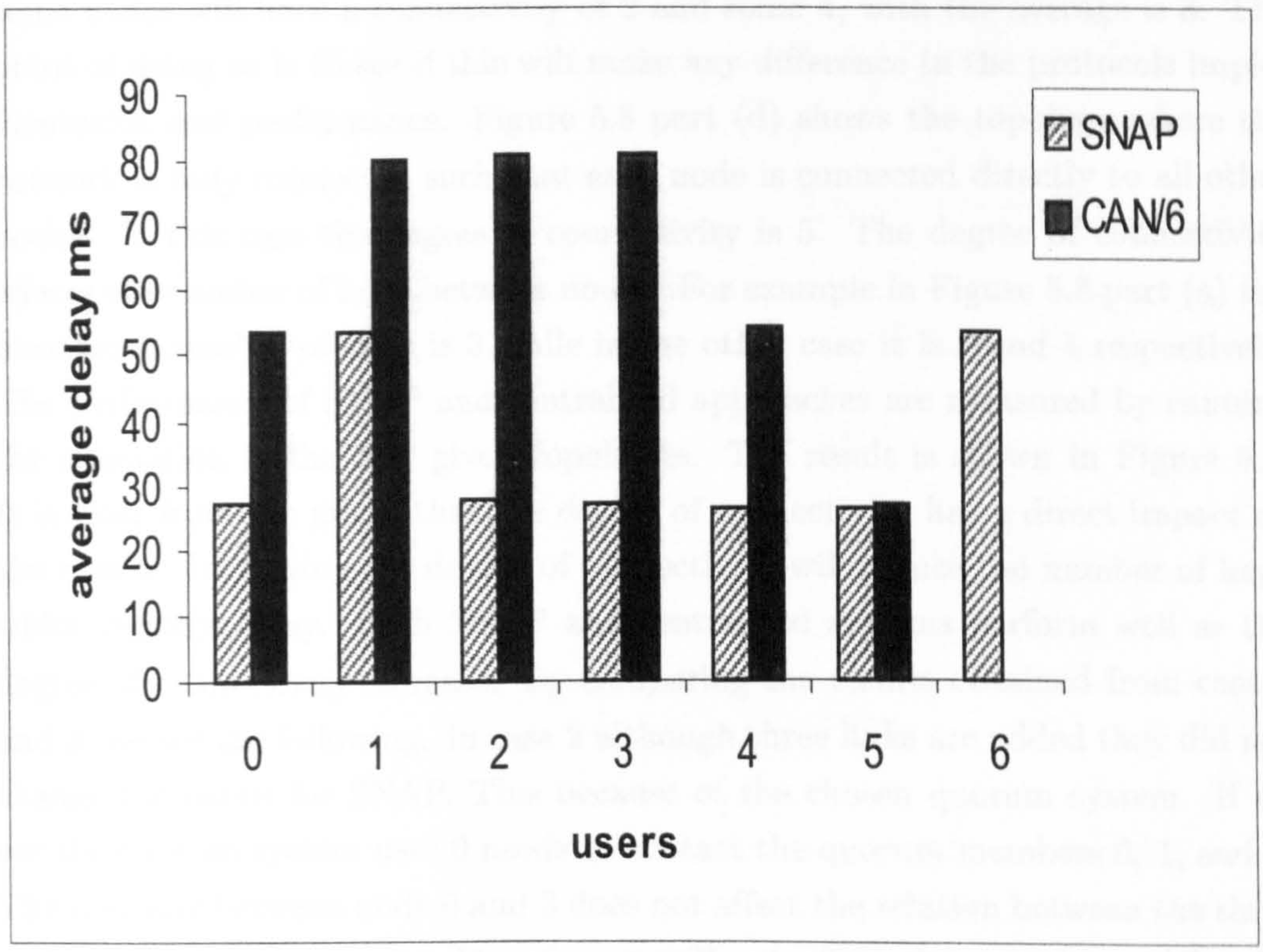


Figure 5.7: CAN is node 6 V local quorum

5.7.3.2 Network connectivity effect

It is important to consider the network design parameters that may affect the implementation of new protocols. The degree of connectivity is one of these parameters. In general, the delay is affected by the number of hops between the nodes. This means that the connectivity will have a direct influence on the result. It is crucial to see the performance of SNAP and centralised authentication protocols in the case of variable degree of connectivity. In order to make that investigation a ring topology of 6 nodes was considered. The quorum system has been used Q has four quorums and the quorum size $k = 3$, $Q = \{\{0,1,5\},\{1,2,3\},\{3,4,5\},\{0,2,4\}\}$. Figure 5.8 shows the topology with different degree of connectivity. In Figure 5.8 part (a) the degree of connectivity is 2. In part (b) and (c) a 3 new links have been added in two different ways. In the first one each node has a degree of connectivity of 3 but in the second one

some nodes will have a connectivity of 2 and some 4, with the average is 3. The point of doing so is to see if this will make any difference in the protocols implementation and performance. Figure 5.8 part (d) shows the topology where the network is fully connected such that each node is connected directly to all other nodes. In this case the degree of connectivity is 5. The degree of connectivity affects the number of hops between nodes. For example in Figure 5.8 part (a) the maximum number of hops is 3 while in the other case it is 2 and 1 respectively. The performance of SNAP and centralised approaches are measured by running the simulation in the four given topologies. The result is shown in Figure 5.9. It is clear from the graph that the degree of connectivity has a direct impact on the results. Increasing the degree of connectivity will reduce the number of hops which reduces delay. Both SNAP and centralised systems perform well as the degree of connectivity increase. By comparing the results obtained from case 2 and 3, we see the following. In case 2 although three links are added they did not change the result for SNAP. This because of the chosen quorum system. If we see the quorum system user 0 needs to contact the quorum members 0, 1, and 5. The new link between node 0 and 3 does not affect the relation between the three nodes in the first quorum. Therefore it will not affect the experience delay for SNAP users. This is shows the importance of mapping the logical construction of the quorum system to the physical topology. With respect to case 3 the new added links serve the users on the given quorum system and affect the results.

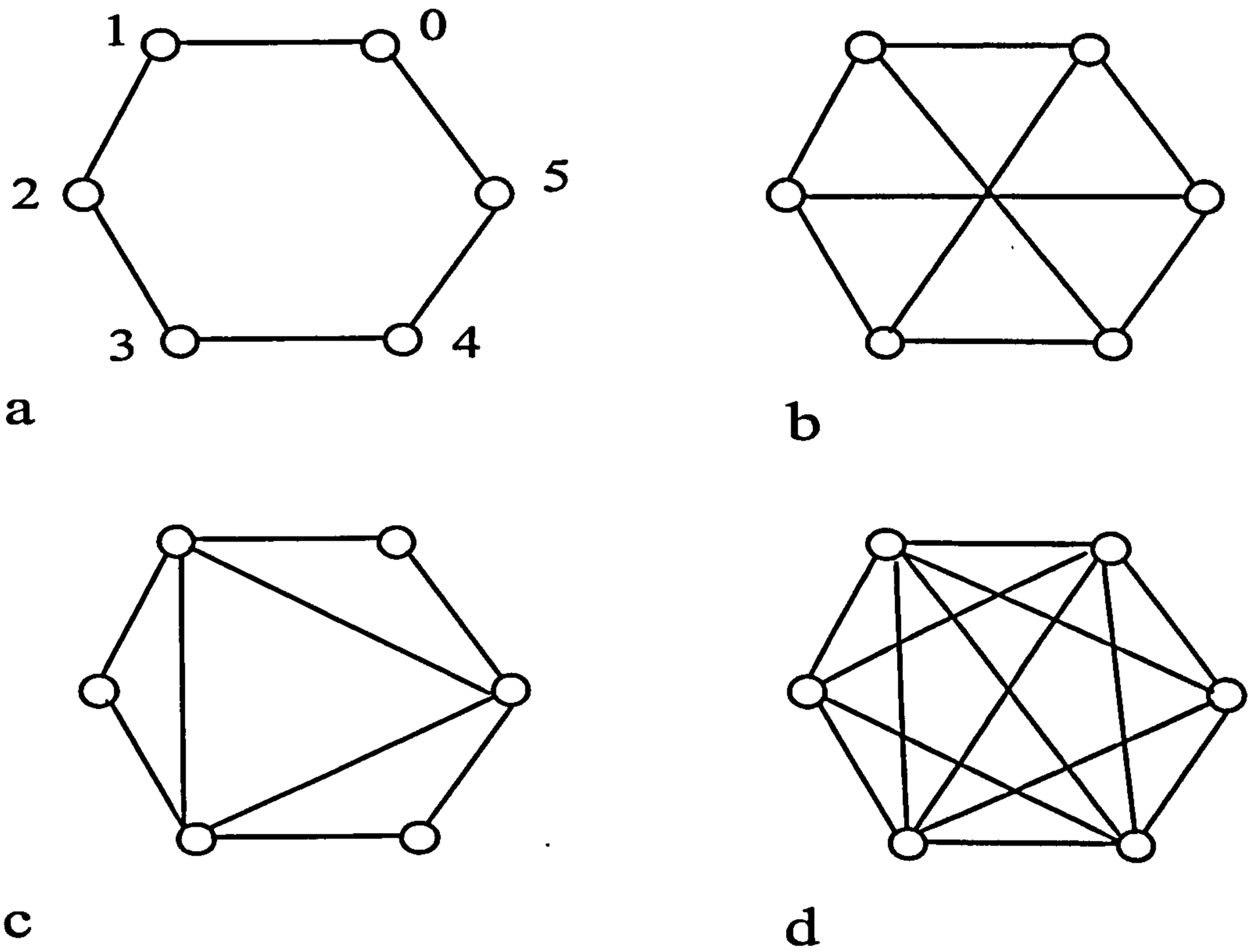


Figure 5.8: Topologies with different connectivity

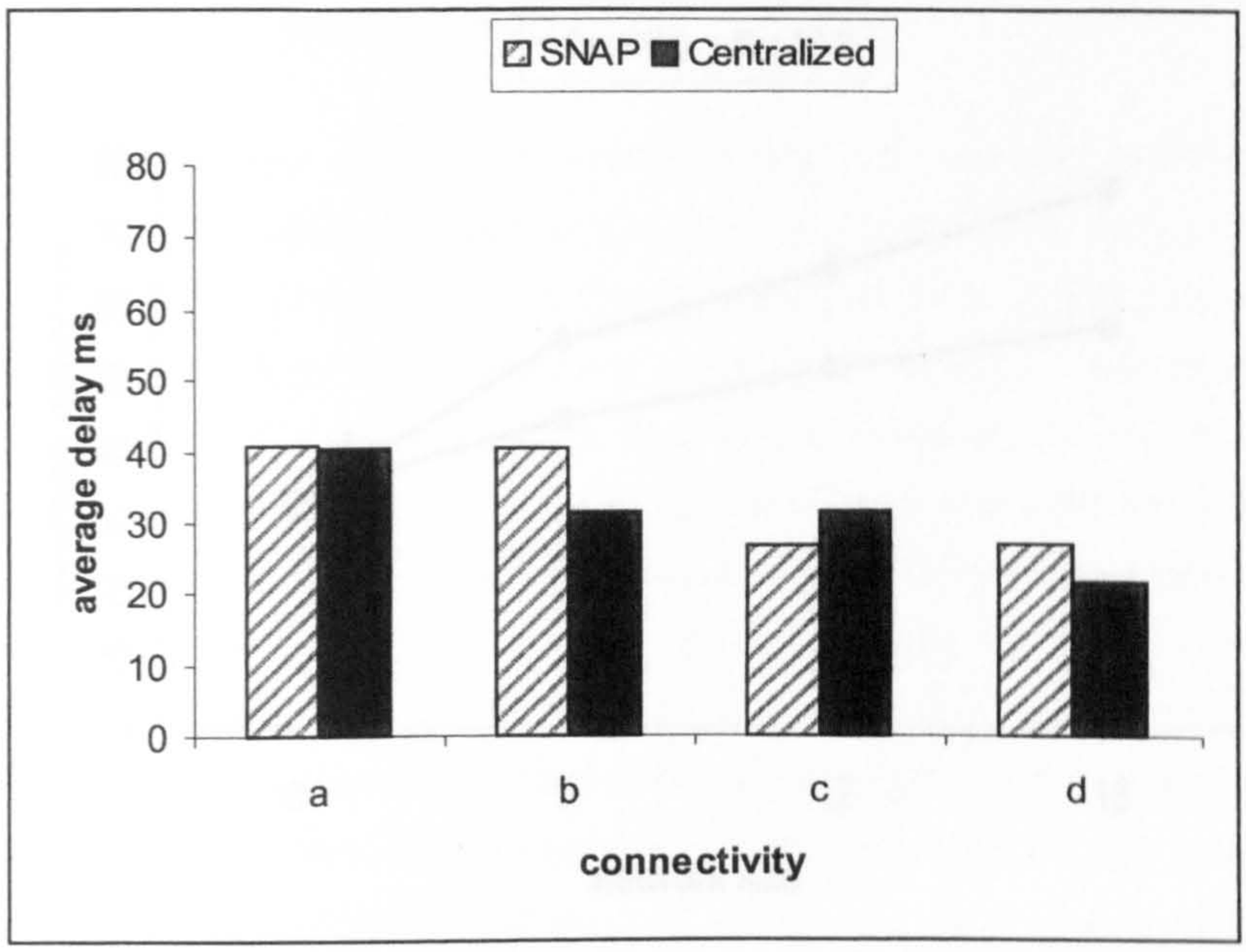


Figure 5.9: Impact of connectivity

5.7.3.3 Effect of Network size

The network size is another issue to be considered. In order to see if SNAP is scalable, different network sizes were simulated. The results show that increasing the network size may introduce an increased delay if SNAP is used. This is obvious since contacting one node is different from contacting k nodes and if the network is large the k nodes may not be neighbors. In addition the quorum size will usually increase with the network size taking into consideration the intersection property between quorums. Figure 5.10 shows the impact of the network size with different quorum sizes in each case. For SNAP the quorum sizes k for the given networks were 3, 5, 6, and 7 respectively. Figure 5.11 shows the average delay experienced by users where the network size is 16 nodes. In this figure the increased delay caused by SNAP is marginal except for one or two users. Those users belong to certain quorums and their quorum members are in different locations. The advantages of SNAP may be worth that increased delay particularly in critical applications

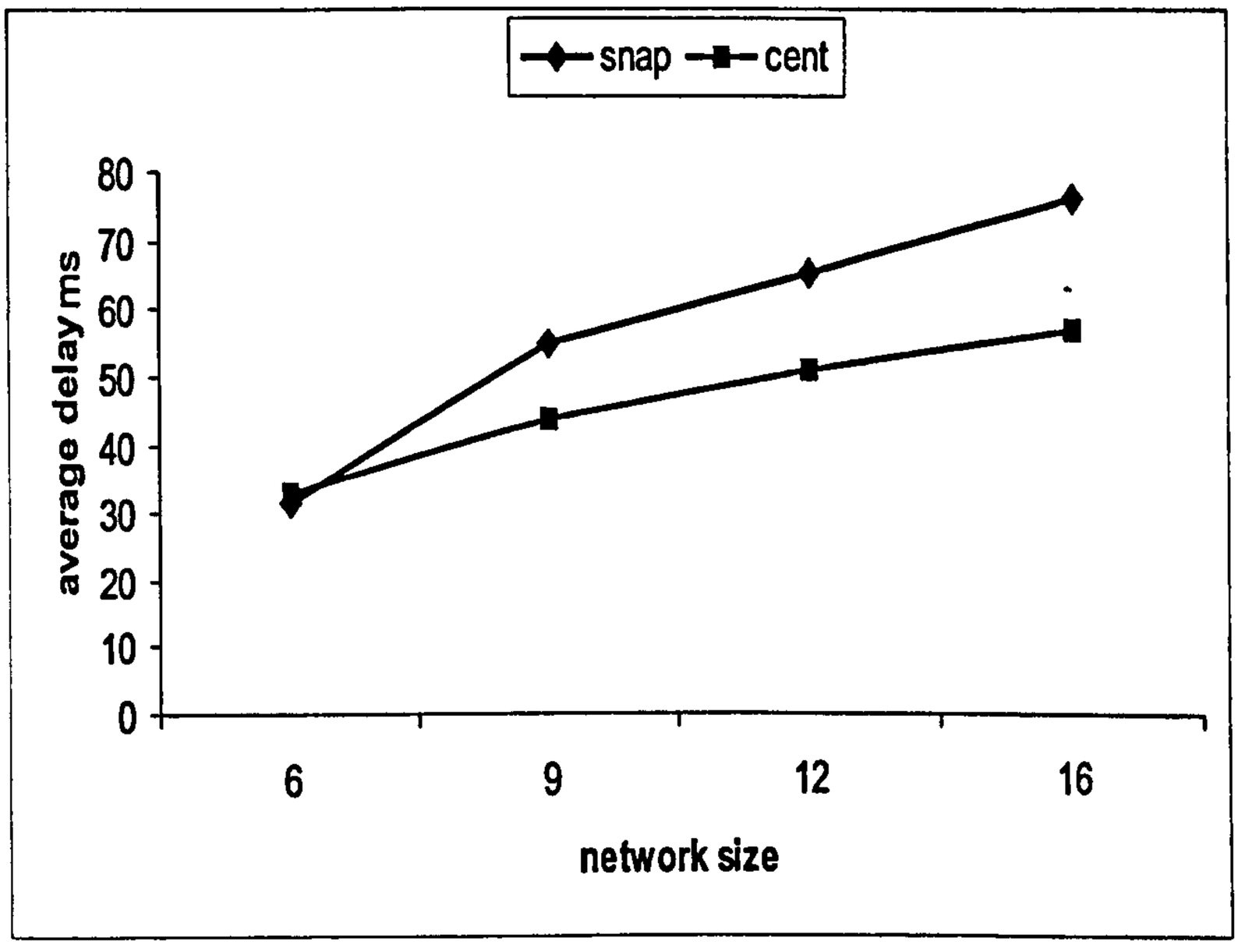


Figure 5.10: The network size impact

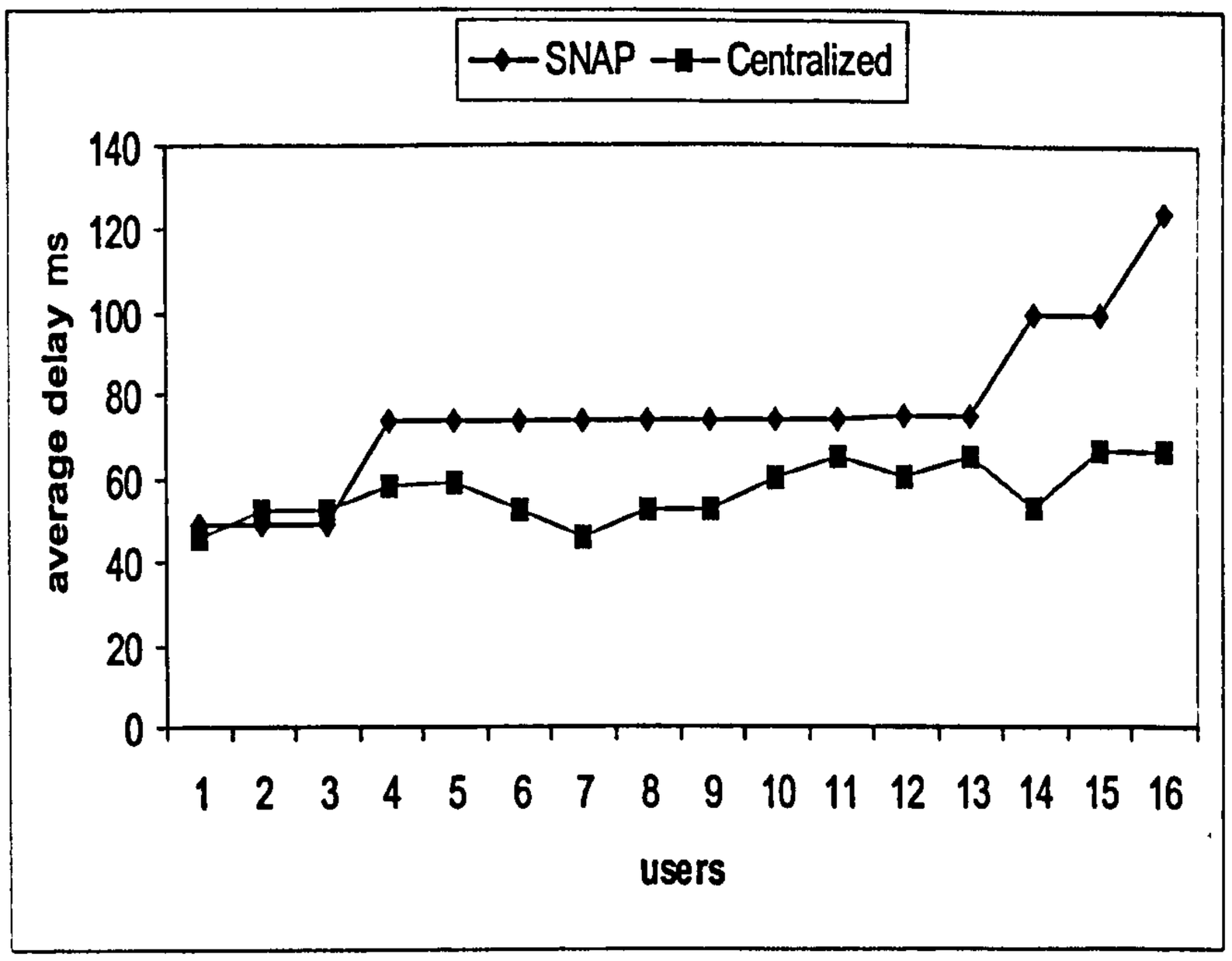


Figure 5.11: Average delay for all users where n =16

5.7.3.4 Effect of Quorum size

The quorum size is one of the measurements used to compare different quorum systems constructions [67]. Increasing the quorum size leads to increased security. The attacker needs to compromise all the quorum members in order to reconstruct the secret, so increasing the quorum size makes the attacker's job more difficult. The negative side is the resultant high load and communication overhead. If the quorum size is large, this means that the user needs to send numbers of messages to all the quorum members which increases traffic. However, there is a trade off between increasing the security and minimizing the load and overhead. In general it is recommended to built quorum system that have small quorum size. Having small quorums increases the number of quorums in the system. This will decrease the required access time but may increase the revocation process based on the revocation mechanism. Figure 5.12 shows the importance of minimizing the quorum size k. It is clear that SNAP can perform well even with large network size. It is possible to have k very small but the cost is to design very poor quorum system. For example a quorum system can have one node that is a member in all quorums which satisfies the intersection property but has very poor availability as described in Section 4.9.3.

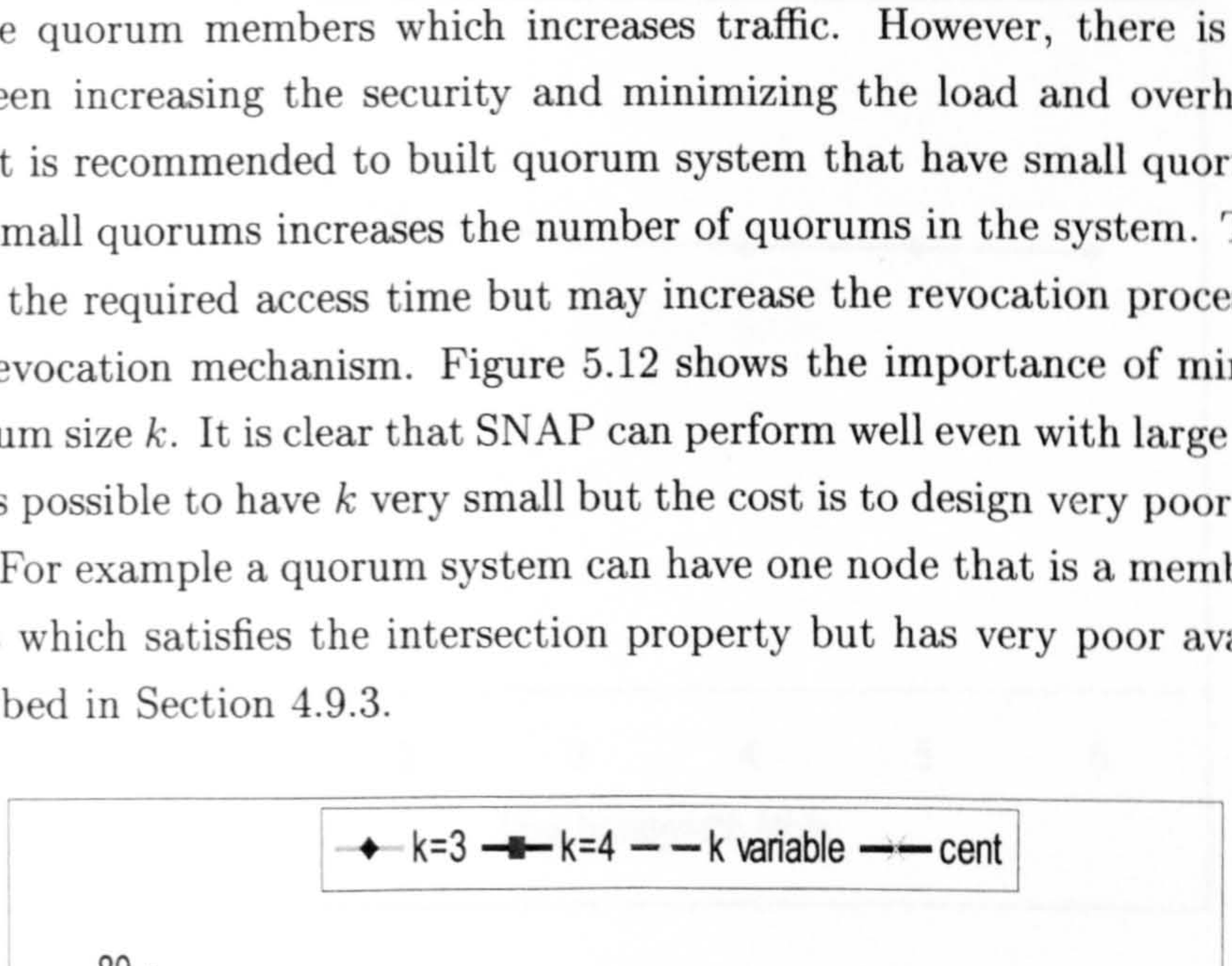


Figure 5.12: Impact of quorum size

5.7.3.5 Impact of link rate

The link bandwidth can have a direct impact on the experienced delay. Figure 5.13 shows the average delay experienced by SNAP users at different link bandwidth. It is clear that increasing the link bandwidth will help to decrease the delay.

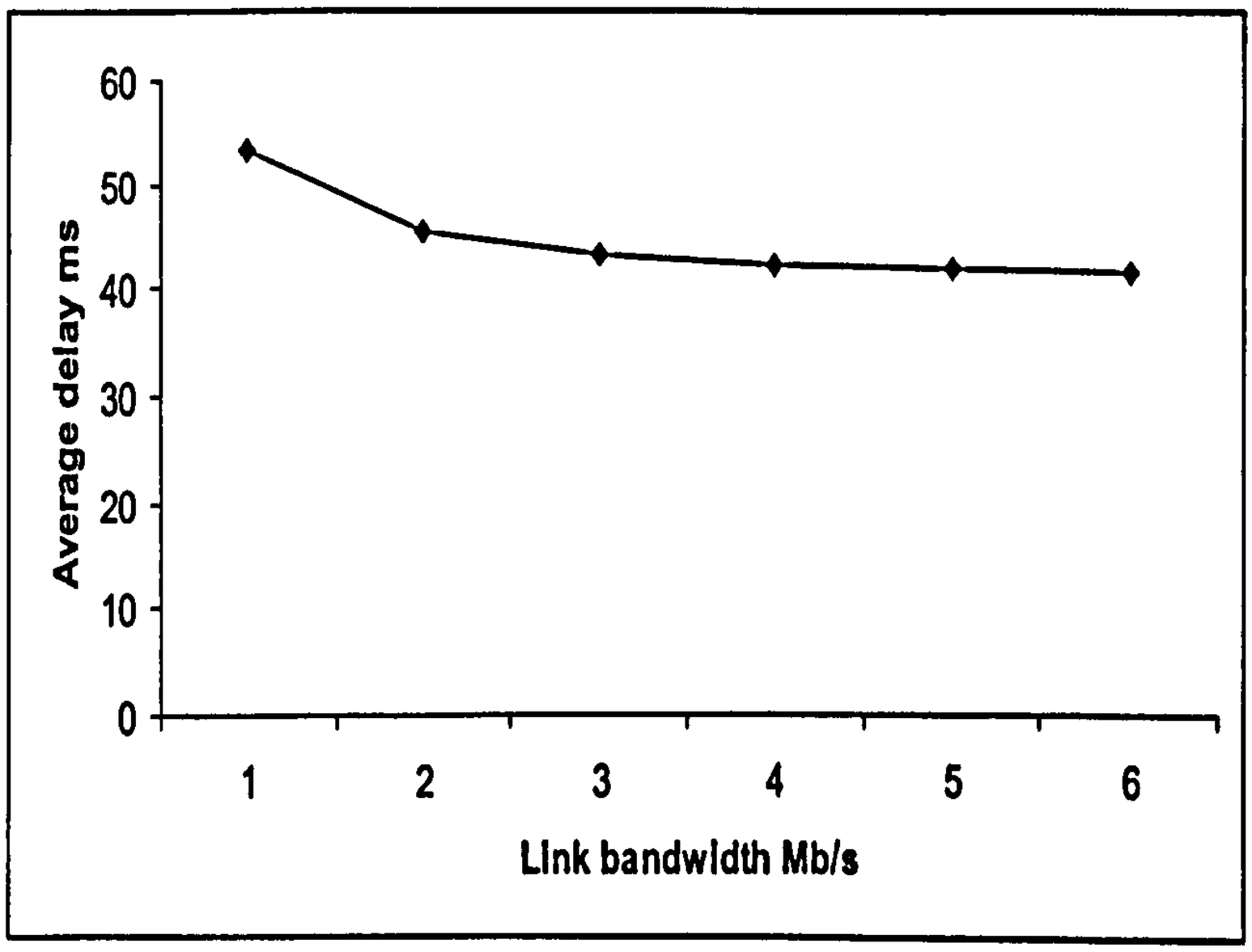


Figure 5.13: Impact of link bandwidth

5.7.3.6 Packet size affect

The size of the packet is important. As the packet size increases so does the delay. It is recommended that packet size should be kept to a minimum. In general the security protocols used have large packets; for example, the recent RSA encryption keys are 2048 bits[9]. In this section the impact of packet size is investigated. Figure 5.14 shows the performance of SNAP with different authentication packet sizes versus different peak rates. It is clear that the packet with a large size introduces more delay.

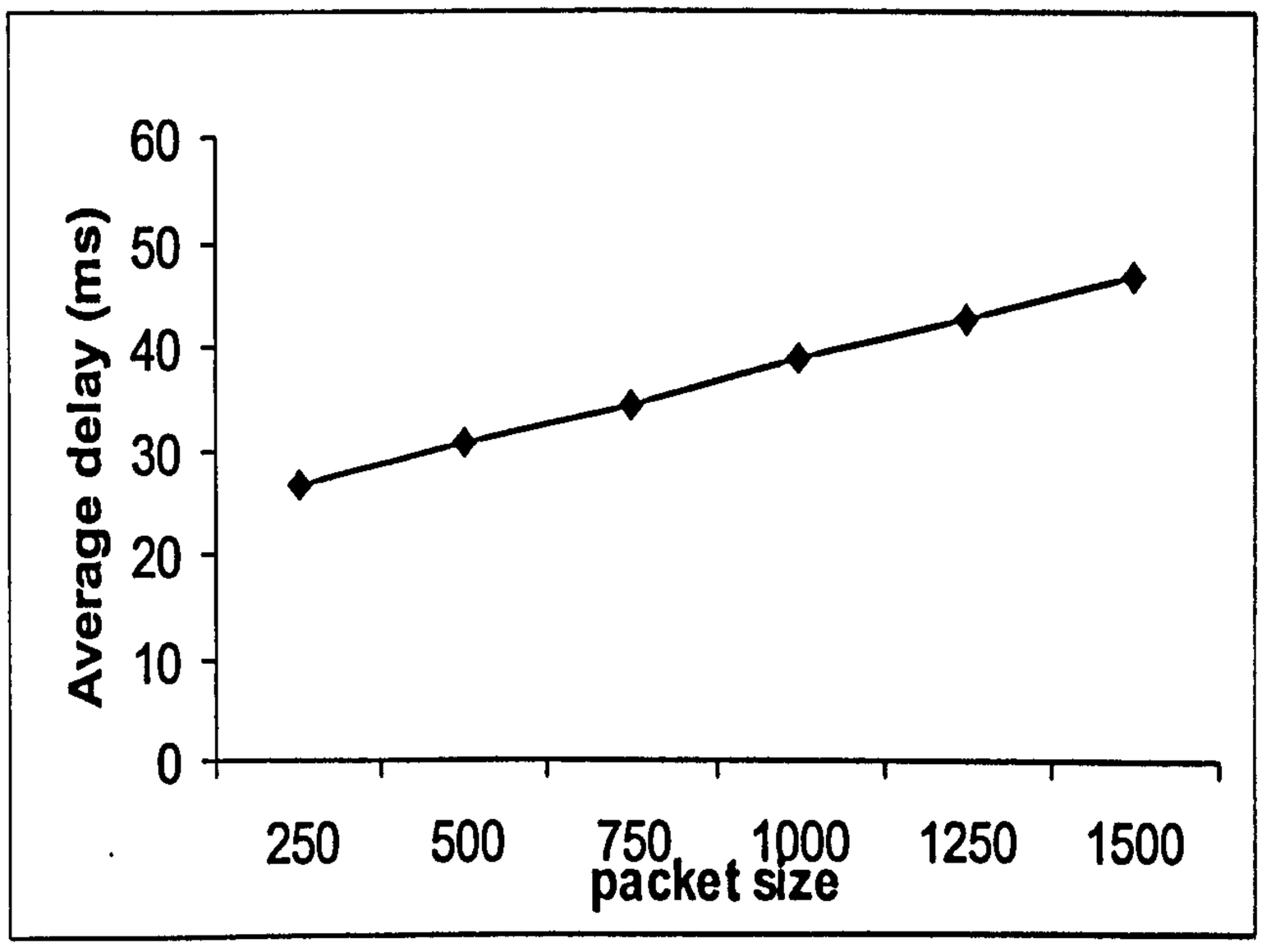


Figure 5.14: The impact of packet size

5.7.3.7 Quorum construction mapping

In the literature review all the quorum systems which are based on logical structures such as grid, CWlog, tree, etc deal with a system that has n nodes. In these studies the nodes are structured logically based on the given logical structure to form a quorum system. Since we are dealing with networks there are certain elements that must be considered. The major issue is the relation between the network nodes i.e the connectivity. It is very important to take this into consideration in order, to obtain an efficient quorum system. Mapping the logical quorum structure to the physical topology such that users contact local quorums is advantageous. The Tables 5.4 to 5.7 show four different mappings of a certain quorum system for the topology given in Figure 5.2 where $n = 12$. It shows the importance of selecting the best mapping of quorum system into physical topology. It is clear that mapping M2 performs better than all the other three mappings since in this mapping users contact local quorums.

Quorum	Members
Q1	1,2,5,11,6,10
Q2	2,3,4,8,9,20
Q3	0,4,5,6,7,8
Q4	0,1,3,7,9,11

Table 5.4: Mapping M1

Quorum	Members
Q1	0,2,3,4,5,6
Q2	5,6,7,8,9,10
Q3	1,3,4,8,9,11
Q4	0,1,2,7,10,11

Table 5.5: Mapping M2

Quorum	Members
Q1	3,4,5,6,8,9
Q2	0,1,5,6,7,11
Q3	1,2,3,9,10,11
Q4	0,2,4,7,8.10

Table 5.6: Mapping M3

Quorum	Members
Q1	2,3,5,6,9,10
Q2	0,2,4,7,8,10
Q3	1,3,4,8,9,11
Q4	0,1,5,6,7,11

Table 5.7: Mapping M4

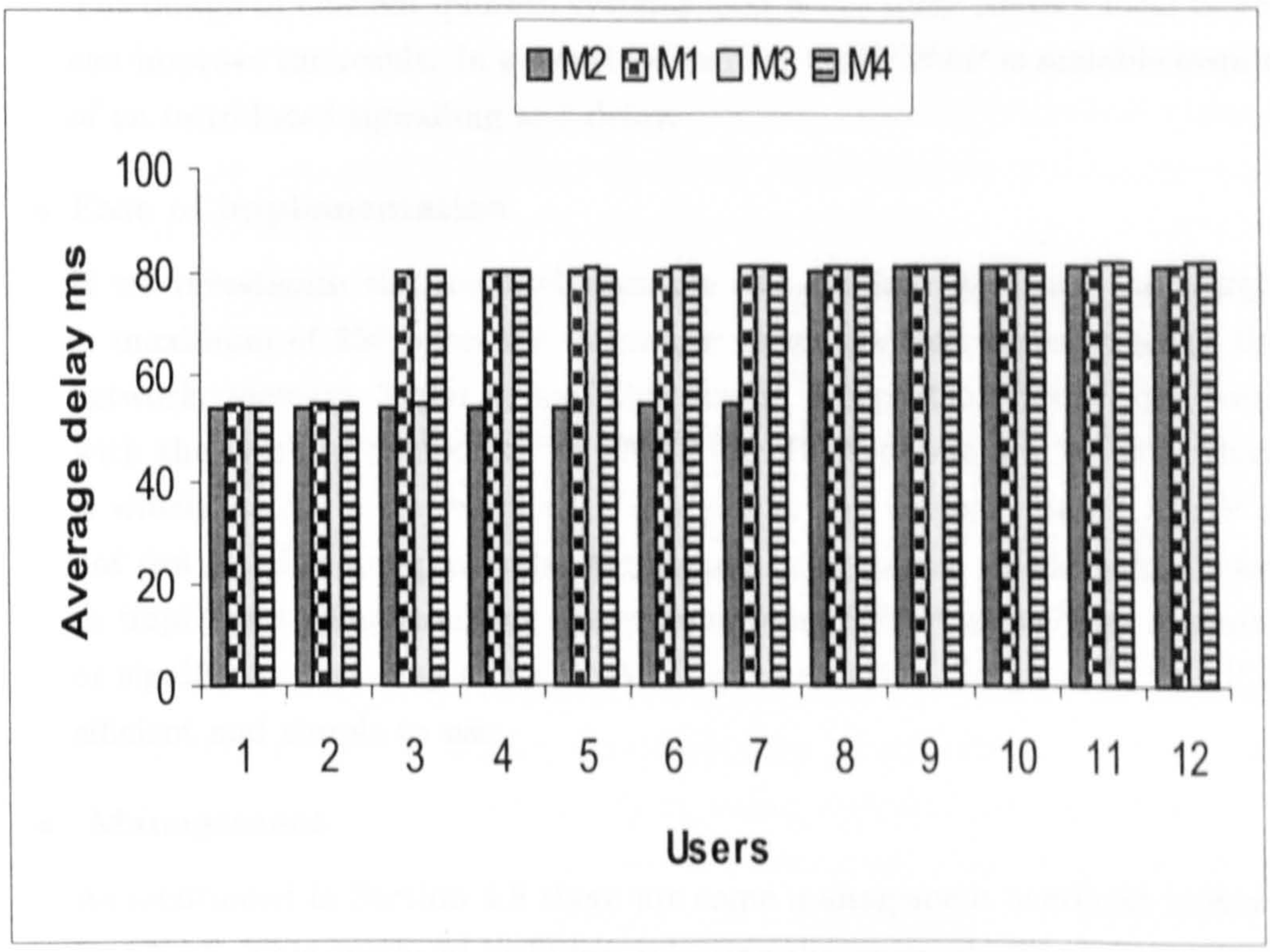


Figure 5.15: Impact of different mapping

5.7.4 Network Requirements

This section describes how SNAP meets the network requirements introduced in Chapter 2.

- **Scalability**

  In this regard there are two metrics to measure SNAP scalability: signalling overhead and delay. As mentioned in Section 4.4.2 SNAP generates k signals compared to centralized systems and the complexity is $O(N)$ which is linear and this means that SNAP is scalable. With respect to delay we can evaluate SNAP scalability by increasing the network size and see what will happen. The results in Section 5.7.3.3 show that increasing the network size leads to an expected increased delay. If we investigate the results in Figure 5.10 we can find that the increase in the delay is linear which is acceptable.

The design of efficient quorum systems that make users contact local nodes can improve the result. In general we can say that SNAP is scalable inspite of an introduced signalling and delay.

- **Ease of implementation**

  If we investigate the required message size we find it is not that large. A maximum of 256 bytes for the major messages that travel through the network, message 3 and 4, and this size is acceptable. SNAP can work with the existing protocols. In SNAP the RSA public key is used which is widely used. It can work with any public key infrastructure. SNAP is not restricted to any particular algorithms or protocols which make it easy to implement. The resulting delay from using SNAP is not that dilemma or significant that may affect SNAP implementation. In general, SNAP is efficient and simple to use.

- **Management**

  As mentioned in Section 4.8 there are some management overhead induced by SNAP. This overhead includes: selecting the secret sharing scheme, quorum system, quorum size, update and revoke mechanisms, etc. However, this management overhead is not affecting the end users and is handled by the NSM.

- **Reliability**

  SNAP can maintain its function inspite of the failure of some networks nodes. This feature is enhanced by using quorum systems since one of the major advantages of quorum systems is increasing the system reliability such that any quorum can work on behalf of the whole system. In SNAP, if one node or a quorum of nodes are down the user can contact another quorum and access the system. In addition, if one message got lost in SNAP the user or the sending party will not be harmed. He needs to wait for certain time (window) then retransmit the message again or contact another quorum.

- **Network Security**

  With respect to this issue, the security protocol can maintain the network secure, if one node is compromised the whole network is not compromised.

SNAP achieve this feature as it is come from the protocol nature which is distributed. In other words, if the attacker compromised one node he only will be able to get that node's share which will not affect the whole system. In order for the attacker to affect the whole network he needs to compromise at least all the members of one quorum which is difficult task. In addition SNAP can avoid or minimize the major network threats: denial of service and unauthorized access as described in chapter 4. Therefore, SNAP can increase the network security.

5.8 Conclusions

This chapter has investigated the performance of SNAP by considering some network parameters such as connectivity, network size, quorum size, etc. It has been shown that SNAP can introduce an extra signalling overhead but signalling effect can be neglected. However, this is the cost of having very secure systems. There are some cases where SNAP may perform better than a centralised approach if it is used to provide a more localized distributed authentication where the user can contact local quorums. Finally, it is necessary to study the performance of any security protocol before the protocol deployment.

Chapter 6

Mobile Distributed Authentication Protocol

6.1 Introduction

As an example of how SNAP could be implemented in a practical system, a mobile network has been considered. The demand on using wireless communication is rapidly increasing. The first generation of mobile networks provided communication using analog techniques. In the second generation (2G) digital communications were used, which provided not only voice service but also text messaging. In the third generation mobile networks (3G), enhanced services will emerge such as multimedia communication, web browsing, email and e-commerce. Already, users can access their bank accounts using their mobile handsets. This means that mobile networks are providing critical applications and the security is a paramount requirement. It is crucial to be sure that the user is authorized to access the network and the service.

Most existing authentication protocols (for example that used in the GSM mobile network) are centralized [92]. Depending on a single entity is undesirable as it has security, trust and availability issues. This chapter proposes an extension of SNAP for use in mobile networks. GSM networks are used as the example. Therefore, a new protocol, GSM-Secure Network Access Protocol (G-SNAP) is proposed. As with SNAP, in G-SNAP, the authentication procedure and the

PAGE NUMBERS CUT OFF

IN

ORIGINAL

network access control is handled by a quorum of authentication centers. This gives the advantages of SNAP avoiding the problem of trust being focussed on a single entity which becomes unconditionally trusted, resulting in increased risk and perhaps being unsuitable for the more complex business models in 3G systems

Section 6.2 introduces authentication in GSM networks. Section 6.3 describes the proposed protocol G-SNAP. Section 6.4 provides analysis and simulation results. A comparison between G-SNAP and centralized approaches is discussed in Section 6.5.

6.2 Authentication in GSM Networks

In GSM, authentication is achieved by checking the validity of a subscriber's SIM card. An authentication algorithm (termed A3) is stored on the SIM card and also in the authentication center (AuC) on the network. The process is challenge-response based. The A3 algorithm uses two input parameters: the secret key, K_i, which is stored in the SIM card and in the network, and a random number (RAND), which is transmitted to the mobile station as a challenge. The A3 algorithm uses K_i and RAND to calculate a response (SRES). The mobile station will send back the SRES to the network as a response to the challenge. The network uses the same RAND, K_i, and A3 to produce an SRES, which is checked against the response from the mobile station [40]. Figure 6.1 depicts the authentication on GSM networks.

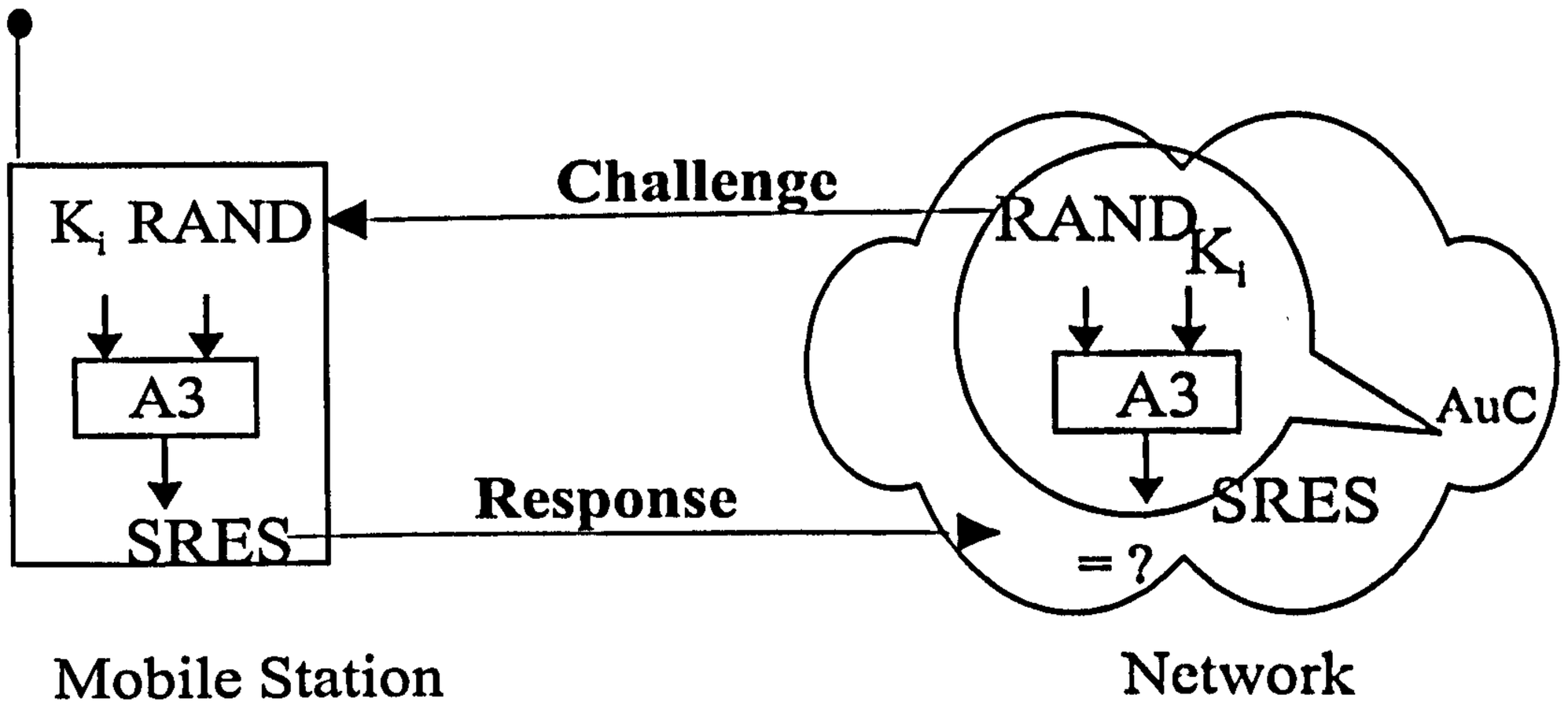


Figure 6.1: Authentication in GSM Networks

6.2.1 Drawbacks

- A challenge-response procedure such as GSM uses a strong authentication process to access network resources. The major drawback of the GSM authentication process comes from its centralized nature. The authentication center in the network controls the authentication, and as such, if the authentication center is attacked or compromised, then unauthorized users may obtain access to network resources.

- A centralized system also has availability issues. Authorized users will be unable to obtain access if that entity breaks down, denying service to users.

- A primary concern is one of trust. In GSM networks, the authentication center is unconditionally trusted. Such a centralized trust model is not recommended, as focussing the trust on a single entity will increase the risk, which might affect the system security.

- The communication between all users and the central authentication node involves a lot of signalling traffic and high load on the authentication node.

6.3 G-SNAP

G-SNAP is an extension of SNAP (described in Chapter 4). The proposed G-SNAP algorithm seeks to address GSM centralized authentication deficiencies by using a distributed approach. In order to perform authentication or access control in a GSM network using G-SNAP, the mobile station contacts a quorum of authentication centers with a quorum size k to obtain permission. Figure 6.2 depicts the mobile distributed model. In G-SNAP, the first (local) authentication server (local AuC) which receives the mobile station signal will start the authentication process. The local AuC participates in the negotiation but does not control nor take the network access decision on its own. Figure 6.3 depicts the required signalling for the user to obtain the required shares from a quorum of k AuC's. The G-SNAP works as follows:

1. The local AuC receives a signal from the MS (mobile station) to register or to use the network.

2. The local AuC will ask the MS to obtain permission or to be authenticated by a quorum of AuCs by providing k shares $(S_1, ..S_k)$ from any quorum.

3. The MS will start independently communicating with the AuCs, which are already known by the SIM card of that MS or given by the local AuC, and ask for the corresponding shares. If the MS does not receive a response from one AuC in a certain time it will try another quorum, up to l quorums.

4. Each AuC will check if the MS is authorized and then will send the corresponding share if appropriate

5. After receiving the required k shares by the SIM card of the MS, the MS will send the concatenated secret shares as one message to the local AuC.

6. The local AuC will reconstruct the AC using a reconstruction function. If it is correct, it will give the MS permission to access the network or alternatively output a rejection message.

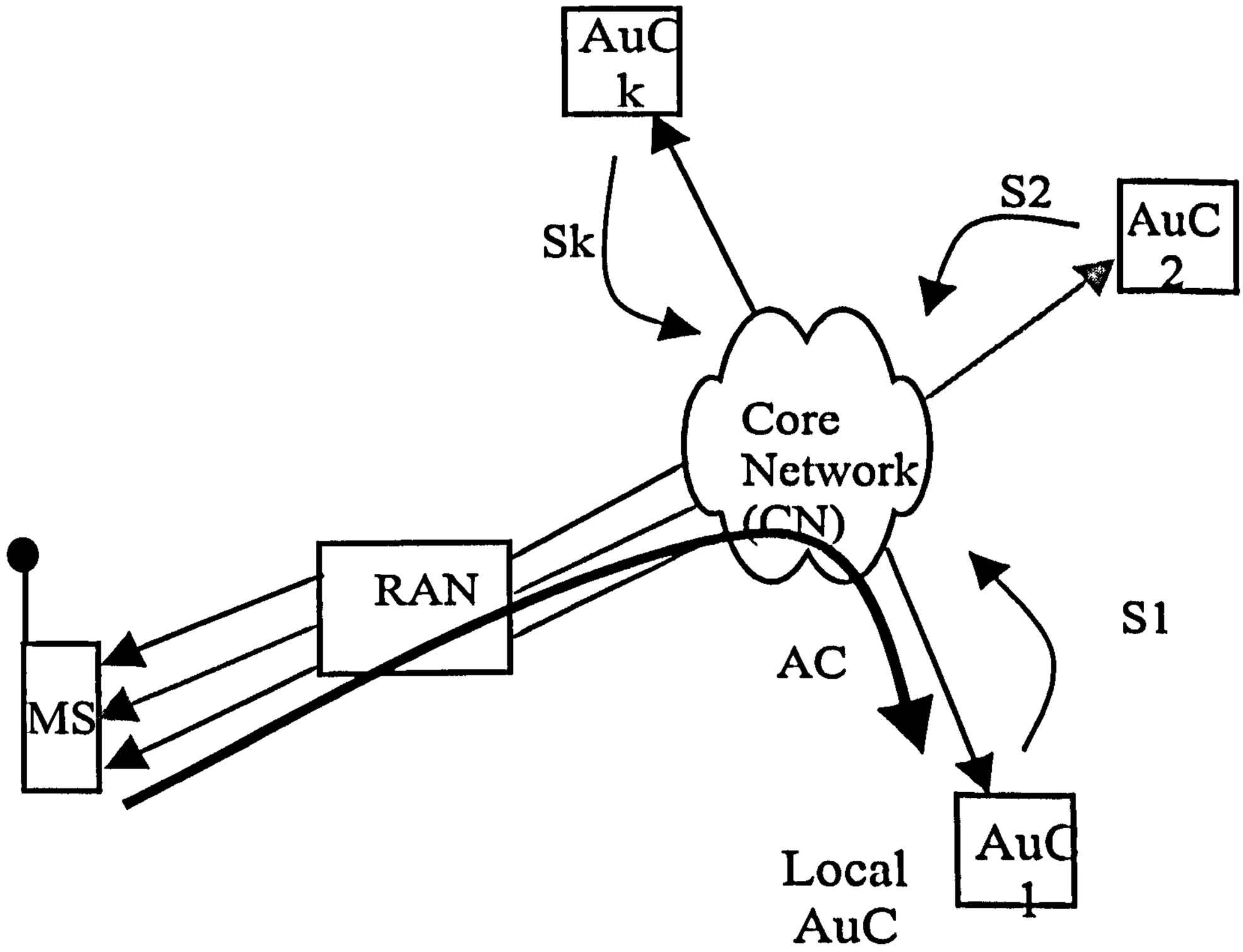


Figure 6.2: Distributed authentication model

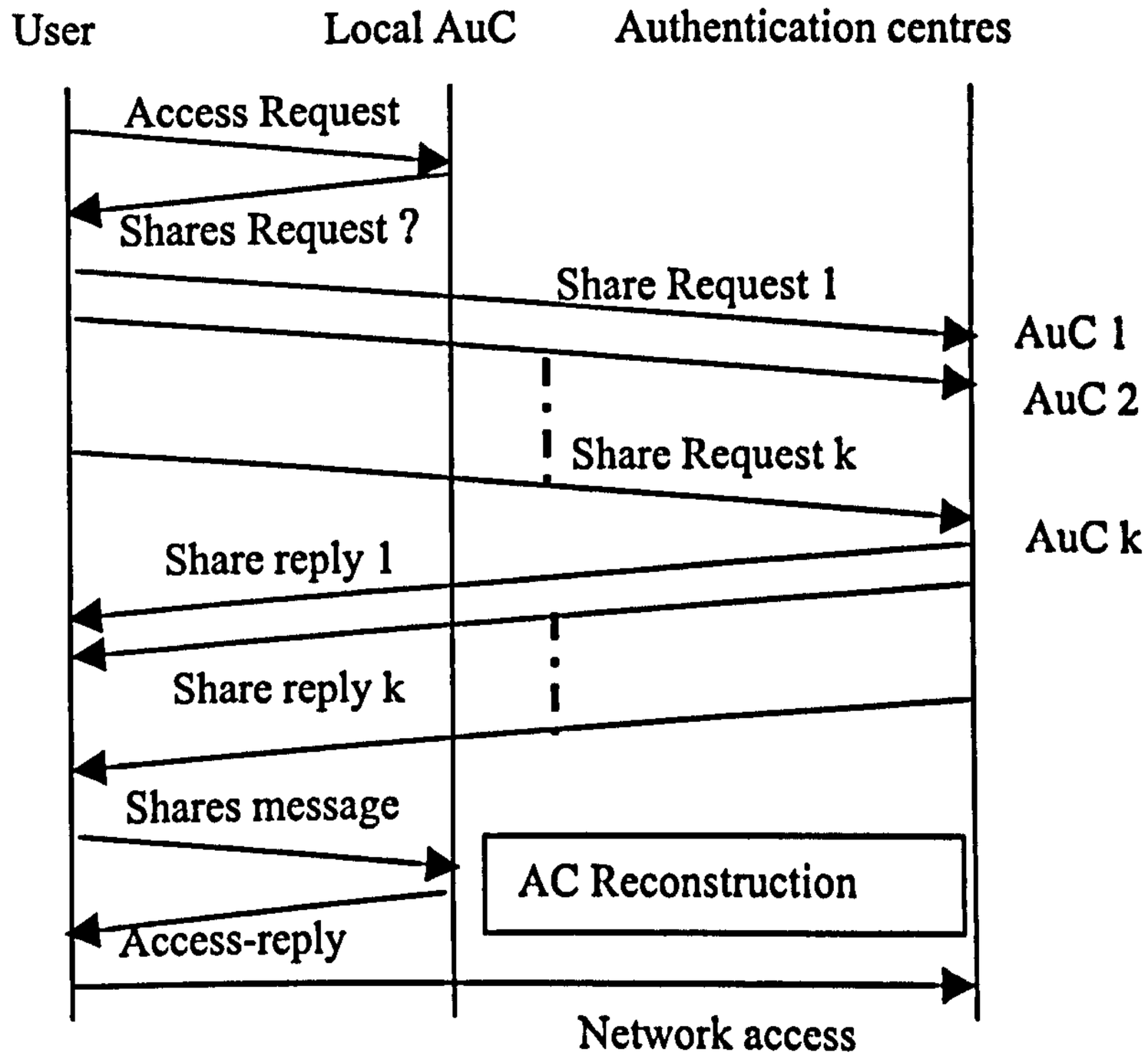


Figure 6.3: G-SNAP Signaling

6.4 Simulation analysis

A similar approach to that used for studying SNAP was applied to G-SNAP with a few distinctions to make the results more valid for GSM core network. The conducted simulation in this chapter was done using the network simulator version 2 (NS-2) [89]. In this simulation the background traffic packet size was reduced to 250 bytes to consider the bandwidth limitations on wireless networks. The signalling link is $1Mb/s$ and the link delay is 10 ms. In order to study the performance of G-SNAP, an arbitrary core network of 9 MSC nodes was used and depicted in Figure 6.5. The quorum system for the given topology is shown in Table 6.1. The quorum system is based on SuperGrid quorum system described in Chapter 4. As shown in the table the quorum size k is 4 nodes.

The simulation model involves two main steps: the network is loaded by gener-

ating background traffic to load all the links. After a certain time, whenever the network has been loaded, G-SNAP traffic will start generating the authentication packets.

In GSM networks the AuCs are connected to the mobile switching centers (MSCs). All the MSCs are in the core network and are *linked by a fixed* network. The G-SNAP architecture is illustrated *in Figure 6.4. Each node contacts its neighbor* using a full duplex link. In this simulation each node represents an authentication center in the MSC.

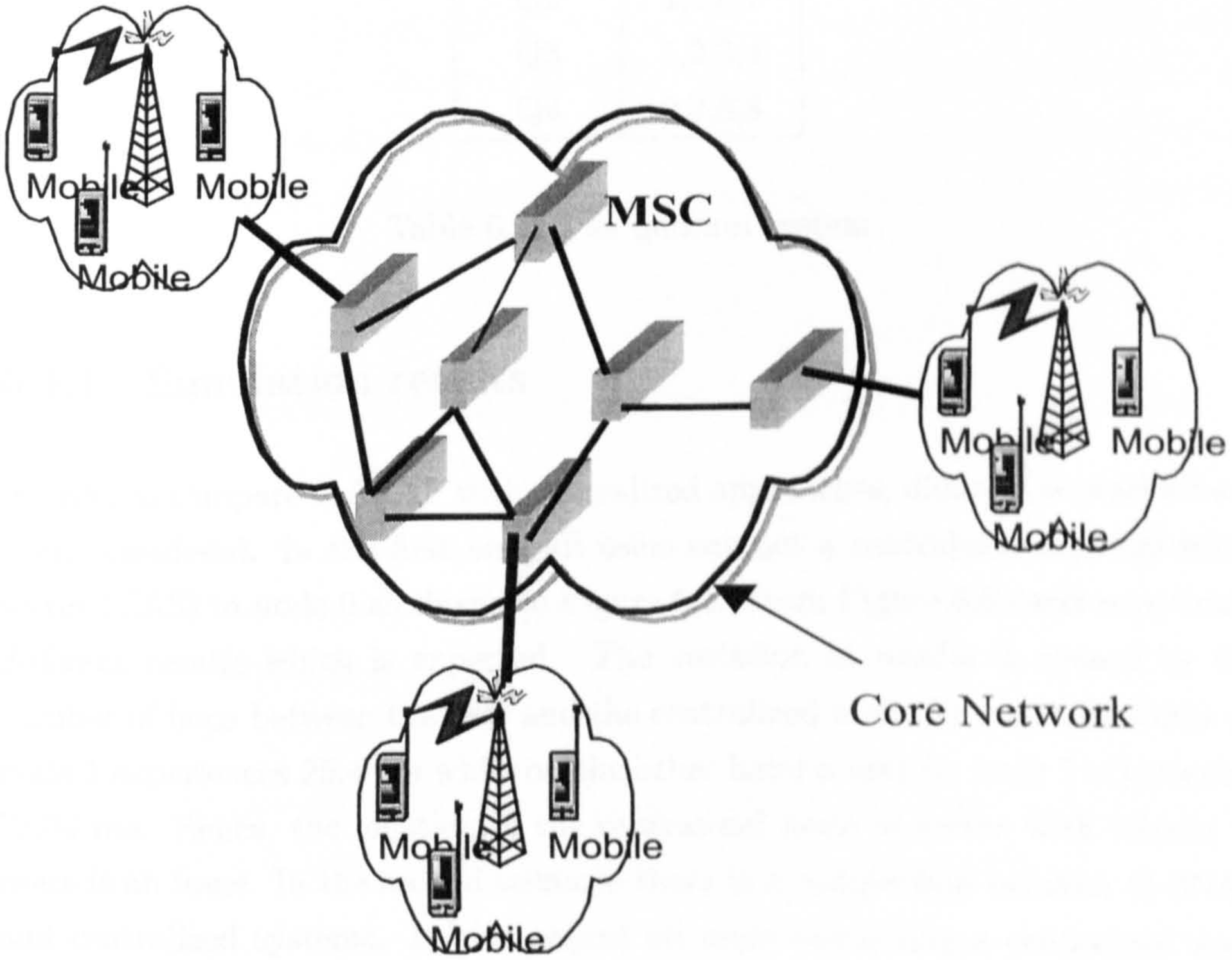


Figure 6.4: G-SNAP architecture

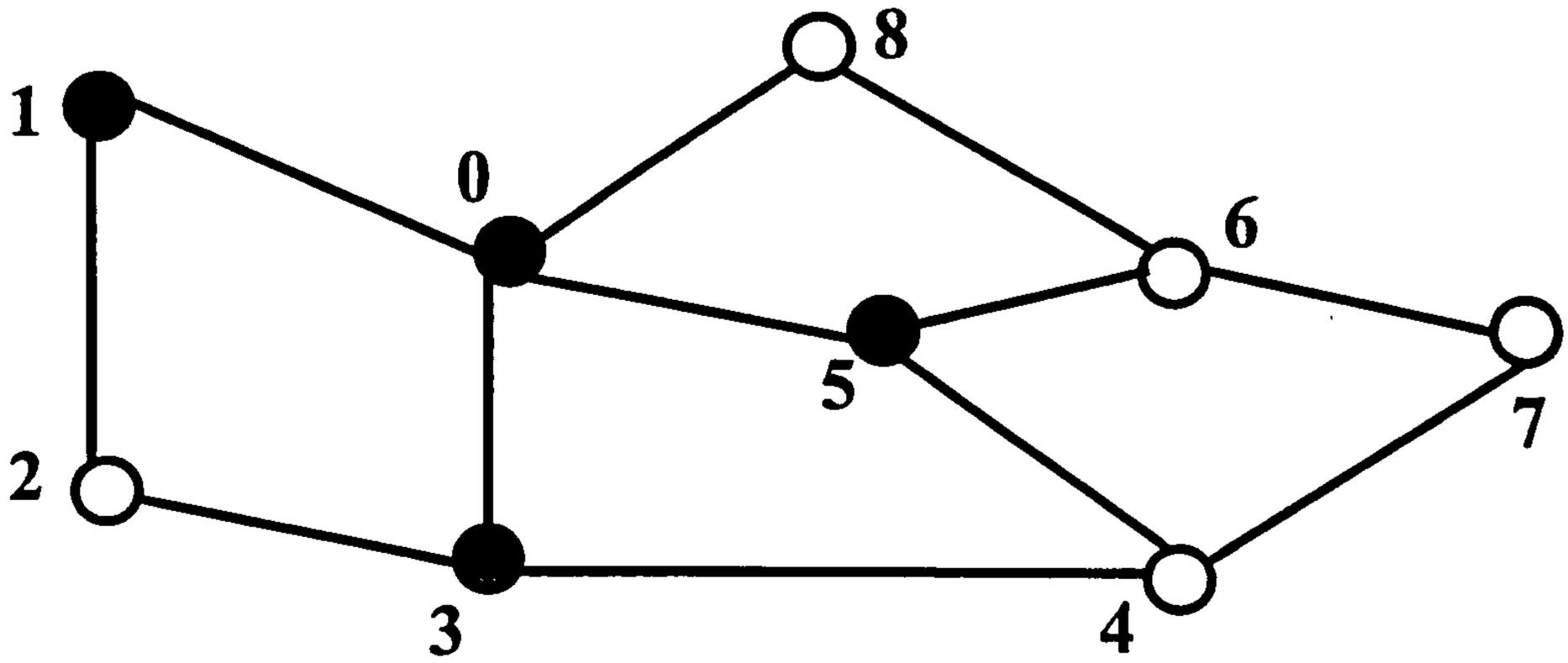


Figure 6.5: The core network topology and one quorum is shaded (Q1)

Quorum	Members
Q1	0,1,3,5
Q2	4,5,6,7
Q3	1,2,3,4
Q4	0,2,6,8

Table 6.1: The quorum system

6.4.1 Simulation results

In order to compare G-SNAP with centralized approaches, different scenarios have been considered. In the first one, all users contact a centralized authentication server (CAS) in node 0 as shown in Figure 6.6. From Figure 6.6 users experience different results which is expected. The variation in results is caused by the number of hops between the user and the centralized node, for example a user on node 1 experiences 25.4 ms while on the other hand a user on node 7 experiences 75.04 ms. Hence, the location of the centralized node or server with respect to users is an issue. In the second scenario there is a comparison between G-SNAP and centralized systems. In this regard all users contacting a centralized node (CAS 0) versus all users contacting one selected quorum Q1. From Table 6.1, Q1 has four member nodes $\{0, 1, 3, 5\}$ which means that each user needs to contact

all these nodes and wait for the last response. Figure 6.7 depicts the results and it is obvious that G-SNAP introduces more delay compared to a centralized approach. The difference in delay between the approaches is marginal. In the third scenario, a comparison between contacting local quorums such that each user contacts his local quorum versus all users contacting a selected randomly quorum (Q1) is conducted. Figure 6.8 shows the results. It is evident that contacting local quorums is better and the experience delay is less than contacting a single quorum. This shows the importance of distributing users to their local quorums. The last scenario, shows a comparison between three things, users contacting a centralized node (CAS 2), users using G-SNAP contacting Q1 and users using G-SNAP and contacting their local quorums (see Figure 6.9). It is clear that G-SNAP performs well if local quorums are used. Table 6.2 shows the average delay for the conducted scenarios. From the table contacting node 0 (CAS 0) is the lowest value than contacting local quorums. This mean that G-SNAP can perform better than centralized systems based on the location of the centralized node. Table 6.3 shows the average delay of using the centralized approach which represents the total average delay of contacting all nodes as centralized nodes compared to the delay caused by G-SNAP using local quorums. It evident that G-SNAP is not introducing a significant delay which may affect the protocol implementation. If we study these results we can come up with the following observations:

1. The delay increases if the number of hops between nodes is increased.

2. In most of the cases some users will experience exactly the same delay, either using a centralized authentication center or using G-SNAP (although traffic load will increase).

3. The location of the authentication node with respect to the user has a direct impact on results in both centralized and distributed approaches.

4. Designing an efficient quorum system and distributing the users on local quorums is crucial and can lead to good results.

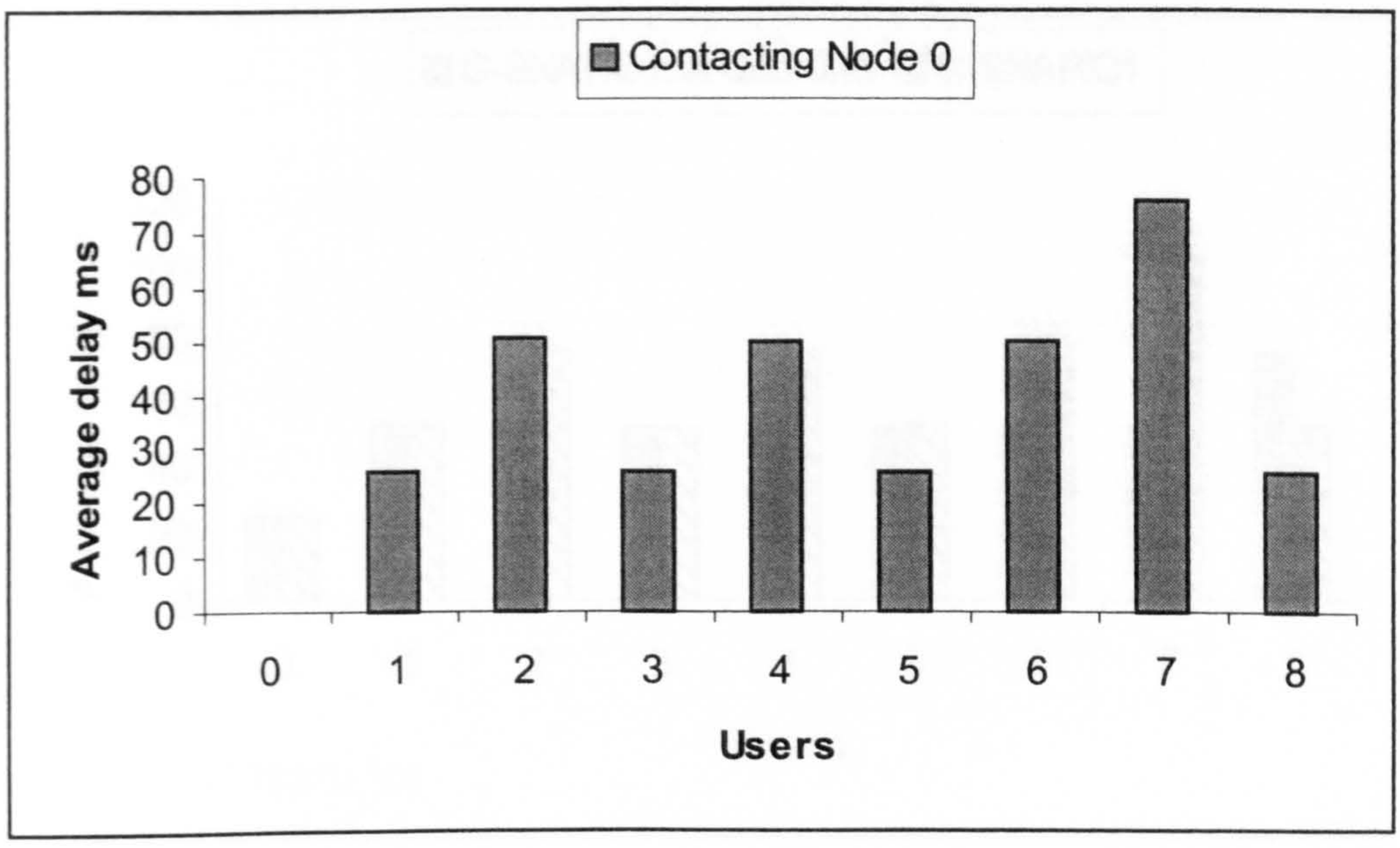


Figure 6.6: Users contacting centralized node (node 0)

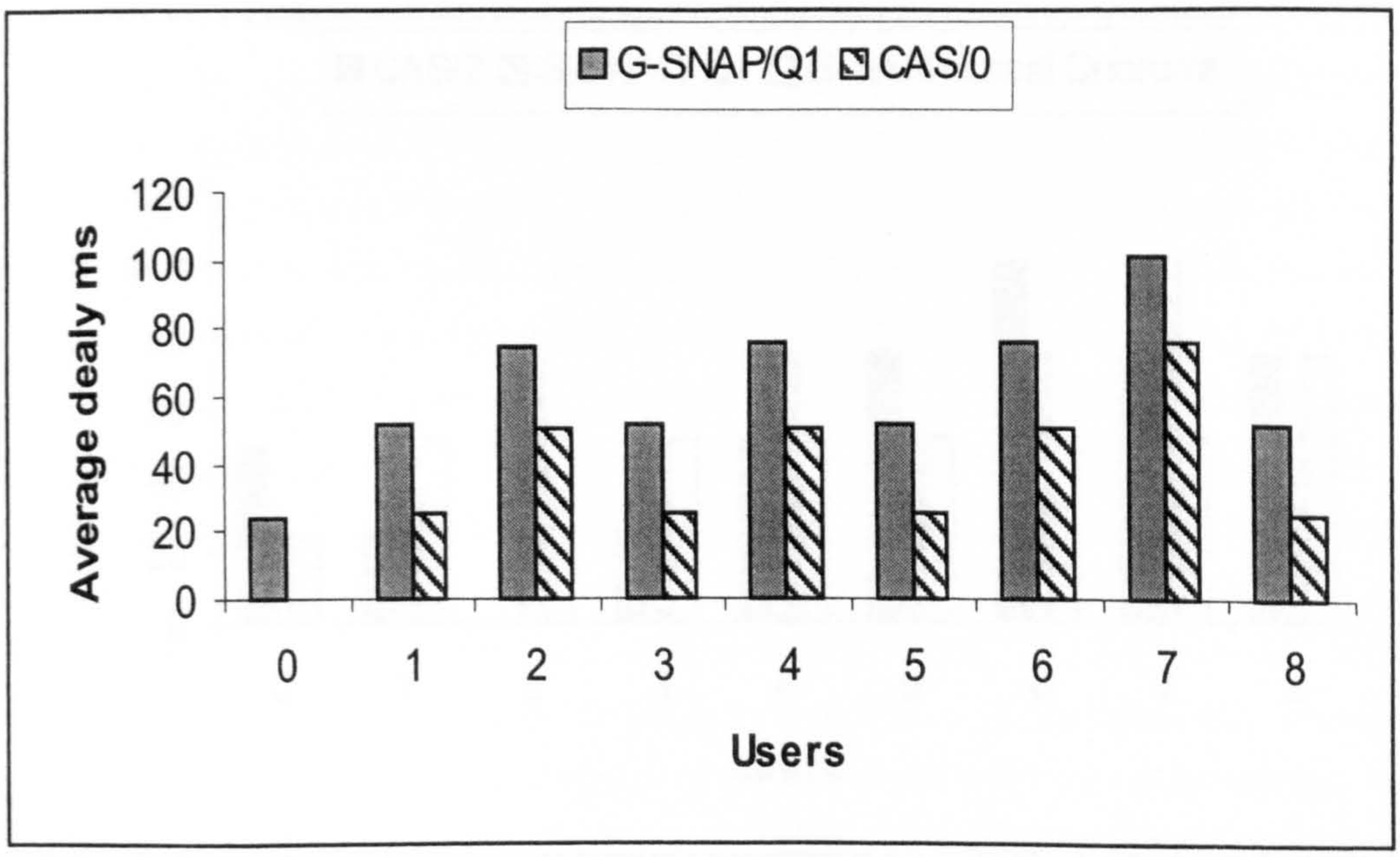


Figure 6.7: The CAS is node 0 V quorum 1

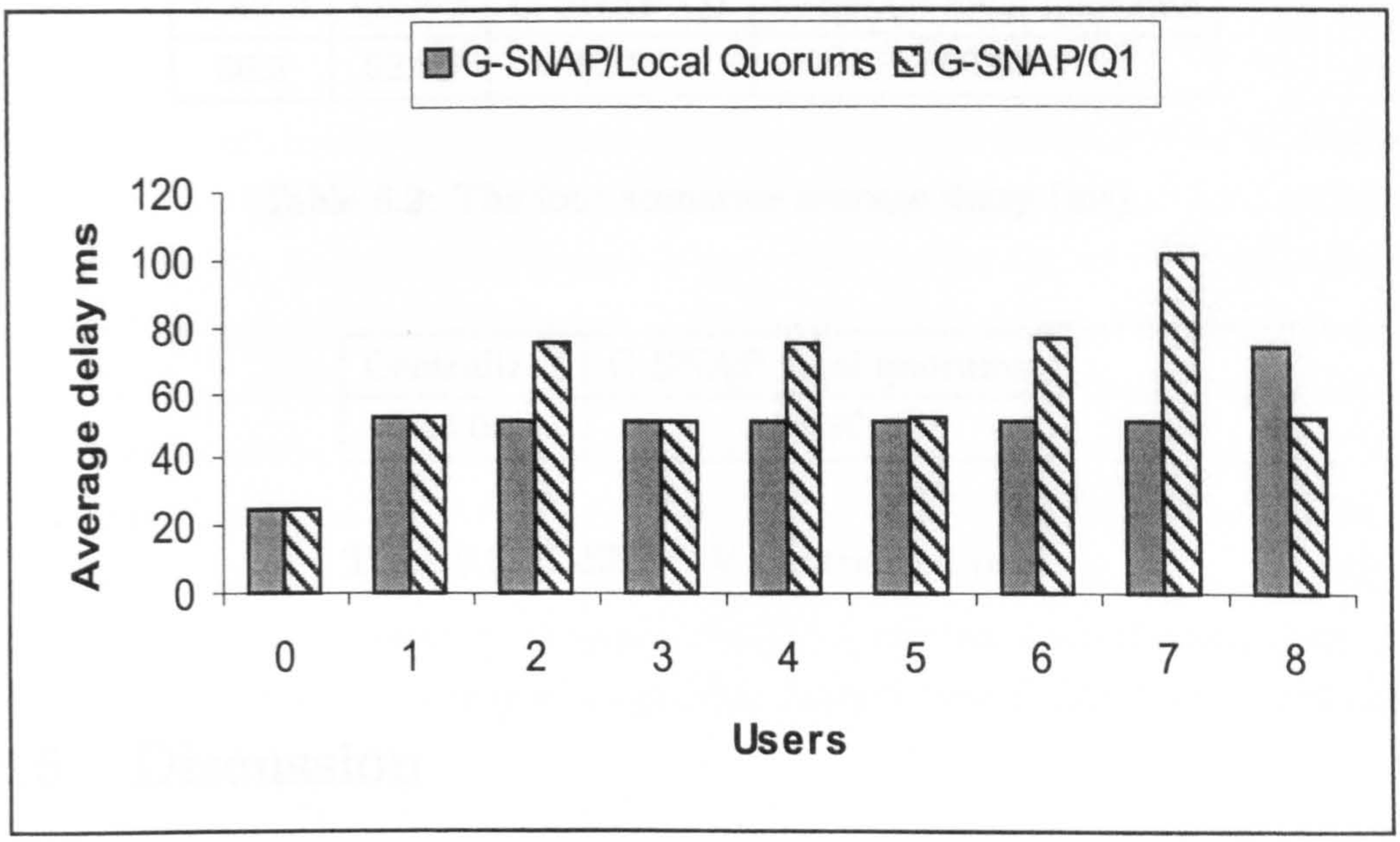


Figure 6.8: Quorum 1 V local quorums

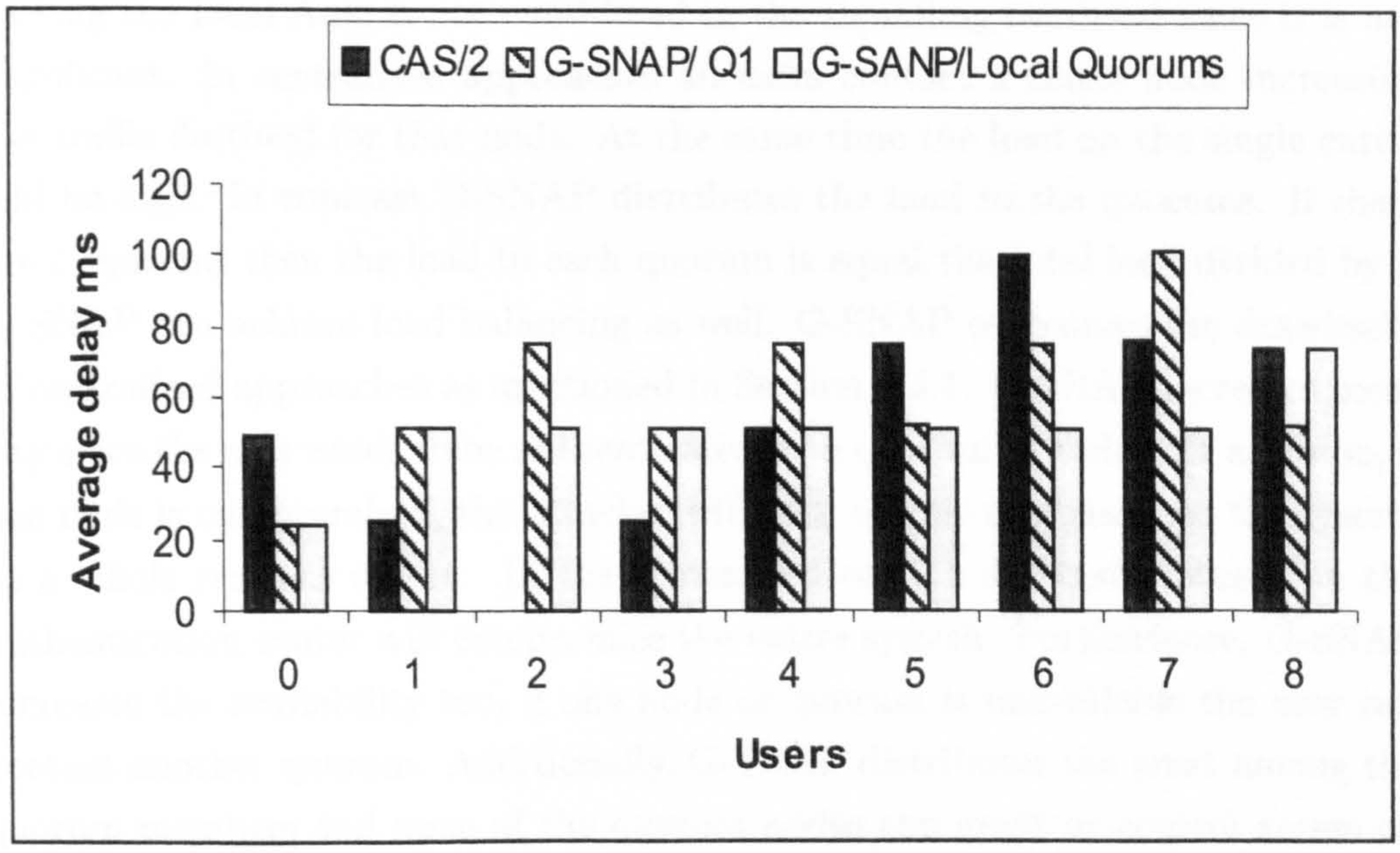


Figure 6.9: Quorum 1 and local quorums V CAS 2

CAS 0	CAS 2	G-SNAP Q1	G-SNAP local quorums
36.3	52.96	62.03	50.87

Table 6.2: The four scenarios average delay (ms)

Centralized	G-SNAP local quorums
45.02	50.87

Table 6.3: G-SNAP V Centralized (ms)

6.5 Discussion

In this discussion a comparison between G-SNAP and centralized approaches is investigated. In order to determine the signalling overhead caused by G-SNAP, the number of messages sent to the authentication nodes is required. In G-SNAP the user must contact k nodes to collect the required shares. Hence, signalling overhead is increased by a factor of k compared to centralized approaches. Contacting the local AuC is not considered in the signalling overhead since it is not significant. In centralized approaches all users contact a single node increasing the traffic destined for that node. At the same time the load on the single entity will be high. In contrast G-SNAP distributes the load to the quorums. If there are l quorums then the load to each quorum is equal the total load divided by l. G-SNAP can achieve load balancing as well. G-SNAP overcomes the drawbacks of centralized approaches as mentioned in Section 6.2.1. G-SNAP increases security since the user needs to be authenticated by a quorum of nodes. In addition, if one node is compromised, the attacker will only receive one share, so the system as a whole remains secure. In the centralized case, a successful attack on the authentication center will compromise the entire system. Furthermore, G-SNAP increases the availability too, if one node or quorum is unavailable the user can contact another quorum. Additionally, G-SNAP distributes the trust among the quorum members and none of the quorum nodes can grant or control access on its own. In high security applications the advantages of G-SNAP would appear to outweigh the additional signalling overhead.

6.6 Conclusion

There is a need to have multi-party authentication protocols such that more than one server controls the authentication process. This results in increased security and availability, and also distributed trust, which overcomes the centralized authentication approaches deficiencies. The importance of this will increase with 3G networks, which are more distributed in nature and have more complex business arrangements. G-SNAP is a protocol which achieves these objectives. The performance evaluation of G-SNAP compared to centralized approaches has been discussed in this chapter. Although G-SNAP does introduce more signalling, the delay impact on users is marginal. G-SNAP can be more efficient than the centralized approach if it is used to provide more localized distributed authentication.

Chapter 7

Conclusions

Security is important given the rapid increase in the size, connectivity, complexity and importance of networks. As the deployment of critical applications such as e-commerce, banking applications, electronic voting and so on increases, security has an ever more crucial role. Although networks and distributed systems provide several advantages including information sharing and deployment of critical applications, the opportunities for attack are increased as well.

Authentication is one of the major security mechanisms and as such, is required in most of the applications in order to restrict access to resources to only authorized users. Hence, authentication can minimise or prevent significant networks threats: DoS and unauthorized access. Given the importance of authentication, a number of different protocols have been proposed and implemented. However, these protocols typically suffer from certain common drawbacks, most resulting from their centralized nature. These drawbacks include security breach at single point of failure, reduced availability, trust focused on a single entity and very high load on centralized node. The main contribution of this thesis was to propose a novel distributed authentication protocol, SNAP.

SNAP is based on secret sharing schemes that posses a quorum access structure since it was found that secret sharing schemes and quorum systems leads to an increased security, increased availability and distributed trust. The key authentication properties have been introduced and it has been shown that SNAP emphatically satisfies these properties. It was also shown that relative to tradi-

tional centralized authentication protocols, SNAP increases the security, increases the availability and distributes the trust. Having shown that SNAP achieved the required objectives and can be considered an attractive alternative to centralized authentication protocols, the performance of an implementation of SNAP to evaluate the cost, to ensure that there are no implementation pitfalls was investigated. The performance metrics considered include the signalling overhead, load distribution and management overhead.

The study shows that SNAP introduces an increased signalling compared to centralized systems. The required signalling exceeds centralized schemes by a magnitude of k, where k is the quorum size. Therefore, the signalling overhead is dependent on the quorum size, suggesting smaller quorums should be used. With respect to load distribution, SNAP outperformed centralized systems since the load can be spread between quorums by distributing the users among the available quorums. SNAP introduces an increased management overhead to update and revoke shares and to maintain the system. However, the introduced management overhead affects only the NSM and not the end users.

The implementation of SNAP was investigated using certain parameters. These parameters included network connectivity, network size, packet size, quorum size, quorum mapping and link bandwidth. The delay was the measured parameter in all cases. The study shows the following: the connectivity between network nodes affects the relation (distance) between the quorum members. As the network connectivity increased, the delay decreased for both SNAP and centralized schemes as expected. A rise in network size, however, increased the delay linearly for SNAP and centralized schemes. The quorum size is a key parameter and if it is reduced then the delay reduced. The packet size effect is measured since most of the security protocols require large packet size. Increasing the packet size increases the delay. The link bandwidth result showed that increasing the link bandwidth decreases the delay. Quorum systems are usually organized based on certain logical structures, meaning the physical topology is not considered. Mapping a virtual quorum system to a given physical topology is a key issue since it was found that different mappings produce different results.

Another contribution of this thesis is the development of G-SNAP. G-SNAP is an extension of SNAP to mobile networks. G-SNAP is a distributed authenti-

cation protocol such that a quorum of authentication centers participate in the authentication process. G-SNAP was described and evaluated. It was found that G-SANP performed satisfactory, with negligible signalling delay with respect to centralized systems.

In closing, it has been shown that SNAP and G-SNAP are better suited to authentication requirements for future networks due to their design philosophy and underlying architecture. Encouragingly, this increased suitability incurs only a negligible penalty, suggesting SNAP and G-SNAP are very feasible, practical and desirable network security authentication protocols.

Chapter 8

Future work

In many ways, this thesis has raised as many questions as it has answered. Therefore, there are some interesting avenues for further work. Some of these are briefly discussed in the following sections.

8.1 SNAP Extension work

As described in the thesis SNAP is an authentication protocol that is based on secret sharing schemes that have quorum access structure. Therefore, it will be interesting to investigates the features of secret sharing schemes and quorum systems to design another security protocols. This means that SNAP concept can be generalized to develop the following protocols:

8.1.1 A Quorum based Key Management Protocol

Key management is one of the important issues in security. It is represent a corner stone in most of security applications and mechanisms. These include encryption, authentication, digital signature, non-repudiation, exchange of session keys etc. For example, to encrypt a message the user needs to obtain the other party encryption key, to sign a message he needs to have the other party decryption key and so on. In most cases, a trusted third party is involved in key manage-

ment protocol. However as mentioned in Section 2.7.1, this model has certain drawbacks. Therefore, the potential application of SNAP to key management is an attractive prospect. For example, each user contact a quorum of servers to obtain the required parts of the second party key and using the quorum access structure the user will be able to reconstruct the required secret key. In this case the security, availability are increased and the trust is distributed.

8.1.2 A Quorum based Signature protocol

In this regard, the message is signed by a quorum of users or servers such that each member partially participates in the signature process. Obtaining a qualified set of these partial signatures leads to the whole message signed by all members. An example of this is the signing of a large cheques which is signed by more than one qualified person. The quorum based signature protocol will be more beneficial in critical financial applications. In addition it can be used in critical decision where the decision letter or the required message is required to be signed by several authorized people.

8.1.3 SNAP Analytical Model

The performance of SNAP was measured using simulation. It is difficult to have an analytical model for SNAP in mesh networks since there are different parameters that are involved in the study. However, the study of a SNAP analytical model in a small ring topology would be advantagous.

8.1.4 SNAP Further Results

The study of SNAP was implemented using certain topologies. Therefore it would be interesting to study the performance of SNAP on certain real life topologies. In addition, in the obtained results the link bandwidth and the link delay for all links were uniform. Running the simulation with different link bandwidth and link delay is also of interest.

8.1.5 Quorum Systems Issues

Since SNAP is based on quorum systems two examples of quorum constructions were used in the study. The performance of SNAP using several quorum constructions may lead to new findings. In addition, the conducted result on quorum mapping was considered one construction. Further investigation of quorum mapping using several quorum construction is required.

8.1.6 G-SNAP

In the thesis, G-SNAP is proposed for GSM networks, therefore it is important to see how to implement G-SNAP in UMTS networks. In addition, the conducted results for G-SNAP considered the core network (CN) part in GSM networks. It would then be of interest to investigate the performance of G-SNAP in wireless networks by considering the radio link characteristics part in the simulation. This will leads to have holistic picture of G-SNAP performance.

Bibliography

[1] N. Ltd, How Many Online, http://www.nua.ie/surveys/how_many_online, May 2003.

[2] E. Rabinovitch, The Neverending Saga of Internet Security: Why? How? And What To Do Next, *IEEE Comm. Mag*, May 2001.

[3] C. Douligeris and D. Wheeler, Telecommunication Networks Security, *IEEE Communication Magazine*, pp. 40–41, October 2002.

[4] J. Steiner, B. C. Neuman, and J. L. Schiller, An authentication service for open networks, *In Proc. Winter USENIX Conference, Dallas, TX, USA*, 1988.

[5] R. M. Needham and M. D. Schorder, Using encryption for authentication in large networks of computers, *Communication of the ACM*, vol. 21, no. 12, pp. 993–999, December 1978.

[6] A. Levi and M. U. Caglayan, The Problem of Trusted Third Party in Authentication and Digital Signature Protocols, *The Twelfth International Symposium on Computer and Information Sciences, 27 - 29 October, Antalya, Turkiye*, 1997.

[7] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC, 1996.

[8] R. Oppliger, *Internet and Intranet Security*, Artech House , London, 1998.

[9] B. Schneier, *Applied Cryptography*, 2nd edition, Katherine Schowalter publish, 1996.

[10] M. Rozenblit, *Security for Telecommunications Network Management*, Wiley-IEEE Press, march 2001.

[11] W. Stallings, *Network Security Essentials: Applications and Standards*, Prentice-Hall, 2000.

[12] D. Khan, *The Code Breakers*, Sphere Books Limited, London, 1968.

[13] R. Rivest, A. Shamir, and L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, *Communication of the ACM*, vol. 21, pp. 120–126, 1978.

[14] V. hassler, X0509 and LDAP Security: A Comparative Overview, *IEEE Network*, Nov./Dec. 1999.

[15] Cisco, Security Technologies, http://www.cisco.com/univercd/cc/td/doc/cisintwk /ito_doc/security.htm, 2002.

[16] M. Curtain, Introduction to network security, http://www.interhack.net/pubs/network-security, March 1997.

[17] Institute of Electrical and Electronics Engineers, IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries, New York, NY, http://www.sei.cmu.edu/str/indexes/glossary/reliability.html, 1990.

[18] Appgate, Trusted Networks White Paper, www.appgate.com, April 2001.

[19] Attrition, Network Security White Paper, http://www.attrition.org/m̃odify/texts/security/whtpap2.html, 2001.

[20] C. Yeun, *Design,analysis and application of cryptographic techniques*, Ph.D. thesis, Royal Holloway University of London, 2000.

[21] R. Housley, W. Ford, and D. Solo, Internet X.509 public key infrastructure-Certificate and CRL profile, RFC 2459, January 1999.

[22] C. Adamsand and S. Farrell, Internet X.509 public key infrastructure- Certificate management protocols, RFC 2510, March 1999.

[23] M. Branchaud, *A survey of public-key infrastructures*, Master's thesis, McGill University, Montreal, March 1997.

[24] C. Mitchell, PKI standards, Tech. rep., Royal Holloway, University of London, November 2000.

[25] U. Blumenthal, Key Derivation for Network Management Applications, *IEEE Network*, May/June 1997.

[26] L. Vigano', Security Protocols, Tech. rep., Institute for Information, Albert-Ludwigs-University Freiburg, November 2001.

[27] M. Burrows, M. Abadi, and R. Needham, A Logic of Authentication, from Proceedings of the Royal Society, Volume 426, Number 1871, 1989.

[28] P. Lin and L. Lin, Security in Enterprise Networking: A Quick Tour, *IEEE Comm. Mag.*, pp. 56–61, January 1996.

[29] S. Bellovin and M. Merritt, Cryptographic Protocol for Secure Communications, *U.S. Patent # 5,241,599*, 1993.

[30] ITU recommendation X.509, Information technology-Open Systems Interconnection- The Directory: Public-key and attribute certificate frameworks, 2000.

[31] R. Perlman, An Overview of PKI Trust Models, *IEEE Network*, Nov./Dec. 1999.

[32] P. Wing and B. O'Higgins, Using Public-Key Infrastructure for Security and Risk Management, *IEEE Comm. Mag*, pp. 71–73, Septemeber 1999.

[33] S. Kent, Privacy enhancement for Internet electronic mail, Part II: certificate-based key management, RFC 1422, February 1993.

[34] D. Chadwick, Merging and Extending the PGP and PEM Trust Models-The ICF-TFI Trust Model, *IEEE Network*, May/June 1997.

[35] S. Boeyen, T. Howes, and P. Richard, Internet X.509 public key infrastructure- operational protocols, RFC 2559, April 1999.

[36] T. Ylonen, T. Kivinen, M. Saarinen, T. Rinne, and S. Lehtinen, SSH protocol architecture, Internet draft-draft-ietf-secsh-architecture-11.txt, November 2001.

[37] SSH Communications Security Corp., Secure Shell, www.ssh.fi/support/documentation/online/ssh.

[38] T. Ylonen, T. Kivinen, M. Saarinen, T. Rinne, and S. Lehtinen, SSH Authentication protocol, Internet draft-draft-ietf-secsh-architecture-13.txt, November 2001.

[39] SSH Communications Security Corp., SSH: Authentication, www.ssh.com.

[40] J. Dunlop, D. Girma, and J. Irvine, *Digital mobile communications and the TETRA system*, John Wiley & Sons, 1999.

[41] D. R. Stinson, An explication of secret sharing schemes, *Designs, Codes and Cryptography*, vol. 2, pp. 357–390, 1992.

[42] G. J. Simmons, An Introduction to Shared Secret and/or Shared Control Schemes and their Application, *in Contemporary Cryptology: The Science of Information Integrity, IEEE Press*, pp. 441–97, 1991.

[43] M. van Dijk, On the Information Rate of Perfect Secret Sharing Schemes, *reprint, IEEE*, pp. 489–489, 1994.

[44] A. Shamir, How to share a secret, *ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[45] G. Blakley, Safeguarding Cryptographic Keys, *AFIPS Conference Proceedings*, vol. 48, pp. 313–317, 1979.

[46] M. Ito, A. Saito, and T. Nishizeki, Secret Sharing Scheme Realizing General Access Structure, *J. Cryptology*, vol. 6, pp. 15–20, 1993.

[47] J. Benaloh and J. Leichter, Generalized secret sharing and monotone functions, *Advances in Cryptology, CRYPTO '88 Lecture Notes in Computer Science, Springer-Verlag*, vol. 403, pp. 27–35, 1990.

[48] R. Capocelli, A. D. Santis, L. Gargano, and U. Vaccaro, On the size of shares for secret sharing schemes, *in Advances in Cryptology-Crypto'91 Proceedings, Lecture Notes in Computer Science, Springer-Verlag, Berlin*, pp. 101–113, 1991.

[49] C. Blundo, A. D. Santis, L. Gargano, and U. Vaccaro, On the Information Rate of Secret Sharing Schemes, *Lecture Notes in Computer Science*, vol. 740, pp. 148–167, 1993.

[50] A. Beimel and B. Chor, Universally Ideal Secret Sharing Schemes, in E. F. Brickell, (ed.) *Proc. CRYPTO 92*, pp. 183–195, Springer-Verlag, 1992.

[51] H. Krawczyk, Secret Sharing Made Short, *In Proc. of Crypto*, pp. 136–146, 1993.

[52] C. Blundo and D. R. Stinson, Anonymous Secret Sharing Schemes, *Discrete Applied Mathematics*, vol. 77, pp. 13–28, 1997.

[53] C. Blundo, A. Cresti, A. D. Santis, and U. Vaccaro, Fully Dynamic Secret Sharing Schemes, *Lecture Notes in Computer Science*, vol. 773, 1994.

[54] K. Kulesza and Z. Kotulski, On secret sharing schemes with extended capabilities, www.ippt.gov.pl/žkotulsk/zegrze2002.pdf, 2002.

[55] S. Shieh and H. Sun, On constructing secret sharing schemes, *in Proceedings of the 1994 IEEE International Conference on Computer Communications, Networking for Global Communications (INFOCOM'94)*, pp. 1288–1292, 1994.

[56] Lee, Yeh, Chen, and Ku, A Share Assignment Method to Maximize the Probability of Secret Sharing Reconstruction under the Internet, *IEICE, TRANS. INF. & SYST.*, vol. E83-D, no. 2, pp. 190–199, February 2000.

[57] G. J. Simmons, W. Jackson, and K. Martin, The Geometry of Shared Secret Schemes, *Bulletin of the ICA*, vol. 1, pp. 71–88, 1991.

[58] C. Blundo, A. D. Santis, D. R. Stinson, and U. Vaccaro, Graph Decompositions and Secret Sharing Schemes, *Lecture Notes in Computer Science*, vol. 658, 1993.

[59] H.-M. Sun and S.-P. Shieh, Secret Sharing in Graph-Based Prohibited Structures, in *INFOCOM (2)*, pp. 718–724, 1997.

[60] M. Naor and A. Wool, Access Control and Signatures via Quorum Secret Sharing, *IEEE Transactions on Parallel and Distributed Systems*, vol. 9, no. 9, 1998.

[61] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch, Verifiable secret sharing and achieving simultaneity in the presence of faults, *In 26th IEEE Symposium on Foundations of Computer Science*, pp. 282–395, 1985.

[62] M. A. Stadler, Publicly Verifiable Secret Sharing, *Lecture Notes in Computer Science*, vol. 1070, 1996.

[63] B. Schoenmakers, A Simple Publicly Verifiable Secret Sharing Scheme and Its Application to Electronic Voting, *Lecture Notes in Computer Science*, vol. 1666, pp. 148–164, 1999.

[64] Y. Desmedt and Y. Frankel, Threshold cryptosystems, *In Proc. CRYPTO '89, volume 435 of LNCS ,Springer-Verlag*, vol. 435, pp. 307–315, 1989.

[65] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, Robust Threshold DSS Signatures, *Lecture Notes in Computer Science*, vol. 1070, 1996.

[66] M. Reiter and K. P. Birman, How to securely replicate services, *ACM Transactions on Programming Languages and Systems*, vol. 16, no. 3, pp. 986–1009, May 1994.

[67] M. Naor and A. Wool, The Load, Capacity, and Availability of Quorum Systems, *SIAM Journal on Computing*, vol. 27, no. 2, pp. 423–447, 1998.

[68] L. Gong, Increasing Availability and Security of an Authentication Service, *IEEE Journal on Selected Areas in Communications*, vol. 11, no. 5, pp. 657–662, 1993.

[69] R. Cramer, I. Damgard, and U. Maurer, General secure multi-party computation from any linear secret sharing scheme, *http : //citeseer.nj.nec.com/cramer99general.html*, 1999.

[70] A. Kumar, Hierarchical Quorum Consensus: A New Algorithm for Managing Replicated Data, *IEEE Transactions on Computers*, vol. 40, no. 9, pp. 969 – 1003, September 1991.

[71] S. Y. Cheung, M. H. Ammar, and M. Ahamad, The Grid Protocol: A High Performance Scheme for Maintaining Replicated Data, *Knowledge and Data Engineering*, vol. 4, no. 6, pp. 582–592, 1992.

[72] A. Kumar, An Efficient SuperGrid Protocol for High Availability and Load Balancing, *IEEE Transactions on Computers*, vol. 49, no. 10, pp. 1126 – 1133, October 2000.

[73] D. Agrawal and A. El-Abbadi, An efficient and fault-tolerant solution for distributed mutual exclusion, *ACM Transactions on Computer Systems*, February 1991.

[74] D. Gifford, Weighted voting for replicated data, *Proceedings of 7th ACM Symposium on Operating System Principles*, pp. 150–162, 1979.

[75] H. Garcia-Molina and D. Barbara, How to assign votes in a distributed system, *Journal of the ACM*, vol. 32, no. 4, pp. 841–860, october 1985.

[76] M. Maekawa, A $\sqrt{N}$ algorithm for mutual exclusion in decentralized systems, *ACM Transactions on Computer Systems*, vol. 3, no. 2, pp. 145–159, May 1985.

[77] D. Beaver and A. Wool, Quorum-based secure multiparty computation, *In Proc. of EUROCRYPT'98, LNCS 1403, Springer Verlag*, pp. 375–390, 1998.

[78] D. Peleg and A. Wool, Crumbling walls: A class of practical and efficient quorum systems, *Distributed Computing*, vol. 10, no. 2, pp. 8798, 1997, 1997.

[79] Z. Haas and B. Liang, Ad hoc mobility management with uniform quorum systems, *IEEE/ACM Trans. on Networking*, vol. 7, no. 2, pp. 228–240, April 1999.

[80] M. Neilsen, *Quorum Structures in distributed systems*, Ph.D. thesis, Dept. Computing and Information Sciences, Kansas State University, 1992.

[81] M. Liu, D. Agrawal, and A. Abbadi, the implementation of the quorum concensus protocol, 1995.

[82] M. Herlihy and J. D. Tygar, How to Make Replicated Data Secure, in *CRYPTO*, pp. 379–391, 1987.

[83] Y. Amir and A. Wool, Evaluating Quorum Systems over the Internet, in *Symposium on Fault-Tolerant Computing*, pp. 26–35, 1996.

[84] L. Zhou, F. Schneider, and R. van Renesse, COCA: A Secure Distributed On-line Certification Authority.

[85] D. Malkhi and M. Reiter, Byzantine quorum systems, pp. 569–578, 1997.

[86] L. Zhou, F. Schneider, and R. van Renesse, APSS: Proactive secret sharing in asynchronous systems.

[87] B. Bhansali, Man in the middle attack, http://www.giac.org/practical/gsec/Bhavin_Bhansali_GSEC.pdf, 2001.

[88] A. Tanenbaum, *Computer Networks*, Prentice Hall, 3rd ed., 1996.

[89] U. UC Berkeley, Network Simulator version 2 (NS-2), http://www.isi.edu/nsnam/na/ns-documentation.

[90] M. Greis, Tutorial for the network simulator "ns", http://www.lkn.ei.tum.de/studium/veranstaltungen/scn/ns2/tutorial/nsintro.html

[91] A. Law and W. Kelton, *Simulation modeling and analysis*, McGraw-Hill International series, third edition ed., 2000.

[92] A. Mehrotra, *GSM System Engineering*, Artech house, Inc., 1997.