ABSTRACT

Title of Dissertation: SECURING MULTI-LAYER COMMUNICATIONS: A SIGNAL PROCESSING APPROACH

Yinian Mao, Doctor of Philosophy, 2006

Dissertation directed by: Professor Min Wu
Department of Electrical and Computer Engineering

Security is becoming a major concern in this information era. The development in wireless communications, networking technology, personal computing devices, and software engineering has led to numerous emerging applications whose security requirements are beyond the framework of conventional cryptography. The primary motivation of this dissertation research is to develop new approaches to the security problems in secure communication systems, without unduly increasing the complexity and cost of the entire system.

Signal processing techniques have been widely applied in communication systems. In this dissertation, we investigate the potential, the mechanism, and the performance of incorporating signal processing techniques into various layers along the chain of secure information processing. For example, for application-layer data confidentiality, we have proposed atomic encryption operations for multimedia data that can preserve standard compliance and are friendly to communications and delegate processing. For multimedia authentication, we have discovered the potential key disclosure problem for popular image hashing schemes, and proposed mitigation solutions. In physical-layer wireless communications, we have discovered the threat of signal garbling attack from compromised relay nodes in the emerging cooperative communication paradigm, and proposed a countermeasure to trace and pinpoint the adversarial relay. For the design and deployment of secure sensor communications, we have proposed two sensor location adjustment algorithms for mobility-assisted sensor deployment that can jointly optimize sensing coverage and secure communication connectivity. Furthermore, for general scenarios of group key management, we have proposed a time-efficient key management scheme that can improve the scalability of contributory key management from $O(\log n)$ to $O(\log(\log n))$ using scheduling and optimization techniques.

This dissertation demonstrates that signal processing techniques, along with optimization, scheduling, and beneficial techniques from other related fields of study, can be successfully integrated into security solutions in practical communication systems. The fusion of different technical disciplines can take place at every layer of a secure communication system to strengthen communication security and improve performance-security tradeoff.

SECURING MULTI-LAYER COMMUNICATIONS: A SIGNAL PROCESSING APPROACH

by

Yinian Mao

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2006

ACKNOWLEDGEMENTS

I am eternally grateful to my advisor, Professor Min Wu, for her guidance by instilling me with the quintessence of scientific research; for her unwavering support and constant encouragement for my research endeavors; for her being my role model in endless pursuit of academic and professional excellence; and for her immense patience in teaching the unteachable and mentoring the unthinkable.

I am also deeply indebted to Professor K. J. Ray Liu, Professor M. Kıvanç Mıhçak, and Professor Yan Lindsay Sun for their help and support during my graduate studies. Professor Liu brought vision and energy to help propel my research exploration. Professor Mıhçak and Professor Sun are resourceful mentors to learn from and wonderful collaborators to work with.

I would like to thank my colleagues at University of Maryland multimedia group: Guan-Ming Su, Shan He, Hongmei Gou, and Ashwin Swaminathan. They have kept me in good company during the past five years, provided a helping hand when needed, and contributed unselfishly during collaborations.

Finally, I give my heartfelt gratitude to my mother, Ms. Youqin Zhou. She gave me unconditional love and support, made enormous efforts and countless sacrifices, in order for me to have the opportunity to pursue my Ph.D. study. I dedicate this thesis to her.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Motivation

Security is becoming a major concern in this information era. The development in wireless communications, networking technology, personal computing devices, and software engineering has led to numerous emerging applications whose complex security requirements are beyond the framework of end-to-end cryptography. The complexity lies in the fact that emerging electronic devices often contain multiple functional layers and components, each performing different functionality while interacting with each other. On the system level, applications that use such electronic devices nowadays could involve a large number of users, who are often required to perform tasks in a coordinated way with heterogeneous communication capability. The primary motivation of this dissertation research is to discover new approaches to the security problems in emerging communication technologies and multimedia applications, without unduly increasing the complexity and cost of the entire system.

The issue of information security arises in different forms and may cause damages of different severity. For example, piracy in digital video, audio and software incurs huge damage to the economy. A 2001 study shows that the estimated annual revenue losses due to piracy by the Recording Industry Association of America, the Motion Picture Association of America, and the software industry are 3 billion, 4.2 billion, and 11 billion US dollars, respectively. The impact of identity theft on the economy is even larger. A comprehensive study by the US Federal Trade Commission found that there were nearly 10 million victims of identity theft in 2002, and identity theft cost businesses and individuals nearly 50 billion dollars in that year alone [19].

In addition to economic damages, security problems in the emerging communication and multimedia systems may also lead to the compromise of personal privacy, and corporate or state proprietary information. Such information can be exploited by adversaries for malicious purposes. For example, electronic patient record has the potential to allow health care provider and patients to interact more efficiently. However, such records may also be illegally sold to insurance companies, police departments, employers, drug companies. Such concerns have been recently raised by academic experts toward the patient record system "Connecting for Health", which was deployed by the National Health Service of the United Kingdom [5]. An astonishing security breach in electronic records recently occurred at the United States Department of Veteran Affairs, which could leak private information of up to 26.5 million veterans and active-duty service-members [1]. In addition, security issues also exist during information transmission. For instance, many communications system rely on wireless radio transmission, which is very flexible in terms of deployment, but can be easily eavesdropped and interfered. Applications

[1]`http://www.firstgov.gov/veteransinfo.shtml`

using radio communications, such as ad-hoc network and sensor network, may also be deployed in hostile environment. For these applications, a comprehensive security framework is required to ensure information confidentiality and authenticity, detect system intrusion, and trace security breaches.

1.2 Background

In this section, we introduce general background related to security in communication systems and multimedia applications. We focus on various aspects where existing security solutions cannot meet the requirements of the emerging applications, and provide general philosophies on how to resolve such issues.

In the past, the security in communication systems have been addressed mainly by cryptography. To achieve various security goals, cryptographic primitives, such as encryption tools, authentication tools, and cryptographic random number generators are designed [76][115]. The role of encryption tool is to protect the confidentiality of data, by either encrypting the data one block at a time (i.e. block-cipher), or performing bit-by-bit encryption (i.e. stream cipher)[115]. For authenticating the content of a message, hash functions can be used. A hash function is a one-way function producing a fixed-length binary sequence from a variable-length input message [76]. Based on the cryptographic primitives, cryptographic protocols can be developed to specify how to use the cryptographic primitives in order to achieve a security objective [52]. Usually a cryptographic protocol specifies a sequence of actions to be taken by the protocol participants, as well as certain conditions to be satisfied for the protocol to be successfully carried out. One such example is the key management protocol. Since most (symmetric-key) cryptographic primitives require a secret key as input, the information sender and receiver usually need to agree upon the same secret key before such cryptographic operations as encryption and decryption can be performed. The key management protocol achieves this goal by either using a trusted third party, called the key distribution center, or using public-key cryptographic primitive such as the Diffie-Hellman algorithm [52].

Traditionally, communication system and upper layer multimedia applications are separately considered in system design. With the advances in communications and networking, and multimedia signal processing, more integrated approaches have been proposed for multimedia communications. For instance, unequal error protection techniques for transmitting multimedia over error prone channels have been proposed to exploit the different importance in various layers of media representation [121]. For multicasting multimedia content over heterogeneous network, efficient media transcoding schemes have been proposed to adapt the bit-rate of media content according to available bandwidth, processing power, or display resolution [138]. At the same time, more communication system designs take into account the nature of data being transmitted and the applications that utilize the communication channel. For example, different streams of data, such as voice, video, image, or text are assigned different priorities in the communication system. For delay-sensitive applications, such as interactive voice, more network resources will be reserved to achieve better quality of service. In such an integrated multimedia communication system, incorporating a security component is a challenging task. If we encrypt the multimedia content using a contemporary cipher directly, the encryption will remove the bit stream syntax contained in the compressed media. However, doing so may hinder the capability of rate adaptation in network transmission, or applying unequal error protection when the media is transported through error prone channels [72]. This is because most of the bit-rate adaptation schemes and unequal error protection schemes rely on the syntax and structure in the coded media content to identify which part of the media stream is more important. Verifying the authenticity of the media content is equally challenging as encryption. Since the media content can go through rate adaptation or be corrupted by transmission noise/packet loss, the bit-by-bit representation of media content may be changed. Ideally, such graceful quality degradation does not indicate malicious modification of the content, and the received content should still be considered authentic in principle. Unfortunately, these goals cannot be achieved by directly applying cryptographic hash to media authentication. In this dissertation, we will introduce techniques to address these challenges in secure multimedia communications.

One special class of security problems is how to trace an adversary or compromised user within a group of users. We introduce an abstraction of the traitor tracing in the cooperative communication scenario as follows. Let us consider a group of nodes that can communicate with each other. One node s sends a message M to a destination node d, through a number of relay nodes $r_1, r_2, ..., r_n$. Each of these relay nodes obtains a message from sender s containing (almost) the same content, and may or may not forward this message to d. There are one or multiple adversaries within the group of relay nodes. These adversaries can modify the message M to M' and send M' to the destination d. Upon receiving a message M, d may want to find out the following information: (1) from which relay the message M was sent, and (2) whether the message has been tampered. When combining the answer to both questions, we can detect the tampering and relate it to the compromised relay node(s). This is especially useful in tracing information tampering in relay communications, such as in wireless ad hoc networks. In such networks, the wireless communication channel and the malicious nature of attacks can make the task of traitor tracing very challenging. The message s sent to each relay node r_i could contain the same content plus the relay node's unique ID. However, if the relay node(s) is adversarial, it (they) will have every incentive to remove such ID information to avoid being identified. Another example is that the received message at d is a superposition of relay messages from multiple relay nodes through wireless channel. The relay signals are combined by physical laws related to microwave signal propagation. In this situation, it is very hard for the destination node to distinguish which part of the received message is contributed by which relay node. It is clear that solving such a traitor tracing problem not only requires cryptographic tools, but also involves signal processing to take into account the nature of signal transmission over the physical communication channel.

In an electronic system that involves communication, signal processing, and system security, there are tradeoffs among security, performance and functionality. Since the security subsystem is a part of the overall system, it will interact with other components in the system. The functionality and performance of all system components may be mutually constrained and affected. The security-performance or security-functionality tradeoffs are very important because, as secure communication systems become more complex, the interaction and interdependency between the security subsystem and other system components are also becoming more intricate. Sometimes adding a security component may lead to incompatibility with other system components or substantial performance loss. In order to achieve security and improve the performance in the overall system, one can seek to optimize the security protocol when the security sub-system is stand-alone and such optimization will not drastically affect other system components. Otherwise, one should seek to optimize the tradeoff between the security component and other performance aspects in the system design, which can lead to a more balanced system performance. One example in security-performance tradeoff is the tradeoff between key manage protocol and other system performance aspects. Such a tradeoff has an especially significant impact to the overall system performance in group communications. First, the complexity and cost of the key management should have good scalability with respect to the communication group size. Second, the key management scheme should also take into consideration the memory and computation resources available at each individual node, such as in sensor networks or other networked embedded systems. Third, the key management scheme often interacts with other system performance criterions. Given a practical system, how to jointly optimize the performance of key management together with other system performance aspects is an important issue open for investigation.

1.3 Organization and Contribution

This dissertation focuses on identifying, modelling, and solving the security problems in a single layer or across several layers of a secure communication system. The main challenge in solving such security problems is that, the issues in application security are often coupled with system constraints, and directly applying existing cryptographic tools cannot solve such emerging security problems without violating the constraints. The contributions of this dissertation can be summarized in three aspects. Given an application and its security requirements, we first model and formulate the security problems and system constraints into a common framework. Based on such a framework, we propose new security solutions and evaluate its strength using existing cryptographic tools and signal processing algorithms. When performance versus security tradeoffs are involved, we introduce quantitative metrics that capture the tradeoff within the system formulation, and improve the tradeoff using signal processing and optimization techniques.

The nature of this research is to view security issues in secure communication systems from cross-layer, inter-disciplinary perspectives and jointly consider system security, system constraint, and system performance. To that end, not only signal processing theory and algorithms, but also optimization techniques, scientific computing algorithms, and other related knowledge from different disciplines can be employed in solving security problems and optimizing security versus performance tradeoff. In this dissertation, we show such design principle through a number of examples in different layers of secure communication systems. In Chapter 2, we focus on application-layer communication security and introduce a joint signal processing and cryptographic approach to multimedia encryption. We also discuss the security and robustness tradeoff in image hashing by adapting the concept of unicity distance pioneered by Shannon [101]. In Chapter 3, using signal processing techniques for physical-layer wireless communications, we discuss how to trace malicious attacks from adversarial relays in cooperative wireless communications, which is an emerging communication paradigm for ad hoc networks. In Chapter 4, we look into the joint optimization of key establishment and sensing coverage for secure sensor network design and deployment, and propose a coordinated sensor deployment framework that can improve the sensing and secure communication tradeoff. For general scenarios of group key management, in Chapter 5, we investigate how to optimize the time efficiency of rekeying operations in contributory key management, using dynamic scheduling and cost amortization. Through these typical scenarios, we demonstrate the potential and benefits of introducing signal processing in formulating and solving security problems in different layers of a secure communication system. Finally, conclusions and future perspectives are drawn in Chapter 6.

Chapter 2

Multimedia Encryption and Content Authentication

Signal processing has played a significant role in developing coding and communication technologies for digital multimedia, paving ways to many opportunities for people around the world to acquire, utilize, and share multimedia content [93]. To allow for wider availability of multimedia information and successful commercialization of many multimedia related services, assuring that multimedia information is used only by authorized users for authorized purposes has become essential. This chapter focuses on jointly utilizing signal processing and cryptography to protect the confidentiality and achieving access control of multimedia information, as well as authenticating the media content received through communication channels.

In a typical use of multimedia illustrated in Fig. 2.1, the owner of the multimedia content wants to distribute the content through networks or archive it for future use. With the sophistication of heterogeneous networks and the growing amount of information being generated, it is becoming less efficient for the content owners to manage the distribution or archiving process all by themselves.

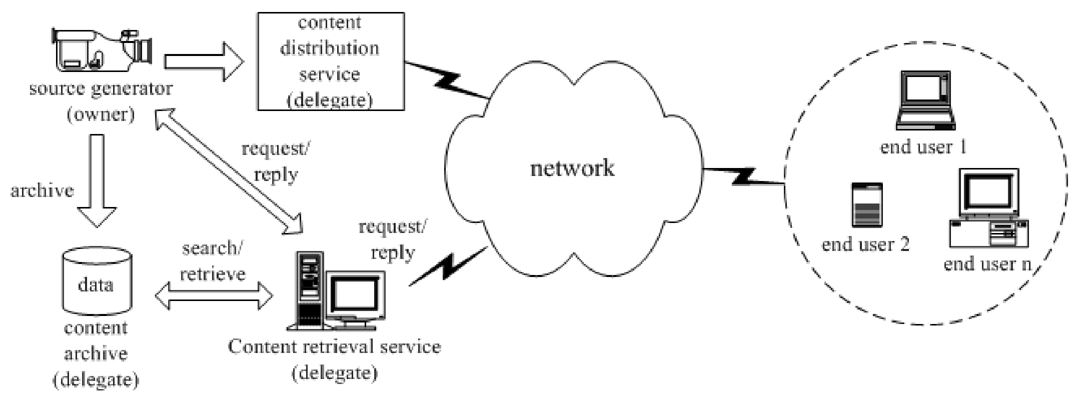


Figure 2.1: Typical usage of multimedia content

As a result, third-party service providers, equipped with specialized servers, huge disk space, and abundant bandwidth resources, will serve as *delegates* for content owners to perform content distribution, archiving, search and retrieval. On one hand, the delegate service providers often need to process the received media data, such as adapting the rate of the media data according to the available bandwidth. On the other hand, the owner may not want to reveal the media content to these delegates because of security and privacy concerns. One such example is the privacy-preserving data retrieval using untrusted server [40].

A common way to achieve content confidentiality is to encrypt the entire multimedia data using a cipher, such as DES, AES, or RSA [76][115]. However, many types of processing, such as rate adaptation for multimedia transmission in heterogeneous networks [58] and DC-image extraction for multimedia content searching [137], cannot be applied directly in the bitstream encrypted by these generic encryption tools or their simple variations. This implies that the delegates are still have to hold the decryption keys to decrypt the content, process the data, and then re-encrypt the content. Since revealing decryption keys to potentially untrustworthy delegates is often not in line with the security requirements of many applications, generic encryption alone is inadequate in the delegate service scenario.

Another unique security issue with multimedia delivery is the authentication of received media content. Since the media content may undergo changes during network communication, using a traditional cryptographic hash for authentication may "reject" media contents that has undergone format changes, resolution reduction, or bit/packet errors, which are not intentional tampers to the media content [117][37]. For this reason, an ideal hash should authenticate based on the media content, not the exact bit-by-bit representation. At the same time, the hash should also use a secret key as input and contain randomization that cannot be easily forged or attacked by other means. In addition to image authentication, robust image hashing can also be used in non-oblivious watermarking for image and video [24][14]. Instead of using the original image, the hash can provide partial information of the image content to be used in the watermark detection phase. In video fingerprinting, video frame hashes have been used for temporal registration purpose [41]. For content-based multimedia retrieval, image hashes can be used as a concise ID for each image. Thus fast and effective comparison of image content can be achieved by comparing the distance of hash vectors [62]. In all these applications, the image hashing scheme needs to be resilient to a set of authenticate modifications and sometimes minor intentional distortions to the original image content. These modifications include rotation, scaling, noise corruption, and common image filtering.

The first step towards addressing the aforementioned issues is to design flexible multimedia encryption schemes that can handle delegate processing and achieve access control by content and quality [65], as well as content-based multimedia authentication tools. In this chapter, we first focus on the design of encryption tools for multimedia that can *encrypt once and securely process in many ways* using the existing multimedia signal processing techniques. To achieve this goal, we jointly consider signal processing and cryptography in our exploration of multimedia encryption. In particular, we investigate the possible domains in which encryptions can be applied, including the sample domain, the quantized transform domain, the intermediate bitplanes, and the bitstream domain. We propose and analyze two atomic encryption operations tailored for multimedia signals, namely, a generalized index mapping encryption tool with controlled overhead and an intra-bitplane encryption tool compatible with fine granularity scalable coding. A video encryption system, which incorporates the proposed operations and other relatively straightforward extensions of generic encryption, is then studied. The resulting system takes into consideration the structure and syntax of multimedia sources and protects the content confidentiality during delegate processing. Then we investigate the security and robustness tradeoff exhibited in image hashing schemes, by adapting the unicity distance concept pioneered by Shannon [101]. Through two examples, we show that the security of typical image hashing schemes can be quantitatively analyzed. Such security analysis framework can provide guidance in the design of multimedia hashing schemes.

2.1 Background and Preliminaries

We examine in this section the possible domains in which encryption can be applied to multimedia, along with a review of prior work. Using a widely adopted multimedia coding framework, we illustrate the candidate domains for applying encryption to multimedia in Fig. 2.2.

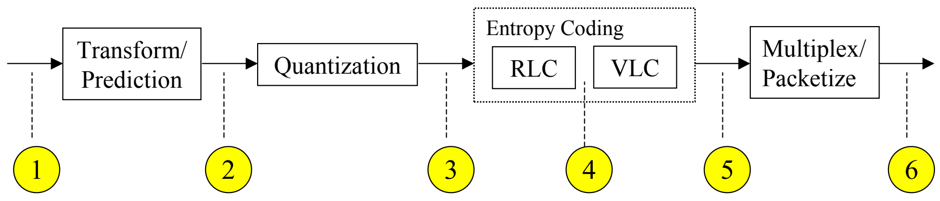


Figure 2.2: Candidate domains to apply encryption to multimedia

2.1.1 Encryption Before and After Coding

According to Fig. 2.2, there are two straightforward places to apply generic encryption to multimedia. The first possibility is to encrypt multimedia samples before any compression (i.e. Stage 1 in Fig. 2.2). The main problem with this approach is that the encryption often significantly changes the statistical characteristics of the original multimedia source, resulting in much reduced compressibility. It is worth noting a novel approach has been recently proposed to efficiently compress encrypted data [49]. By employing distributed source coding theory, this new method achieves the same compression gain as compressing the unencrypted data in the case of ideal Gaussian source. The compression gain, however, would be reduced for more general source that is common in practice, and it cannot easily support many other forms of delegate processing.

The second possibility is to apply generic encryption to the encoded bitstream after compression (i.e., Stage 5 and 6 in Fig. 2.2) [94]. This approach introduces little overhead, but may destroy the structures and syntax readily available in the unencrypted bitstream. Such structures, often indicated by special header/marker patterns, would enable many kinds of processing in delegate service providers and intermediate network links, such as bandwidth adaptation, unequal error protection, and random access [136, 131, 121, 126].

As headers and markers are special bit patterns in a compressed bitstream for a variety of purposes [121], a simple way to realize syntax-aware encryption is only to encrypt the content-carrying fields of the compressed multimedia bitstream, such as the fields of motion vectors and DCT coefficients in MPEG video, and keep the structure and headers/markers of the bitstream unchanged [125][140]. However, this method can only preserve limited syntax from unencrypted media to facilitate a fixed set of processing. Other advanced features in multimedia signal processing, such as the fine granular scalable (FGS) coding [102, 95, 58], cannot be easily preserved using this approach. Bitrate overhead also occurs in the range of 1%-8% due to block padding and the added side information. Furthermore, each tailored encryption technique may also require different handling by delegate processing units. This is not always realistic, because the processing units are likely coming from different vendors and the barrier to standardization seems insurmountable due to market considerations [7].

After applying generic encryption, some parts of the encrypted media data could become identical to certain headers/markers. This emulation problem can bring potentially serious troubles to delegate service providers and intermediate processing modules [126] when the multimedia data go through certain network protocols, transcoding, and error recovery. One possible remedy to header emulation is bit-stuffing [132], a technique that is widely adopted in the packetization stage of network communications [111].

2.1.2 Encryption at Intermediate Stages of Coding

Recently, there have been interests in studying how to encrypt multimedia data in such a way that the encrypted data can still be represented in a meaningful, standard-compliant format [86]. They are particularly useful for secure delegate services and multimedia communications that prefer handling media streams compliant to certain multimedia coding standard, such as JPEG or MPEG-1/2/4 standard [126][127]. The encryption is performed in the intermediate stages illustrated in Fig. 2.2. For example, at Stage-2, the motion vectors in video can be encrypted by applying DES to their codeword indices [126]. At Stage-3, DC and selected AC coefficients in each block of a JPEG image or an MPEG video frame can be shuffled within the block [112], or across blocks but within the same frequency band [141]. At Stage-4, the entropy codeword can be spatially shuffled within the compressed bitstream [127]; the Huffman codewords of coefficients and/or motion vectors can be encrypted by alternating between several Huffman codebooks in a cryptographically secure fashion [130]. At Stage-5, only intra-coded frames and blocks of an MPEG video are selected and encrypted using a classic DES cipher [133] or its variations [94]. Some of these schemes are also known as *selective encryption* [127, 133, 130], i.e., they encrypt only portions of multimedia data stream that carry rich content, in hope of alleviating the problem of high computational complexity and the potential bitrate overhead. However, we believe that the computational complexity in encryption is not a major concern given the fast-growing computation power and the efficient implementation of established generic encryption tools. In contrast, the protection of content confidentiality under different processing scenarios, such as delegate services and communications, is of paramount urgency. Unfortunately, few existing work has thoroughly considered these scenarios.

In Section 2.2 and Section 2.3, we propose two general atomic encryption operations using index mapping and constrained shuffling to achieve confidentiality protection in delegate services and other processing scenarios. The rationale is to ensure that the encrypted bitstream still complies with the state-of-the-art multimedia coding techniques. We are particularly interested in how much price in terms of security and compressibility these techniques have to pay to achieve standard-compliant encryption, and we answer this question through analysis and simulations.

2.1.3 Content-based Authentication using Image Hashing

Image hash is a content-based concise representation of an image. Just as the traditional cryptographic hash, image hash can be used to verify the authenticity of image content. To this end, the generation of the image hash must be randomized according to a secret key, which is to ensure that an unauthorized user will not be able to forge a hash without the key. What makes image hash different from the cryptographic hash is that, an image usually allows different representations for approximately the same content. For example, an image may be compressed to lower bit-rate or corrupted by noise during transmission. After such distortions, the bit-by-bit representation of the image has been changed while the content has been preserved. In order to verify the authenticity of the image content under such situations, an ideal image hashing scheme should be robust against moderate distortions, such as rotation, scaling, filtering, compression, and additive noise, etc. In the literature, a number of robust image hashing schemes have been proposed in recent years [117][37][108][79]. In Section 2.6, we show that the robust image hashing schemes should be used in a proper way, otherwise an adversary would be able to exploit the robustness constraint in image hashing schemes, deduce the hashing key, or forge a valid hash for a completely new image. We will present our analysis under the framework of unicity distance proposed by Shannon.

The concept of unicity distance was pioneered by Claude Shannon in investigating encryption systems[101]. The basic idea is that, when a secret key is used to encrypt multiple messages, the amount of uncertainty in the encryption key (or the cipher text for a new message) reduces with the increased number of observed clear text-cipher text pairs. When the number of observed cryptograms is large enough, one can almost surely determine the secret key from the observations. The number of observations (or some normalized form) is called the unicity distance. Thus the unicity distance quantifies how many clear text-cipher text pairs are required to almost surely determine the secret key.

Shannon demonstrated the concept of unicity distance using a two-letter substitution cipher in his paper[101]. The field of cryptography has been advanced significantly since Shannon's discovery. Most contemporary block ciphers nowadays require a minimum key length that is 64 bits or more. For a message block of 128 bits or more, one would require an enormous number of clear text-cipher text pairs to deduce the secret key. Therefore it is hard to demonstrate the unicity distance in contemporary ciphers. However, for perceptual hashes that are constrained by the robustness requirement, we found that demonstrating unicity distance only requires a few dozen of observed image-hash pairs. These results are presented in Section 2.6.

2.2 Generalized Index Mapping with Controlled Overhead

Unlike generic data encryption where the encryption output can take values over the entire data space, joint signal processing and cryptographic encryption requires that the encrypted output should satisfy additional constraints. These constraints are essential to preserve the structure, syntax, and standard compliance that enable delegate processing and leads to communication friendliness. A format-compliant encryption scheme was proposed in [126] by assigning a fixed-length index to each variable length codeword (VLC), encrypting the concatenated indices, and then mapping the encrypted indices back to codeword domain to form an encrypted bit-stream. This prior approach would work well with such codes as the Huffman codes and the Golomb-Rice codes, which associate each symbol coming from a finite set with a unique codeword of integer length, but it cannot be directly applied to VLCs that allow fractional codeword length per symbol, such as the arithmetic codes. In addition, this prior encryption work incurs a substantial amount of bitrate overhead, and analytic study has not been provided regarding the overhead. In this section, we construct and analyze an encryption tool that can overcome these two problems.

2.2.1 Generalized Index Mapping

We extend the index encryption idea to apply encryption directly to symbols that take values from a finite set before getting into VLC codeword domain. Examples include working with quantized coefficients and quantized prediction residues (Stage #3 in Fig. 2.2), as well as run-length coding symbols (Stage #4 in Fig. 2.2).

The encryption process to produce a ciphertext symbol $X^{(enc)}$ from a clear-text symbol X is shown as follows:

$$X^{(enc)} = Enc(X) \triangleq T^{-1}[\mathcal{E}(T(X))], \tag{2.1}$$

where $\mathcal{E}(\cdot)$ is a core encryption primitive such as AES or one-time pad [115], and $T(\cdot)$ represents a codebook that establishes a bijective mapping between all possible symbol values and indices represented by binary strings. The goal of this bijection is to produce fixed-length indices that will be passed to subsequent encryption or decryption. The decryption process has a similar structure:

$$X = Dec(X^{(enc)}) \triangleq T^{-1}[\mathcal{D}(T(X^{(enc)}))], \tag{2.2}$$

where $\mathcal{D}(\cdot)$ is a core decryption primitive corresponding to $\mathcal{E}(\cdot)$.

As a simple example, we consider encrypting a string of symbols coming from a finite set {A, B, C, D}. The symbol sequence to be encrypted is “ABBDC”. We first assign a fixed-length index to each symbol:

$$\begin{array}{ccccccc} A & \rightarrow & [00], & & B & \rightarrow & [01], \\ C & \rightarrow & [10], & & D & \rightarrow & [11]. \end{array}$$

We then convert the symbol sequence to an index sequence “00 01 01 11 10”, and encrypt the index sequence using an appropriate encryption primitive such as a stream cipher (the one-time pad) with a random bit-stream [0100 1011 1001 ...]. Finally we convert the encrypted index sequence “01 01 11 00 00” back to symbol sequence “BBDAA”. After encryption, any appropriate VLC coding can be applied to the encrypted symbol sequence. It is worth noting that in such an encryption one input symbol can be mapped to different encrypted cipher-text. For instance, in the previous example the symbol B has appeared in the clear-text sequence twice, the first time it was mapped to B and the second time to D.

When processing a large sequence of symbols, the encryption method by index mapping tends to make the encrypted symbols uniformly distributed, which is good in terms of security [115]. However, the entropy of the encrypted symbols is increased from the unencrypted ones. Since the compressibility of a sequence of symbols using entropy coding depends on the entropy of the source symbols [23], the index-mapping encryption would bring a bit-rate overhead in compression. Next, we discuss how to quantify and control this overhead.

2.2.2 Analysis and Control of Overhead

In the following analysis, we investigate the impact of the index encryption on the compressibility of the source symbols, which can be quantified by the changes in average code length before and after encrypting a sequence of symbols.

Case-1

We consider compressing the source symbols using a default entropy codebook as provided by many multimedia standards. The default codebook is obtained from a set of representative training samples and is used most of the time for the simplicity of implementation. We denote the probability mass function of the symbols prior to encryption by $\{p_i\}$, that of the symbols after encryption by $\{q_i\}$, and the code length designed for distribution $\{p_i\}$ by $\{l_i\}$. If encryption is performed on an index drawn from the full range of symbol values, the distribution of ciphertext symbols, q, will be uniform over the entire range. Alternatively, if we partition the range of symbol values into mutually exclusive subsets $\{S_j\}$ and restrict the outcome of the encryption of a symbol $x \in S_j$ to be within the subset S_j, i.e., $Enc(x) \in S_j$, the distribution q will be a piecewise uniform approximation of p, as illustrated in Fig. 2.3.

Consider $\{q_i\}$ to be a piecewise uniform approximation of $\{p_i\}$, i.e., for each subset S_j from a non-overlapping partition of the symbols' range, we have $\sum_{i \in S_j} p_i = \sum_{i \in S_j} q_i = |S_j| q^{(S_j)}$, where $q^{(S_j)} \triangleq q_i$ for all $i \in S_j$, and $|\cdot|$ denotes the cardinality of a set. Assuming that encryption changes the symbol distribution from $\{p_i\}$ to the above $\{q_i\}$ and the same codebook of code length $\{l_i\}$ is used both before and after encryption, we can show that the changes of the expected code length δL is

$$\delta L = \sum_i (q_i - p_i) l_i = D(p||q) + D(q||r) - D(p||r), \tag{2.3}$$

where $D(\cdot||\cdot)$ represents the Kullback-Leibler divergence, and r represents a probability distribution of $\{r_i \triangleq P(R = i) = 2^{-l_i} / \sum_k 2^{-l_k}\}$. The derivation is presented in Appendix-A.

If we partition the symbol range S into more than one subset and restrict the encryption output to be in the same subset as the input symbol, the complexity of a brute-force attack for each symbol is reduced [1] from $2^{|S|}$ to $2^{|S_j|}$, where S_j is the subset to which the symbol belongs. On the other hand, the overhead is also reduced because in the Kullback-Leibler divergence sense the distance from the original distribution p to the piecewise uniform distribution q is closer than that to a completely uniform distribution. Thus by controlling the set partitioning, we can adjust the tradeoff between the security and the overhead. In addition, Eq. (2.3) suggests that the optimality of the codelength $\{l_i\}$ designed for probability distribution $\{p_i\}$ affects the changes of the expected code length after encryption. If the code is optimal for $\{p_i\}$, i.e., $l_i = -\log p_i$, then $D(p||r) = 0$ and Eq. (2.3) is reduced to $\delta L = D(p||q) + D(q||p)$.

[1] This is equivalent to encrypting fewer bits of the indices. As to be discussed later in this section, symmetric set partitioning can protect the sign bit, which is important to resist attacks.

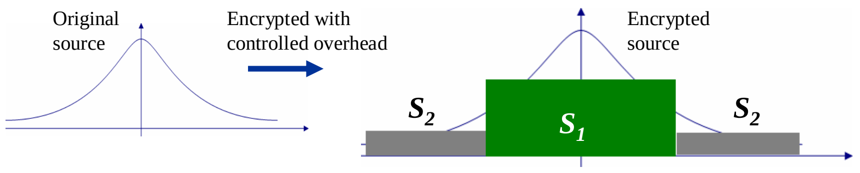


Figure 2.3: Index mapping within subsets gives piecewise constant approximation of the distribution.

Case-2

We consider compressing the source symbols using adaptive or universal entropy coding that adjusts itself to the source's distribution. Arithmetic coding and Lempel-Ziv coding are two such examples. Assuming that the adaptive entropy coding can achieve the entropy bound for any source distribution, we exploit the piece-wise constant property of $\{q_i\}$ and show in Appendix-A that the change of the average codeword length is

$$\delta L = H_{\{q_i\}} - H_{\{p_i\}} = D(p||q), \tag{2.4}$$

where $H_{\{x_i\}}$ is the entropy of a discrete random variable following the distribution $\{x_i\}$. Similar to the first case, this result also indicates that if we partition the symbol range S into more than one subset and restrict the encryption output to be in the same subset as the input symbol, the distribution of encrypted source, $\{q_i\}$, can better resemble that of the original source, $\{p_i\}$, leading to reduced overhead in compression.

The final result of the relative bitrate overhead (η) also depends on the ratio of the size of the content to be encrypted (B_1) to the overall size of the stream (B). That is,

$$\eta = \frac{B_1^{(e)} - B_1}{B} \times 100\% = \eta_e \frac{B_1}{B} \times 100\%, \tag{2.5}$$

where $B_1^{(e)}$ denotes the size of the encrypted part and η_e is the relative overhead for the part being encrypted. Even if η_e is large as in the case of prior work [126], the overall overhead can be constrained if only a relatively small part of the stream is encrypted. With our proposed technique of set partitioning, the overall overhead can be controlled both through reducing η_e and through maintaining a low B_1/B ratio by selectively encrypting only a portion of the stream (such as the perceptually significant coefficients).

2.2.3 Set-Partitioning in Index Mapping

As we have seen, the set partitioning technique can control the bitrate overhead introduced by the index mapping encryption, trading off the resistance against brute-force attacks. The choice of partition also affects the security against estimation attack. Since the encryption flattens the distribution within each subset, no particular clue of the exact unencrypted source value can be inferred from its encrypted value. The best estimate of an unencrypted value X in terms of the mean square error has to resort to its conditional statistical distribution in the subset S_j. It can be shown that, by observing $X^{(enc)} \in S_j$, the minimum mean square error (MMSE) estimate $\hat{X}$ for X is

$$\hat{X} = \text{argmin}_t E(|X - t|^2 | X^{(enc)}) = E(X|S_j), \tag{2.6}$$

and the corresponding mean square error is

$$E(|\hat{X} - X|^2 | S_j) = Var(X|S_j). \tag{2.7}$$

It has been known that many variables in efficient multimedia representation, including the quantized transform coefficients and prediction residues, follow a symmetric zero-mean distribution such as a Gaussian distribution or a Laplacian distribution. For these variables, if the subset partition is symmetric around zero, as shown in the example of Fig. 2.3, the MMSE estimate of the original value given the encrypted one will always be zero, regardless of the subset. This means that the sign of the original value is not inferrable. Since the sign, or more generally, the phase, is known to carry important information of a signal [84], an attacker can hardly get any useful information from such MMSE estimation. However, if the partition is not symmetric, for example, as in Fig. 2.4, the MMSE estimates still preserve the sign and the approximate energy of the original value for each subset. Such a choice of partition can leak a substantial amount of information to an attacker after estimation, and therefore should be avoided.

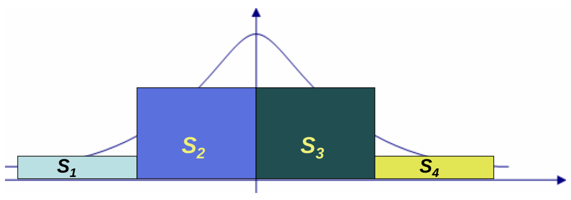


Figure 2.4: Set partition with subsets asymmetric to zero

For several popular entropy coding techniques such as those in JPEG and MPEG, the choice of set partition can be tailored to eliminate the overhead. For example, the JPEG standard employs run-length coding, where the run of zero coefficients are coded before the encoding of a non-zero coefficient. If we only encrypt the values of non-zero coefficients and group all coefficient values with the same code-length into the same subset, the code-length of the encrypted value will be identical to that of the unencrypted one. No bitrate overhead will be introduced in this particular case.

2.2.4 Examples and Discussions

As an example, we encrypt the DC prediction residues of the JPEG representation of the 512×512 *Lena* image using the index mapping approach. In JPEG, the DC coefficient in each block collectively captures the coarse information of an image and is differentially encoded to reduce the redundancy. For natural images, DC differential residues are approximately Laplacianly distributed with very small probability outside the range of $[-63, 64]$. We encode both the unencrypted and the encrypted prediction residues with the default Huffman table in the JPEG standard. Without encryption, the average code length for encoding DCs is 5.78 bits. In the first encryption experiment, we apply the proposed generalized index encryption to the DC differential residues within $[-63, 64]$ without set partitioning. The index encryption is realized via XORing with a one-time pad, resulting in an average code length of 8.60 bits, or an overhead of 2.82 bits. In the second encryption experiment, we partition the symbol range of $[-63, 64]$ into two subsets $[-31, 32]$ and $[-63, -32] \cup [33, 64]$, and restrict the input and output of index encryption to be in the same subset. Fig. 2.5 shows the encryption result of the Lenna image [2]. With set partitioning, the overhead in average code length is reduced from 2.82 bits to 1.53 bits.

[2]Encrypting DC alone is not secure enough as an attacker can still get the edge information by setting the DCs to constant and observing the resulting image. We only encrypt DC in this experiment for the purpose of demonstrating the proposed approach as one potential building block. We will show in Section 2.5 that a complete encryption system should encrypt both DCs and other information.

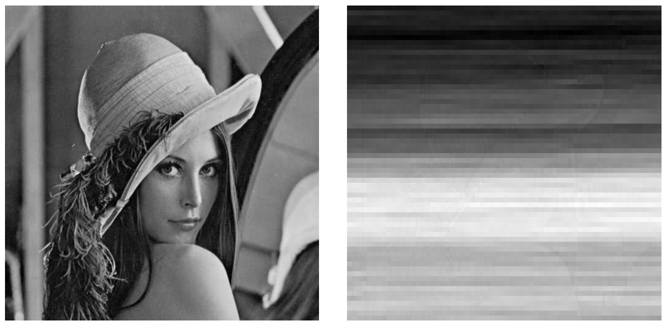


Figure 2.5: Encryption results using the Lenna image based on generalized index mapping of DC differential residues: (left) original, (right) encrypted.

2.3 Constrained Shuffling

Random permutation or shuffling is a common cryptographic primitive operation. The temporal characteristic of audio and video data as well as the spatial characteristic of visual data make permutation a natural way to scramble the semantic meaning of multimedia signals. Even before the arrival of digital technology, an early work by Cox *et al.* builds an analog voice privacy system on a subband representation framework and permutes time segments of subband signals across both time and frequency [25]. More sophisticated coding techniques have been employed by modern digital coding systems. Thus to control the bit-rate overhead and allow for delegate processing, random permutation should be performed in a constrained way and in appropriate domains. In this section, we use the encryption of scalable video with fine granularity as an example to illustrate the proposed constrained shuffling technique, which is known as the *intra bitplane shuffling.* This encryption technique is compatible with fine granularity scalable coding and provides a tool for access control of multimedia content at different quality levels.

2.3.1 Intra Bitplane Shuffling (IBS)

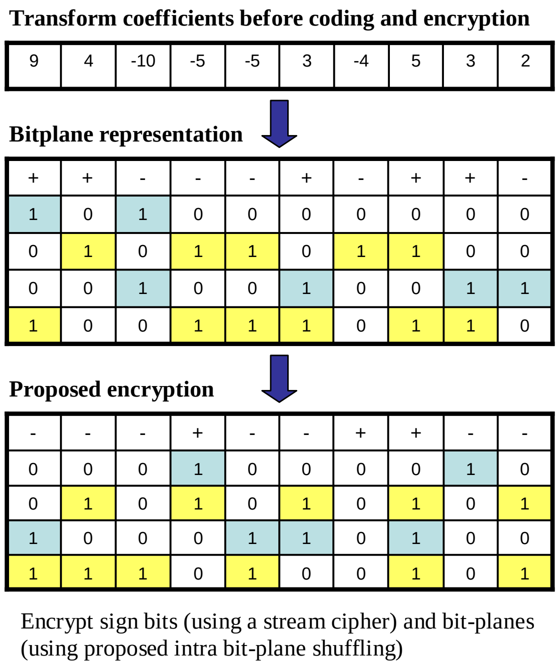
Fine granularity scalability (FGS) is desirable in multimedia communications to provide a near-continuous tradeoff between bitrate and quality. FGS is commonly achieved by bitplane coding, as used in the embedded zero-tree wavelet (EZW) coder [102] and the MPEG-4 FGS coder [95]. We shall use the MPEG-4 FGS to illustrate the concept and the approach can be extended to other FGS coders. As surveyed in [58], MPEG-4 FGS is a functionality provided by the MPEG-4 streaming video profile. A video is first encoded into two layers, namely, a base layer that provides a basic quality level at a low bit rate and an enhancement layer that provides successive refinement. The enhancement layer is encoded bitplane by bitplane from the most significant bitplane to the least significant one to achieve fine granularity scalability. Each bitplane within an image block is represented by (R_i, EOP_i) symbols, where R_i is the run of zeros before the i-th "1", and EOP_i is an end-of-plane flag indicating whether the current "1" is the last bit with value 1 in the current bitplane. The run-EOP symbols are encoded using variable-length codes and interleaved with sign bits.

To provide access control to the FGS encoded enhancement layers, the index-based encryption discussed in Section 2.2 can be applied to each run-EOP symbol, and the overhead can be analyzed using Eq. (2.3) or (2.4). We now present an alternative encryption by shuffling each bitplane according to a set of cryptographically secure shuffle tables.

Fig. 2.6 illustrates the proposed intra-bitplane shuffling. We perform random shuffling on each bitplane of n bits and the shuffled bitplane will then be encoded using the run-EOP approach. For example, the first unencrypted bitplane [3] in

[3]We use "the first bitplane" to denote the MSB bitplane throughout this chapter.

Fig. 2.6 "1 0 1 0 0 0 0 0 0 0" has $n_1 = 2$ bits of value "1" out of a total of $n = 10$ bits, which will lead to $\binom{n}{n_1} = 45$ different permutated patterns. In addition to bit-plane shuffling, the sign bit s_i of each coefficient is randomly flipped according to a pseudo-random bit b_i from a one-time pad, *i.e.*, the sign remains the same when $b_i = 0$ and changes when $b_i = 1$.



Transform coefficients before coding and encryption
Bitplane representation
Proposed encryption
Encrypt sign bits (using a stream cipher) and bit-planes (using proposed intra bit-plane shuffling)

9	4	-10	-5	-5	3	-4	5	3	2

+	+	-	-	-	+	-	+	+	-
1	0	1	0	0	0	0	0	0	0
0	1	0	1	1	0	1	1	0	0
0	0	1	0	0	1	0	0	1	1
1	0	0	1	1	1	0	1	1	0

-	-	-	+	-	-	+	+	-	-
0	0	0	1	0	0	0	0	1	0
0	1	0	1	0	1	0	1	0	1
1	0	0	0	1	1	0	1	0	0
1	1	1	0	1	0	0	1	0	1

Figure 2.6: Illustration of intra bitplane shuffling

2.3.2 Analysis of Overhead

An important property of shuffling is that the set of elements before and after shuffling are identical. For intra-bitplane shuffling, this implies that the number of "1"s is preserved. Thus the number of run-EOP symbols representing the encrypted bitplane is unchanged, ensuring no overhead coming from the increase in the number of run-EOP symbols. We denote by N_d the number of occurrences that the run of zeros before a "1" equals to d after shuffling. Assuming each of the

$\binom{n}{n_1}$ shuffles is equally likely for a specific n-bit bitplane with a total of n_1 bits of "1", we show in the Appendix-B that N_d has expected value

$$E(N_d|n_1) = \left(\prod_{k=0}^{d-1}(1 - \frac{n_1}{n-k})\right)\frac{{n_1}^2}{n-d}. \tag{2.8}$$

We therefore arrive at an expected histogram $\{E(N_d|n_1)\}$ versus d, which suggests the likelihood of getting each possible run length.

Shuffling can also be done in a block larger than the block for run-EOP encoding. For example, we can shuffle a macroblock of $n = 256$ bits and perform run-EOP encoding in a smaller block of $n_B = 64$ bits. In this case, the expected zero-run histogram within an encoding block will become

$$E(N_d|n_1) = \prod_{k=0}^{d-1}(1 - \frac{n_1}{n-k}) \cdot \frac{n_1}{n-d} \cdot \left[n_1 - \frac{(n-n_B)(n_1-1)}{n-d-1}\right],$$

which will be reduced to Eq. (2.8) if $n = n_B$. From the histogram before and after encryption we can arrive at the expected overhead per symbol.

As a proof-of-concept, we experiment on the FGS encoded enhancement layer of two video sequences in QCIF format (176×144 pixels per frame). One is 10 frames from the "*Foreman*" sequence, and the other is 100 frames the "*Carphone*" sequence. We use intra-bitplane shuffling to encrypt each bitplane of the enhancement bitstream, where shuffling is performed on a macroblock ($n = 256$) followed by encoding block by block ($n_B = 64$). The expected histograms $\{E(N_d|n_1)\}$ is presented in Fig. 2.7, which also shows that the experimental results match the above analytic results very well.

To compare different encryption approaches, we use the "*Foreman*" sequence and encrypt the first three most significant bitplanes, which provides sufficient visual scrambling on the enhancement layer. We have found that for the "*Foreman*" sequence, the intra bitplane shuffling approach gives an overhead of 7.0%, while the overhead by the index-mapping approach in Section 2.2 is 14.3%. In general, the bitrate overhead for each bitplane by the proposed shuffling approach depends on the overhead of each run-length symbol and the number of symbols (n_1), as reflected in the plot of the overhead contributed by each bitplane in Fig. 2.8. The arc shape is a result of an increase in symbol number from MSBs to LSBs and a decrease in overhead per symbol. From Fig. 2.8 we can also see that some videos, such as "*Foreman*", have very few number of "1"s in the MSB. Therefore the overhead from encrypting the MSB is very small. This also suggests that the MSB alone does not contribute to the perceptual quality significantly in these sequences, and more bitplanes should be protected.

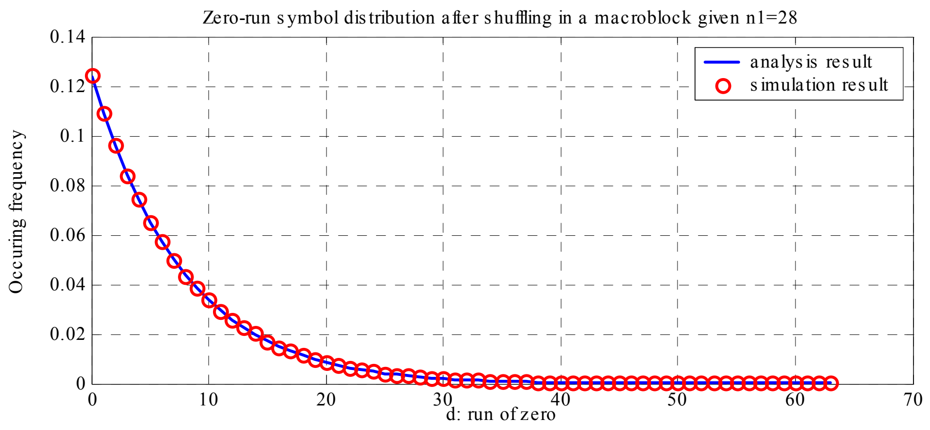


Figure 2.7: Expected histograms of zero-run lengths after intra-bitplane shuffling: solid line indicates the analytic result, and circles indicate the experimental result; the number of "1"s per macroblock (n_1) is 28 bits, which is the average from the 2nd MSB bitplane of the "Foreman".

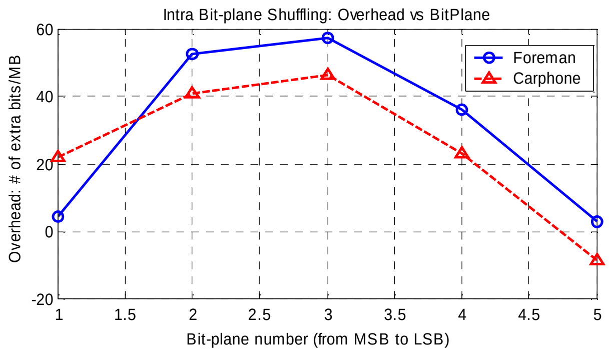


Figure 2.8: Overhead of each bitplane by intra-bitplane shuffling algorithm.

2.3.3 The Security of IBS

The security of intra bitplane shuffling is affected by how shuffle tables are generated and used. Shuffle tables can be generated from a cryptographically strong pseudorandom sequence using a classical linear-complexity algorithm [55]. Different shuffle tables should be used for different bitplanes, which forces attackers to resort to a brute-force search to simultaneously guess the permutation table used in multiple bitplanes. Similarly, the shuffle tables should be updated frequently without reuse. The amount of brute-force trials for finding the exact clear-text of an enhancement frame is proportional to

$$\prod_{i=1}^{N_{blk}} \prod_{j=1}^{N_{bp}} \binom{n}{n_1^{(i,j)}},$$

where $n_1^{(i,j)}$ is the number of "1"s in the j-th bitplane of the i-th block, and N_{blk} and N_{bp} represent the number of blocks and bitplanes, respectively. As an example, we consider recovering exactly the second bit-plane in the "*Foreman*" sequence. This bit-plane is important because adding it to the base-plus-MSB video improves the PSNR from 29dB to 33.4dB. In the second bit-plane, the average number of "1"s in each shuffled 8x8 block is about 7 in the "*Foreman*" sequence. Suppose each shuffled block has 7 bit of "1"s in a bit-plane with QCIF size, the number of brute-force trials an attacker has to perform is proportional to $\binom{64}{7}^{396}$, which is equivalent to guessing approximately 10^5 encrypted bits. In this situation, a brute-force attacker would rather guess the cryptographic key used in generating shuffling tables, which is 128 bits long.

Another security aspect is how many bit-planes should be encrypted in order to provide sufficient protection. From our study using a number of video sequences, we found that when the sign information and the first three bit-planes are unknown, adding lower bit-planes to based layer video will degrade the quality of based layer video both perceptually and in terms of PSNR. This is because without the higher bit-planes and the sign information, the lower bit-planes behave like random noise. A similar observation has also been introduced in the streaming video literature [12]. Another observation from the FGS literature is that the first three bit-planes, especially the second bit-plane, contribute most to the refinement of the quality [58]. Hence we believe that encrypting the first three bit-planes along with the sign information can provide sufficient protection for most multimedia applications.

2.3.4 Other Forms of Constrained Shuffling

Since shuffling auditory signals temporally or visual signals spatially can easily make the shuffled signal unintelligible, random shuffle among self-contained coding units (such as the macro-blocks in compressed video), has been a popular tool for multimedia encryption and appears in various forms [112, 94, 141, 127]. We refer to this encryption method as *coded block shuffling* (CBS). A major drawback for block shuffling alone lies in the fact that the information within a block is perfectly retained. An attacker can exploit the correlation across the blocks (such as the continuity of edges and similarity of colors and textures) and reassemble the shuffled blocks with a moderate number of trials [4]. The reassembly effort can be significantly smaller than the brute force search as many unlikely search directions are pruned [85]. Therefore, block shuffling alone is often not a secure encryption operation. We will incorporate block/macroblock shuffling as a complementary building block to our two proposed operations and explore their combinations in the design of an encryption system in Section 2.5.

2.4 Attack Modelling and Security Measurement

In this section, we introduce a notion of *multimedia-oriented security* to evaluate the security against approximation recovery. To illustrate this concept, we shall use the security of visual data as an example and propose two visual security scores. The principles behind these security scores can be extended to auditory and other types of multimedia.

2.4.1 Approximation Recovery

One simple and common way to evaluate the security is to count the number of brute-force trials in order to break the encryption, which is proportional to $\min\{|\text{ clear-text space }|, |\text{ key space }|\}$, where $|\cdot|$ denotes cardinality. Aside from the brute-force search, there are also notions of security that quantify the security of a system in terms of the amount of resources needed to break it [9] [10]. How-

[4]This is usually true when the block is large enough. For small blocks, such as block of size 2x2, the correlation information between blocks is hard to exploit.

ever, the traditional all-or-nothing situation in generic data security is not always appropriate for measuring the security of multimedia encryption [112, 133, 141]. Beyond the exact recovery from ciphertext, it is also important to ensure partial information that is perceptually intelligible is not leaked out from the ciphertext. Many forms of multimedia data, such as image, video, audio and speech, contain inherent spatial and/or temporal correlation. The encrypted multimedia content may be approximately recovered based on the syntax, context, and the statistical information known as *a priori* [70]. This is possible even when the encrypted part is provably secure according to some generic security notions. For example, in MPEG-4 video encryption [127], when motion vector fields are encrypted and cannot be accurately recovered, a default value 0 can be assigned to all motion vector fields. This approximation sometimes results in a recovered frame with fairly good quality for frames having a limited amount of motion. Additionally, the statistical information, neighborhood patterns, and/or smoothness criterion can help estimate an unknown area in an image [122] and automatically reorder shuffled image blocks [85]. Although these estimations may not be exact, they can reveal perceptually meaningful information once the estimated signal is rendered or visualized.

As an example, we have shown in Fig. 2.5 the experimental result of encrypting the DC prediction residue of the *Lena* image. Although the directly rendered version of the encrypted *Lena* image in Fig. 2.5 is highly obscured, an attacker can obtain edge information by setting the DCs to a constant and observing the resulting image shown in Fig. 2.9. We can see that the edge and contour of the approximated *Lena* image is clearly comprehensible, which suggests that it is necessary to encrypt other components in addition to DCs.

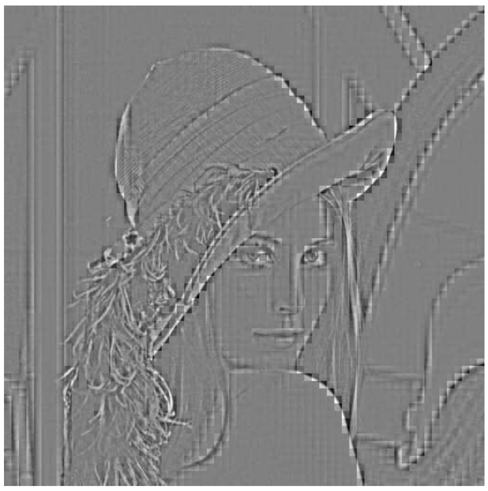


Figure 2.9: Approximated *Lenna* image by setting all DC coefficients to 0.

Since the value of multimedia content is closely tied with its perceptual quality, such value composition should be reflected in access control and confidentiality protection [66][67]. From the considerations presented above, we propose to evaluate the security of multimedia encryption using the following framework. After encryption, the encrypted media is first undergone some approximation attacks. We then use perceptual similarity scores to measure the amount of information leakage about the original media data through the approximated media. The results can indicate the security of the encryption scheme against the approximation attacks. Next, we discuss the methods and tradeoffs of measuring visual similarity for encryption applications.

2.4.2 Visual Similarity Scores

Studies on human visual system have shown that the optical characteristic of eyes can be represented by a low-pass filter [47], and that human eyes can extract coarse visual information in images and videos in spite of a small amount of noise and geometric distortion. The important information extracted by human visual system includes spatial-luminance information and edge and contour information [36]. Motivated by these studies, we design a luminance similarity score and an edge similarity score to reflect the way that human perceives visual information. These scores can quantitatively measure the perception-oriented distance between the clear-text copy of multimedia and the attacker's recovered copy from the encrypted media. The proposed scores are inspired by the recent work on automated image quality measurement [123][124] that incorporated human perceptual properties.

Luminance Similarity Score To capture the coarse luminance information, we introduce a block-based luminance similarity score. We assume that two given images are preprocessed to be aligned and scaled to the same size. These two images are first divided into blocks in the same way, using 8×8 or 16×16 non-overlapping blocks. Then the average luminance values of the i-th block from both images, y_{1i} and y_{2i}, are calculated. We define the luminance similarity score LSS as

$$LSS \triangleq \frac{1}{N} \sum_{i=1}^{N} f(y_{1i}, y_{2i}). \tag{2.9}$$

Here, the function $f(x_1, x_2)$ for each pair of average luminance values is defined as

$$f(x_1, x_2) \triangleq \begin{cases} 1 & \text{if } |x_1 - x_2| < \frac{\beta}{2}; \\ -\alpha \text{ round}(\frac{|x_1-x_2|}{\beta}) & \text{otherwise,} \end{cases}$$

where the parameters α and β control the sensitivity of the score. Since the images under comparison may be corrupted by noise during transmission or be mis-aligned by a few pixels, such noise and perturbation should be suppressed during similarity estimation. The resistance to minor perturbation and noise can be achieved by appropriately choosing the scaling factor α and the quantization parameter β . In our experiments, α and β are set to 0.1 and 3, respectively. A negative LSS value indicates substantial dissimilarity in the luminance between the two images.

Edge Similarity Score The edge similarity score measures the degree of resemblance of the edge and contour information between two images. After the images are partitioned into blocks in the same way as in the LSS evaluation, edge direction classification is performed for each block by extracting the dominant edge direction and quantizing it into one of the eight representative directions that are equally spaced by 22.5 degrees, as shown in Fig. 2.10. We use indices 1 to 8 to represent these eight directions, and use index 0 to represent a block without edge. Denoting e_{1i} and e_{2i} as the edge direction indices for the i-th block in two images, respectively, the edge similarity score (ESS) for a total of N image blocks is computed as follows:

$$ESS \triangleq \frac{\sum_{i=1}^{N} w(e_{1i}, e_{2i})}{\sum_{i=1}^{N} c(e_{1i}, e_{2i})}. \tag{2.10}$$

Here, $w(e_1, e_2)$ is a weighting function defined as

$$w(e_1, e_2) \triangleq \begin{cases} 0 & \text{if } e_1 = 0 \text{ or } e_2 = 0, \\ |cos(\phi(e_1) - \phi(e_2))| & \text{otherwise,} \end{cases}$$

where $\phi(e)$ is the representative edge angle for an index e, and $c(e_1, e_2)$ an indicator function defined as

$$c(e_1, e_2) \triangleq \begin{cases} 0 & \text{if } e_1 = e_2 = 0; \\ 1 & \text{otherwise.} \end{cases}$$

The score ranges from 0 to 1, where 0 indicates that the edge information of the two images is highly dissimilar and 1 indicates a match between the edges in the two images. A special case arises when the denominator in Eq. (2.10) is zero, which happens when both input images are "blank" without any edge. We assign an ESS score of 0.5 to this special case. In our experiments, the input images are partitioned into non-overlapping 8x8 blocks, and the Sobel operator is used for edge detection [47]. The dominant edge direction of a block is determined by a majority voting inside the block according to the number of pixels associated with each representative direction by the Sobel operator.

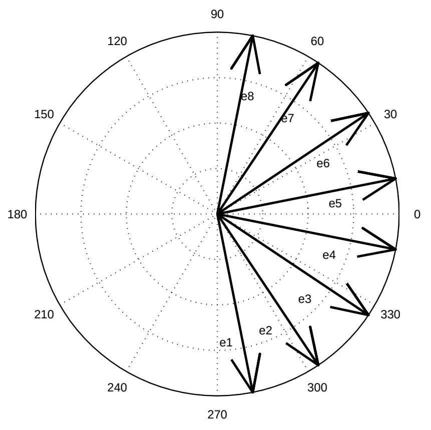


Figure 2.10: Eight representative edge directions used in ESS score evaluation

2.4.3 Evaluation Framework for Multimedia-Oriented Security

When evaluating the image similarity, we first calculate the ESS and LSS scores between the attacked/approximated image and the original image, and then compare the scores with two pre-determined thresholds, ESS_{th} and LSS_{th}, respectively. An encrypted image/video is said to *pass* the similarity test against a certain attack if both the ESS and the LSS are lower than the thresholds. In our experiments, we set ESS_{th} to 0.5 and LSS_{th} to 0.

The proposed similarity scores exhibit a tradeoff between capturing coarse semantic information and texture details of images. The sensitivity of the two similarity scores to image details can be controlled by the size of the block partition. When small blocks (e.g. 4×4) are used, a small amount of noise or geometric distortion can result in scores that indicates dissimilarity for two similar images. We refer to this type of mis-classification as a *miss*. From security point of view, such a miss would lead to a security breach. When blocks with larger size (e.g. 32×32) are used, the scores tend to identify some images as similar when their details are different. We refer to this type of mis-classification as a *false alarm*. As preventing information leak is the main concern in many access control applications, usually there are relatively stringent requirements on keeping misses as low as possible, while allowing to tolerate a moderate amount of false alarms. Given these considerations, we suggest using 8×8 or 16×16 blocks in block partition.

The two similarity scores are intended to measure the amount of information leakage through an attacked image. To this end, other types of perception-based similarity scores, such as robust image hashing [108], can also be incorporated to measure the image similarities, especially on the luminance similarity aspect. These hashing methods measure image similarity through the similarity of hash vectors using the normalized Hamming distance [78] and the receiver-operating-characteristics (ROC) formulation [108].

The design philosophy of the LSS and ESS scores can be extended to other types of multimedia, such as audio and speech. Similar to the design of audio hash [78], we can first segment an auditory signal into a set of temporal frames, and analyze the components in various frequency ranges from each frame. The results from each interested frequency range (such as low frequency corresponding to LSS, and high frequency corresponding to ESS) can be examined and compared to arrive at an auditory similarity measurement.

2.5 Application Example: Video Encryption System Design

In this section, we present a framework for a video encryption system that employs the building blocks proposed in this chapter and from the literature. Using this encryption system, several example configurations are presented and the encrypted videos are compared in terms of the security against brute-force and approximation attack, the friendliness to delegate processing, and the compression overhead.

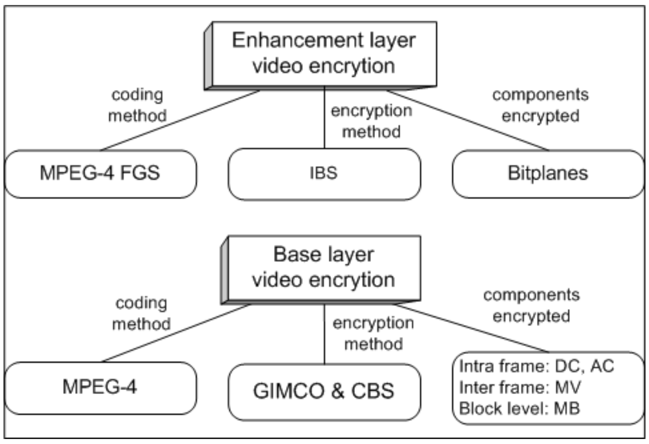


Figure 2.11: Video encryption system description. The encryption system is divided into two layers and for each layer candidate encryption components and methods are listed.

2.5.1 System Setup

Designed with scalable video in mind, the video encryption system has two layers as shown in Fig. 2.11. The base-layer video is coded with the MPEG-4 standard and the enhancement layer with the MPEG-4 FGS standard. The size of the group of pictures (GOP) is set to 15 and all predicted frames are set to P frames. For each layer, we provide candidate encryption methods and components to be encrypted. Our experiments are conducted on a Dell workstation with 1.8GHz Pentium IV CPU and 512MB RAM.

The two encryption operations proposed earlier in this chapter, namely, the generalized index mapping with controlled overhead (GIMCO) and the intra bitplane shuffling (IBS), lend themselves naturally as building blocks for this system. A third building block is the coded block shuffling discussed in Sec. 2.3.4. The index mapping encryption can be applied to intra block DC/AC coefficients and inter block motion vector (MV) residues, the intra bitplane shuffling encryption can be applied to FGS bitplanes, and the coded block shuffling can be applied to macroblock (MB) coding units. We use AES [26] with a 128-bit key to generate the pseudo-random numbers for all encryptions.

2.5.2 Bitrate Overhead Study for Index-Mapping Encryption

As discussed in Section 2.2, the index-mapping based encryption introduces bitrate overhead. The overhead can be controlled by carefully choosing the set of components to encrypt and by using the set partitioning technique. In this part we study the bitrate overhead under different encryption settings. A test video with 4000 frames in QCIF format is constructed by concatenating nine classic video clips, including *Carphone, Claire, Foreman, Grandma, Miss America, Mother-Daughter, Salesman, Suzie,* and *Trevor*. After an encryption range is chosen for each component, the range can be further partitioned into two subsets. For example, the encryption range $[-63, 64]$ for DC residue can be partitioned into $[-31, 32]$ and $([-63, -32] \cup [33, 64])$. We also tested to encrypt the first two non-zero AC coefficients along the zig-zag scanning order in each intra block. The encryption range of motion vector residue is $[-16, 15.5]$ with half pixel accuracy, which is also the MPEG-4 standard coding range. The encryption ranges of AC coefficients and MV residue can also be partitioned into subsets in a similar fashion as that of DC encryption. Since the MV's are predictively coded, the encryption of MV in one frame is able to propagate the scrambling effect to future frames, suggesting that encrypting MV of a subset of frames would be sufficient. For comparison, we include one experiment in which MV's from the first two P frames in a GOP are encrypted, and another experiment in which all MV's are encrypted. The latter serves as an upper bound for the bitrate overhead by MV encryption.

The above mentioned components are encrypted individually and the compressed file sizes are shown in Table 2.1. From Table 2.1, we can see that the compression overhead incurred by encrypting DC prediction residue ranges from 3% to 7%, depending on the encryption range and whether set partitioning is used. Encrypting AC and MV will generally incur an overhead larger than encrypting DC, especially when the motion vectors from all P frames are encrypted. However, limiting the encrypted MV's to the first two P frames in each GOP can reduce this overhead to around 5%. Also shown throughout Table 2.1 is that set partitioning is an effective way to control the overhead. These results provide a basis for designing a video encryption system, which is presented in the next subsections.

2.5.3 Base-Layer Video Encryption

In this part, we present experimental results for base-layer video encryption and analyze the security for different configurations. Four video clips, from fast-motion to slow-motion, are used in our experiment. They are the *Football*, the *Coast-*

Table 2.1: Index Mapping Encryption Overhead Comparison

Component Encrypted	*Encryption Range*	*Set Partition*	*File Size (MBytes)*	*Relative Overhead*
None	-	-	1.88	-
DC	[-127,128]	no	2.01	6.9%
DC	[-127,128]	2 sets	1.98	5.3%
DC	[-63,64]	no	1.98	5.3%
DC	[-63,64]	2 sets	1.94	3.2%
AC	[-64,64]	no	2.12	12.8%
AC	[-64,64]	2 sets	2.04	8.5%
AC	[-32,32]	no	2.04	8.5%
AC	[-32,32]	2 sets	1.99	5.9%
MV	[-16,15.5]	no	2.36	25.5%
MV	[-16,15.5]	2 sets	2.26	20.2%
MV of 2 P frames	[-16,15.5]	no	1.98	5.3%
MV of 2 P frames	[-16,15.5]	2 sets	1.96	4.3%

guard, the *Foreman*, and the *Grandma*, and each is 40 frames long. Encryption is performed under the settings shown in the left column of Table 2.2, where the encryption of DC, AC, and/or MV is based on the proposed generalized index mapping. The DC and AC encryption ranges are chosen as [-63,64] and [-32,32] with set partitioning, respectively; the motion vector encryption is applied to the first two P frames in each GOP. Additionally, macro-block shuffling in the compressed bitstream is applied to every frame. Each encrypted video complies with the syntax prescribed in the MPEG-4 standard. We also consider approximation attacks to these settings to emulate an adversary's action, and list them in the right column of Table 2.2. The security for these encryption configurations are discussed below in details.

Security Against Exact Recovery by Exhaustive Search

To accurately recover an original I frame from the encrypted one, an attacker needs to recover all the DC coefficients. For P frames, recovering the values of motion vectors is also necessary. In the above configuration, each DC coefficient and motion vector component has 6 bits encrypted. From the discussion in Section 2.4, each I frame in QCIF format has 2376 equivalent DC bits encrypted and the encrypted motion vectors in a P frame is equivalent to 1188 bits. Since a 128-bit key is used, the security against exact recovery by exhaustive search is determined by the cryptographic primitive with a 128-bit key.

Visual Security Against Approximation Recovery

To evaluate the visual security for our encryption system, we first encrypt the test video and then apply approximation attacks to the encrypted video. After that

Table 2.2: Encryption and attack settings for security analysis

Encryption System Settings	*Approximation Attack Settings*
(E1) encrypting intra block DC residue by index mapping;	(A1) set all intra block DC coefficients to 0;
(E2) encrypting inter block MV residue in the first two P frames of a GOP, and all intra block DC residues;	(A2) set all intra block DC coefficients to 0 and set the encrypted motion vector values to 0;
(E3) encrypting all the components in E2, plus the first two (in the zig-zag scan order) non-zero AC coefficients of intra block;	(A3) including all the approximations in A2, plus set the encrypted AC coefficients to 0;
(E4)–(E6) correspond to E1–E3 plus macro-block shuffling in the compressed bit-stream, respectively;	(A4)–(A6) the same as A1–A3, respectively.

we obtain the ESS and LSS scores of the approximated video and compare them with the thresholds, $ESS_{th} = 0.5$ and $LSS_{th} = 0$. An encryption is considered not secure enough when either score is above the corresponding threshold.

Fig. 2.12 and Fig. 2.13 show the video encryption results under different settings for the *Football* and *CoastGuard* clips. The results presented are for Y components as they carry most of the information about the video. Visual examination suggests that encrypting DC alone still leaks contour information after approximation attacks, while extending encryption to MV and/or some ACs helps diffuse the contour to reduce the information leakage. Furthermore, shuffling self-contained coding unit such as macroblocks, coupled with the above value encryption, can scramble the content to a completely unintelligible level. Table 2.3 lists the average

ESS and LSS (averaged over the total 40 frames) of the videos after approximation recovery. From the average LSS and ESS scores we can see that, when coded block shuffling is not used as an encryption tool, only the LSS score is below its security threshold of 0, and the ESS score is around or above its threshold of 0.5. This indicates that the encryption leaks out shape information and is not secure enough, which we can also observe from Fig. 2.12 and Fig. 2.13. Once the coded block shuffling is incorporated in the encryption, the ESS and LSS indicate that the encryption is secure against approximations. These results concur with the visual examination.

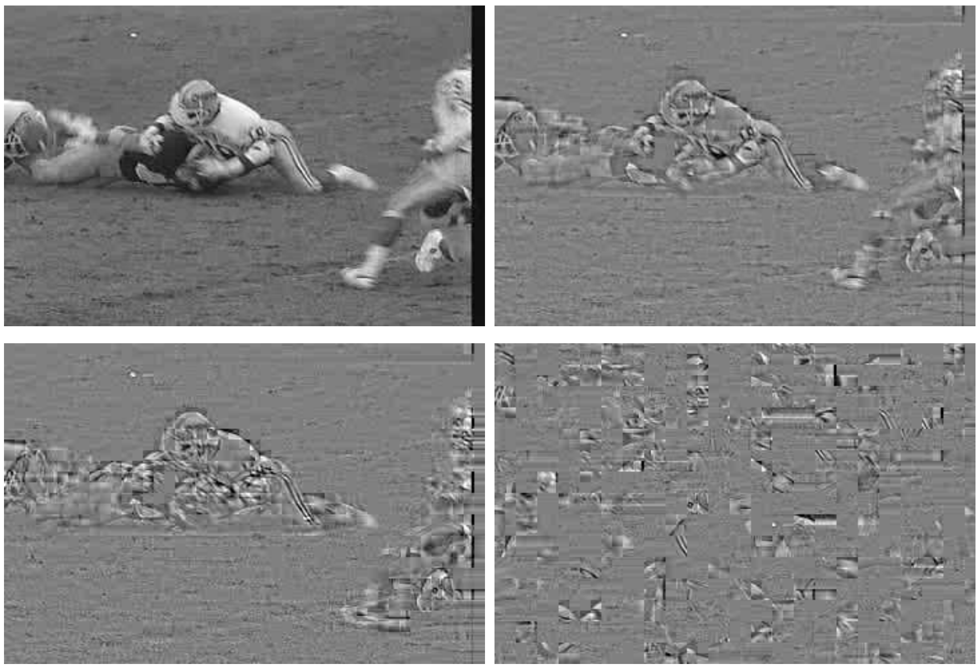


Figure 2.12: Encryption results for *Football*. Approximation attacks are performed after encryption. The encryption-approximation settings are: (top left) unencrypted; (top right) E1-A1; (bottom left) E2-A2; (bottom right) E5-A5.

To examine the detailed ESS scores, we plot the frame-by-frame ESS score of *Coast-guard* under different encryption-attack settings in Fig. 2.14. The top curve is from the attacked video with DC encrypted only, which confirms that encrypting DC alone still leaves some contour information unprotected. The two middle curves are the results involving MV encryption for inter-blocks and AC encryption for intra-blocks, where the ESS scores are low at the beginning of a GOP and increase substantially toward the end of the GOP. This is because as it approaches the end of a GOP, motion compensation becomes less effective and the compensation residue provides a significant amount of edge information. Such observation suggests that if we can only afford the bitrate overhead to encrypt two P frames in a GOP, the two encrypted P frames should be interleaved, such as choosing the 1st and the 8th P frames in a GOP of 15 frames. On the other hand, by incorporating the shuffling of macroblock coding units, the resulting ESS measurements are consistently around 0.1 or lower.

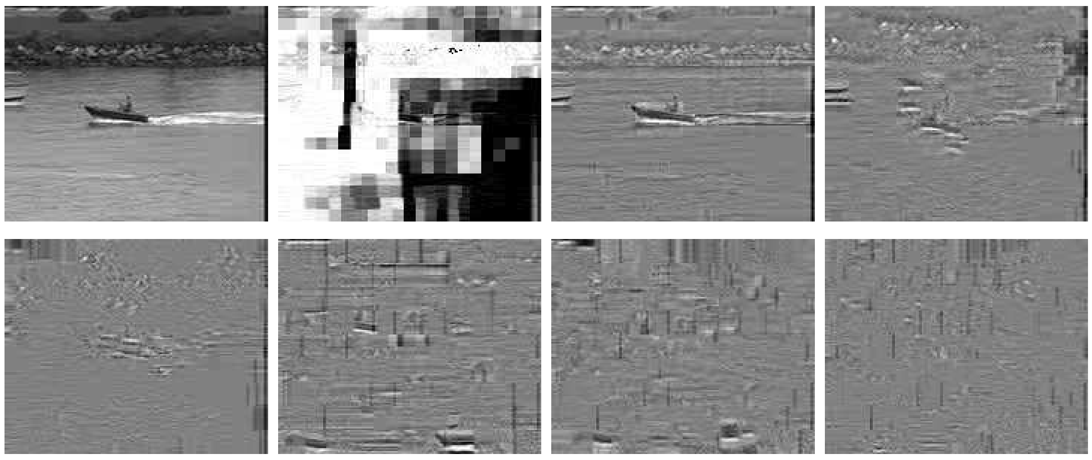


Figure 2.13: Encryption results for *Coast-guard*. The encryption-approximation settings are: (top row, left to right) unencrypted, E1, E1-A1, E2-A2; (bottom row, left to right) E3-A3, E4-A4, E5-A5, E6-A6.

Table 2.3: Perception based security measures for video encryption

	Football		*Grandma*		*Coastguard*		*Foreman*	
Settings	*ESS*	*LSS*	*ESS*	*LSS*	*ESS*	*LSS*	*ESS*	*LSS*
E1-A1	0.70	-0.78	0.64	-2.13	0.79	-1.18	0.71	-1.42
E2-A2	0.53	-0.85	0.46	-2.13	0.43	-1.19	0.43	-1.48
E3-A3	0.53	-0.86	0.30	-2.13	0.40	-1.20	0.40	-1.48
E4-A4	0.12	-0.93	0.05	-2.13	0.07	-1.20	0.07	-1.47
E5-A5	0.13	-0.92	0.05	-2.13	0.06	-1.21	0.06	-1.45
E6-A6	0.12	-0.92	0.04	-2.13	0.04	-1.20	0.05	-1.47

Relative Overhead

Table 2.4 lists the compression overhead for four videos under each encryption settings. We can see that the overhead is low for high-complexity, fast-motion video such as the *Football* and the *Foreman*, and relatively high for low-complexity, slow-motion video such as the *Grandma* and the *Coastguard*. As we go from the setting E1 to E3, more components are encrypted and thus the overhead increases. We also see that the coded block shuffling approach does not introduce overhead, as shown in setting E4 to E6. Overall, the overhead of 4–11% by the *E1, E2, E4,* and *E5* is comparable to that of a direct adaptation of generic encryption to multimedia as discussed in Section 2.1.1. Considering both security and compression overhead, we have found that the *E5* setting provides a very good tradeoff. This setting is a combination of block shuffling and selective value encryption via generalized index mapping.

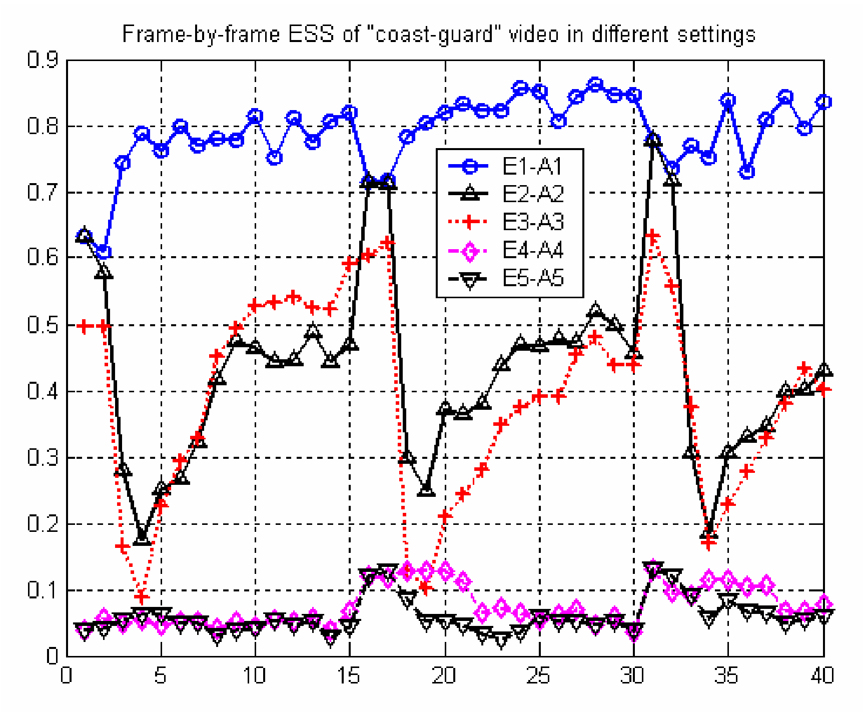


Figure 2.14: Frame-by-frame *ESS* of *Coastguard* video sequence under different settings. The corresponding settings are listed in Table 2.2.

2.5.4 Protecting FGS Enhancement Layer Video

We use 10 frames from the *Foreman* video sequence to demonstrate the protection of the enhancement data while preserving the FGS characteristics from the source coding. The proposed intra bitplane shuffling encryption is applied within each 8x8 block and the sign bit of each coefficient is encrypted using a stream cipher. To allow for a better visual examination of the protection effects on the enhancement data, we combine the encrypted FGS bitplanes with a clear-text base layer.

For most natural images, the coded DCT coefficients have decreasing dynamic range versus the frequency. As such, we emulate an approximation attack, whereby for each significant bitplane, all the “1”s of the bitplane is put to the lowest possible frequency bins. A total of six encryption-attack settings are used, namely:

Table 2.4: Relative Compression Overhead of the Encrypted Videos

	Football	*Foreman*	*Coastguard*	*Grandma*
E1 and E4	1.29%	1.75%	3.15%	6.96%
E2 and E5	3.88%	6.41%	8.74%	11.11%
E3 and E6	6.47%	9.62%	11.54%	24.61%

(a) to shuffle the first bitplane with clear-text base layer,

(b) to approximate the bitplane of (a) with the correct signs,

(c) to approximate the bitplane of (a) with random signs,

(d) to shuffle the first two bitplanes,

(e) to approximate the bitplanes of (d) with the correct signs,

(f) to approximate the bitplanes of (d) with random signs.

Fig. 2.15 shows the encrypted and attacked versions of the *Foreman* FGS video. The first row shows the encryption and approximation results using MSB only (settings (a), (b) and (c)), and the second row shows the results using the first two bit-planes (settings (d), (e) and (f)). The blocky artifacts in the settings (d), (e) and (f) are clearly more pronounced than in the settings (a), (b) and (c). This suggests that using more encrypted bit-planes will worsen the approximation attack results. Within each row, the visual difference is very subtle. This observation verifies that without knowing the decryption key to the enhancement layer, an attacker cannot obtain a more refined video than the base-layer video using approximations from the encrypted enhancement layer.

Table 2.5 lists the corresponding average PSNR, LSS and ESS of the videos under the six encryption-attack settings. From the table, we can see the approximation recovery can only reduce a little luminance error in terms of LSS and PSNR in the approximated video compared to the encrypted video, and the edge similar-

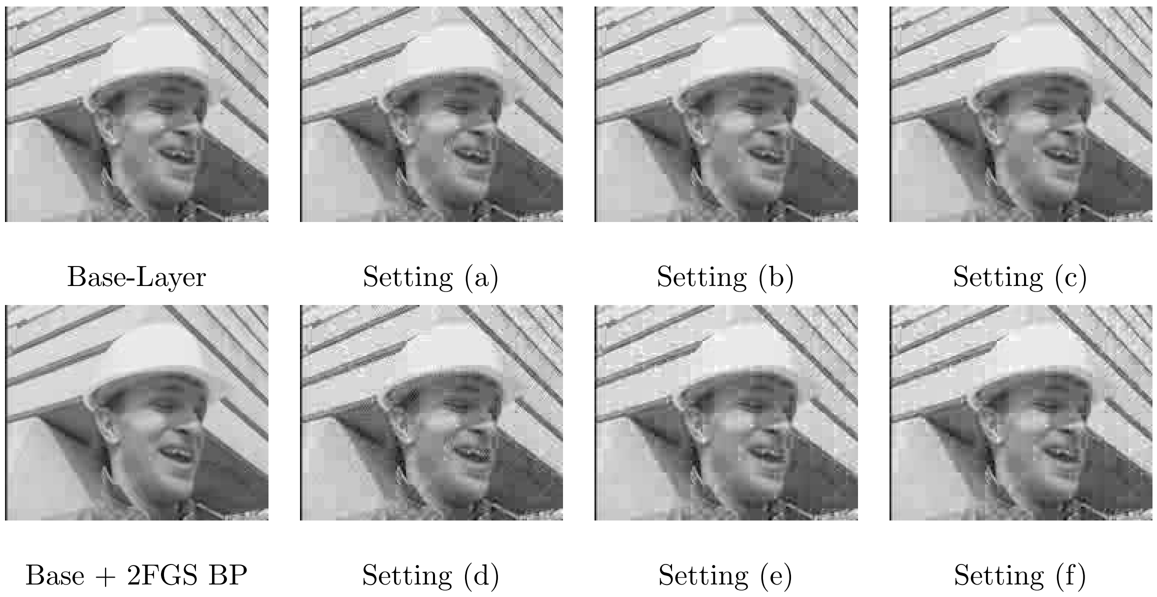


Figure 2.15: Encryption results for *Foreman* FGS video. Top row, left to right: base layer picture; and the encryption-attack settings (a), (b), (c). Bottom row, left to right: base layer plus 2 clear-text FGS bitplanes; and the encryption-attack settings (d), (e), (f).

ity in terms of ESS remains imperfect and has little improvement after attack. This demonstrates that the proposed method can protect the premium quality version of the content in a FGS compatible way, with a separate key from base-layer video encryption.

2.6 The Security in Media Authentication using Image Hashing

In addition to media encryption, multimedia content authentication is equally important in secure multimedia communications. In this section, we focus on the security of image hashing schemes for content authentication. we first adapt

Table 2.5: Intra Bitplane Shuffling and Approximation Attack

	(a)	(b)	(c)	(d)	(e)	(f)
PSNR (dB)	28.59	28.76	28.74	27.39	27.87	27.50
LSS	0.28	0.34	0.34	0.28	0.34	0.29
ESS	0.85	0.85	0.85	0.85	0.85	0.85

the unicity distance concept to evaluate the security of image hashing. Using the unicity distance framework, we analyze the security for two recently proposed image hashing schemes [110][117]. Our analysis shows that the secret hashing key can be disclosed with high accuracy when the key is re-used for several dozen of times. This result determines the maximum number of key reuse in the investigated hashing schemes. One way to mitigate such key disclosure is to generate a new hashing key using a master key for each different image.

2.6.1 An Analysis of Image Hashing

Most image hashing schemes consists of two steps: randomized feature extraction and postprocessing of image features. To achieve robustness and security, the feature extraction stage is particularly important. If the image feature can be generated without the secret key, an attacker can try to create a different image that produces the same image feature. Since this feature will result in the same image hash, the forged image becomes an artificial *pre-image* of the given hash. Hence, randomization techniques must be incorporated into the feature extraction stage to prevent hash forgery [37][110]. On the other hand, the robustness of image hashing greatly relies on the robustness of image features. A robust image feature extraction scheme must withstand minor distortions to the image, such as filtering, compression, noise corruption, and moderate geometric changes involving rotation, scaling and translation.

The post-processing of image feature involves quantization and compression of the feature. Further randomization, such as a permutation of the hash vector values, can be introduced in this step. However, to fully preserve the robustness obtained in image feature, the newly introduced randomness at this stage is usually limited [109]. For example, if a cryptographic cipher is applied on the image feature, the ultimate hash will dramatically change even if some feature values are changed by one bit of information. Various image hashing schemes mainly differ in the way randomized features are extracted. This is because usually similar post-processing technique can be applied to almost all randomized feature extraction schemes. For these reasons, we will concentrate on the feature extraction stage and assume the randomized image feature can be observed as the hash output. At the end of this paper, we will further discuss how to model and estimate the randomization in the post-processing stage.

Let us denote an image hash function by $h(\cdot)$, the input image by I, the secret key by K, and the resulting hash vector by $\mathbf{v}$. When the same key K is used to generate n image hashes, we observe pairs of $(I_1, \mathbf{v}_1)$, $(I_2, \mathbf{v}_2)$, $\cdots$, $(I_n, \mathbf{v}_n)$. The conditional entropy of the secret key K with respect to these observations is $H(K|\{(I_j, \mathbf{v}_j)\}_{j=1}^{n})$. According to information theory, in general, such conditional entropy will decrease with the increase of n. In our investigation of image hashing security, we mainly focus on how the uncertainty in the secret key decreases in robust image hashing schemes. Our exploration takes the following form. We randomly choose a hashing key, and generate multiple hashes using some natural images. The hash and image pairs are then provided to a key estimation algorithm.

This algorithm outputs an estimated hashing key. This estimation is gradually refined with the increased number of observed image-hash pairs. We expect to observe the estimated key becoming closer and closer to the actual key, until finally the two keys can be regarded functionally identical for hash generation purpose.

The "functional identicality" for two keys with only a few bit differences is a unique property in perceptual hashing schemes with robustness constraint. The secret key for most image hashing schemes can be considered as a random variable/vector. Due to the robustness constraint in constructing the image hash, for the same input image, adding a small perturbation vector to the key vector usually results in a hash that is similar to the original hash. Such an observation indicates that we can measure the distance d_K between the estimated the hashing key and the actual hashing key, and use d_K to predict the distance d_H between the resulting hash vectors that use these two keys and the same image as input. Conversely, we can also use d_H to predict the distance d_K. These two observations form the basis of the key estimation algorithms that we will present later. In the subsequent discussions, we use two kinds of vector norms to measure the distance in key vector and hash vector, namely, the L^1 norm and the L^2 norm.

Next, we analyze the security for two image hashing schemes. The first scheme was proposed by us in [110]. This scheme is one of the image hashing schemes with the best performance in terms of robustness and security. We are also the first to propose an analytical framework for quantitatively evaluating the hash security based on a single input image and output hash [110]. The second scheme we analyze was proposed by Venkatesan *et al.* in [117]. The authors of [117] are among the first to propose a perceptual hashing algorithm and to recognize that randomized feature generation is important to the robustness and security of image hashing. Analyzing these two representative hashing schemes can provide insight towards the security and limitation of robust image hashing algorithms.

2.6.2 Unicity Distance for Polar Fourier Transform Hash

Our recently proposed robust and secure image hashing scheme [110] is based on Fourier transform of image in polar coordinates. To obtain the hash, the FFT of an image is first taken, and the Fourier coefficients are represented in polar coordinates with zero frequency at the origin. Along the normalized magnitude coordinate ρ, m equally-spaced magnitude $\rho_1, \rho_2, \cdots, \rho_m$ are chosen ranging from 0 to 0.4. Then for each chosen magnitude ρ_i, Fourier coefficients are circularly summed to obtain one component in the feature vector, f_i. The circularly summed Fourier coefficients form a vector $\mathbf{f} = [f_1, f_2, ..., f_m]^T$. To randomize this feature vector, a linear transformation is performed on the vector $\mathbf{f}$ to obtain the randomized feature $\mathbf{r}$

$$\mathbf{r} = [r_1, r_2, \cdots, r_m]^T = \mathbf{K}\mathbf{f}.$$

Here the matrix $\mathbf{K}$ is a m-by-m key matrix. Each matrix element in $\mathbf{K}$ is a Gaussian distributed random variable with mean zero and variance one. We can also decompose this key matrix into m row vectors. Each row $\mathbf{k}_i^T$ is a component key with dimension 1-by-m. Thus we have

$$\mathbf{K} = [\mathbf{k}_1, \mathbf{k}_2, \cdots, \mathbf{k}_m]^T, \quad r_i = \mathbf{k}_i^T \mathbf{f}.$$

When n image-hash pairs are observed, we have $(\mathbf{f}_1, \mathbf{r}_1)$, $(\mathbf{f}_2, \mathbf{r}_2)$, $\cdots$, $(\mathbf{f}_n, \mathbf{r}_n)$. Define $\mathbf{F} = [\mathbf{f}_1, \mathbf{f}_2, \cdots, \mathbf{f}_n]$, and $\mathbf{R} = [\mathbf{r}_1, \mathbf{r}_2, \cdots, \mathbf{r}_n]$. We can obtain

$$\mathbf{R}_{m\times n} = \mathbf{K}_{m\times m}\mathbf{F}_{m\times n}. \tag{2.11}$$

There are two situations when trying to solve the above linear equation array. When $n \geq m$ and m linearly independent columns of $\mathbf{F}$ exists, we can extract the m columns of $\mathbf{F}$ as $\overline{\mathbf{F}}$ and the corresponding m columns of $\mathbf{R}$ as $\overline{\mathbf{R}}$. Now that $\overline{\mathbf{F}}$ is invertible, we solve the linear equation $\overline{\mathbf{R}} = \mathbf{K}\overline{\mathbf{F}}$ to obtain $\mathbf{K} = \overline{\mathbf{R}} \left(\overline{\mathbf{F}}\right)^{-1}$. If we ignore the numerical inaccuracy in the solution, such a solution will give an exact estimate of the secret key.

When $n < m$ or the columns of $\mathbf{F}$ does not constitute an invertible matrix, the problem in Eqn.(2.11) becomes solving an under-determined system and a number of existing solutions can be applied. For example, we can use the least-square method to obtain an estimate of the key matrix $\widehat{\mathbf{K}}$. Each row $\widehat{\mathbf{k}}_i^T$ in $\widehat{\mathbf{K}}$ corresponds to a hash component key. The L_1 and L_2 norm of the estimation error for the component key is $d_1 = ||\widehat{\mathbf{k}}_i - \mathbf{k}_i||_1$, and $d_2 = ||\widehat{\mathbf{k}}_i - \mathbf{k}_i||_2$, respectively. We expect both d_1 and d_2 to decrease with the increase of n. Since each column $\mathbf{f}_i$ of in matrix $\mathbf{F}$ is generated from a different image, we expect the matrix $\mathbf{F}$ to have full column rank with high probability. Usually when $n = m$, the key should be uniquely determined.

Experimental Validations

To verify our analysis, we conduct key estimation experiments using the method discussed above. We set $m = 64$ as in [110]. According to the statistics gathered from several hundred natural images, we generate the circular sum of Fourier transform coefficients at different radii values $\mathbf{f} = [f(\rho_1), f(\rho_2), \cdots, f(\rho_m)]^T$. For each randomly generated key vector $\mathbf{k}$, we perform key estimation using different numbers ($1 \leq n \leq 64$) of observed image-hash pairs. Then we compute the key estimation error $d_1(n)$ and $d_2(n)$. We repeat the above experiment for 5000 times and compute the average key estimation error as a function of n. In Fig. 2.16 we show that the key estimation error remains approximately constant when $1 \leq n \leq 40$. When $n > 40$ the error gradually decreases. When $n = 64$, the error becomes zero and we are able to obtain the exact key. This experiment shows that the unicity distance for polar Fourier transform hash is equal to $m = 64$.

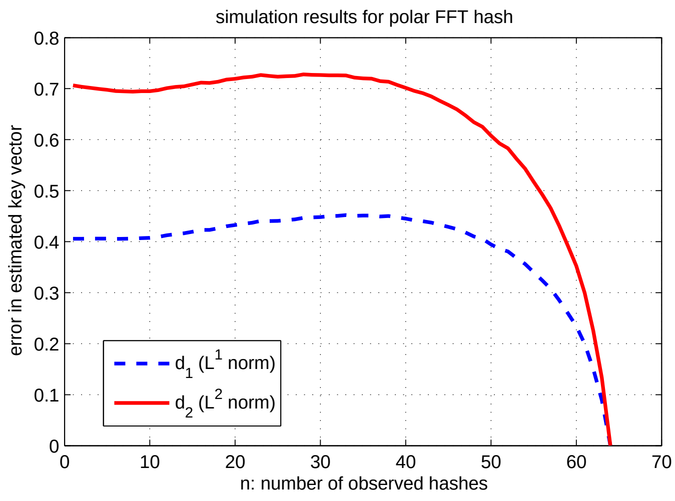


Figure 2.16: Average key estimation error for polar FFT hash. The results are averaged over 5000 runs.

2.6.3 Unicity Distance for Hash using Randomized Spatial Statistics

The second representative scheme on image hashing that we investigate was proposed by Venkatesan *et al.* in [117]. This hashing scheme uses randomized spatial statistics from input image as image feature. To generate one feature component, the secret key is used to produce a four-tuple parameters $p = (x, y, h, w)$ of a rectangle. Here (x, y) is the coordinate of the upper left corner of the rectangle, and h and w correspond to the height and width, respectively. With these parameters, a rectangular region in the input image is determined. The randomized spatial statistics can be either the mean or the variance of this region. To obtain a feature vector, the parameters for multiple random rectangles are generated. In this scheme, the equivalent secret key can be regarded as the four-tuple parameter set (x, y, h, w). Next, we show how to formulate the problem of estimating these parameters as a numerical optimization problem.

Suppose the mean (or variance) of luminance value of the selected image region is used as the randomized image feature as in [117]. We denote the process of computing the mean value as a function

$$v = h(p, I).$$

Here p indicates the secret key (rectangle parameters), I the input image, and v the regional statistics. Our goal is to estimate the secret parameter set p based on the observation of image-hash pairs. As the content of natural images are mostly smoothly varying, the function $h(p, I)$ varies smoothly with parameter p for any given I. If an estimated parameter $\hat{p}$ produces hash value $\hat{v} = h(\hat{p}, I)$, we can use the distance between hash value v and $\hat{v}$ as an indication of how accurate the estimate $\hat{p}$ is. For one image-hash pair, this may not be possible since many image regions will produce very close regional statistics v. When multiple hashes are generated using the same key but different images, we can observe (I_1, v_1), (I_2, v_2), $\cdots$, (I_n, v_n). Let $\mathbf{v} = [v_1, v_2, ..., v_n]^T$. The hash value obtained using the estimated key $\hat{p}$ is $\hat{\mathbf{v}} = [\hat{v}_1, \hat{v}_2, ..., \hat{v}_n]$, where $\hat{v}_i = h(\hat{p}, I_i)$. We formulate the key estimation problem as follows:

Given (I_1, v_1), (I_2, v_2), $\cdots$, (I_n, v_n), find $\hat{p}$ that minimizes the normalized correlation η between $\mathbf{v}$ and $\hat{\mathbf{v}}$, where

$$\eta = \frac{\mathbf{v}^T \hat{\mathbf{v}}}{||\mathbf{v}|| \cdot ||\hat{\mathbf{v}}||}. \tag{2.12}$$

To efficiently solve the above problem, we devised an iterative search algorithm that consists of two stages: the initialization stage and search refinement stage. We consider all the input images are of the same size; otherwise, a preprocessing step can be used to achieve this effect.

Table 2.6: Twelve sets of increments for parameter update

	1	2	3	4	5	6	7	8	9	10	11	12
Δ_x	$-d$	$-d$	0	d	d	d	0	$-d$	$-d/2$	$d/2$	0	0
Δ_y	0	$-s$	$-s$	$-s$	0	s	s	s	0	0	$-s/2$	$s/2$
Δ_w	0	0	0	0	0	0	0	0	$d/2$	$-d/2$	0	0
Δ_h	0	0	0	0	0	0	0	0	0	0	$s/2$	$-s/2$

Iterative Search Algorithm

Our proposed searching algorithm is similar in spirit to the hierarchical block matching algorithm used in video compression [90]. Different spatial regions are tested to find an initial rectangle that can approximate the actual rectangle area used in hash generation. Then the initial rectangle is successively refined through a hierarchical search to obtain the final rectangle estimation.

Initialization: The initialization stage is to obtain the approximate location and size of the rectangle. We partition the entire image area in several ways. The first partition is an non-overlapped partition into 4-by-4 equal size rectangles. The second is an 8-by-8 non-overlapped uniform partition. The third partition is a 4-by-4 overlapped partition, with the width and height of partition rectangle being 2/5 of the image width and height, respectively. The horizontal and vertical overlap between neighboring rectangles are 1/5 of the image width and height, respectively. These partitions give a collection of 96 initial sets of parameters. Next, for each set of initial parameters, we compute the regional statistics using all input images and the normalized correlation with the given image hashes as in Eqn.(2.12). The parameter set $t = (t_x, t_y, t_w, t_h)$ that achieves the minimum correlation among 96 partitions is chosen as the initial input to the next stage.

Iterations: In this stage, we update the rectangle parameter iteratively. Each iteration tries to update the existing parameter to obtain a smaller normalized correlation η. The parameters are updated by adding an increment or decrement as

$$x' = x + \Delta_x,\ y' = y + \Delta_y,\ w' = w + \Delta_w,\ h' = h + \Delta_h.$$

We use twelve sets of parameter increments in our algorithm, as shown in Table 2.6. These parameter increments correspond to translating the current rectangle in eight directions to its neighbor areas, combined with horizontal (or vertical) expansion (or shrinking) of the current rectangle. The incremental parameters d and s are obtained from the current rectangle parameters as

$$d = \theta w/2 \ \text{ and } \ s = \theta h/2.$$

Here θ is the step-size factor. We start with $\theta = 1$. If none of the updated parameters can result in a smaller normalized correlation η, we decrease θ by half and recompute the updated parameters. This process is repeated until we can find an updated parameter set that reduces η in (2.12). This iteration process is illustrated in Fig. 2.17.

Termination conditions: The iteration stage will terminate when one of the following three conditions are met. The first condition is that the normalized correlation approaches 1 within a pre-determined margin ϵ. The second condition is that when the incremental parameters d or s becomes smaller than 1, which indicates that we cannot find a better parameter value within one pixel accuracy.

The third condition is that the number of iterations reaches a pre-determined maximum number.

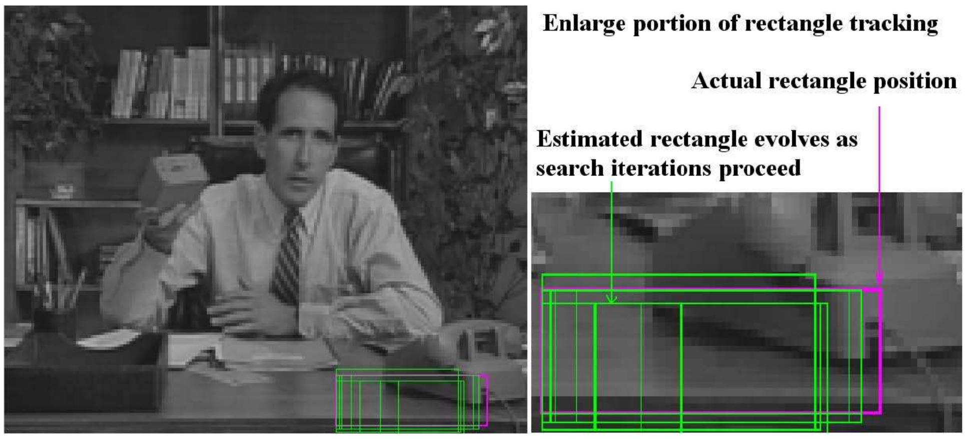

Enlarge portion of rectangle tracking
Actual rectangle position
Estimated rectangle evolves as search iterations proceed

Figure 2.17: Illustration of the searching algorithm for the unknown rectangle parameters

Experimental Validations

We have collected 4000 images from 11 video sequences in QCIF format (frame size 176×144) and apply the proposed key estimation test. In each experiment, we randomly choose n images from this collection, generate hashes for these images using a secret key, then perform key estimation. For each fixed n, we repeat the experiment 400 times using different keys and calculate the average estimation error in the rectangle parameters (e_x, e_y, e_w, e_h). Fig. 2.18 shows the absolute value of the average estimation errors for different n values. As we can see, the error reduces quickly as n increases. In Fig. 2.19 we also plot a histogram for the width estimation error when $n = 40$. We can see that about half of the estimation errors have absolute value smaller than 3. Within several pixels' range, the estimated rectangle will produce nearly identical hash values. Since most perceptual hash verification relies on a soft hash distance metric by measuring the normalized Hamming distance [79], the probability of forging a valid hash for any new image becomes very high. Therefore the unicity distance for the hashing using randomized spatial parameters is approximately around 40.

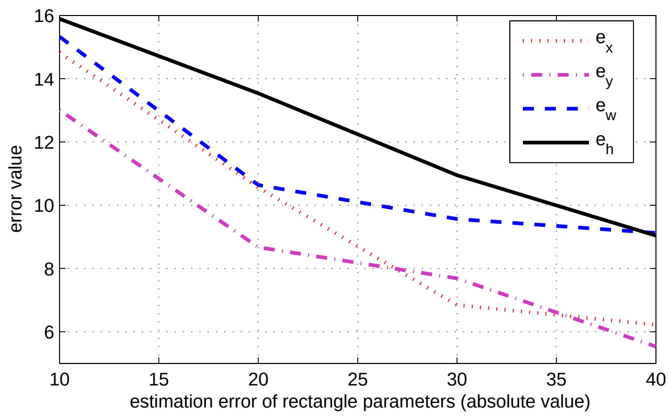


Figure 2.18: Key estimation results for image hash using regional mean.

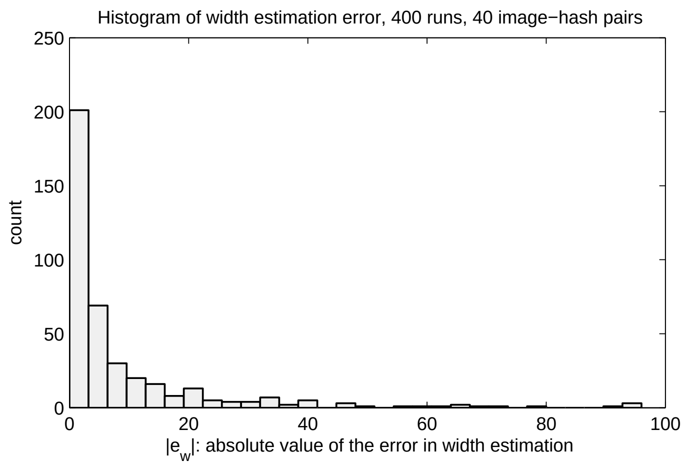


Figure 2.19: Histogram of the width estimation error.

2.6.4 Discussions

Our investigation in the security for robust image hashing shows that the secret hashing key, or its equivalent form, can be revealed with high accuracy when the key is re-used for a few dozen of times. We believe such key disclosure problem exists for all robust image hashing schemes, mainly due to the design limitation that perceptually similar images should have similar hashes, which is a drastically different criterion from cryptographic hash design. One way to mitigate such key disclosure is to use a new hashing key for each different image, because usually the unicity distance is more than 10 observed image-hash pairs. To reduce the increase in key storage cost, the hashing key can be generated by a key derivation function (a hash function) using a master key and a random number (the "salt") as input [52]. The salt value can be transmitted with the image content in clear-text.

We have demonstrated in this section how to employ the unicity distance concept in analyzing the feature extraction stage of image hashing. For operations in the post-processing stage of hash generation, such as the randomized quantization [79] and hash vector permutation, the concept of unicity distance can also be applied. In an unified framework, the randomization parameters in both the feature generation stage and post-processing stage can be searched at the same time. The exhaustive searching space may become very large. However, by exploiting the robustness constraint, incorrect searching directions can be quickly pruned and the effective search space will become much smaller. We believe through correct mathematical modelling and by using signal processing techniques, the proposed analytical framework based unicity distance concept can be extended to other robust perceptual hashing schemes.

2.7 Chapter Summary

In this chapter, we discussed the unique security issues related to multimedia encryption and content authentication. For media encryption, we proposed two atomic encryption operations that can preserve standard compliance and are friendly to delegate processing. We also provided quantitative analysis and demonstrated that a good tradeoff can be made between the security and bitrate overhead. We pointed out the need of quantifying the security against approximation attacks that are unique to multimedia, and have proposed a set of multimedia-oriented security scores to complement the security metrics for generic data. Using video as an example, we presented a systematic study on how to integrate different atomic operations to build a video encryption system that can achieve a good tradeoff among security, friendliness to delegate processing, and bitrate overhead.

For media authentication, we adapted the concept of unicity distance to evaluate the security of image hashing. Our explorations on two recent image hashing schemes shows that the secret hashing key can be disclosed with high accuracy when the key is re-used for a few dozen of times. One way to mitigate such key disclosure is to generate a new hashing key using a master key for each different image.

2.8 Appendix: Derivations

2.8.1 Overhead Analysis for Generalized Index Mapping

In this appendix section, we present the derivations for the overhead analysis of generalized index mapping as discussed in Section 2.2. We start with the a default entropy codebook being used to encode encrypted data and prove Eq. (2.3).

Recall that r denotes a probability distribution of $\{r_i \triangleq P(R=i) = 2^{-l_i}/\sum_k 2^{-l_k}\}$. Representing the constant in the denominator as c, we have $l_i = -\log r_i - \log c$. Expanding δL leads to

$$\begin{aligned}
\delta L &= \sum_i (q_i - p_i) l_i && (2.13)\\
&= -\sum_i (q_i - p_i)\log r_i - \sum_i (q_i - p_i)\log c && (2.14)\\
&= (-\sum_i q_i \log r_i + \sum_i q_i \log q_i) + \\
&\quad\ (-\sum_i q_i \log q_i + \sum_i p_i \log p_i) + \\
&\quad\ (-\sum_i p_i \log p_i + \sum_i p_i \log r_i) && (2.15)\\
&= D(q||r) + D(p||q) - D(p||r), && (2.16)
\end{aligned}$$

In the above derivation, the second summation in (2.14) is zero; and the second term in (2.16) is obtained by exploiting the piece-wise constant property of q such that $\sum_{i\in S_j} p_i = \sum_{i\in S_j} q_i = |S_j| q^{(S_j)}$, leading to

$$\begin{aligned} -\sum_i q_i \log q_i &= -\sum_j (\sum_{i\in S_j} q_i) \log q^{(S_j)} \\ &= -\sum_j (\sum_{i\in S_j} p_i) \log q^{(S_j)} \\ &= -\sum_j (\sum_{i\in S_j} p_i \log q_i) \\ &= -\sum_i p_i \log q_i. \end{aligned}$$

Using the same technique, we can prove Eq. (2.4) for the overhead in the adaptive entropy coding case. That is,

$$\begin{aligned} \delta L &= H_{\{q_i\}} - H_{\{p_i\}} \\ &= -\sum_i q_i \log q_i + \sum_i p_i \log p_i \\ &= -\sum_i p_i \log q_i + \sum_i p_i \log p_i \\ &= D(p||q). \end{aligned}$$

2.8.2 Overhead Analysis for Intra-bitplane Shuffling

In this appendix section, we analyze the zero-run symbol distribution after intra-bitplane shuffling as discussed in Section 2.3.

As we have seen in the main text, a macro-block in MPEG4 FGS coding consists of four 8×8 luminance blocks, and zero-run symbols are formed within each block. Intra bitplane shuffling can be done within a block or within a macroblock. For generality, we refer to the block to which shuffling is applied as *shuffling block* and denote its size as n; and similarly, the block to which run-length encoding is applied as *coding block* and its size n_B. In the case of shuffling within an 8×8 block, the two blocks are identical and $n = n_B = 64$; while for the case of shuffling within a macroblock and run-length encoding within an 8×8 block, we have $n_B = 64$ and $n = 256$. We assume that there are n_1 "1"s in the shuffling block, hence the total number of zero-run symbols is n_1 both before and after shuffling. We derive the zero-run symbol histogram conditioned on n_1.

We use an indicator function $I_k^{(d)}$ to denote the following event of the bitplane of interests for coefficients in a coding block: when ones appear in both kth and $(k+d+1)$th position and zeros in between (i.e. the bits from $(k+1)$th position to $(k+d)$th position forms a zero-run symbol with run-length d), $I_k^{(d)} = 1$; otherwise, $I_k^{(d)} = 0$. The range of k to be considered for $I_k^{(d)}$ is between 0 and $(n_B - d - 1)$, where $I_0^{(d)} = 1$ indicates the first zero-run symbol of the block has run d. The expected number of symbols with zero-run being d in a coding block can be obtained as the follows:

$$\begin{aligned}
E(N_{B,d}|n_1) &= E\left[\sum_{k=0}^{n_B-d-1} I_k^{(d)}|n_1\right] \\
&= \sum_{k=1}^{n_B-d-1} \Pr(I_k^{(d)} = 1|n_1) + \Pr(I_0^{(d)} = 1|n_1) \\
&= (n_B - d - 1) \times \frac{\begin{pmatrix} n-d-2 \\ n_1 - 2 \end{pmatrix}}{\begin{pmatrix} n \\ n_1 \end{pmatrix}} + \frac{\begin{pmatrix} n-d-1 \\ n_1 - 1 \end{pmatrix}}{\begin{pmatrix} n \\ n_1 \end{pmatrix}} \\
&= \prod_{i=0}^{d-1}\left(1 - \frac{n_1}{n-i}\right) \times \frac{n_1}{n-d} \times \left[n_1 - \frac{(n-n_B)(n_1-1)}{n-d-1}\right].
\end{aligned}$$

When $n = n_B$, the results can be simplified as

$$E(N_d|n_1) = \prod_{i=0}^{d-1}\left(1 - \frac{n_1}{n-i}\right) \times \frac{n_1^2}{n-d}.$$

Chapter 3

Tracing Malicious Relay in Cooperative Wireless Communications

Wireless communication systems have enjoyed a tremendous growth in the past decade [96]. Recently, cooperative communications is proposed as an emerging wireless communication paradigm that explores a new dimension of diversity to combat the unfriendly wireless communication environment. Such a new communication paradigm also introduces new information security issues, especially in the area related to detecting and verifying signals simultaneously transmitted from multiple cooperating nodes. In this chapter, we show that using signal processing techniques and cryptographic tools, we are able to trace the adversarial relay nodes who try to garble the relay signals in a cooperative communication system. This shows the promising potential of signal processing in cooperative wireless communication security.

Consider a wireless communication example where node A is transmitting to node B, and the direct link may be obstructed by large objects or hilly terrains. In such a scenario, two other nodes C and D that are near to node A and B can serve as relay nodes to improve the communication quality. In an cooperative communication system, each node may simultaneously be an information source, a destination, and a relay node. To achieve cooperation, the transmission of a node can usually be divided into two phases. The first phase is to transmit information generated from itself while listening to other source nodes. The second phase is to transmit relay information [56][99][48]. Among the strategies employed by the relay nodes, amplify-and-forward and decode-and-forward are two most common strategies [56]. In amplify-and-forward, the relay nodes simply boost the energy of the signal received from sender and re-transmit to the receiver. Such a strategy may also amplify the noise in the received signal at the relay nodes. In decode-and-forward, the relay nodes will perform physical layer decoding (demodulation plus signal detection) and then forward the decoding result to the destination. When multiple relays are available, more sophisticated relay strategy can be employed. For example, multiple relay nodes can serve as multiple transmit antennas to employ a space-time code [2][113] in transmitting the relay signals [57][99][100][48]. Such schemes can improve the communication quality by exploiting the space and time diversity.

While most of these prior arts focused on the communication aspects of such cooperative system, there have also been serious concerns regarding the cooperation motivation and security in such systems. The first concern is from autonomous ad hoc network perspective, where a centralized control does not exist and each node is an autonomous entity. In such a scenario, nodes may lack motivation to cooperate and behave selfishly, such as avoiding packet forward in order to preserve

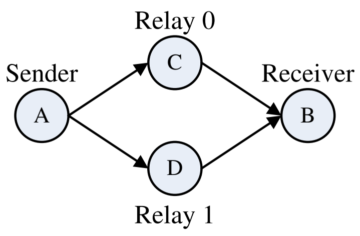


Figure 3.1: Two nodes, C and D, try to forward the message from node A to B.

its own energy and prolong its lifetime [139]. We classify such “selfish” behavior as *passive non-cooperation*. In the literature, passive non-cooperation have been addressed by using credit system [142], or reputation propagation system [77]. The second concern is from wireless communication perspective. Wireless communications use a shared medium and is susceptible to malicious denial-of-service attacks. For example, a powerful attack against wireless communications is signal jamming, where an adversary tries to corrupt communication by actively transmitting radio signal and interfering with the receiver [134]. We classify such attacks as *active malicious attack*, which is the focus of this paper.

In recent years, a number of works have addressed the problem of malicious signal jamming in general wireless communications. Xu *et al.* proposed a scheme to detect jamming attacks based on packet delivery ratio and signal strength [134], and proposed to use channel surfing and spatial retreat to avoid jamming attack [135]. Noubir *et al.* proposed to improve node connectivity and transmission reliability under jamming attack using directional antennas, node mobility, and error correction code [61][83]. Another traditional anti-jamming techniques is spread-spectrum, which spreads the energy of one bit information onto a wide frequency spectrum. Most of these counter-jamming techniques have one common attribute: they try to prevent the attacker from getting into the sender/receiver’s multiple access channel (MAC). In frequency division multiple access, the MAC channel can be defended by frequency hopping; in code division multiple access, the MAC channel is protected by each user's spread-spectrum code; and node mobility and the use of directional antennas provide a spatial multiple access channel that is designated to the source/destination in a particular spatial region. The result is that the jamming signal energy is significantly attenuated in the MAC channel, thus an improved signal-to-noise-ratio (SNR) can be obtained at the receiver.

The involvement of multiple relay nodes with sophisticated relay rules in cooperative communications poses a challenge to detecting and avoiding the malicious attacks [73]. The complication brought by user cooperation is that most schemes require nodes share their multiple access channels with other nodes [57][2][99][48]. This is because using an exclusive MAC channel between a sender and a receiver would make it difficult to benefit from node cooperation. However, in an adversarial environment, some relay nodes could be compromised. Using the captured communication device, the compromised nodes can maliciously modify the relay information, injecting falsified information, or choose not to relay. In such a situation, the counter attack techniques based on MAC channel exclusion will not be effective, and the security enforcement for cooperative wireless system becomes a challenging and delicate task.

In this paper, we are interested in the counter attack strategies against malicious relay nodes in a cooperative wireless system. The goal of the malicious relays is to corrupt the communications between the information source and the receiver. These malicious relay nodes would exploit the weakness in user cooperation rules, especially in the multi-node relay situation, and disguise themselves as legitimate relays. We first study the attack behavior and investigate the impact of such attacks to the receiver. To defend against such malicious attacks, we propose a cross-layer scheme for detecting and tracing adversarial nodes, which involves both inserting pseudo-random tracing markers in the transmitted symbol stream at the sender side, and an adaptive signal detection and processing at the receiver side. Note that in addition to detecting the jamming, our scheme can also pinpoint the adversarial relay node with very high probability. This allows further actions to be taken to exclude the adversarial node from the network.

3.1 System Model and Proposed Framework

3.1.1 System Setting

Let us consider the simple decode-and-forward situation introduced in Fig. 3.1. Sender node A transmit signals to both the receiving node B and relay nodes C and D. The relay nodes obtain the signal $\mathbf{y_r} = \mathbf{x} + \mathbf{n_r}$. Assuming the signal-to-noise-ratio (SNR) at the relay nodes are high enough, they decode the received signal symbol and normally should follow the forwarding rule. In this paper, we mainly consider relay nodes using coded cooperation, such as the user cooperation schemes described in [57][48] using space-time coding [2][113]. Nonetheless, the analysis about attack and defense can also be applied to non-coded forwarding strategy such as in [99][100], i.e., the relay nodes will directly transmit the received signal without coding.

As an example, we consider two relay nodes employing an orthogonal space-time code [2], described in Table 3.1, for message forwarding. After decoding, the relay nodes obtain the message bits. Suppose the signal constellation set $\mathcal{M}$ consists of $M = 2^k$ symbols. Each relay node takes two decoded symbols $\mathbf{s_0}$ and

Table 3.1: Space-Time Code Used by Two Relay Nodes

	relay node 0	relay node 1
t	$\mathbf{s_0}$	$\mathbf{s_1}$
$t+T$	$-\mathbf{s_1^*}$	$\mathbf{s_0^*}$

$\mathbf{s_1}$ from consecutive symbol durations, and transmit according to the cooperation rule in Table 3.1. Denote the symbol duration by T. At the receiving side, the received signals at two consecutive symbol durations t and $t+T$ are

$$\begin{aligned} \mathbf{r_0} &= \mathbf{h_0 s_0} + \mathbf{h_1 s_1} + \mathbf{n_0} \\ \mathbf{r_1} &= -\mathbf{h_0 s_1^*} + \mathbf{h_1 s_0^*} + \mathbf{n_1} \end{aligned} \tag{3.1}$$

Here $\mathbf{h_0} = \alpha_0 e^{j\theta_0}$ and $\mathbf{h_1} = \alpha_1 e^{j\theta_1}$ are complex channel gain, and $\mathbf{n_0}$ and $\mathbf{n_1}$ are complex Gaussian noise. Throughout this paper, we will use such base-band equivalent expression for analysis. We assume the channel conditions $\mathbf{h_0}$ and $\mathbf{h_1}$ are known at the receiver side, but not at the relay node. This is achieved by using proper channel estimation at the receiver side with pilot symbols that are inserted frequent enough relative to the channel variations [2].

3.1.2 Attack Modelling

We now examine the attack strategies employed by an malicious relay node against the cooperative communication system. The goal of the malicious relay node is to corrupt the communication between the sender and receiver. In the meantime, the attacker also tries to disguise his/her identity as a malicious adversary and to corrupt communication for as long as possible [39]. This can be done by cleverly choosing the attack technique and the portion of the relay signals to attack. Generally, we assume that the adversarial node uses the same transmission device as a normal relay node. The attacker will only corrupt communications when there is relay message. This strategy is to preserve the energy of the attacker's device as well as to disguise attacker's identity. When there is a relay message, the adversarial relay nodes can do one of the following: (1) transmit nothing, which is the passive non-cooperation; (2) transmit garbled signal, which is the active malicious attack. We discuss the details of these two cases subsequently.

Transmit Nothing

When both relay nodes are adversarial and choose not to transmit, such an event can be detected by comparing the received signal energy with ambient noise energy. Thus we shall focus on the situation when only one relay node (say, the relay node 1) is adversarial and do not forward message. When the relay node 1 does not transmit at all, the received signal from Eqn. (3.1) becomes

$$\begin{aligned}\mathbf{r_0} &= \mathbf{h_0 s_0} + \mathbf{n_0},\\ \mathbf{r_1} &= -\mathbf{h_0 s_1^*} + \mathbf{n_1}.\end{aligned}$$

This situation is also described in [2] as *soft failure*, i.e., one of the relay nodes fails to function. In this situation, the receiver still can detect the received signals. Here we describe the maximum likelihood detector, which is the optimal detector in Gaussian noise. First, the receiver builds the following two combined signals

$$\begin{aligned}\mathbf{c_0} &= \mathbf{h_0^* r_0} = \alpha_0^2 \mathbf{s_0} + \mathbf{h_0^* n_0}\\ \mathbf{c_1} &= -\mathbf{h_0 r_1^*} = \alpha_0^2 \mathbf{s_1} + \mathbf{h_0 n_1^*}\end{aligned}$$

It is clear that $\mathbf{c_0}$ only contains signal $\mathbf{s_0}$, and $\mathbf{c_1}$ only contains signal $\mathbf{s_1}$. Thus the optimal detector for $\mathbf{s_0}$ only depends on $\mathbf{c_0}$

$$\hat{\mathbf{s}}_\mathbf{0} = \mathbf{argmin}_{x_i \in \mathcal{M}} \left\{ (\alpha_0^2 - 1)|\mathbf{x_i}|^2 + d^2(\mathbf{c_0}, \mathbf{x_i}) \right\}. \tag{3.2}$$

Table 3.2: Forwarding Rules by Compromised Relay-1

	relay node 0	relay node 1
t	$\mathbf{s_0}$	$\mathbf{s_2}$ (instead of $\mathbf{s_1}$)
$t+T$	$-\mathbf{s_1^*}$	$\mathbf{s_3^*}$ (instead of $\mathbf{s_0^*}$)

Here $\mathcal{M}$ is the entire signal constellation set and $d(\cdot,\cdot)$ denotes Euclidean distance. The optimal detector for signal $\mathbf{s_1}$ takes the same form as (3.2), only $\mathbf{c_0}$ is replaced by $\mathbf{c_1}$.

From the above discussion we can see that, the attacker's strategy to transmit nothing will not prevent communications because the transmitted signals can still be detected. However, it will degrade the receiver's performance because of the loss of diversity.

Transmit Garbled Signal

Instead of transmitting the valid information, the adversarial relay node can arbitrarily change some signal symbols and transmit such garbled signals. In order not to be detected as an malicious attacker, the adversarial node will transmit the valid symbols from signal constellations and according to the rule stipulated by the space-time(ST) code. Without loss of generality, we analyze the damage of such attacks for the situation where the relay node 1 is adversarial. Table 3.2 describe the strategy used the relay node 1. In this table, the adversarial relay node randomly picks two signal symbols $\mathbf{s_2}$ and $\mathbf{s_3^*}$ to transmit, while the cooperative relay (node 0) transmits the information according to the relay message and the ST code. There are two difficulties at the receiver side. First, at the physical layer that uses the conventional signal detector, the receiver may not be able to correctly perform signal detection. Second, even if the receiver can separately decode the signals transmitted from relay node 0 and 1, only by looking at the physical layer symbols, it is very hard to tell which node is a friendly relay and which is a malicious attacker. It is possible either relay node 0 or 1 is garbling the transmitted signals, or even both.

Ambiguity in Conventional Signal Detector

To analyze the difficulty in signal detection using the conventional signal detector, we first review the detection rule for the space-time code in Table 3.1. When both relay nodes send signals according to the ST code, the conventional maximum likelihood signal detector is equivalent to minimizing the following quantity

$$\min_{x_i,x_j\in\mathcal{M}} \left\{(\alpha_0^2+\alpha_1^2-1)(|\mathbf{x_i}|^2+|\mathbf{x_j}|^2)+d^2(\mathbf{c_0},\mathbf{x_i})+d^2(\mathbf{c_1},\mathbf{x_j})\right\}. \qquad (3.3)$$

Here $\mathbf{c_0}$ and $\mathbf{c_1}$ are computed from the received signals in (3.1) and channel gain

$$\mathbf{c_0}=\mathbf{h_0^*r_0}+\mathbf{h_1r_1^*},\quad \mathbf{c_1}=\mathbf{h_1^*r_0}-\mathbf{h_0r_1^*}.$$

When the transmitted signal is garbled as in Table 3.2, the combined signals using the conventional decoding rule produces the following result:

$$\begin{aligned}\mathbf{c_0}^{(g)} &= \mathbf{h_0^*}(\mathbf{h_0s_0}+\mathbf{h_1s_2}+\mathbf{n_0})+\mathbf{h_1}(-\mathbf{h_0s_1^*}+\mathbf{h_1s_3^*}+\mathbf{n_1})^* \\ &= \alpha_0^2\mathbf{s_0}+\mathbf{h_1h_0^*s_2}-\mathbf{h_0^*h_1s_1}+\alpha_1^2\mathbf{s_3}+(\mathbf{h_0^*n_0}+\mathbf{h_1n_1^*})\end{aligned} \qquad (3.4)$$

Comparing to $\mathbf{c_0}=(\alpha_0^2+\alpha_1^2)\mathbf{s_0}+(\mathbf{h_0^*n_0}+\mathbf{h_1n_1^*})$, which is the combined signal without signal garbling, we can see that $\mathbf{c_0}^{(g)}$ contains the same noise signal $(\mathbf{h_0^*n_0}+\mathbf{h_1n_1^*})$, but the deterministic part has been significantly changed from $(\alpha_0^2+\alpha_1^2)\mathbf{s_0}$ to $(\alpha_0^2\mathbf{s_0}+\mathbf{h_1h_0^*s_2}-\mathbf{h_0^*h_1s_1}+\alpha_1^2\mathbf{s_3})$. Such garbling can easily lead to a detection error, as is shown in the following example.

Example: Suppose the signalling scheme is QPSK, and the information symbols are chosen as $\mathbf{s_0}=-\mathbf{s_3}$ and $\mathbf{s_1}=-\mathbf{s_2}$. Furthermore, consider that identical channel coefficients $\mathbf{h_0} = \mathbf{h_1} = \alpha e^{j\theta}$. From (3.4) we obtain the combined signal as

$$\begin{aligned} \mathbf{c_0}^{(g)} &= \alpha^2(\mathbf{s_0} + \mathbf{s_2} - \mathbf{s_1} + \mathbf{s_3}) + (\mathbf{h_0^* n_0} + \mathbf{h_1 n_1^*}) \\ &= 2\alpha^2 \mathbf{s_2} + (\mathbf{h_0^* n_0} + \mathbf{h_1 n_1^*}) \end{aligned} \tag{3.5}$$

Under the maximum likelihood detection rule, the receiver outputs the signal constellation that is closest to $\mathbf{c_0}^{(g)}$ in Euclidean distance. Hence the detection result will most probably be $\mathbf{s_2}$, while the actual signal sent is $\mathbf{s_0}$. This illustrates the ambiguity in the conventional signal detector when encountering a adversarial relay node.

3.1.3 Proposed Framework

There are two main challenges to distinguish the malicious relay from cooperative nodes at the information receiver. First, traditional cryptography at the application layer may be able to detect the attack, but cannot pinpoint the source of the attack. This is because the received signals from possibly multiple relays are superimposed with each other and corrupted by noise. At the physical layer, we need to separately detect the signal symbols from each of the relay paths. Second, we notice that such symbol-by-symbol detection is only feasible under certain conditions, and the detected symbol may have low reliability. Therefore it is necessary to aggregate the multiple symbol detection results for reliably distinguishing adversary from cooperator. At the same time, the receiver needs to obtain some side information about the "ground truth" of the relay signals. This side information is used to compare with the received signals for identifying the malicious relay.

We propose a cross-layer framework for tracing malicious relay, as shown in Fig. 3.2. The sender and receiver set up a secret key before sending messages. We refer to this key as the *tracing key*. This key is unknown to the relay nodes.

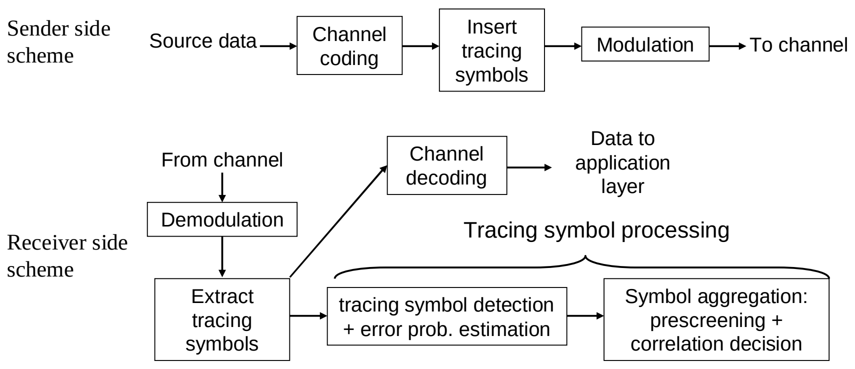


Figure 3.2: Schematic of the tracing scheme for malicious relay.

During communication, the information sender inserts a small number of pseudo-random signalling symbols at random locations in the symbol stream. We refer to the inserted symbols as the *tracing symbols*. Both the insertion location and the inserted tracing symbols are generated using a cryptographically secure function with the tracing key. Upon receiving the relay signals, the receiver uses the tracing key to find out the location of the tracing symbols, extract them, and apply signal detection. To process the detected tracing symbols, receiver also compute the "ground truth" of the tracing symbols using the tracing key and compare them with the detected tracing symbols from the relay path. Such a comparison can tell whether a relay node is adversarial or cooperative. The details of the tracing scheme consists of two parts: (1) how to detect the garbled tracing symbols, and (2) how to aggregate the detection results from multiple tracing symbols to achieve a reliable decision. These two components are presented in the next two sections.

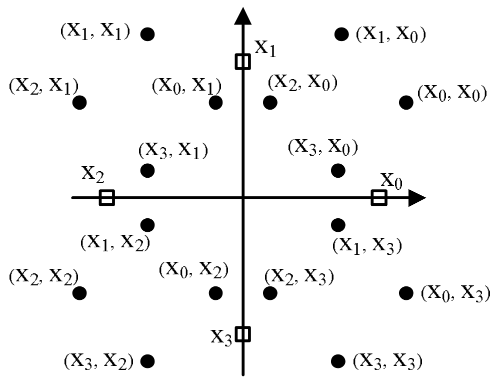


Figure 3.3: The pair-wise combined signal constellations. Notation $(\mathbf{x_i}, \mathbf{x_j})$ indicates the signal point is formed by the combination $\mathbf{h_0}\mathbf{x_i} + \mathbf{h_1}\mathbf{x_j}$, where $\mathbf{h_0} = 1/2$ and $\mathbf{h_1} = e^{j\pi/4}$.

3.2 Detecting Garbled Signals

3.2.1 Resolving Ambiguity Using One Receive Antenna

In this part, we discuss how the receiver with a single receive antenna can resolve the ambiguity in the garbled signal and estimate the information sent respectively by the two relay nodes. Consider the following example. The channel gain $\mathbf{h_0} = 1/2$ and $\mathbf{h_1} = e^{j\pi/4}$. QPSK is chosen as the signalling scheme with constellations $\mathcal{M} = \{\mathbf{x_0}, \mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3}\}$. The received signal, by combining the signals from two relay nodes and the additive noise, takes the form $\mathbf{r} = \mathbf{h_0}\mathbf{s_0} + \mathbf{h_1}\mathbf{s_1} + \mathbf{n}$, where $\mathbf{s_0}, \mathbf{s_1} \in \mathcal{M}$ and $\mathbf{n}$ is the complex Gaussian noise. If we are only concerned with the deterministic part of the signal, let $\mathbf{y} = \mathbf{h_0}\mathbf{s_0} + \mathbf{h_1}\mathbf{s_1}$ and we can see that $\mathbf{y}$ can take 16 distinct constellations in the complex signalling plane, as shown in Fig. 3.3. In this figure, the combined signal constellations are shown together with the original QPSK constellations.

Rewrite the received signal as $\mathbf{r} = \mathbf{y} + \mathbf{n}$. We can detect the signal $\mathbf{y}$ (corrupted in noise) as if the original signal contains a 16-point constellation. Under the complex Gaussian noise assumption, the maximum likelihood detector is equivalent to the minimum distance detector. Thus the detection result in the two dimensional signal plane is the signal constellation closest to the received signal $\mathbf{r}$.

Due to the smaller separation and geometrical irregularity in the combined signal constellations (Fig. 3.3), the probability of error in signal detection would naturally increase, and the closed form expression of the error probability cannot be obtained in general. Here we outline a procedure for detecting combined signal and computing the probability of error:

(a) Suppose the original signal constellation is $\mathcal{M}$ and the condition of the two channels are known as $\mathbf{h_0}$ and $\mathbf{h_1}$. We first find the combined signal constellations $\mathcal{Y} = \{\mathbf{y_k} : \ \mathbf{y_k} = \mathbf{h_0 x_i} + \mathbf{h_1 x_j}, \ \mathbf{x_i}, \mathbf{x_j} \in \mathcal{M}\}$.

(b) In the two-dimensional signal plane, find the Voronoi diagram $\mathcal{V}$ associated with the signal constellations $\mathcal{Y}$. The Voronoi cell V_i delimits the areas that are closer to a signal $\mathbf{y_i}$ than any other signal constellations. If the received signal $\mathbf{r} \in V_j$, the detection output is $\mathbf{y_j}$.

(c) Estimate the error probability P_e for each received signal $\mathbf{r}$. This procedure is presented in details below.

Error Probability in Combined Signal Constellation

Denote the probability that a signal $\mathbf{y_i} \in \mathcal{Y}$ is sent while $\mathbf{y_j} \in \mathcal{Y}$ is detected by $\Pr(\mathbf{y_j}|\mathbf{y_i})$. According to Bayes rule, the probability that an error occurred when $\mathbf{y_j}$ is detected is

$$\Pr(e|\mathbf{y_j}) = \frac{\Pr(\text{error and detect } \mathbf{y_j})}{\Pr(\text{detect } \mathbf{y_j})} = \frac{\sum_{\mathbf{y_i} \in \mathcal{Y}, i \neq j} \Pr(\mathbf{y_j}|\mathbf{y_i}) \Pr(\mathbf{y_i})}{\sum_{\mathbf{y_i} \in \mathcal{Y}} \Pr(\mathbf{y_j}|\mathbf{y_i}) \Pr(\mathbf{y_i})}. \tag{3.6}$$

We first simplify the above equation by noticing that each constellation point

$\mathbf{y_i} \in \mathcal{Y}$ is sent with equal prior in most practical applications, i.e., $\Pr(\mathbf{y_i}) = 1/|\mathcal{Y}|$ for all i. Second, if the Voronoi cells V_i and V_j are not neighbors, the probability $\Pr(\mathbf{y_j}|\mathbf{y_i})$ and $\Pr(\mathbf{y_i}|\mathbf{y_j})$ are most likely to be small. Such terms can be ignored in the estimation of (3.6). Third, if V_i and V_j are neighbors, then their shared boundary line must be perpendicular to and dissect the line segment from $\mathbf{y_i}$ to $\mathbf{y_j}$. Without considering the interference from other constellation points and under complex Gaussian noise, we can approximate $\Pr(\mathbf{y_j}|\mathbf{y_i}) \approx \Pr(\mathbf{y_i}|\mathbf{y_j})$. Such approximation is also true when the change in the probabilities $\Pr(\mathbf{y_j}|\mathbf{y_i})$ and $\Pr(\mathbf{y_i}|\mathbf{y_j})$ introduced by the geometry of other constellation points are small. This allows us to rewrite (3.6) as

$$\Pr(e|\mathbf{y_j}) = \sum_{\mathbf{y_i} \in \mathcal{Y}, i \neq j} \Pr(\mathbf{y_i}|\mathbf{y_j}). \tag{3.7}$$

In the above equation, $\mathbf{y_j}$ is the detected signal, and $\mathbf{y_i}$ runs through all constellation points. The conditional error probability in (3.7) can be computed using Monte-Carlo method in two steps.

Step-1: Generate samples $\mathbf{t} = \mathbf{y_j} + \mathbf{e}$, where $\mathbf{e}$ is drawn from noise distribution $\mathcal{N}$ with noise variance according to SNR. Use the detection algorithm proposed above to obtain the detected symbol $\mathbf{r}$ and compare them with the ground truth $\mathbf{y_j}$.

Step-2: Repeat Step-1 for N times and count the number of times with incorrect detection N_e. When N is large, the probability of error due to receiver side noise can be estimated by $P_e^N \approx N_e/N$.

Error Probability in a Single Relay Stream

Suppose the transmitted symbols by the two relays are $\mathbf{x_0}$ and $\mathbf{x_1}$, respectively. Denote the combined signal by $\mathbf{y} = \mathbf{h_0 x_0} + \mathbf{h_1 x_1}$. The detected combination sym-

Table 3.3: Channel Conditions with Two Receive Antennas

	relay node 0	relay node 1
Rx 0	$\mathbf{h_0}$	$\mathbf{h_1}$
Rx 1	$\mathbf{h_2}$	$\mathbf{h_3}$

bol $\hat{\mathbf{y}}$ corresponds to a pair of detected symbols $\hat{\mathbf{x}}_\mathbf{0}$ and $\hat{\mathbf{x}}_\mathbf{1}$. A detection error in $\hat{\mathbf{y}}$ can be due to one of the the two cases: (1) either $\hat{\mathbf{x}}_\mathbf{0}$ or $\hat{\mathbf{x}}_\mathbf{1}$ is incorrect, but not both; (2) both $\hat{\mathbf{x}}_\mathbf{0}$ and $\hat{\mathbf{x}}_\mathbf{1}$ are incorrect. We observe that situation (1) occurs with much higher probability than (2). In most cases, the two channel coefficients $\mathbf{h_0}$ and $\mathbf{h_1}$ have different magnitudes. Usually the signal symbol associated with the channel having a smaller-magnitude coefficient (or stronger attenuation) has a much higher chance to be incorrectly detected. We assume the relay channels are independent, and the probability for each relay channel to be attenuated more than the other channel is 1/2. Thus for two relay channels, we can approximate the error probability of the tracing symbols from a single relay stream as

$$p_e = P_e^N/2. \tag{3.8}$$

In the above detection algorithm, the detection complexity increases to $\mathcal{O}(M^2)$, where M is the number of signals in the original constellation. The detection rule has removed the time diversity originally in the ST code scheme. The error probability depends on the channel coefficients $\mathbf{h_0}$ and $\mathbf{h_1}$, which influence the geometry of the combined signal constellations.

3.2.2 Resolving Ambiguity Using Two Receive Antennas

When the receiving side has more than one receive antenna, the signal detection for garbled signals can take advantage of the additional antenna. We assume that

Table 3.4: Received Signals at the Receive Antennas

	Rx 0	Rx 1
t	$\mathbf{r_0}$	$\mathbf{r_2}$
$t+T$	$\mathbf{r_1}$	$\mathbf{r_3}$

the channel conditions between the two relay nodes and the two receive antennas are known at the receiver, as shown in Table 3.3, and the channel variation is negligible for adjacent symbol durations. The signals sent by the two relay nodes are according to Table 3.2. The received signals at the two time slots are shown in Table 3.4. The signals received at the first time slot are

$$\begin{cases} \mathbf{r_0} &= \mathbf{h_0 s_0} + \mathbf{h_1 s_2} + \mathbf{n_0}, \\ \mathbf{r_2} &= \mathbf{h_2 s_0} + \mathbf{h_3 s_2} + \mathbf{n_2}. \end{cases} \tag{3.9}$$

The signals received at the second time slot are

$$\begin{cases} \mathbf{r_1} &= -\mathbf{h_0 s_1^*} + \mathbf{h_1 s_3^*} + \mathbf{n_1}, \\ \mathbf{r_3} &= -\mathbf{h_2 s_1^*} + \mathbf{h_3 s_3^*} + \mathbf{n_3}. \end{cases} \tag{3.10}$$

We observe that only the signals $\mathbf{s_0}$ and $\mathbf{s_2}$ appear in (3.9), which corresponds to the received signals at the first time slot. Similarly, equation array (3.10) has the same structure as (3.9). From now on we will only focus on the signal detector for $\mathbf{s_0}$ and $\mathbf{s_2}$ in (3.9). The detector for $\mathbf{s_1}$ and $\mathbf{s_3}$ can be obtained similarly. Let us define

$$\begin{aligned} d_1 &= d(\mathbf{r_0}, \mathbf{h_0 x_i} + \mathbf{h_1 x_j}), \\ d_2 &= d(\mathbf{r_2}, \mathbf{h_2 x_i} + \mathbf{h_3 x_j}). \end{aligned} \tag{3.11}$$

It can be shown that under uncorrelated Gaussian noise, the maximum likelihood detector for $\mathbf{s_0}$ and $\mathbf{s_2}$ chooses signal constellations $\mathbf{x_i}, \mathbf{x_j} \in \mathcal{M}$ that minimizes the sum $(d_1^2 + d_2^2)$. Expanding $(d_1^2 + d_2^2)$ using (3.9) and (3.11) leads to

$$\begin{aligned}(d_1^2 + d_2^2) = &|\mathbf{r_0}|^2 + |\mathbf{r_2}|^2 + d^2(\mathbf{w_0}, \mathbf{x_i}) + d^2(\mathbf{w_1}, \mathbf{x_j}) \\ &- |\mathbf{w_0}|^2 - |\mathbf{w_1}|^2 - d^2(\mathbf{v}, \mathbf{x_i^* x_j}) + |\mathbf{v}|^2 + |\mathbf{x_i}|^2|\mathbf{x_j}|^2 \\ &+ (\alpha_0^2 + \alpha_2^2 - 1)|\mathbf{x_i}|^2 + (\alpha_1^2 + \alpha_3^2 - 1)|\mathbf{x_j}|^2.\end{aligned} \tag{3.12}$$

Here three auxiliary variables $\mathbf{w_0}$, $\mathbf{w_1}$, and $\mathbf{v}$ are defined as

$$\begin{aligned}\mathbf{w_0} &= \mathbf{h_0^* r_0} + \mathbf{h_2^* r_2}, \\ \mathbf{w_1} &= \mathbf{h_1^* r_0} + \mathbf{h_3^* r_2}, \\ \mathbf{v} &= \mathbf{h_0 h_1^*} + \mathbf{h_2 h_3^*}.\end{aligned} \tag{3.13}$$

Thus minimizing $(d_1^2 + d_2^2)$ is equivalent to minimizing the following quantity

$$\begin{aligned}T = &d^2(\mathbf{w_0}, \mathbf{x_i}) + d^2(\mathbf{w_1}, \mathbf{x_j}) - d^2(\mathbf{v}, \mathbf{x_i^* x_j}) + |\mathbf{x_i}|^2|\mathbf{x_j}|^2 \\ &+ (\alpha_0^2 + \alpha_2^2 - 1)|\mathbf{x_i}|^2 + (\alpha_1^2 + \alpha_3^2 - 1)|\mathbf{x_j}|^2.\end{aligned} \tag{3.14}$$

For PSK signals, the last three summation terms in (3.14) can be removed as their values are fixed. This reduces (3.14) to

$$T_{PSK} = d^2(\mathbf{w_0}, \mathbf{x_i}) + d^2(\mathbf{w_1}, \mathbf{x_j}) - d^2(\mathbf{v}, \mathbf{x_i^* x_j}). \tag{3.15}$$

The optimum detector has the following structure

$$(\mathbf{\hat{s}_0}, \mathbf{\hat{s}_2}) = \mathbf{argmin}_{(x_i, x_j) \in \mathcal{M}} T(\mathbf{x_i}, \mathbf{x_j}). \tag{3.16}$$

The complexity in computing T in (3.14) is $\mathcal{O}(M^2)$. This is because the signal norms $\{|\mathbf{x_i}|\}$ can be pre-computed. Computing $d^2(\mathbf{w_0}, \mathbf{x_i})$ and $d^2(\mathbf{w_1}, \mathbf{x_i})$ takes linear time (w.r.t. M). Only computing $d^2(\mathbf{v}, \mathbf{x_i^* x_j})$ takes $\mathcal{O}(M^2)$ time, which is the dominating factor in the detector complexity.

Similar to the case of one receive antenna, the symbol error probability in the case of two receive antennas can be obtained using Monte-Carlo method. Generally speaking, the symbol error probability of using two receive antennas is lower than that of using only one receive antenna. To see this, let us consider the combined signal constellation sets

$$\mathcal{Y}_1 = \{\mathbf{h_0 x_i} + \mathbf{h_1 x_j} \mid \mathbf{x_i}, \mathbf{x_j} \in \mathcal{M}\} \text{ and}$$

$$\mathcal{Y}_2 = \{\mathbf{h_2 x_i} + \mathbf{h_3 x_j} \mid \mathbf{x_i}, \mathbf{x_j} \in \mathcal{M}\}$$

from the two combining channels in (3.9). With one receive antenna, the detection errors are usually caused by a constellation point $\mathbf{u_1}$ that is close to $\mathbf{t_1}$ in $\mathcal{Y}_1$ with a small Euclidean distance $d(\mathbf{u_1}, \mathbf{t_1})$. Thus noise may drag the received signal closer to $\mathbf{u_1}$. Recall that $\mathbf{u_1}$ is a mapping from a pair of sent signals $(\mathbf{x_i}, \mathbf{x_j}) \in \mathcal{M}$ by channel coefficients $\mathbf{h_0}$ and $\mathbf{h_1}$. At the second receive antenna, the same $(\mathbf{x_i}, \mathbf{x_j})$ pair is mapped by uncorrelated channel coefficients $\mathbf{h_2}$ and $\mathbf{h_3}$ to a signal $\mathbf{u_2}$ in $\mathcal{Y}_2$. The probability that the distance from $\mathbf{u_2}$ to $\mathbf{t_2}$ ($d(\mathbf{u_2}, \mathbf{t_2})$) is also small will be low. Therefore when we have two receive antennas, the signal detection accuracy would naturally increase.

We have shown that with two receive antennas, the receiver is able to detect the potentially garbled signalling symbols from each relay node, with higher accuracy than using one receive antenna. The only "luck" required by such a signal detection scheme is that the channel condition matrix $\mathbf{H}$ is non-singular, i.e.,

$$\det(\mathbf{H}) = \det \begin{pmatrix} \mathbf{h_0} & \mathbf{h_1} \\ \mathbf{h_2} & \mathbf{h_3} \end{pmatrix} \neq 0.$$

Otherwise, it would be easy to see that the situation described in (3.9) reduces to the situation of only having one receive antenna as discussed in Section 3.2.1.

3.2.3 Generalization

In this part, we generalize the proposed signal detection approach to the scenario of having R relay nodes and K receive antennas. Denote the channel coefficient between the i-th receive antenna and j-th relay node by h_{ij}. At any given symbol duration, the transmitted signal symbol from the j-th relay node is denoted by $\mathbf{s_j}$. Since each relay node can potentially be adversarial, we do not impose any constraint on the signal that can be transmitted by a relay node. The receive signal at the i-th receive antenna is

$$\mathbf{r_i} = \sum_{j=1}^{R} \mathbf{h_{ij}} \mathbf{s_j} + \mathbf{n_i} \quad (1 \leq i \leq K). \tag{3.17}$$

The maximum likelihood detector for the signal $(\mathbf{s_1}, \mathbf{s_2}, ..., \mathbf{s_R})$ is

$$(\mathbf{\hat{s}_1}, \mathbf{\hat{s}_2}, ..., \mathbf{\hat{s}_R}) = \mathbf{argmin}_{(\mathbf{z_1}, \mathbf{z_2}, ..., \mathbf{z_R}) \in \mathcal{M}^R} \{ \sum_{i=1}^{K} d^2(\mathbf{r_i}, \sum_{j=1}^{R} \mathbf{h_{ij}} \mathbf{z_j}). \} \tag{3.18}$$

Define

$$\mathbf{w_j} = \sum_{i=1}^{K} \mathbf{r_i} \mathbf{h^*_{ij}}, \ \alpha_j^2 = \sum_{i=1}^{K} \alpha_{ij}^2, \ \text{ and } \mathbf{v}_{jt} = \sum_{i=1}^{K} \mathbf{h_{ij}} \mathbf{h^*_{it}} \ \ (1 \leq j \leq R, 1 \leq t \leq R).$$

After some algebraic manipulations, it can be shown that minimizing (3.18) is equivalent to minimizing

$$T = \sum_{j=1}^{R} d^2(\mathbf{w_j}, \mathbf{z_j}) + \sum_{j=1}^{R} (\alpha_j^2 - 1)|\mathbf{z_j}|^2 - \sum_{j=1}^{R-1} \sum_{t=j+1}^{R} \left\{ d^2(\mathbf{v_{jt}}, \mathbf{z^*_j} \mathbf{z_t}) - |\mathbf{z_j}|^2 |\mathbf{z_t}|^2 \right\}. \tag{3.19}$$

The formulation of the minimization objective function (3.19) has the advantage that it reduces the computation complexity in searching the detection output. Because $\mathbf{z_j}$ takes values only from the signal constellation set $\mathcal{M}$, there for a R-by-M table can be pre-computed for $d(\mathbf{w_i}, \mathbf{z_j})$, $(1 \leq i \leq R, \ 1 \leq j \leq M, \mathbf{z_j} \in \mathcal{M})$. The magnitude $|\mathbf{z_j}|^2|\mathbf{z_t}|^2$ $(1 \le j \le M,\ 1 \le t \le M,\ \mathbf{z_j}, \mathbf{z_t} \in \mathcal{M})$ can be pre-computed to obtain a M-by-M table. The term $d(\mathbf{v_{jt}}, \mathbf{x_1^*x_2})$ can be pre-computed for every possible combinations of $\mathbf{v_{jt}}$ $(1 \le j \le R-1,\ j+1 \le t \le R)$ and $\mathbf{x_1^*x_2}$ $(\mathbf{x_1}, \mathbf{x_2} \in \mathcal{M})$. Thus obtaining a table of size $\frac{R(R-1)}{2} \times M \times M$. Since computing each $\mathbf{v_{jt}}$ (or $\mathbf{w_i}$) term takes K multiplications, the total number of multiplications needed is about $(R^2M^2K/2)$. To search for the optimal $(\mathbf{z_1}, \mathbf{z_2}, ..., \mathbf{z_R}) \in \mathcal{M}^R$, a total of M^R possible combinations need to be computed. With the help of pre-computed tables, the rest of the computation cost would only be the cost for performing additions. If we directly compute the general formulation using (3.18), the total number of multiplications is about (M^RRK), which is significantly larger than $(R^2M^2K/2)$ when R is large. Therefore the efficiency of our proposed algorithm is superior than the brute-force computation of (3.18).

3.3 Cross-Layer Scheme for Tracing Adversarial Relay

The physical layer signal detection scheme, working on a symbol-by-symbol basis, has inevitably low reliability and detection accuracy. In this section, we present the adversary-tracing scheme outlined in Sec. 3.1.3 from a system perspective, which aggregates the detection results of a stream of tracing symbols, and significantly improves the accuracy of tracing the malicious relay.

3.3.1 The Tracing Algorithm

In a secure cooperative communication system, it is expected that the sender applies application-layer encryption and message authentication to protect data confidentiality and integrity. In our proposed scheme, the sender also insert pseudo-random tracing symbols into the data symbol stream. The inserted symbol values and locations can be computed using a cryptographic one-way function with the secret key as input. Such a scheme enables the receiver to recompute the ground truth of the tracing symbols and their locations.

At the receiver side, the receiver extracts the tracing symbols according to the signal detection algorithm presented in the previous section. Since there may be multiple relay nodes transmitting according to a cooperative relay scheme (either coded or non-coded), the detected tracing symbols and the ground truth for each relay stream should be treated separately. Next, we present the tracing scheme using aggregated information from multiple tracing symbols for one relay stream, and this scheme can be repeatedly applied to all the relay streams.

Denote the detected tracing symbols from one relay node at the physical layer by $[\mathbf{s_1}, \mathbf{s_2}, ..., \mathbf{s_n}]$, and the ground truth by $[\mathbf{t_1}, \mathbf{t_2}, ..., \mathbf{t_n}]$. Additionally, a confidence value p_i for each detected tracing symbol s_i, indicating the probability of correct detection for each tracing symbol is also provided. The algorithm for determining whether the relay node is cooperative or adversarial consists of the following steps:

(1) Pre-processing: Remove the detected tracing symbols $\mathbf{s_i}$ (and the corresponding ground truth $\mathbf{t_i}$) whose confidence value p_i is below a pre-determined threshold τ. Hence the two sequence may be shortened in this step.

(2) Symbol mapping: Assign binary Gray code to the two-dimensional signal constellations. After the mapping, use antipodal signal to represent the results, i.e., represent binary bit one by "1" and binary bit zero by "-1". Thus the mapping results are two sequences $[s_1, s_2, ..., s_m]$ and $[t_1, t_2, ..., t_m]$, whose elements take value in $\{-1, +1\}$.

(3) Correlation decision: Compute the normalized correlation

$$\rho = \frac{\sum_i s_i t_i}{\sqrt{\sum_i s_i^2}\sqrt{\sum_i t_i^2}}. \tag{3.20}$$

Then compare it with a threshold value η to make a decision

$$\text{decision} = \begin{cases} \text{cooperative} & \text{if } \rho \geq \eta; \\ \text{adversarial} & \text{if } \rho < \eta. \end{cases} \tag{3.21}$$

In the above algorithm, the pre-processing is to ensure that each tracing symbol involved in the final decision is reliable enough, i.e., with probability of correct detection $P_c \geq \tau$. The mapping from signal constellations to binary data is conventional in digital communication system [92]. Gray coding will ensure that the constellations that are close in Euclidean distance will be mapped to binary strings with small Hamming distance.

The above algorithm can be implemented sequentially, i.e., with each received tracing symbol, the correlation value ρ can be updated with a relatively low computation overhead. Suppose k reliable tracing symbols have been received and each tracing symbol maps to one binary bit (i.e. BPSK). Let $C_k = \sum_{i=1}^{k} s_i t_i$, $S_k = \sum_{i=1}^{k} s_i^2$, and $T_k = \sum_{i=1}^{k} t_i^2$. When the $(k+1)$-th tracing symbol arrives, the receiver updates $C_{k+1} = C_k + s_{k+1}t_{k+1}$, $S_{k+1} = S_k + s_{k+1}^2$, $T_{k+1} = T_k + t_{k+1}^2$, and computes the updated $\rho = C_{k+1}/\sqrt{S_{k+1}T_{k+1}}$. Thus as more tracing symbols arrive, the receiver can gradually improve the accuracy of the tracing scheme. Such a property will be discussed in the next subsection in details.

The main costs for such a tracing scheme include: (1) the computation at the receiver side; (2) the bandwidth cost by inserting the tracing symbols into the data stream, which incurs about 1-3 % overhead; (3) the cost of setting up the secret key for the tracing scheme, which can be done at the same time when setting up the application layer secret keys for using cryptographic tools. The mechanism for tracing the adversarial relay is only activated when the receiver detects abnormal behavior from the relay signals. For example, when application layer cryptographic authentication frequently would not pass or decryption results in meaningless data.

3.3.2 Analysis of the Correlation Statistics ρ

In this part, we analyze the statistical property of the correlation statics ρ. Consider N tracing symbols have been received, each with probability of correct detection $p_c = 1 - p_e$, where p_e is the single stream error probability in Eqn.(3.8). The result of symbol detection is independent for each symbol. This is because each tracing symbol is randomly chosen, and the time slot for transmitting each tracing symbol is sparsely and pseudo-randomly spaced. We assume that if an error happens during symbol detection, it is equi-probable to choose each of the $(2^m - 1)$ erroneous detection results, i.e., each with probability $1/(2^m - 1)$. We rewrite Eqn.(3.20) as follows

$$\rho = \frac{1}{N}\sum_{k=1}^{N} C_k, \quad \text{and} \quad C_k = \frac{1}{m}\sum_{i=1}^{m} S_i^{(k)} T_i^{(k)}. \tag{3.22}$$

Here $\{S_i^{(k)}\}_{i=1}^m$ is the binary expansion of the k-th tracing symbol, represented by antipodal signals ± 1, and $\{T_i^{(k)}\}_{i=1}^m$ is the corresponding ground truth generated using cryptographic tools with the secret tracing key. Since $\{C_k\}_{k=1}^N$ are i.i.d. independently distributed, we denote their common mean and variance by μ_c and σ_c^2. Thus we have

$$\mathrm{E}(\rho) = \mu_c \quad \text{and} \quad \mathrm{Var}(\rho) = \sigma_c^2/N. \tag{3.23}$$

Although ρ only takes discrete values, when N approaches infinity, the distribution of ρ will converge to a normal distribution according to central limit theorem. As the mean and variance would suffice to describe such distributions, we derive μ_c and σ_c^2 under two different hypothesis, i.e., whether the relay node is cooperative (H_0) or adversarial (H_1). To make the presentation concise, we will drop the index k in C_k, $S_i^{(k)}$, and $T_i^{(k)}$.

Mean and Variance Analysis

Hypothesis H_0 (Relay node is cooperative): When the relay node is cooperative, the received S_i will only differ from ground truth T_i due to noise corruption. When a detection error occurs, the bit sequence $\{S_i\}_{i=1}^m$ can differ from $\{T_i\}_{i=1}^m$ by flipping j bits, where $j \in \{1, 2, ..., m\}$. Since each of the $(2^m - 1)$ detection error occurs with equal probability, the probability that exact j bits are flipped from $\{T_i\}_{i=1}^m$ to $\{S_i\}_{i=1}^m$ is $p_j = \binom{m}{j}/(2^m - 1)$. Thus the mean of C is

$$\begin{aligned}\mathrm{E}(C|H_0) &= \frac{1}{m}\left\{(1-p_c)[\sum_{j=1}^{m}\frac{\binom{m}{j}}{2^m-1}(m-2j)] + p_c m\right\} \\ &= 1 - \frac{2^m}{2^m-1}(1-p_c).\end{aligned} \tag{3.24}$$

The variance of C can be obtained as

$$\mathrm{Var}(C) = \mathrm{E}(C^2) - [\mathrm{E}(C)]^2 \tag{3.25}$$

The second moment of C can be computed as

$$\begin{aligned}\mathrm{E}(C^2|H_0) &= \frac{1}{m^2}\left\{(1-p_c)\sum_{j=1}^{m}\frac{\binom{m}{j}}{2^m-1}(m-2j)^2 + p_c m^2\right\} \\ &= (1-p_c)\frac{2^m-m}{(2^m-1)m} + p_c\end{aligned} \tag{3.26}$$

Hypothesis H_1 (Relay node is adversarial): When the relay node is adversarial, the transmitted tracing symbol takes value from each of the 2^m constellations with equal probability of $(1/2^m)$ [1]. Thus the mean of C is

$$\mathrm{E}(C|H_1) = \frac{1}{m}\sum_{j=0}^{m} \frac{\binom{m}{j}}{2^m}(m-2j) = 0. \tag{3.27}$$

Again, the variance of C can be obtained from its second moment, which is

$$\mathrm{E}(C^2|H_1) = \frac{1}{m^2}\sum_{j=0}^{m} \frac{\binom{m}{j}}{2^m}(m-2j)^2 = \frac{1}{m} \tag{3.28}$$

Thus the variance of C under hypothesis H_1 is also $\frac{1}{m}$.

Since the variance of the correlation statistics ρ decreases with more received tracing symbols,the two hypothesis can be gradually separated, resulting in improved tracing accuracy. The threshold value η in (3.21) can be dynamically adjusted according to the number of tracing symbols N and the probability p_c.

Distribution Analysis

The close-form probability mass function of the tracing statistics ρ can be derived for some scenarios. We note that such a distribution can be expressed in a unified framework for both H_0 and H_1. Here we give an example when $m = 2$, i.e., the signal plane contains four constellation points. When $m = 2$, C_k in (3.22) can take three values $-1, 0$, and 1. Denote the probability that C_k takes value 1 by p_1, and that it takes value 0 by p_0. When N symbols are reliably detected, ρ can take discrete values from the set $\{-1, -1 + \frac{1}{N}, ..., 0, ..., 1 - \frac{1}{N}, 1\}$. Denote the number of C_k's with value 1 by N_1 and that with value -1 by N_{-1}, and that with value 0 by

[1]Here we have made another simplifying assumption. The adversarial relay can certainly choose to transmit the correct tracing symbol from time to time to avoid being identified. However, in order to interrupt communications, the adversary need to transmit at least a fair portion of garbled signals. The simplified analysis here can be utilized in subsequent discussions when we consider more sophisticated attacks and involving system issues such as channel coding.

N_0. When $\rho = \frac{k}{N}$, this means $N_1 - N_{-1} = k$. We thus can obtain the probability mass function accordingly. When $k \geq 0$:

$$\begin{aligned}\Pr[\rho = \frac{k}{N}] &= \sum_{t=0}^{\lfloor \frac{N-k}{2} \rfloor} \Pr(N_1 = k + t \; ; \; N_{-1} = t; \text{ and } N_0 = N - (k+2t)) \\ &= \sum_{t=0}^{\lfloor \frac{N-k}{2} \rfloor} \binom{N}{k+t} \binom{N-(k+t)}{t} p_1^{k+t} p_0^{N-(k+2t)} (1 - p_1 - p_0)^t \quad (3.29)\end{aligned}$$

When $k < 0$:

$$\Pr[\rho = \frac{k}{N}] = \sum_{t=0}^{\lfloor \frac{N-|k|}{2} \rfloor} \binom{N}{|k|+t} \binom{N-(|k|+t)}{t} p_1^t p_0^{N-(|k|+2t)} (1-p_1-p_0)^{|k|+t}. \quad (3.30)$$

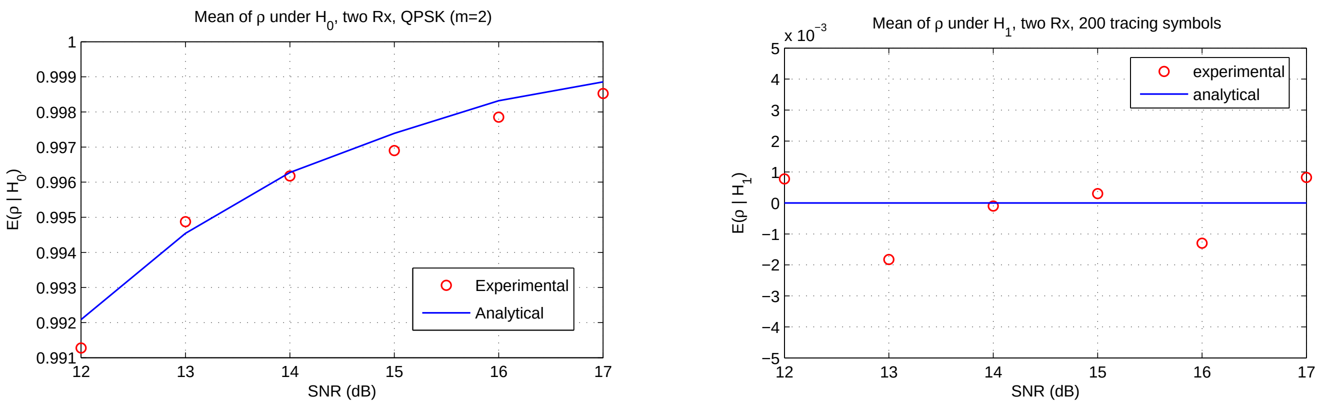
When applying this unified distribution to two hypothesis H_0 and H_1, the probability p_1 and p_0 can be adjusted. For example, under hypothesis H_0, $p_1 = p_c$ and $p_0 = 2(1 - p_c)/3$. Under hypothesis H_1, $p_1 = 0.25$ and $p_0 = 0.5$.

Validations and Discussions

In this part ,we experimentally validate the mean, variance, and distribution of ρ under hypothesis H_0 and H_1.

When the relay node is adversarial (hypothesis H_1), we experimentally compute the mean and variance of ρ using QPSK signalling. The results about mean and variance are shown in Fig. 3.4(b) and Fig. 3.5, respectively. From Fig. 3.4(b) we can see that the experimental mean approaches the analytical value 0; and in Fig. 3.5 we can see that the variance of rho decreases with the increase in the number of tracing symbols.

Approximating the mean and variance for ρ under hypothesis H_0 is a more intricate task. Our analytical mean and variance are obtained for any fixed value of p_c. In practice, ρ is computed from a large number of tracing symbols, each with different p_c values. Nonetheless, we observe that the analytical expression of



(a) Relay node is cooperative (H_0) (b) Relay node is adversarial (H_1)

Figure 3.4: The mean of ρ under hypothesis H_0 and H_1.

$\mathrm{E}(\rho|H_0)$ in Eqn. (3.24) is a linear function of p_c. This allows us to average the estimated p_c over a large number of tracing symbols to obtain the analytical value for $\mathrm{E}(\rho|H_0)$. We compare the analytical estimation obtained using Eqn. (3.24) and the experimental results in Fig. 3.4(a). Fig. 3.4 shows that, for SNR from 12 dB to 17 dB, our analytical approximations closely fit the experimental results, with an error margin on the order of 10^{-3}.

We plot the analytical distribution of ρ in Fig. 3.6 for $N = 100$. To validate our analysis that the distribution of ρ approaches Gaussian as N becomes large enough, we also performed χ^2 test to quantitatively measure how close the distribution of ρ is with respect to a Gaussian distribution. The results in Table 3.5 indicates that with the probability of miss equal 0.1, we can classify ρ as a Gaussian random variable. Such distributions of the tracing statistics ρ under two hypotheses clearly indicate that we are able to distinguish the cooperative relay from malicious ones with high accuracy.

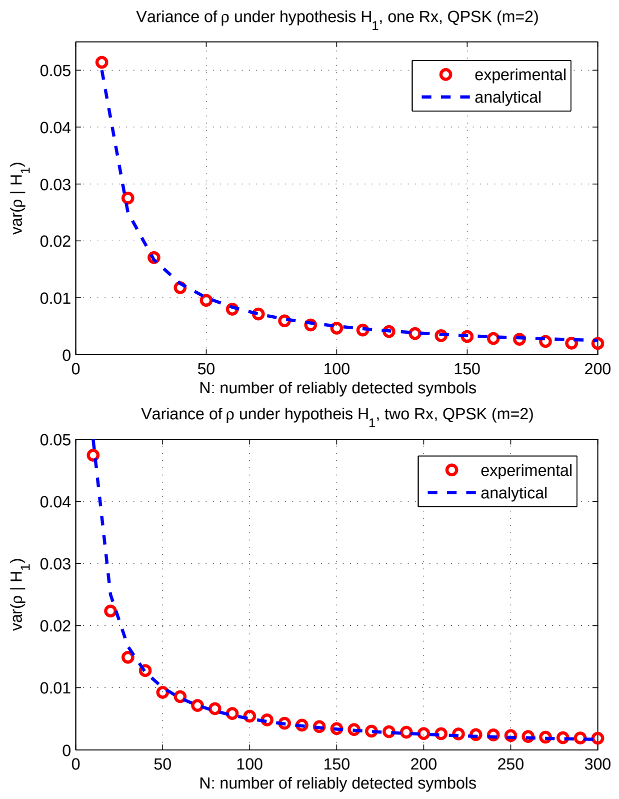


Figure 3.5: The variance of tracing statistics ρ under 10 dB SNR.

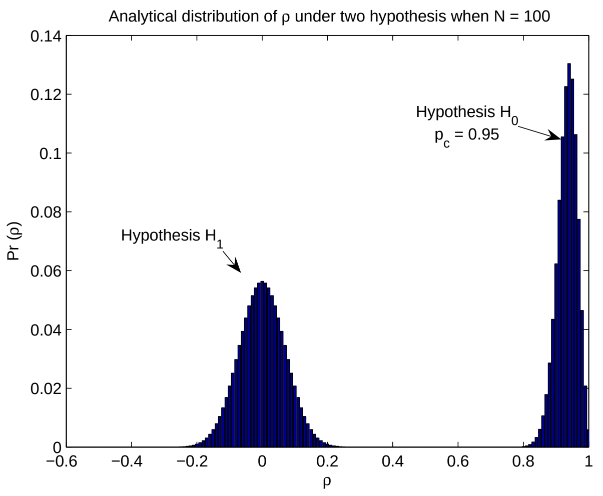


Figure 3.6: The distribution of tracing statistics ρ.

Table 3.5: χ^2 fitting statistics of ρ under H_1

N	20	40	60	80	100	120	140	160
One Rx	3.9847	3.2885	3.3828	4.1831	1.7825	1.2650	4.3765	3.4841
Two Rx	4.2556	6.0017	5.2737	0.4385	1.6241	1.3392	1.7076	0.5940

Note: $\chi^2 < \chi^2_\alpha$ indicates the experimental data fits the analytical Gaussian distribution with *miss probability* α.

Degree of freedom = 5; $\chi^2_{0.05} = 11.07$; $\chi^2_{0.1} = 9.236$.

3.3.3 A Randomized Attack Strategy

Since adversarial relays have every incentive to avoid being caught, a more sophisticated strategy is to "comply" with the transmission rule for part of the symbol stream and randomly jam the rest of the symbols, as long as the goal of corrupting communications can be achieved. Such a strategy results in a correlation statistics (from the partially garbled tracing symbols) that consists of two parts:

$$\rho = \left[\sum_{k=1}^{N_1} C_k^{(c)} + \sum_{l=1}^{N_2} C_l^{(g)}\right] / (N_1 + N_2) \tag{3.31}$$

Here N_1 is the number of symbols that complies with the ground truth and N_2 is the number of garbled symbols. Superscript $^{(c)}$ and $^{(g)}$ indicate whether a symbol is compliant or garbled. Let $N = N_1 + N_2$ and it is straightforward to decompose (3.31) into two parts

$$\begin{aligned} \rho &= \frac{N_1}{N}\left[\sum_{k=1}^{N_1} C_k^{(c)}/N_1\right] + \frac{N_2}{N}\left[\sum_{l=1}^{N_2} C_l^{(g)}/N_2\right] \\ &= \frac{N_1}{N}\rho^{(c)} + \frac{N_2}{N}\rho^{(g)} \end{aligned} \tag{3.32}$$

We can see that $\rho^{(c)}$ follows the distribution in Hypothesis H_0 and $\rho^{(g)}$ that in H_1. It can be seen that the mean (resp. variance) of the overall correlation statistics ρ are weighted combinations from the means (resp. variances) of $\rho^{(c)}$ and $\rho^{(g)}$. Therefore, in general, the mean of such an attacked ρ would increase and the variance would decrease. The increase in the mean results in a smaller "gap" between the attacked statistics and the complied statistics. However, the gap cannot be indefinitely small in practice. This is because the adversary's primary goal is to corrupt communications. When channel coding are applied before symbol formation, if the attacker only garbles a very small portion of the symbol stream, the garbled information would be recovered by the FEC at the receiver side. The minimum portion of the symbol stream that the attacker needs to garble in order to corrupt communications is thus determined by the channel coding rate. Suppose the channel code can correct 1/4 of the symbol errors on average, the attacker needs to corrupt at least 1/4 of the symbol stream, i.e., $N_2/N = 1/4$ in (3.32). Using QPSK signalling $m = 2$ and with $p_c = 0.9$, we can estimate the mean and variance of tracing statistics as

$$\mathrm{E}(\rho) = 0.65 \quad \text{and} \quad \mathrm{Var}(\rho) = 0.261/N.$$

3.3.4 Effects of Channel Estimation Error

In the previous discussions, we have assumed that the exact channel information can be obtained at the receiver side through channel estimation. In practice, the channel estimation will deviate from the actual channel information. This require that the tracing scheme should be adapted according to such deviations. In this part, we examine the effect of channel estimation error on the tracing statistics ρ.

The existing literatures on wireless channel estimation [32][59][98] indicate that with training symbols, the normalized mean square error (NMSE) of the channel estimation can approach the Cramer-Rao Bound (CRB) at high receiver SNR, or when the number of training symbols is large enough. With a moderate number of training symbols and high enough receiver SNR, we assume that the NMSE of channel estimation is equal to the negation of the receiver SNR. We note that such a result can be obtained using various channel estimation methods, such as the Minimum Mean Square Error (MMSE) method [59], the least square (LS) method [32], or the Maximum Likelihood (ML) method [98]. For example, if the receiver side has a SNR of 14 dB, the NMSE of channel estimation will be about -14 dB.

The deviation in channel estimation will result in a deviation in estimating the combined signal constellations, which is used to perform symbol detection. Since the energy of the error is relatively small compared to the energy of channel coefficients, the estimated signal constellations $\mathcal{Y}_E$ can be seen as a slightly shifted and rotated version of the true signal constellations $\mathcal{Y}$. This effect is illustrated in Fig. 3.7, where the actual signal constellations is drawn by $\times$, and the deviated one by $\circ$. To see how a deviated signal constellation will affect symbol detection, we consider two received signal points, C and D, shown in the figure. Point C is generated by sending signal point A_1 with additive noise. Using minimum distance detection, C is detected as point B_1 in the estimated constellations. Since $B_1 \in \mathcal{Y}_E$ and $A_1 \in \mathcal{Y}$ correspond to the same set pair of transmitted symbols in $\mathcal{M} \times \mathcal{M}$, such a result is a correct detection. However, for received signal point D generated by A_2 and detected as B_1, the detection result is incorrect, because the constellation point in $\mathcal{Y}_E$ corresponding to $A_2 \in \mathcal{Y}$ is B_2.

When the channel estimation error is considered, the received signal model can be written as

$$\mathbf{r} = (\mathbf{H} + \mathbf{n}_H)\mathbf{s} + \mathbf{n}_T = \mathbf{H}\mathbf{s} + \mathbf{n}_T + \mathbf{n}_H\mathbf{s}, \tag{3.33}$$

where $\mathbf{n}_H$ is the uncertainty in channel estimation and $\mathbf{n}_T$ represents receiver thermal noise. It can be seen that the second noise term $\mathbf{n}_H\mathbf{s}$ is proportional to the signal power $|\mathbf{s}|$. Thus we refer to the noise introduced by channel estimation error

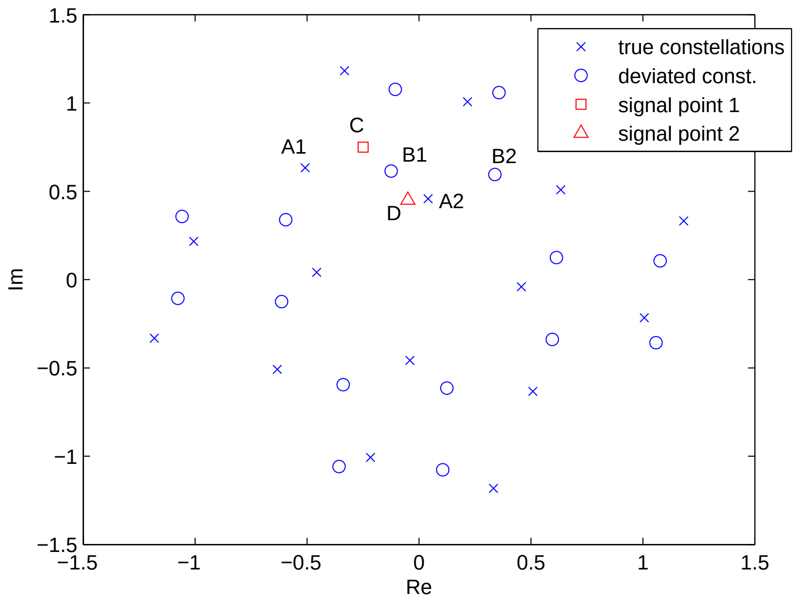


Figure 3.7: Illustration of channel estimation error and symbol detection in combined signal constellation plane. QPSK signalling, $\mathcal{M} = \{1, \jmath, -1, -\jmath\}$. The true signal constellations (indicated by $\times$) $\mathcal{Y} = \{\mathbf{y} : \mathbf{y} = \mathbf{h_0 x_i} + \mathbf{h_1 x_j}\}$, where $\mathbf{h_0} = 0.82 + 0.15\jmath$, $\mathbf{h_1} = -0.36 - 0.19\jmath$, $\mathbf{x_i}, \mathbf{x_j} \in \mathcal{M}$. Because of the error in channel estimation, the estimated $\hat{\mathbf{h}}_\mathbf{0} = 0.84 - 0.12\jmath$ and $\hat{\mathbf{h}}_\mathbf{1} = -0.24 - 0.22\jmath$. The estimated signal constellations (indicated by $\circ$) $\mathcal{Y}_E = \{\mathbf{y} : \mathbf{y} = \hat{\mathbf{h}}_\mathbf{0}\mathbf{x_i} + \hat{\mathbf{h}}_\mathbf{1}\mathbf{x_j}\}$.

as "self-noise" [2].

We now quantitatively analyze the effect of inaccurate channel information to the tracing statistics ρ under hypothesis H_0 and H_1. The analysis under hypothesis H_1 is straightforward. Since the adversary randomly choose one symbol in $\mathcal{M}$ to transmit, at the correlation phase (Eqn.(3.22)), even if the transmitted symbol is incorrectly detected, the effect is still the same as receiving a randomly chosen symbol in $\mathcal{M}$. Thus the mean, variance, and distribution of ρ under H_1 will not change with inaccurate channel information.

Under hypothesis H_0, if we know the mean square error (MSE) of channel estimation, we can then estimate the energy of the self-noise term in (3.33). Assuming the self-noise and thermal noise are independent, we can obtain the aggregated noise energy and the equivalent SNR. This SNR will lead to an equivalent error probability P_e for symbol estimation in the combined signal plane. From here on the rest of previous analysis for tracing statistics ρ can be applied. In practical implementations, the total noise energy including the self-noise can be measured.

3.4 Simulation Results

In this section we present a number of simulation results using the tracing statistics ρ for detecting malicious realy under different Signal-to-Noise-Ratio (SNR). We consider three simulation scenarios. In the first scenario, the malicious relay uses the basic attack strategy and we assume the receiver obtains perfect channel information. In the second scenario, the malicious relay employs the advanced attack strategy, and the receiver has perfect channel information. In the third scenario,

[2]The term "self-noise" was introduced by Dr. Rajiv Laroia during his talk at University of Maryland on Jan. 13, 2006.

the malicious relay uses the advanced attack strategy, and the receiver's channel information is inaccurate, where the inaccuracy is quantified by the normalized MSE of the channel estimation.

In our simulations, The channel conditions are generated using the modified Jakes model in [27]. Each channel follows time-correlated slow Rayleigh fading. Different channels are uncorrelated. QPSK is chosen as the signalling scheme. Every second, a total of physical layer 20K symbols are transmitted. Since the fading is slow, the channel coefficient for each symbol duration is considered constant. Every 64 symbols are grouped as a frame. In each frame the sender will send one, two, or three tracing symbols with equal probability. The time slot to send the tracing symbols are generated uniformly within each frame from 1 to 64.

We consider two types of decision threshold η. The first is a simple average of the analytical expected values under two hypotheses as follows

$$\eta = (\mathrm{E}(\rho|H_0) + \mathrm{E}(\rho|H_1))/2. \tag{3.34}$$

The second decision threshold setting we consider is the Neyman-Pearson setup, where we require the probability of miss is smaller than α and minimize the probability of false alarm. The decision threshold can be computed using the distribution analysis results in Section 3.3.2. Here we demonstrate that even a simple threshold setting as in (3.34) will be very effective in detecting malicious relay. We note that these threshold settings are inherently adaptive. Because the mean and distribution of ρ under two hypothesis are based on the receiver SNR, the attacker's strategy, and the channel estimation error.

Another important system parameter is the threshold probability of correct detection for each tracing symbol, τ (ref. Section 3.3.1). Generally speaking, when τ is large, each tracing symbol will have higher accuracy, thus we will need fewer tracing symbols to obtain a converged tracing statistics ρ. However, the fact that τ is high will reduce the chance that a received tracing symbol can be admitted to compute the final tracing statistics. Since ρ converges with more admitted tracing symbols, a higher τ will take a longer time for the tracing statistics to converge. This is especially a problem at very low SNR because most received tracing symbols have lower probability of correct detection. In our simulations, we set $\tau = 0.9$ when channel estimation is considered perfect, and set $\tau = 0.75$ otherwise.

3.4.1 Basic Attack with Perfect Channel Information

In this experiment, we simulate the tracing scheme with one adversarial relay node randomly transmit signalling symbols, and the other relay node transmit cooperatively. The simulations are performed for SNR in the range of 12 – 17 dB. In Fig. 3.8 we present one realization of the tracing statistics ρ with respect to the number of received tracing symbols, under 13 dB SNR. The adversarial relay randomly chooses one symbol to send and the simulation runs in a 200-frame duration. We can see that using one receive antenna (left figure), only 40% of the tracing symbols are considered reliable, because the threshold probability τ is set rather high at 0.9. However, using two receive antennas (right figure) significantly improves symbol detection probability. All the received tracing symbols are reliable. We can also observe that the tracing statistics from adversarial relay converges to its mean 0; and that from the cooperative relay converges to its mean close to 1. In both cases the convergence rate is fast, requiring only 50-100 reliable tracing symbols. We note that the simulation time duration is about 0.6 second, which indicates that the proposed tracing scheme can obtain a highly confident result

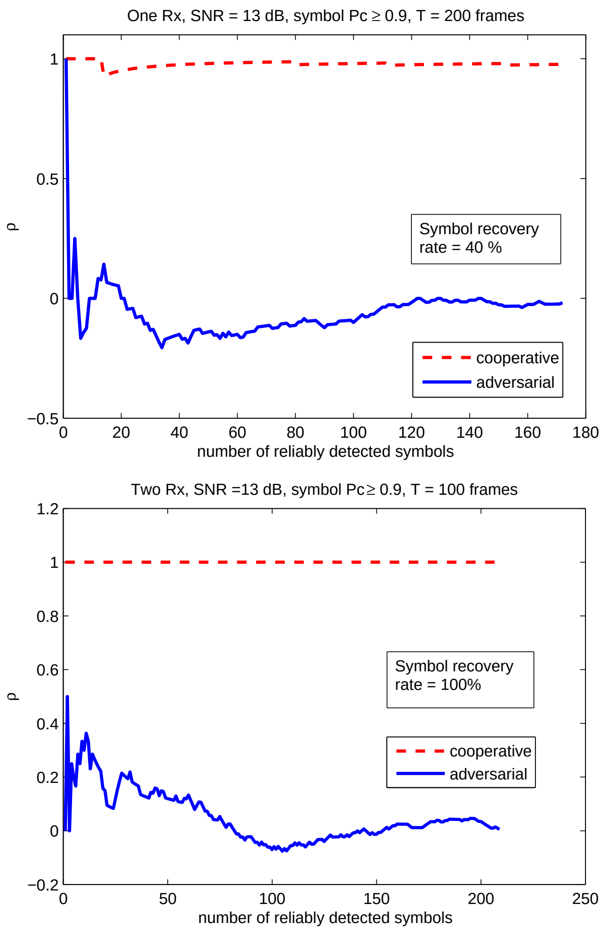
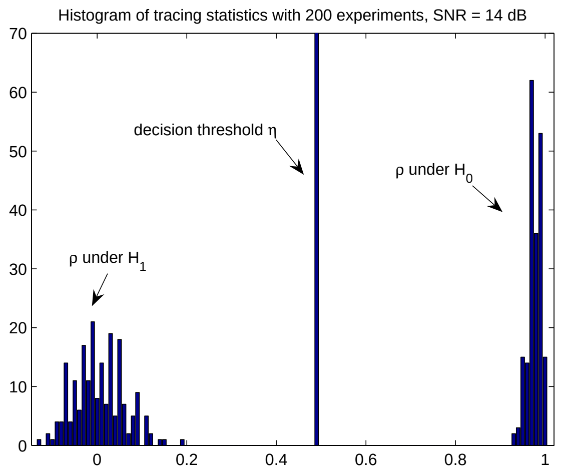
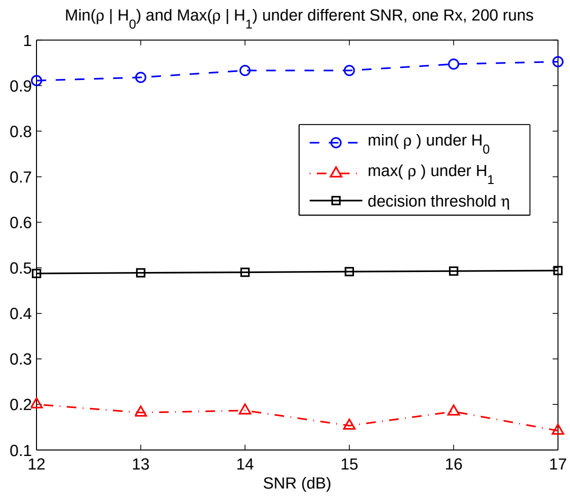


Figure 3.8: Tracing statistics ρ under 13 dB SNR. Left: one receive antenna. Right: two receive antennas.



(a) histogram of ρ under 14 dB SNR



(b) min and max of ρ under different SNR

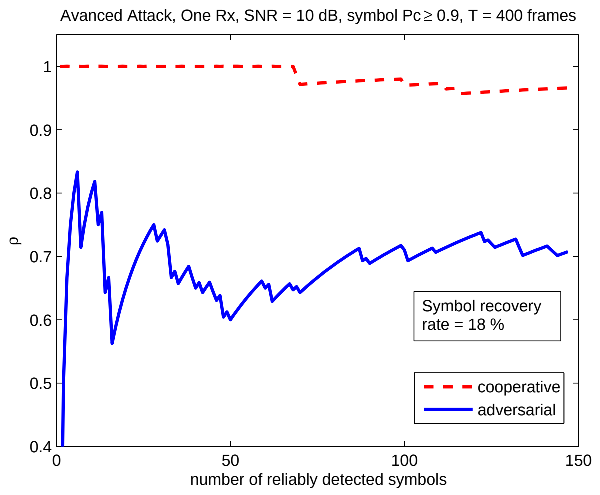
Figure 3.9: Statistics of ρ under basic attack with perfect channel information.

within half a second.

For SNR value ranging from 12dB to 17dB, we perform the simulation 200 times for each integer SNR value. Each simulation run contains one random node that is adversarial and the other that is cooperative. We use the proposed tracing scheme to identify the adversarial and cooperative node. The number of received tracing symbols ranges from 100 to 300 in our simulations. Through all of our simulations, we achieve 100% correctness in detecting both malicious and cooperative nodes. We note that a tracing error can happen by either classifying a cooperative node as adversarial (false alarm), or classifying a malicious node as cooperative (miss). In Fig. 3.9(a) we plot the histogram of ρ under 14 dB SNR. We can see that the decision threshold η computed according to Eqn.(3.34) clearly separates the two classes of ρ values. We note that although the values of η over the 200 experiments are not exactly the same, they are very close as can be seen in the histogram. In Fig. 3.9(b) we show and minimum and maximum values of ρ from 12 dB SNR to 17 dB. We also show the average value of decision threshold η. Again, these decision thresholds perfectly separate the cooperative nodes from malicious ones.

3.4.2 Randomized Attack with Perfect Channel Information

In this subsection we present the tracing scheme under the advanced attack strategy discussed in Section 3.3.3. Similar to [61], we consider that the information sender employs a RS code to protect the transmitted information. Suppose the RS code has parameters $(40, 20, 21)$. This forces the malicious relay to jam at least 1/4 of the symbol stream on average in order to successfully corrupt the communication link. In Fig. 3.10 we plot one realization of the tracing statistics under



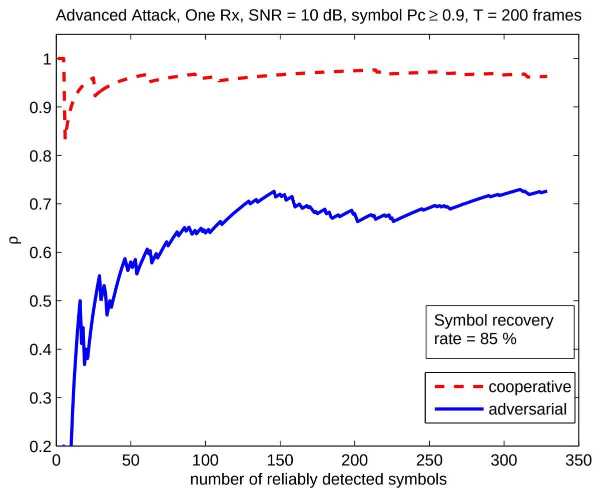
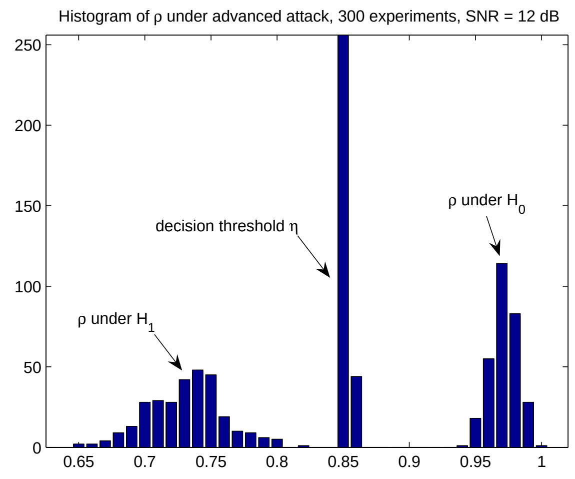
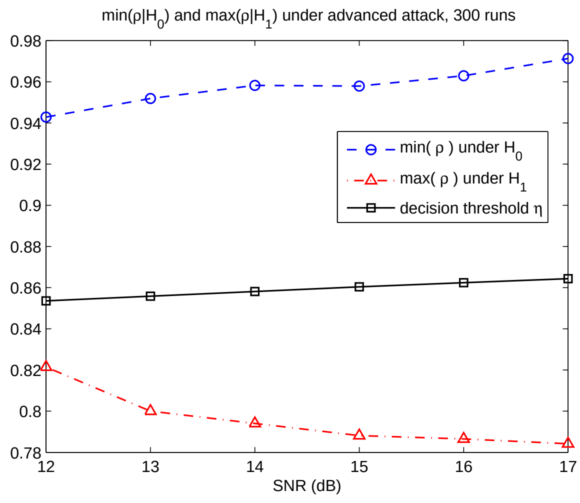


Figure 3.10: Tracing statistics ρ under advanced attack strategy, 10 dB SNR. Left: one receive antenna. Right: two receive antennas.



(a) histogram of ρ under advanced attack, SNR = 12 dB



(b) min and max of ρ under advanced attack

Figure 3.11: Statistics of ρ under advanced attack with perfect channel information.

such a situation. The receiver SNR is 10 dB. For each signalling symbol, the malicious relay randomly choose to transmit a garbled signal with probability 1/4, or to transmit the correct signal symbol with probability 3/4. We can see from the figure that the separation of tracing statistics ρ between the adversarial relay and the cooperative relay becomes smaller, but is still separable. The convergence rate of tracing statistics ρ is slower than in the basic attack.

Under advanced attack, the decision threshold η is adaptively calculated according to Eqn.(3.32). For example, when adversarial relay garble 1/4 or more signalling symbols, the expected value $\mathrm{E}(\rho|H_1) = 3\mathrm{E}(\rho|H_0)/4$. A threshold by simple averaging $\mathrm{E}(\rho|H_0)$ and $\mathrm{E}(\rho|H_1)$ would give $\eta = 7\mathrm{E}(\rho|H_0)/8$. We plot the histogram of ρ under advanced attack in Fig. 3.11(a) with that from a cooperative relay. The histogram for decision threshold η is also shown in the figure. Together these histograms indicate that the tracing statistics ρ under two hypotheses are well separated by the adaptive threshold η. In Fig. 3.11(b) we present the minimum of ρ under hypothesis H_0 and maximum of ρ under H_1. Each data point (min or max) in the figure is obtained from 300 experiments. In all the experiments under all SNR values, we obtained perfect classification, i.e., no false alarm or miss event appeared.

3.4.3 Randomized Attack with Channel Estimation Error

In Eqn.(3.33), we modelled channel estimation error as an additional noise term in the received signal. In our simulation, we intentionally add noise to the actual channel coefficient, and provide the inaccurate channel coefficients to the tracing algorithm. We assume the channel estimation mean square error is a known parameter. In this setting, we explore the Neyman-Pearson threshold setting by

Table 3.6: Experimental occurring frequency of miss and false alarm

SNR (dB)	12	13	14	15	16	17
number of miss	0	0	0	0	0	0
number of false alarm	0	0	0	0	0	0

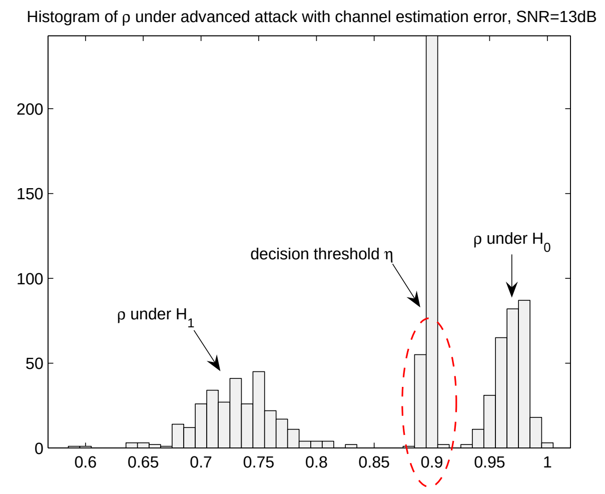
Advanced attack with channel estimation error, Neyman-Pearson setting $P_m \leq 10^{-5}$, 300 runs.

analytically computing the distribution of ρ under H_0. The threshold η ensures that the miss probability $P_f \leq 10^{-5}$. As in the previous simulation setting, the malicious relay randomly garble 25% of relayed signals.

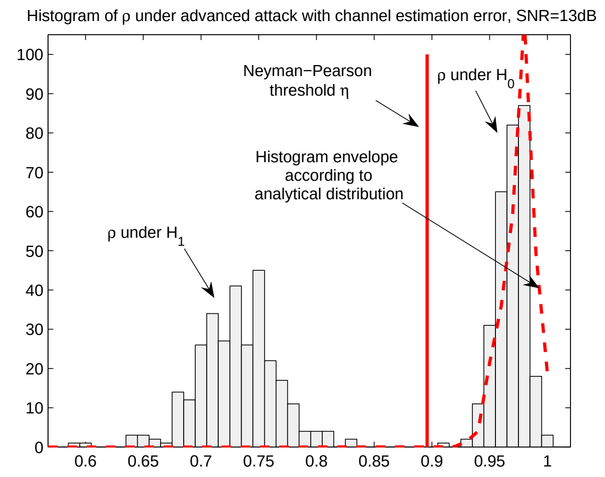
In Fig. 3.12 we the histogram of ρ from 300 experiments. Two methods to compute the Neyman-Pearson threshold are tested. The first method is to compute the threshold based on the estimated symbol error probability from each experiment run, as shown in Fig. 3.12(a). The second method is to use the average symbol error probability from all experiments, as shown in Fig. 3.12(b). In both cases the miss or false alarm occurred. In addition, we plot the analytical distribution for ρ under hypothesis H_0 in Fig. 3.12(b), which is the basis for computing the threshold η. Due to the inaccuracy in estimating symbol detection error, the analytical histogram slightly deviate from the actual one by a mean shift of about 3×10^{-3}. In spite of the slight deviation in estimated distribution, the tracing algorithm performs very well as it achieves perfect detection accuracy. If the tracing scheme can allow a training stage, even such deviation in analysis can be corrected by using the results from training.

3.4.4 Discussions

The cost of the tracing scheme is the extra bandwidth for transmitting the tracing symbols, the computing resource used to detect garbled symbols and aggregate the



(a) Threshold η computed for each experiment run



(b) Threshold η computed using collective information from 300 experiments.

Figure 3.12: Statistics of ρ under advanced attack with channel estimation error.

tracing symbols. The performance of the proposed tracing scheme is quantified by detection probability. Generally speaking, improving the receiver SNR can improve the efficiency and performance of the tracing scheme. This can be achieved in several ways, such as increase transmit power, using multiple receive antennas. With improved SNR, the receiver can achieve a better detection probability using a smaller number of tracing symbols, thus improving efficiency and performance of tracing simultaneously. Another way to improve the detection probability of tracing scheme is to use stronger channel coding at the sender side. Thus the malicious relay has to garble more signals in order to corrupt communications.

Throughout this paper, we have assumed that the pilot symbols for channel estimation are not manipulated by the malicious relay node. In reality, the malicious node can also garble these pilot symbols. However, in cooperative communications, a relay node is chosen only when the channel condition between itself and the receiving node is good enough. As such, in order to be chosen as relays and have the potential to corrupt communications, the malicious nodes cannot choose to significantly manipulate the pilot symbols. Otherwise the channel estimation result will indicate high fluctuation in the channel condition and the malicious node will not be chosen as a relay.

3.5 Chapter Summary

In this paper, we identify the threat of signal garbling by malicious relay nodes in cooperative wireless communication systems. To counter such malicious attack, we proposed a cross-layer tracing scheme that can pinpoint the malicious relay with high accuracy. The proposed tracing scheme employs an adaptive signal detection algorithm, coupled with an statistical tracing symbol aggregation scheme. Our analysis and simulation results show that the proposed tracing scheme can identify the malicious relay with probability of miss and false alarm as low as 10^{-5}, requiring only $1 - 3\%$ of the bandwidth when activated, and obtain the tracing result in a few seconds of signal transmission.

Chapter 4

Coordinated Sensor Deployment for Security and Sensing Performance

Wireless sensor network is an emerging area that integrates sensing, wireless communication, internetworking, distributed signal processing, embedded system and information security all through a set of tiny sensor nodes that are deployed in a designated area. It has a great potential to be widely used in environmental monitoring, building surveillance, industrial manufacturing, and military combat [1, 50]. To achieve security and other functionality using sensor nodes, the protocols for sensor networks must take into consideration the computation, memory, and power constraints of the sensor nodes. In addition, one aspect in sensor network design is often intricately involved with a number of other aspects. Since sensor network systems have inherent complex criteria and objectives, optimizing a single objective may impair the system performance on other aspects. One such example is the the sensor node deployment. The main jobs for most sensor nodes include sensing and communications. In sensor deployment literature, there are works concerning the efficient sensing coverage issue [74, 145, 120], or the secure communication problem [30, 64], but not yet both. As a first step toward developing the theory and algorithms of more *coordinated* sensor deployment, this chapter focuses on jointly considering two important aspects, namely, the sensing coverage and communication security.

A sensor network usually performs its task in the following way. Depending on applications, appropriate type of environmental information in the field are first gathered by the individual sensor nodes and processed; and the necessary information is then relayed to and/or collected by other nodes [35, 46]. The physical characteristics of the sensing and communication devices on board of a sensor impose limits on both the sensing range and the communication range. Therefore, the placement of sensor nodes will have a substantial impact on both the sensing coverage [74] and the communication connectivity [60].

Recently, the security of sensor networks has been brought to the attention of the research community [18][15][39]. As the sensor nodes rely on wireless transmission for communications, malicious adversaries could intercept the communications, modify the data packets, or inject falsified packets. To ensure secure sensor communications, cryptographic mechanisms can be employed to encrypt the data and produce message authentication code (MAC). As symmetric-key cryptography that employ the same cryptographic key in the sending and receiving ends generally have substantially lower computational complexity than the public-key ones, symmetric-key cryptographic tools is generally preferred in practice because of limited computational resources at each sensor node. Furthermore, resource-constrained sensor networks impose stringent constraints on the key establishment scheme. Conventional key management schemes are either centralized by employing a key distribution server, or contributory by using public-key cryptography, and both often require a non-trivial amount of communications. These conventional schemes are not suitable in the sensor network scenarios [34]. To meet the challenge in designing secure sensor networks, key pre-distribution schemes have been introduced to address the special needs in sensor networks [34, 16].

There are two main scenarios that sensor deployment are modelled and studied, depending on whether the locations of sensor nodes can be adjusted after the initial deployment. The first scenario is static deployment, where the location of the sensors will not change once deployed. When efficient sensing coverage is the sole concern, the existing literature suggests that different deployment topologies lead to substantially different efficiency in sensing coverage [20, 53]. In the mean time, researchers focusing on secure sensor communications have recently shown that, if the key pre-distribution can be adapted according to the sensors' locations, we can substantially improve the probability for establishing secure communication links between sensors as well as the security against compromised nodes [30, 64, 143]. However, there is a very limited amount of analysis on how the topology of sensor locations affects both security and coverage issues [71]. In Section 4.2 of this chapter, we present analytic model and experimental validations on several practical topologies in terms of both sensing coverage and ability to establish secure communication links. We shall consider both the case when each sensor can be accurately placed at any desired location, and the case when the actual deployment deviates from the desired location. This investigation will provide important guidelines to sensor deployment for a variety of applications.

The second scenario of sensor deployment considers more advanced sensing devices, where the sensors have the capability of adjusting their locations after being deployed in the field. This is particularly useful when the actual deployment deviates from the desired location. The current literature primarily concerns the development of adjustment algorithms to optimize the sensing coverage. As we shall show in Section 4.3, such adjustment may negatively affect the probability for sensors to establish secure communication links. In [91] and [22], the authors have proposed general frameworks for constrained sensor location adjustment by jointly considering sensing coverage and other performance aspects, such as sensor communications. For communication connectivity, the distance between nodes is a primary factor. However, more complicated factors than distance also play important roles in determining the secure connectivity between nodes. As such, the extension of prior work to incorporate secure communication is not straightforward. The nature of secure sensor communication requires comprehension of the key establish schemes, investigation into deployment topology, and special formulation of the key sharing constraint as part of the location adjustment algorithm. This motivates us to develop new location adjustment algorithms that can jointly optimize the sensing coverage and communication security. We further relate to the first scenario by examining how different topologies for the desired deployment locations affect the overall performance under these security-aware adjustment algorithms.

4.1 Background and Related Works

In this section, we review the background on sensing coverage and secure communications in sensor networks, and briefly survey the related prior works. Throughout the discussion we adopt a simplified mathematical model for sensing coverage. A sensor node located at $\mathbf{x_0}$ has the capability S of sensing for a given location $\mathbf{x}$

$$S(\mathbf{x_0}, \mathbf{x}) = \begin{cases} 1 & \text{if } d(\mathbf{x_0}, \mathbf{x}) \le R_s; \\ 0 & \text{if } d(\mathbf{x_0}, \mathbf{x}) > R_s. \end{cases} \tag{4.1}$$

where R_s is referred to as the sensing radius, and the distance metric $d(\cdot, \cdot)$ is usually the Euclidean distance. $S = 1$ indicates the sensor has the capability to sense and $S = 0$ otherwise. Analogous to the sensing capability, we can simplify the existence of a communication link between two sensor nodes n_0 (located at $\mathbf{x_0}$) and n_1 (located at $\mathbf{x_1}$) using the following model

$$T(\mathbf{x_0}, \mathbf{x_1}) = \begin{cases} 1 & \text{if } d(\mathbf{x_0}, \mathbf{x_1}) \le R_c; \\ 0 & \text{if } d(\mathbf{x_0}, \mathbf{x_1}) > R_c. \end{cases} \tag{4.2}$$

where R_c is referred to as the communication radius. $T = 1$ indicates the link exists and $T = 0$ otherwise.

4.1.1 The Sensing Coverage Problem

Efficient Sensing in Static Deployment

Suppose the sensor nodes with sensing radius R_s can be hand placed in the field to the exact location of our choice. We are interested in the optimal way to place the sensors so that: (1) any location in the field can be covered by at least one sensor; and (2) the nodes can perform sensing in an efficient way. To quantify the efficiency of sensing coverage, we define the sensing efficiency ratio ρ, which is the ratio of two areas

$$\rho = \frac{A_{sep}}{A_{col}}.$$

Here A_{col} is the actual covered area by all the sensor nodes, and A_{sep} is the sum of the area covered by each individual sensor. Apparently, we have $A_{sep} \ge A_{col}$ as

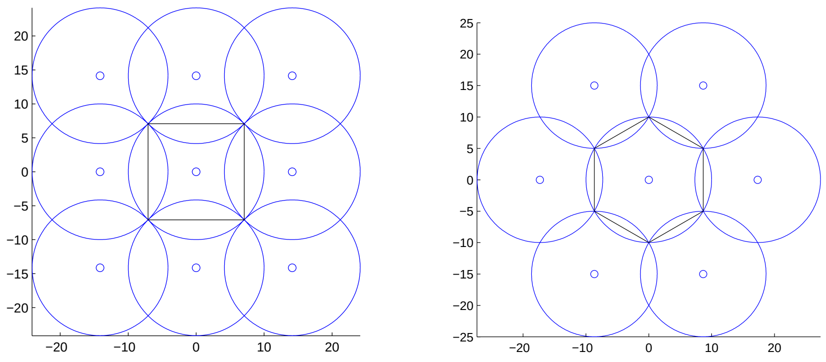

(a) The square lattice. (b) The hexagon lattice.

Figure 4.1: Two possible lattice deployment. Under full coverage requirement, the hexagon lattice has the lowest node density.

the coverage between sensors may overlap, thus $\rho \geq 1$. The closer the efficiency ratio gets to 1, the higher the efficiency. So the problem of optimizing coverage efficiency can be formulated as to minimize the efficiency coefficient ρ subject to the whole area can be fully covered. This problem is traditionally known as the *circle covering* problem [128], where a number of equivalent circles (i.e. circles with the same radius) are placed into a field to completely cover the field area. A sensor node is located at the center of a circle, and the radius of the circle corresponds to the sensing radius.

If the circles are placed in repeated regular patterns, the circle centers form a lattice and the dissecting lines among the centers form a cell pattern. In Fig. 4.1 we show two possible covering layout using the square lattice and the hexagon lattice, respectively. Each layout leads to a specific efficiency ratio ρ, which is also known as the *covering density* or *covering thickness* in the mathematics literatures [20]. For the simplified sensing model of Eqn. (4.1), $A_{sep} = n \cdot \pi R_s^2$, where n is the total number of sensors. Kershner [53] has shown that a lower bound for ρ is $2\pi/\sqrt{27} \approx$

1.21, which is achieved when the center of the circles (i.e. the sensor nodes) form a hexagon lattice. In this case, the distance between any two neighboring nodes is $D = \sqrt{3}R_s$. Fig. 4.1(b) illustrates the geometry of such a placement. Compared to the square lattice placement, which has efficiency ratio $\rho = \pi/2$, the hexagon lattice placement gives much more efficient coverage. Further, sensor placement can be viewed as spatial sampling from signal processing perspective. The literature there also suggests the superiority of hexagonal sampling lattice over the square lattice when the spatial spectrum of a 2-D signal being measured (such as a temperature field) is bandlimited with a circular support.

For the convenience of discussion, we define the following notations. In the square lattice deployment, we denote the distance from a node to its horizontal/vertical neighbor by D_1, and the distance to its diagonal neighbor by D_2. Thus we have $D_2 = \sqrt{2}D_1 \approx 1.41D_1$, and the covering density $\rho = \pi R_s^2/D_1^2$. In the hexagon lattice, we denote the distance from a node to its six neighbors by D_3. Further, if we require that the two lattice have the same node density, we have $D_3 = \sqrt{2/\sqrt{3}}D_1 \approx 1.07D_1$. Throughout this chapter, we use the normalized distance with respect to D_1 as the distance metric, and study the impact of deployment topology on the performance of sensing coverage and secure communications.

Sensor Location Adjustment Algorithms

In recent years, the advances in micro-electromechanical systems (MEMS) have made it possible for tiny sensor devices to walk as microrobots [89]. The locomotion capabilities of sensor nodes have made it possible to improve the sensing coverage after the initial sensor deployment. Consequently, a number of prior works have studied how to adjust the location of sensor nodes to maximize the total coverage in a given area [75, 44, 74, 120, 145]. The total sensing coverage, η, is the percentage of the area covered within the sensing range with respect to the total field area. We can see that $\eta \leq 1$ and a larger η represent a better coverage. Most existing algorithms for sensor location adjustment uses an iterative framework. In each iteration, sensors (or a cluster head) obtain their current locations and the relative locations to their neighbors. Based on these information, each node will compute a new location using the location updating algorithm. The general strategy is to spread out the sensor nodes as evenly as possible. For example, the virtual force algorithm (VFA) proposed in [145] compares the distance between a sensor and its neighbor nodes with a threshold distance. An attractive (resp. repulsive) virtual force is applied to the sensor node if the distance is greater (resp. smaller) than the threshold. The Minmax algorithm proposed in [120] employs the Voronoi cell concept and move the sensors to the center of the minimum-radius circum-circle of its Voronoi. Further, for calculating the sensing coverage, the authors of [74] and [75] have proposed polynomial-time algorithms to calculate the worst case and average case coverage.

From secure communication point of view, however, the location adjustment intended only to maximize sensing coverage may reduce the secure communication connectivity. This is because the secure links established before location adjustment may no longer exist after location adjustment and some sensor nodes can be moved to un-preferred locations in terms of establishing secure communications using pre-distributed key. In Section 4.3, we will present a detailed example to illustrate the limitation of the existing adjustment methods and discuss how to balance the tradeoff between the sensing coverage and the node connectivity using secure links.

4.1.2 Key Pre-distribution for Sensor Networks

As reviewed earlier, one of the critical issues for secure sensor communication is to establish a cryptographic key between two sensors. To accommodate stringent resource constraints in sensor network systems, an increasingly popular approach is to preload each sensor with a set of keys from a large collection of keys. This entire collection of keys is referred to as the *key pool* and the set of the keys loaded by each sensor is referred to as the *key ring*. Once the sensors are deployed in the field, neighboring sensors will follow certain protocol to discover whether they share some secret keys. If so, they will use these shared secret keys to establish a secure communication link. There are two requirements for establishing a secure communication link between two sensor nodes: (1) two nodes should be within communication range; and (2) two nodes should share at least one secret key. The first work on random key pre-distribution [34] was proposed by considering the sensor nodes are randomly deployed into the field. Later, Du *et al.* and Liu *et al.* proposed to incorporate sensor location knowledge into key pre-distribution [30] [64]. The deployment model in these works considers the sensors being deployed at the center of evenly partitioned square cells. Each cell will have its own key pool, and only neighboring cells will have overlap between their key pools. The sensors in each cell will randomly pre-load keys from the key pool of its own cell. Since the key pool of each cell is much smaller compared to the key pool for the entire sensor node collection, neighboring sensors will have a higher chance to share keys. Most recently, Zhou *et al.* identify the improved circular symmetry of the hexagon cell than the square one to reflect the common shape of sensors' communication range, and propose to use hexagon lattice in location-based key pre-distribution [143].

In location-based key pre-distribution, each sensor has a designed deployment location for establishing secure communication link. In practice, these designed locations may not be the same as the locations determined according to the sensing performance. This motivates us to study the impact of practical sensor deployment on establishing secure communications.

4.1.3 Adversary Model and Link Compromise Probability

In addition to the ability to establish secure links between nodes, resilience against node compromise is another important security aspect to be examined. As sensor nodes may be deployed in adversarial environment, a deployed node could be captured by adversaries. We assume the adversaries will try to capture the deployed sensor nodes and use the pre-loaded keys in the captured nodes to eavesdrop secure links among sensor nodes that are not compromised. In measuring such a potential threat, the probability of link compromise due to node compromise is an important security metric considered by previous key pre-distribution works [34, 16, 63, 30]. To allow fair comparison on different deployment topologies, we should require each topology to have the same link compromise probability when the same number of nodes are compromised, and then compare the connectivity of secure communication. We have constructed a probabilistic model to compute and approximate the link compromise probability in the location-based key pre-distribution scheme [30] using lattice deployment. It can be shown that if the compromised nodes are statistically uniformly distributed among all nodes and each node carries the same number of keys, then the link compromise probability is approximately the same for the location-based scheme using the square lattice, the hexagon lattice, and the basic scheme using random deployment, up to the first-order Taylor expansion [34]. The detailed derivation can be found in the Appendix. With this finding, we can construct a fair comparison between deployment topologies. Our study shows that, for fixed-size key ring, the group-based scheme using structured/lattice deployment usually can achieve a better connectivity than the random deployment.

4.2 Lattice-Structured Deployment

In this section, we jointly examine the sensing coverage and communication security under the static sensor deployment scenario. Given that a very limited amount of study has been done in the literature on how the sensor topology affects both security and coverage issues, we focus on analyzing the impact of deployment topology on the performance of sensing coverage and efficiency as well as to the ability of establishing secure communications. We will consider two main deployment topologies, namely, the square lattice and the hexagon lattice.

4.2.1 Fundamental Relations Between Deployment Lattices

As the first example to illustrate the impact of sensor deployment topology on the establishment of secure communications links, we consider the simple case of sensors being placed exactly at the desired location. We deploy sensors under a square lattice and a hexagon lattice, respectively, and employ the basic key pre-distribution scheme [34], where each node has the same probability to share a secret key with any other node. We denote the key sharing probability by P_{share}, and use the same node densities in the two lattice deployment, which is the number of nodes per unit area. As the communication radius R_c increases from 0 to $1.6D_1$, each sensor can gradually have more reachable neighbors, and this in turn will affect the number of secure links per node. Because the distance

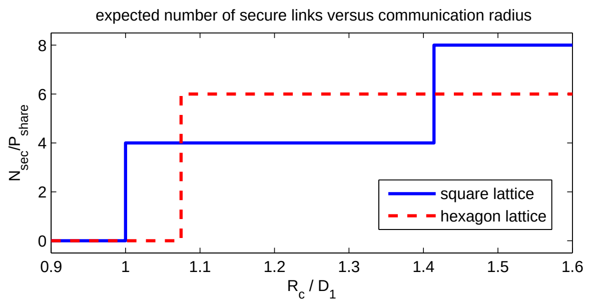


Figure 4.2: Expected number of secure links versus communication radius using the basic key pre-distribution scheme.

between a node and its eight neighboring nodes in a square lattice is not circularly symmetric [143], the number of neighbor nodes that can be reached is a two-step function of the communication range. That is, as the communication range increases, four vertical and horizontal neighbors of the center nodes (also known as the *4-way connection* [47]) will be reached first, before the other four neighbors on the diagonal directions being reached. This can be seen from Fig. 4.2, where we show the relation of the expected number of secure links per node, normalized by the key sharing probability, with respect to the normalized communication radius. The result for a hexagonal lattice, on the other hand, is a one-step function, owing to the circular symmetry between a center node and its six neighbors. Under the same node density, Fig. 4.2 shows that hexagon lattice achieves a better topology when the communication radius R_c is between 1.07 and 1.41 times of D_1; and outside this range, square lattice achieves a better connectivity. Later in this chapter, we shall see several more examples reflecting this fundamental relations between the square and hexagon deployment lattice.

4.2.2 Secure Connectivity Under Perturbed Deployment Lattice

While Fig. 4.2 depicts the trend for the secure communication connectivity using square and hexagon lattices in the ideal situation, sensors may not be deployed with high accuracy at the designed lattice locations in practice. Such inaccuracy may be caused by measurement error (if sensors are deployed by human), or by wind speed (if sensors are deployed by vehicle or airborne methods). Suppose a sensor node is designed to be deployed at location (x_0, y_0) in the field. The actual deployment location (x, y) can be modelled as

$$x = x_0 + r_x; \quad y = y_0 + r_y.$$

Here the deviation terms r_x and r_y are zero-mean random variables. One can model these deviation terms as Gaussian distributed [30] or uniformly distributed [63] random variables with variance σ^2 as the deviation parameter.

Taking the deployment deviation into consideration, we investigate the impact of deployment topology on the connectivity of secure communication. Here we choose the location-based key pre-distribution in [30] and the Gaussian deployment deviation model and compare the square lattice deployment used in [30] with the hexagon lattice deployment. In the hexagon lattice deployment, each node is surrounded by six neighbor nodes. By using location-based key pre-distribution, the key pool for any given node, referred to as the center node, has 1/6 overlap with each of its six neighbors' key pools. Thus the the center node will have equal probability to share keys with each neighbor node. We denote the probability that the center node can still be a neighbor with one of its neighbor node under Gaussian deployment deviation by $\Pr(\mathit{neighbor})$, and the probability that the two nodes can share a key by $\Pr(share)$. As the deployment deviation is independent of key distribution, the probability that a designed neighbor in the hexagon lattice can establish a secure link with the center node is $\Pr(hexlink) = \Pr(neighbor)\Pr(share)$. Because of the geometrical symmetry, the expected number of secure links for the center node is

$$\mathrm{E}(N_{sec}^{hex}) = 6\Pr(hexlink).$$

Similarly, we can compute expected number of secure links per node in the square lattice deployment. In the square lattice, we refer to the horizontal/vertical neighbors of a node as type-A neighbors and the diagonal neighbors as type-B neighbors. Denote the probability that a node can establish a secure link with one of its type-A neighbors as $\Pr(sqlinkA)$, and that with type-B neighbors as $\Pr(sqlinkB)$. The expected number of secure links per node is

$$\mathrm{E}(N_{sec}^{sq}) = 4\Pr(sqlinkA) + 4\Pr(sqlinkB).$$

In Fig. 4.3 we show the expected number of secure links per node under Gaussian deployment deviation. Each node carries 100 keys and each key pool contains 1200 keys. The neighbor probability and key sharing probability can be computed using the model in [31]. In Fig. 4.3, in both square and hexagon lattice deployment, the expected number of secure links increases with the normalized communication radius R_c/D_1. The hexagon lattice achieves a slightly higher connectivity over the range of 0.9 and 1.7 in the normalized communication radius, exhibiting a similar trend as in Fig. 4.2. This suggests that the communication radius, deployment topology, deviation parameter, and the number of pre-loaded keys per sensor all play a role in establishing secure links. It is also worth noticing that while the numerical gain in connectivity by hexagon lattice over the above mentioned communication range is small, its practical impact is non-trivial. Within this com-

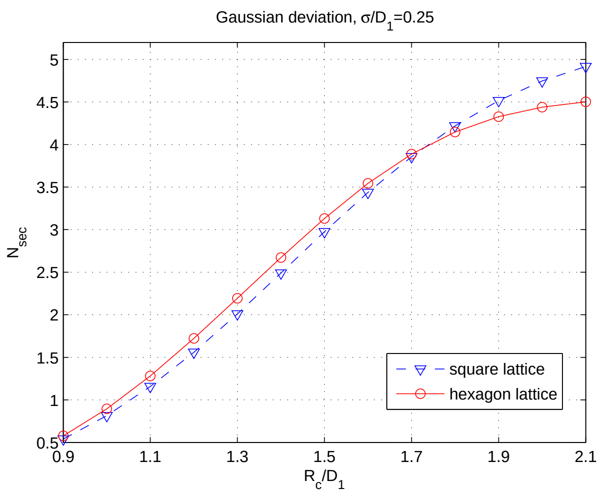


Figure 4.3: Expected number of secure links per node versus communication radius. Shown here are the analytical values under Gaussian deployment deviations.

munication range, the average node degree increases from 0.5 to around 4, and the sensors gradually change from isolated nodes to connected components, where the boundary value for the average links per node is around 2. This phenomenon will be illustrated later in Section 4.3. To achieve the same connectivity, the square lattice would require a larger communication range. As the power consumption of wireless communications is related to the communication range by a power law (from the $2nd$ to the 4-th power, depending on the propagation environment [97]), a lower requirement on communication range with lower power consumption while maintaining communication connectivity is more desirable in many sensor network designs. This makes hexagon deployment lattice attractive for power-limited applications.

4.3 Security-Aware Sensor Location Adjustment

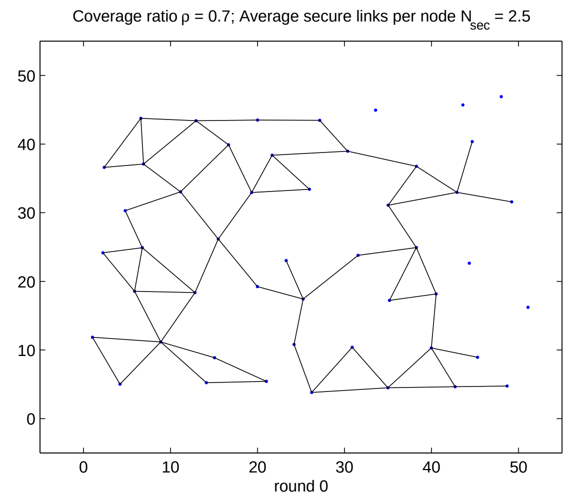
The static deployment strategy described in the previous section considers the sensor deployment as a one-time task. Once the sensors are deployed in the field, their locations are fixed and cannot be further adjusted. In recent years, a number of works on practical sensor deployment have considered movement-adjusted sensor deployment for improving sensing coverage [120, 145, 44]. In this section, we investigate the impact of location adjustment in sensor deployment on secure communications. We propose two new location updating algorithms for sensor deployment that jointly consider sensing coverage and secure communications.

4.3.1 Improving Secure Connectivity Using the Virtual Force Framework

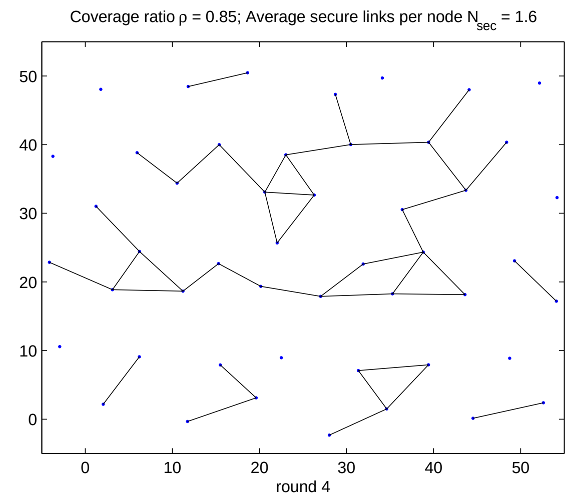
Effect on Secure Connectivity by the Existing Approach

When secure communications is required for sensor nodes, the existing location adjustment algorithms may negatively affect the establishment of secure communication links. As an example, we examine the establishment of secure communication links when the sensors are moved by the Virtual Force [145] location updating algorithm. The Virtual Force algorithm adjusts the sensor locations based on the relative distance from a sensor to its neighbors compared to a pre-determined threshold d_{th}. Suppose a node n_i has a neighbor node n_j and their distance is d_{ij}. The virtual force applied by n_j to n_i is

$$\overrightarrow{F}_{ij} = \begin{cases} w_A(d_{ij} - d_{th}) \cdot \overrightarrow{v}_{ij} & \text{if } \; d_{ij} \geq d_{th}; \\ (w_R/d_{ij}) \cdot \overrightarrow{v}_{ji} & \quad d_{ij} < d_{th}. \end{cases} \tag{4.3}$$



(a) Initial deployment locations



(b) Locations after 4 VFA iterations

Figure 4.4: Impact of location adjustment to the establishment of secure links using VFA

Here $\overrightarrow{v}_{ij}$ is the unit-length pointing from the location of n_i to that of n_j. The total virtual force $\overrightarrow{F}_i$ on n_i is the aggregated virtual force from all of its neighbors, i.e.,

$$\overrightarrow{F}_i = \sum_{j=1}^{Nb} \overrightarrow{F}_{ij}. \tag{4.4}$$

After computing the virtual force for each node, the node n_i is moved to the direction specified by the aggregated virtual force $\overrightarrow{F}_i$ with a step size equal to its magnitude $|\overrightarrow{F}_i|$.

Fig. 4.4 shows an example of location updating using the VFA and its impact on secure communications. Initially, 49 sensors are deployed into a 60×60 area with a hexagon lattice pattern. A uniform distributed deployment deviation is applied to the initial locations, with the deployment variance $\sigma^2 = 4/3$. This initial deployment is shown in Fig. 4.4(a) with the established secure links marked as lines connecting the sensor nodes. In this example, the sensing radius is 5 and the communication radius is 9. The sensing coverage achieved by the initial deployment is $\eta = 0.7$ and the average number of secure links per node is $N_{sec} = 2.5$. Next, we apply the VFA to update the sensor locations. After four iterations, the sensing coverage has been improved to $\eta = 0.85$, while N_{sec} has been reduced significantly to $N_{sec} = 1.6$, implying most of the nodes are no longer connected. This is illustrated in Fig. 4.4(b). At the initial deployment, most nodes form a connected component using the secure links; after four iterations, about half of the nodes are no longer connected with the largest connected group, which reduces the capability of secure communications between the sensor nodes. Our study shows that such a phenomenon is common in sensor location update using virtual force type of schemes. To maintain a comparable sensing coverage while improving secure connectivity, we propose a modified sensor location updating algorithm based on the virtual force framework. We call the modified algorithm *VFSec*, indicating that secure communications is one of the main factors in updating the sensor locations.

VFSec Algorithm

As we have seen, there is a tradeoff between the sensing coverage and secure connectivity. For balancing this tradeoff, we define an additional performance metric as

$$\Gamma = w_1\eta + w_2 N_{sec}. \tag{4.5}$$

The weights w_1 and w_2 are chosen such that $w_1\eta$ and $w_2 N_{sec}$ are approximately in the same value range, so as to achieve a desired tradeoff. Since the sensing coverage is always within $[0,1]$, and the average number of secure links per node is around 3 in most of our experiments, we choose $w_1 = 1$ and $w_2 = 1/3$ in our experiments.

Our algorithm uses the combined performance metric Γ to measure the optimality of sensor locations, which balances the tradeoff between coverage and secure communications. During each iteration of location adjustment, the algorithm tries to keep the distance between those nodes that can establish secure links closer. To achieve this, we add a new term of virtual force, $\overrightarrow{F}^{sec}$, to the total virtual force. The virtual force $\overrightarrow{F}^{sec}_{ij}$ applied to a node n_i by a neighbor node n_j is as follows

$$\overrightarrow{F}^{sec}_{ij} = \begin{cases} w_s(d_{ij} - D_{sec}) \cdot \overrightarrow{v}_{ij} & \text{if } D_{sec} < d_{ij} < R_c \\ & (n_i,\ n_j) \text{ share key;} \\ 0 & \text{otherwise.} \end{cases} \tag{4.6}$$

In computing $\overrightarrow{F}^{sec}_{ij}$, D_{sec} is a threshold distance smaller than the communication radius R_c, d_{ij} is the distance between node n_i and n_j, w_s is the weight assigned to the added virtual force, and $\overrightarrow{v}_{ij}$ is the unit-length vector pointing from the location of n_i to that of n_j. The total virtual force for secure link applied on n_i is $\overrightarrow{F}_i^{sec} = \sum_j \overrightarrow{F}_{ij}^{sec}$. This force is added to distance-based forces $\overrightarrow{F}_{ij}$ in Eqn.(4.4) to compute the total virtual force $\overrightarrow{F}_i$. To update the sensor location, the sensor node n_i is moved along the direction of $\overrightarrow{F}_i$ by a distance equal to the magnitude of vector $|\overrightarrow{F}_i|$. The complete algorithm is described in Algorithm 1.

Simulation Results and Discussions

To study the performance of VFSec, we have performed three experiments and compared the VFA and VFSec in terms of sensing coverage and secure link establishment. Throughout these experiments, we set the communication radius R_c as twice the sensing radius R_s. This is to ensure that even when the sensing range is very small and two neighboring sensors are barely disjointly placed (i.e. the distance between two neighboring sensor nodes is $2R_s$), it is still possible to establish a communication link between the two sensors. In all the experiments, we choose $w_s = 0.2$ and $D_{sec} = 0.6R_c$ based on heuristics. Both the VFA and VFSec algorithms are run for seven iterations.

In the first experiment, we place 36 sensors nodes uniformly in a 50×50 area. Using VFA and VFSec, the locations of the sensors are adjusted. The sensing coverage and the number of secure links per node are recorded. We repeat such experiment 400 times and computed the average coverage and the number of secure links per node under different sensing and communication radius. From the results shown in Fig. 4.5 we can see that, the proposed VFSec algorithm can improve the average number of secure links by approximately 15-20%, with a small reduction in the sensing coverage by approximately 2-5%. In addition, the VFSec consistently

Algorithm 1 VFSec algorithm

```
**Input:** sensor locations $\{(x_i, y_i)\}_{i=1}^n$ and key index set $\{K_i\}_{i=1}^n$
**Output:** new locations $(x_1^{opt}, y_1^{opt})$, ... $(x_n^{opt}, y_n^{opt})$
/* Initialization */
Compute $\Gamma^{opt}$ using Eqn.(4.5) with $(\{(x_i, y_i)\}_{i=1}^n, \{K_i\}_{i=1}^n)$
$(x_i^{opt}, y_i^{opt}) \longleftarrow (x_i, y_i)$ for $1 \leq i \leq n$
/* Iteration */
for $i = 1$ to MAX-ITERATION do
  for each sensor node $n_i$ do
    Calculate $\overrightarrow{F}_{ij}$ using the formulation in [145]
    Calculate $\overrightarrow{F}_{ij}^{sec}$ using (4.6)
    $\overrightarrow{F}_i \longleftarrow \sum \overrightarrow{F}_{ij} + \sum \overrightarrow{F}_{ij}^{sec}$
  end for
  /*Update sensor locations*/
  $(x'_1, y'_1) \longleftarrow (x_i + F_{ix}, y_i + F_{iy})$ for $1 \leq i \leq n$
  Compute $\Gamma$ using Eqn.(4.5) with $(\{(x'_i, y'_i)\}_{i=1}^n, \{K_i\}_{i=1}^n)$
  if $\Gamma > \Gamma^{opt}$ then
    $(x_i^{opt}, y_i^{opt}) \longleftarrow (x'_i, y'_i)$ for $1 \leq i \leq n$
  end if
end for
```

achieves a better performance in terms of the overall performance metric Γ in Eqn.(4.5).

In the second experiment, we compare the VFSec and VFA using square lattice deployment under Gaussian deployment deviation. We fix the node density and the deviation parameter $\sigma = 0.4D_1$. From Fig. 4.7, we can see that VFSec improves the average number of secure links per node, with a small compromise in the sensing coverage. In this experiment, we have excluded the boundary nodes of the square deployment area in computing the sensing coverage and number of secure links. Thus the results can be viewed as if the performance is evaluated in an infinitely large area. In spite of the difference in accounting the performance, the results in Fig. 4.7 shows the same trend as in Fig. 4.5.

In the third experiment, we compare the square and hexagon lattice deployment using the corresponding location-based key pre-distribution. We use the proposed VFSec algorithm for location updating and the results are obtained for different communication and sensing radius under small deployment deviation ($\sigma/D_1 = 0.2$). Fig. 4.6 shows that the hexagon and square lattices achieve comparable sensing coverage. In terms of the average number of secure links per node, the hexagon lattice achieves a better performance when the normalized communication radius R_c/D_1 is in the approximate range of $[1, 1.5]$; outside this range, the square lattice performs better. Such a result again shows that there is no all-time winner in terms of deployment lattice, as is shown in the step function connectivity graph in Fig. 4.2 for the ideal hexagon and square lattices. When designing secure sensor networks, the deployment lattice as well as system parameters, such as the communication radius, should be taken into consideration.

Implementation Issues

Similar to the VFA algorithm, the VFSec can be performed by the cluster head of the sensor nodes. As indicated in Algorithm 1, the cluster head needs to collect the sensors' key index sets and their current locations. The key indices are ID's assigned to secret keys, which is used in the shared key discovery phase in key pre-distribution schemes [34][16]. The iterations for updating sensor locations are performed by the cluster head and only the final results obtained are sent back to the sensor nodes. The actual location adjustment is performed only once by each sensor.

When the cluster head is not available, the algorithm can be performed by the individual sensors only based on its neighborhood information. In this case, Algorithm 1 must be adjusted to suit the distributed implementation. The sensors need to perform movement adjustment after each iteration. At the same time, computing and comparing the global performance metric Γ^{opt} as in Algorithm 1 would not be feasible; and the number of iterations must be limited to reduce the power consumption in sensor movement.

4.3.2 Sensor Location Adjustment Based on Vector Quantization

One limitation of the VFSec algorithm is that in order to achieve a better secure connectivity, the sensing coverage is somewhat sacrificed. The reason is that the virtual force based approach simplifies the problem in a two-dimensional area to a set of vectors. In this part, we propose a new approach for updating sensor locations that can explore more freedom in the two-dimensional space to jointly optimize sensing coverage and secure communications.

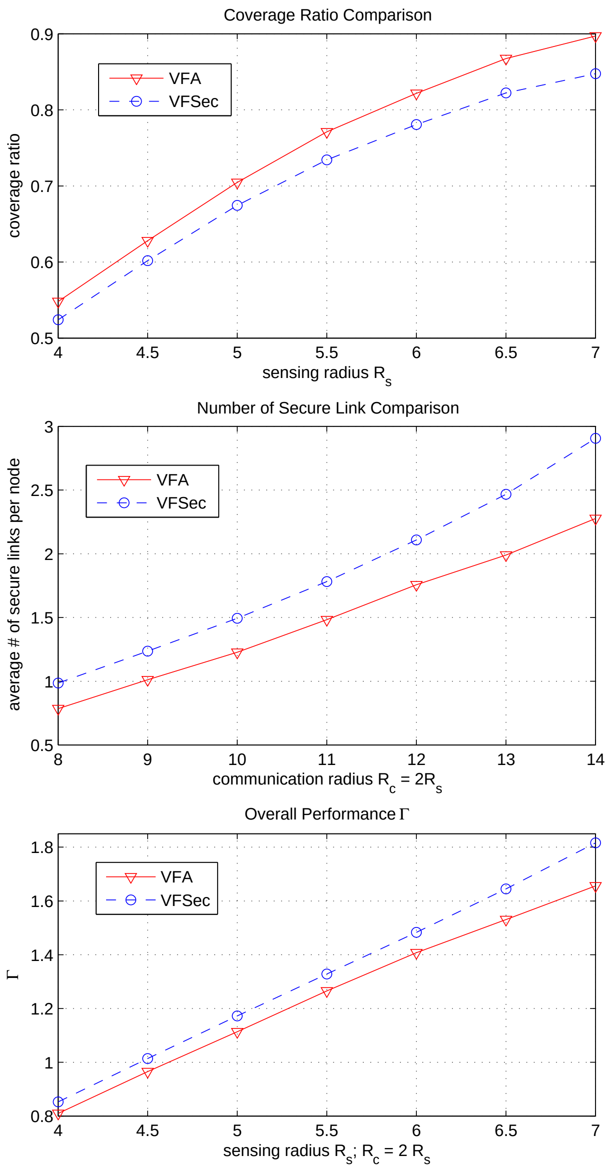


Figure 4.5: Comparison of VFA and VFSec with uniform random initial deployment.

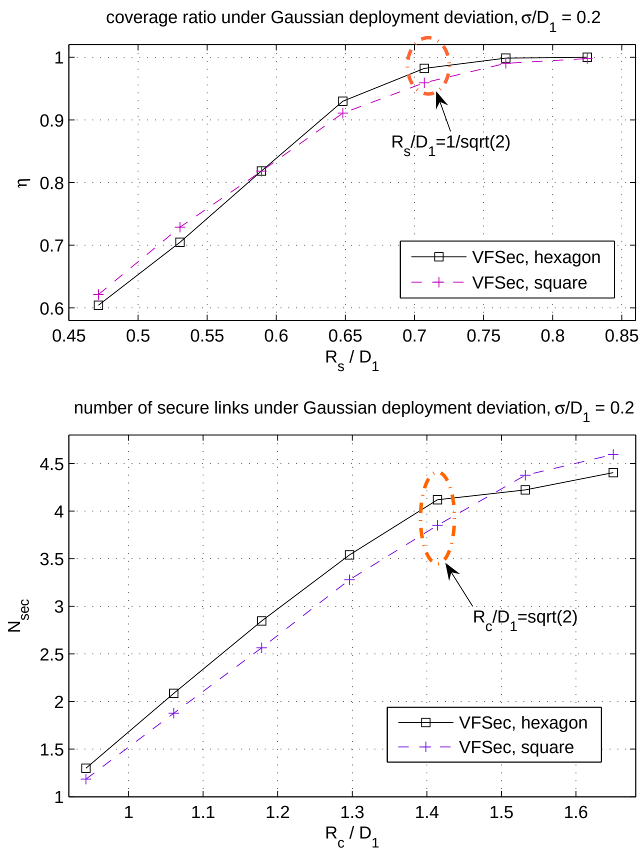


Figure 4.6: Comparison of deployment lattice using VFSec under Gaussian deployment deviation.

The problem of covering a region using distributed sensor nodes is analogous to the vector quantization problem in signal compression [38]. In the sensing coverage problem, each node can sense its nearby region with certain accuracy. The goal is to maximize the total coverage given a limited number of sensors. In the quantization problem, each point in the k-dimensional space is associated with a representative point in the codebook. The goal is to use a limited number of points to represent all points in the region with minimum error. In two-dimensional space, if the input signal is statistically uniformly distributed, the minimum-error quantization lattice and the most efficient covering lattice are the same hexagon lattice [20] as we have seen in Fig. 4.1(b). This has motivated us to employ insights in the vector quantization literature to explore solutions for sensor deployment.

The Weighted Centroid Algorithm

Several prior works have proposed location updating algorithms that are similar to the two-dimensional vector quantization solution [74][120]. In particular, the MinMax algorithm proposed in [120] computes the Voronoi cell V for each sensor node n, and move the sensor to the minmax location $\mathbf{x}_{minmax}$ so that the maximum distance from the new location to any point in the cell V is minimized, *i.e.*,

$$\mathbf{x}_{minmax} = \arg\min_{\mathbf{x}}\{\max_{\forall \mathbf{y}\in V} d(\mathbf{x},\mathbf{y})\}.$$

It has been shown in [120] that the minmax location is the center of the minimum-radius circum-circle of the Voronoi cell associated with each node.

Inspired by these works, we propose a new approach for updating sensor locations based on the Lloyd-Max quantization algorithm [38]. We consider that the sensor has a communication range R_c and can know the locations of its neighbors and its own location [74]. Furthermore, the proposed approach will allow the

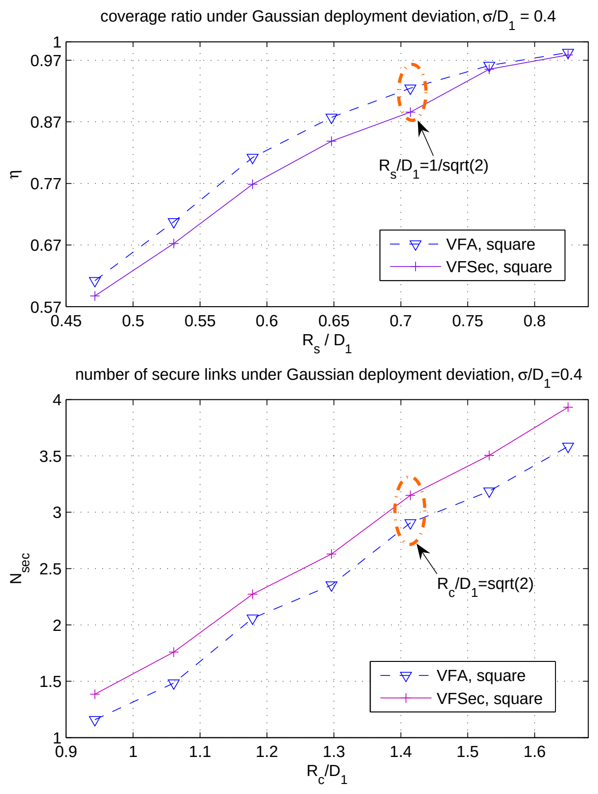


Figure 4.7: Comparison of VFA and VFSec using square deployment lattice under Gaussian deployment deviation.

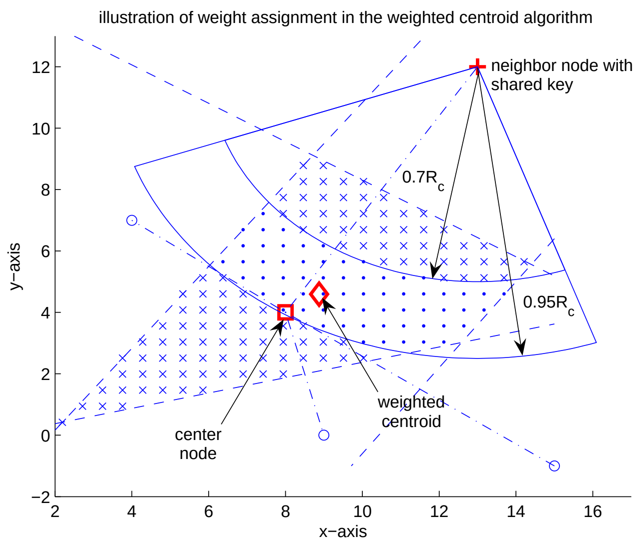


Figure 4.8: Illustration of the weight assignment in the weighted centroid algorithm.

sensors to take secure communication as a factor in updating locations.

Our proposed algorithm aims at minimizing the weighted average distance of a sensor node to the points in its Voronoi cell. We choose a weighted square distance as the distance metric. Suppose in the two-dimensional space, there are N points uniformly distributed at locations $\{(x_i, y_i)\}_{i=1}^N$ inside the Voronoi cell formed of a sensor node located at (x_0, y_0) and its neighbors. Each point is associated with a weight w_i. Then the weighted square distance D_w is

$$D_w = \frac{1}{N}\sum_{i=1}^{N} w_i[(x_0 - x_i)^2 + (y_0 - y_i)^2]. \tag{4.7}$$

From the classic vector quantization results [38], we know that given the set of points $\{(x_i, y_i)\}_{i=1}^N$ and the weight $\{w_i\}_{i=1}^N$, the optimal value for (x_0, y_0) that minimizes the weighted distance D_w is

$$x_0^{opt} = \frac{\sum_{i=1}^N w_i x_i}{\sum_{i=1}^N w_i}; \quad y_0^{opt} = \frac{\sum_{i=1}^N w_i y_i}{\sum_{i=1}^N w_i}. \tag{4.8}$$

For the dual problem, we know directly from the definition of Voronoi cell that, for any point p located inside a voronoi cell V_i of sensor n_i, the node n_i is closer to the point p than any other node outside the Voronoi cell V_i. Thus we have naturally obtained an iterative algorithm for location updating.

The proposed algorithm works as follows. In each iteration, each sensor n_i discovers its neighbors and generates its Voronoi cell V_i according to the neighbor locations. Next, the sensor node generates a set of uniformly distributed grid points $\{(x_i, y_i)\}_{i=1}^{N}$ inside V_i and assign weight to each point. Then the node will compute its new location (x_0', y_0') that can minimize the weighted square distance D_w according to Eqn.(4.8). The simplest weight assignment is to assign equal weight of one to all points. When different weights are assigned to the sampling grid points, the solution (x_0^{opt}, y_0^{opt}) is the *centroid* of the Voronoi cell with respect to weight assignment $\{w_i\}$. Therefore we refer to the algorithm as the weighted centroid (WTC) algorithm and describe it in Algorithm 2.

To jointly consider secure communication and sensing coverage, we propose the following weight assignment procedure. For each sensor node n_i, after the Voronoi cell has been formed and the grid points are generated, the base weight for each grid point inside the cell is 1. If the node n_i already has a secure link with a neighbor node n_j, each grid point that falls into the ring area centered at n_j, and between the radius $0.7R_c$ and $0.95R_c$ will be assigned an extra weight of $w_{sec} = 0.5$. An algorithmic description of the weight assignment procedure is presented in Algorithm 3. In Algorithm 3, Nb refers to the number of neighbors of a center node n_c; and the function Sec(n_i, n_j) is an indicator function, which returns $true$ if node n_i and n_j has a secure communication link and $false$ otherwise.

In Fig. 4.8 we illustrate the weight assignment procedure. In this figure, the

Algorithm 2 The Weighted Centroid Algorithm

```
Input: sensor location $(x_0, y_0)$, neighbor locations $\{(x_i, y_i)\}_{i=1}^{Nb}$
Output: movement vector $\vec{v}$
Compute Voronoi cell $V$
Generate uniform grid points $\{(x_i, y_i)\}_{i=1}^{N} \in V$
Assign weight $\{w_i\}_{i=1}^{N}$ using Alg. 3
Compute updated location $(x_0', y_0')$ using (4.8)
Compute the movement vector $\vec{v} \longleftarrow [(x_0', y_0') - (x_0, y_0)]$
/* adjustment for stability */
if $|\vec{v}| > R_s/2$ then
  $\vec{v} \longleftarrow R_s\vec{v}/(2|\vec{v}|)$
end if
```

Algorithm 3 The Weight Assignment Procedure

```
Input: neighbor locations $\{\mathbf{x_i}\}_{i=1}^{Nb}$, sampling points $\{\mathbf{p_i}\}_{i=1}^{N}$, $R_c$, and $w_{sec}$
Output: weight vector $\{w_i\}_{i=1}^{N}$
$w_i \longleftarrow 1$ for $1 \le i \le N$
for $i = 1$ to $Nb$ do
  for $j = 1$ to $N$ do
    if Sec$(n_i,\, n_c)$ and $0.7R_c \le d(\mathbf{x_i}, \mathbf{p_j}) \le 0.95R_c$ then
      $w_i \longleftarrow w_i + w_{sec}$
    end if
  end for
end for
```

center node is shown as a square, its neighbor nodes are shown as circles, and the node that already has a secure link with the center node is shown as a plus sign. The Voronoi cell is shown as the shaded area. The grid points are shown either as cross or as dots, where a dot indicates that grid point is inside the ring area between radius $0.7R_c$ and $0.95R_c$ of its secure communication neighbor. The weighted centroid is shown as a diamond in Fig. 4.8. We can see that the updated location is within the center of the ring area, at the same time tends to cover more areas in the Voronoi cell.

The choice of the ring area to be within $[0.7R_c, 0.95R_c]$ is due to the joint consideration of sensing and communications. When the center node is far away from its neighbor, the ring-based weighting tend to pull the center node towards its neighbor. When the center node is too close to its neighbor, the ring-based weighting tend to push the center node away from its neighbor. Thus this weight assignment maintains the communication connectivity between the center node and its neighbors, at the same time avoids too much wasteful overlaps between their sensing regions.

Simulation Results and Discussions

We study the performance of the weighted centroid (WTC) algorithm using several experiments. We compare it with the performance of the MinMax algorithm proposed in [120], which is known as one of the best schemes in sensor location updating for improving sensing coverage.

In Fig. 4.9 and Fig. 4.10 we compare the sensing coverage ratio and the average number of secure links per node achieved by the proposed WTC algorithm and by MinMax, with respect to the normalized sensing/communication radius.

We set communication radius $R_c = 2R_s$. The initial deployment uses hexagon lattice and the key pre-distribution uses location-based scheme. Each node has preloaded 100 keys. Both algorithms are run locally by each sensor for four iterations. In these experiments we have excluded all boundary nodes, which allows the results to be interpreted as the expected performance in an infinitely large deployment field. The comparison results can be summarized as follows: (1) when the sensing/communication radius is small, MinMax out perform WTC in both sensing coverage and the average number of secure links; (2) as the sensing/communication radius becomes moderately large, WTC outperforms MinMax in both performance categories; (3) when the sensing/communication radius becomes large enough, the performances of the two schemes will converge.

These results can be interpreted from resource allocation and optimization perspectives [103]. The minmax criterion employed by the MinMax algorithm emphasizes *fairness*, *i.e.*, even when a point in Voronoi cell is very far away from the current sensor location, the location adjusting algorithm tries to cover that point. In contrast, the criterion employed by WTC emphasizes *efficiency*. It tries to minimize the weighted square distance from the sensor node to all points in its Voronoi cell, which is a more greedy philosophy compared to the minmax criterion. In sensor networks, the sensing and communication range are valuable resources to be allocated to the deployment field. The results shown in Fig. 4.9 and Fig. 4.10 indicate that, with a resources-scarce situation (relative to the resource needed for a full coverage/connectivity), the MinMax is a better criterion, with a moderate enough amount of resources, WTC outperforms MinMax. To quantify the demarcation for resource-scarce and resource-abundant situations, we note that in the ideal hexagon lattice, the normalized sensing radius needs to be

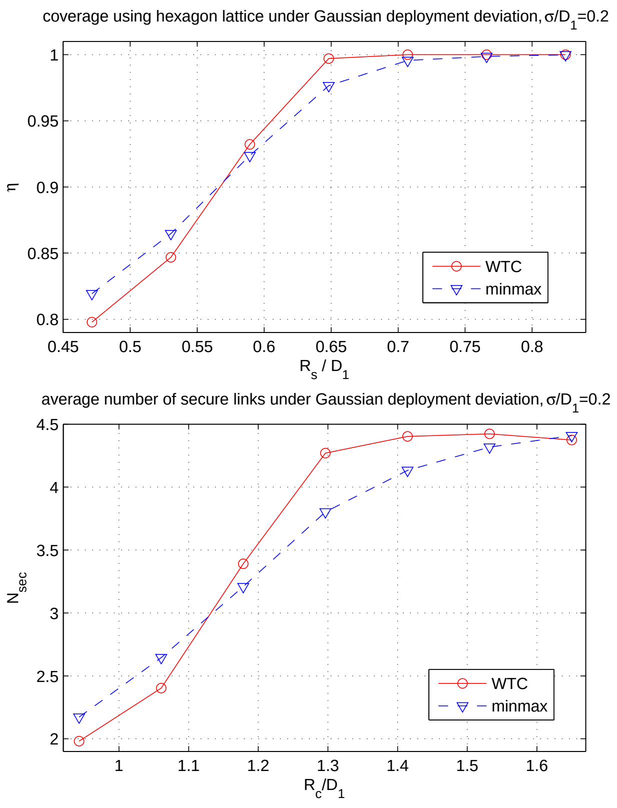


Figure 4.9: Comparison of the weighted centroid and minmax algorithm, small Gaussian deployment deviation, hexagon lattice deployment and location-based key pre-distribution.

at least $R_s/D_1 = \sqrt{2/\sqrt{27}} \approx 0.62$ to achieve full coverage, and the normalized communication radius needs to be at least $R_c/D_1 = \sqrt{2/\sqrt{3}} \approx 1.07$ to achieve full connectivity with the neighbors, which is a pre-requisite for establishing secure links. As shown in Fig. 4.9 and Fig. 4.10, usually the proposed WTC outperforms the MinMax algorithm when the normalized sensing and communication radius are beyond their respective thresholds of 0.62 and 1.07. In practical situations, since the resource budgets are known prior to the design of sensor networks, dynamically determining which criterion to use will best serve the purpose of improving sensing coverage and establishing secure links.

In a separate experiment, we simulated the WTC and MinMax algorithms under random deployment with uniform distribution over the entire field. We place a total of 49 sensors into a 60×60 area and use the basic key pre-distribution scheme for establishing secure links. In this experiment, as it is not possible to exclude the boundary nodes in calculating the average node degree, the average number of secure links drops significantly when compared to the previous lattice-based experiments. In spite of the change in accounting the performance, the simulation results presented in Fig. 4.11 shows the same trend as Fig. 4.9 and Fig. 4.10 in that, the WTC algorithm achieves better performances in both performance categories in the resource-abundant situations, and performs worse than the MinMax in the resource-scarce situations.

Implementation Issues

In the WTC algorithm, the grid points are chosen to discretize the computation of the centroid instead of using a continuous integration over the Voronoi cell. As power consumption is a major concern in sensor networks, the grid points

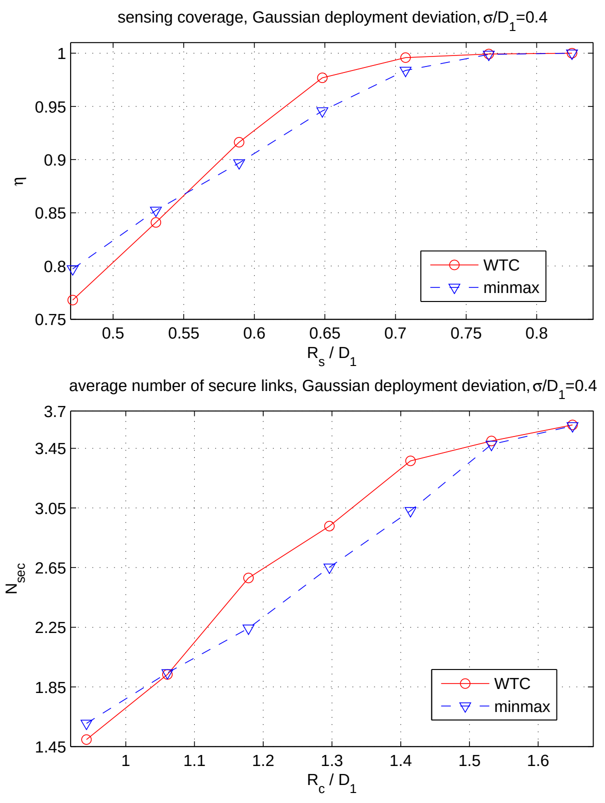


Figure 4.10: Comparison of the weighted centroid and minmax algorithm, large Gaussian deployment deviation, hexagon lattice deployment and location-based key pre-distribution.

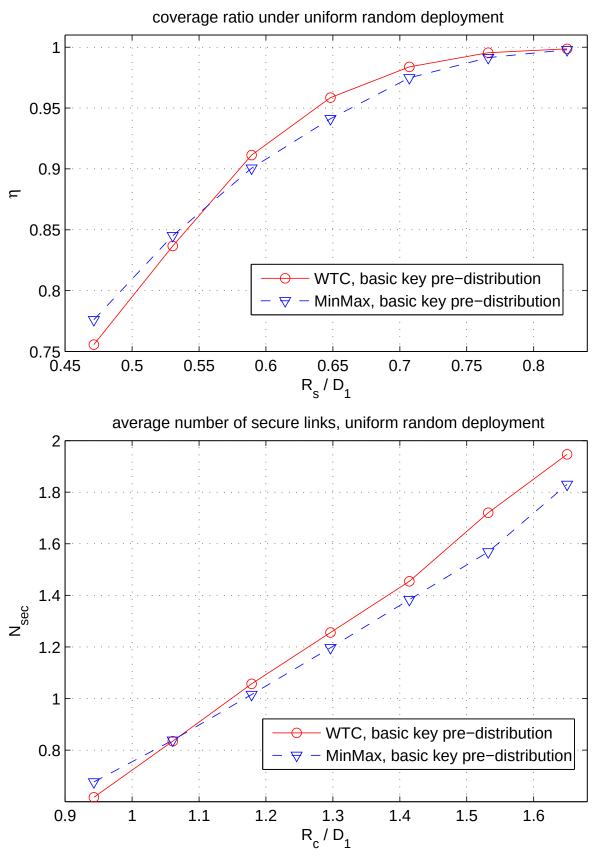


Figure 4.11: Comparison of the weighted centroid and minmax algorithm, uniform random deployment with basic key pre-distribution.

can be chosen as sparse or dense according to the power budget, thus trading off computation accuracy with energy. If the only goal of location adjustment is to maximize the coverage, the same weight can be assigned to all w_i's, *i.e.*, $w_i = 1$ for all points.

Unlike the VFSec algorithm, the WTC is more suitable to be performed by individual sensors. This is because computing both the Voronoi cell and the weighted centroid can be done locally. The simulation results have shown that the locally computed location updates using WTC and MinMax can outperform the location updates run by a cluster head using the VFA and VFSec. However, using the grid-based method in computing the weighted centroid, performing WTC generally requires more computation than performing schemes based on virtual force. One way to reduce the computation is to decompose the weighted centroid in Algorithm 2 into a weighted summation of several uniform-weight centroid problem, and compute the centroid of uniform-weighted polygon (or ring area) using the close-form solution provided in [22] and other related work. We plan to further look into this aspect in our future work.

4.4 Chapter Summary

In this chapter, we have investigated the impact of sensor deployment on the performance of sensing coverage and secure connectivity. For static sensor deployment, we have investigated the hexagon and square lattice topology and compared them with the random deployment. We show that the two lattice topology exhibits range-dependent performance and there is no all-time winner in the context of secure connectivity. For designing secure sensor networks, the system parameters, such as sensing and communication range, should be jointly considered with the deployment topology.

When sensor locations can be adjusted after the initial deployment, we have proposed two sensor location updating algorithms, the VFSec and the WTC algorithm, to jointly optimize sensing covering and secure connectivity. The simulation results show that the WTC algorithm outperforms the existing algorithms in both secure connectivity and sensing coverage under moderate to abundant node density, while VFSec achieves a superior tradeoff in both performance categories than the existing virtual force based algorithms.

4.5 Appendix: A Model for Link Compromise Probability

In this appendix, we analyze the link compromise probability in the location-based key pre-distribution scheme in [30] and compare it with that of the basic scheme [34].

Let us denote the size of the key pool in the basic scheme by P, and there are x nodes compromised. For a given link e in the basic scheme, it has been shown in [34] that the probability that the link e is compromised is

$$P_{basic} = \Pr(e|x) = 1 - (1 - \frac{m}{P})^x.$$

Next, we consider in the group-based scheme [30], the size of the group key pool is S and the number of sensor groups is N. We require the total number of distinct keys in the group based scheme to be the same as in the basic scheme. As each distinct key appears in exactly two group key pools, we have $P = (NS)/2$. In the group-based scheme, suppose the given link e uses key K_e. This key K_e is in the key pools of exactly two groups, denoted by G_1 and G_2. Only when a compromised node $n^{(c)}$ is from one of the two groups, the link e can potentially be compromised by $n^{(c)}$. When the compromised nodes are i.i.d. uniformly distributed among all groups, denoting the probability that a compromised node $n^{(c)}$ falls into group G_1 or G_2 by p, we have

$$p = \Pr(n^{(c)} \in \{G_1 \cup G_2\}) = 2/N.$$

In the group-based scheme, the probability that link e is compromised given x nodes are compromised is

$$\begin{aligned} P_{group} &= \Pr(e|x) \\ &= \sum_{k=0}^{x} \Pr(e|(k \text{ out of } x) \in \{G_1 \cup G_2\}) \cdot \\ &\quad \Pr((k \text{ out of } x) \in \{G_1 \cup G_2\}) \\ &= \sum_{k=0}^{x} [1 - (1 - \frac{m}{S})^k] \binom{x}{k} p^k (1-p)^{x-k}. \end{aligned}$$

For function $f(\epsilon) = (1-\epsilon)^a$ with $\epsilon \approx 0$, the first-order approximation at $\epsilon = 0$ using Taylor expansion is $f(\epsilon) \approx 1 - a\epsilon$. As the key pool size P and the group key pool size S are much larger than the key ring size m, both $\frac{m}{P}$ and $\frac{m}{S}$ are close to zero, we can apply first-order approximation to both $(1 - \frac{m}{S})^k \approx 1 - \frac{m}{S}k$ and $(1 - \frac{m}{P})^x \approx 1 - \frac{m}{P}x$. Thus we arrive at

$$\begin{aligned} P_{basic} &\approx 1 - (1 - x \cdot \frac{m}{P}) \approx x \cdot \frac{m}{P} \\ P_{group} &\approx \sum_{k=0}^{x} \frac{m}{S} k \binom{x}{k} p^k (1-p)^{x-k} \qquad (4.9) \\ &= mpx/S \end{aligned}$$

Since we have $S = 2P/N$ and $p = 2/N$, substituting these into Eqn. (4.9), we obtain

$$P_{group} \approx x \cdot \frac{m}{P} \approx P_{basic}.$$

This shows that the link compromise probability of the basic scheme and the group-based scheme are approximately the same with the fixed key ring size m.

Chapter 5

Optimizing Time Efficiency in Contributory Group Key Management

The complexity of communication security protocols is often determined by multiple factors such as the type of cryptographic primitive used, the connectivity and reliability of the communication links, the number of user who intend to communicate, etc. Among these factors, the scalability of communication security protocol with regard to the user group size is especially important. Such scalability will greatly influence the system efficiency and user satisfaction, and emerging communication applications may involve user groups ranging from thousands to millions [42][69][88][129]. In this chapter, we introduce scheduling and optimization techniques into the design phase of a special class of communication security protocol, group key management protocols. We demonstrate that with these techniques, we can significantly improve the scalability of contributory group key management, hence improve system efficiency and satisfy users' demands.

An important aspect of communication security is content confidentiality and access control [82], which becomes a necessity in a wide range of applications, including bank transactions, teleconferencing, and data collection in sensor networks [87][34]. For secure group-oriented applications, access control is a challenging task due to the potentially large group size and dynamic membership. To achieve confidentiality in group communications, usually a key known to all group members is used to encrypt the communication content [13][51]. This key is usually referred to as the *group key* [42, 88, 129]. In a group with dynamic membership, the group key needs to be updated upon each user's join to prevent the new user from accessing the past communications. Similarly, upon each user's departure, the group key needs to be updated to prevent the leaving user from accessing the future communications. Thus group members need to agree upon the same key management protocol for key establishment and update. Sometimes the group key management protocol is also referred to as the *group key agreement*.

In Chapter 4, we have seen the probabilistic key pre-distribution in sensor networks, which does not guarantee that two nodes who try to communicate share a secret key. In deterministic key management, a group key management scheme follows either a centralized or a contributory approach and can maintain a shared key between information sender and recipient. The centralized approach uses a central key server to generate and distribute keys for all group members [17][119][129], whereas in the contributory approach, each group member contributes his/her own share to the group key [105, 54, 29]. Since contributory schemes do not rely on a central key server, they become necessary in situations where: (a) a central key server cannot be established, such as in Ad Hoc networks, (b) group members do not trust another entity to manage their private keys, or (c) members and server do not share any common knowledge about each other's secret keys beforehand. Contributory schemes remove the need of the key server at the expense of performing computationally expensive cryptographic primitives, such as modular multiplication and exponentiation [51][33]. This poses a challenge to the design of efficient key agreements.

In the literature, many group key management protocols have been proposed [129, 88, 42, 119, 17, 105, 54, 29, 107, 81, 80, 6, 118, 114, 106, 144, 68]. The early designs of contributory key agreements mostly consider the efficiency of key establishment [45][104][8]. Among them, Ingemarsson *et al.* first introduced a conference key distribution system (CKDS) based on a ring topology [45]. Later, Burmester and Desmedt proposed a key distribution system (BD) that takes only three rounds to generate a group key [11]. Steiner *et al.* extended the two-party Diffie-Hellman (DH) protocol and proposed group Diffie-Hellman protocols GDH.1/2/3 [104]. Becker and Willie studied the minimum communication complexity of contributory key agreements and proposed the *octopus* and 2^d-*octopus* protocols [8], which have proven optimality for key establishment. While achieving efficiency in key establishment, most of these early schemes encounter high rekeying complexity in either member join or departure. Recent research on key management became more aware of the scalability issue. As a means to improve scalability, tree-based approach for group rekeying was first presented in the centralized scenario by Wallner *et al.* in [119] and Wong *et al.* in [129], independently. Later, tree-based schemes were also proposed for the contributory setting by Kim *et al.* in their TGDH scheme [54], and by Dondeti *et al.* in their DISEC scheme [29]. The tree-based schemes use a logical *key tree* to organize the keys belonging to the group members and achieve a rekeying complexity of $O(\log n)$ [129][54][29][114], where

n is the group size. In addition, [54] and [29] also pointed out that the rekeying cost is related to both the key tree structure and the location of member join or departure in the key tree, and suggested a balanced key tree to reduce the rekeying cost based on heuristics. In [144], Zhu *et al.* proposed two schemes to optimize the rekeying cost in centralized key management. The key tree structure is reorganized according to the temporal patterns of the group members, or the packet loss probability along the route from the key server to each member.

In this chapter, we investigate the time efficiency of contributory key agreement. The time efficiency is measured by the processing time in group key establishment and update. In order to participate in the group communications, a joining user has to wait until the group keys are updated. Since computing cryptographic primitives and exchanging rekeying messages are time-consuming, such waiting time is not negligible. Similarly, the amount of time needed to recompute a new group key reflects the latency in user revocation. Thus from a quality of service (QoS) perspective, the rekeying time cost is directly related to users' satisfaction and a system's performance. Traditionally, the rekeying time complexity is analyzed only for one join or departure event. The design rationale of our scheme is to look into the combination of multiple events, and optimize the time cost over the dynamics of group membership. To improve the time efficiency, we design a new key tree topology with join and exit subtrees, which are small subtrees located close to the root of the key tree. With this key tree topology, we propose a set of algorithms to handle the key update for join and leave events. In particular, we show through analysis that the sizes of join and exit trees should be at the log scale of the group size. The resulting scheme is called *Join-Exit Tree (JET) Group Key Agreement.* Analytical results show that the proposed scheme achieves an average asymptotic time cost of $O(\log(\log n))$ for a join event, and also $O(\log(\log n))$ for a departure event when group dynamics are known *a priori*. In addition to the improved time efficiency, our scheme also has low communication and computation complexity.

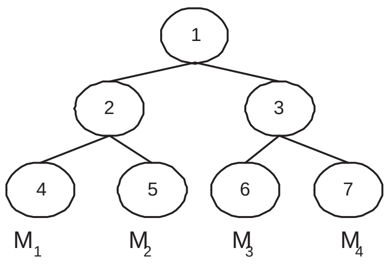
5.1 Efficiency Aspects in Contributory Key Agreement

5.1.1 Background on Tree-based Contributory Key Management

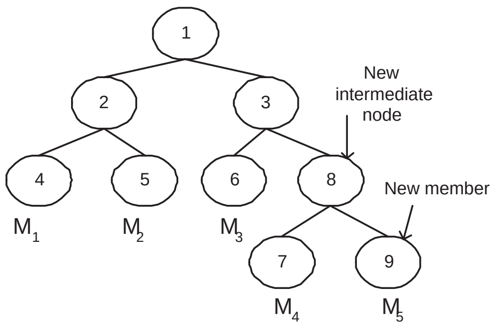
We briefly review rekeying operations for join and leave events in tree-based contributory key agreements [29][54], which use the two-party DH protocol [28] as a basic module.

In a tree-based key agreement, three types of keys are organized in a logical key tree, as illustrated in Fig. 5.1(a). The leaf nodes in a key tree represent the private keys held by individual group members. The root of the tree corresponds to the group key. All other inner nodes represent subgroup keys, each of which is held by the group members that are descendants of the corresponding inner node. We denote the i-th group member by M_i, and the key associated with the j-th node in the key tree by K_j. In addition, g and p are the exponentiation base and the modular base for the DH protocol, respectively.

To establish a group key, the keys in the key tree are computed in a bottom-up fashion. Users are first grouped into pairs and each pair performs a two-party DH to form a sub-group. These sub-groups will again pair up and perform the two-party DH to form larger sub-groups. Continuing in this way, the final group



(a) a key tree



(b) user join

Figure 5.1: Notations for a key tree.

key can be obtained. An example is shown in Fig.5.1(a) with four group members, and member M_i has private key r_i. The group key K_1 corresponding to node 1 is computed in two rounds as

$$K_1 = g^{(g^{r_1 r_2} \bmod p)(g^{r_3 r_4} \bmod p)} \bmod p.$$

In a user join event, the new user will first pair up with an insertion node, which could be either a leaf node or an inner node, to perform a two-party DH. Then all the keys on the path from the insertion node to the tree root are updated recursively. An example is shown in Fig.5.1. When member M_5 joins the group, node 7 in Fig.5.1(a) is chosen as the insertion node. Then M_4 (node 7) and M_5 (node 9) perform a DH key exchange to generate a new inner node 8 in Fig. 5.1(b), followed by the key updates on the path *node* 8 $\rightarrow$ *node* 3 $\rightarrow$ *node* 1.

Upon a user's departure, the leaving user's node and its parent node will be deleted from the key tree. Its sibling node will assume the position of its parent node. Then all the keys on the path from the leaving user's grandparent node to the tree root are updated from the bottom to the top.

5.1.2 Time-Efficiency Issues in Contributory Key Agreements

The time efficiency of DH-based contributory group key agreement is usually evaluated by the number of rounds needed to perform the protocol during a key update [45, 104, 54, 29]. However, in some schemes, the number of operations may be different in distinct rounds. For example, in GDH.2 [104], i modular exponentiations are performed in the i-th round. To address this problem, the notion of "simple round" was introduced in [8], where every party can send and receive at most one message in each round. In our work, we apply the notion of simple round in the tree-based contributory schemes. In each round, each user can perform at most one two-party DH operation. With the new definition of round, we propose performance metrics for time efficiency below.

Average Join/Leave Time We define the *user join time* as the number of rounds to process key updates for a user join event. The average user join time, denoted by T_{join}, is defined as

$$T_{join} = \frac{R_{join}}{N_{join}}, \tag{5.1}$$

where R_{join} is the total number of DH rounds performed for N_{join} join events. Similarly, the *user leave time* is defined as the number of rounds to process key updates for a user leave event. The average user leave time, denoted by T_{leave}, is defined as

$$T_{leave} = \frac{R_{leave}}{N_{leave}}, \tag{5.2}$$

where R_{leave} is the total number of DH rounds performed for N_{leave} leave events. Let $N = N_{join} + N_{leave}$ and $R = R_{join} + R_{leave}$. The overall average processing time T is defined as

$$T = \frac{R}{N}, \tag{5.3}$$

where T can also be interpreted as a weighted average of T_{join} and T_{leave} as $T = \frac{N_{join}}{N} T_{join} + \frac{N_{leave}}{N} T_{leave}$.

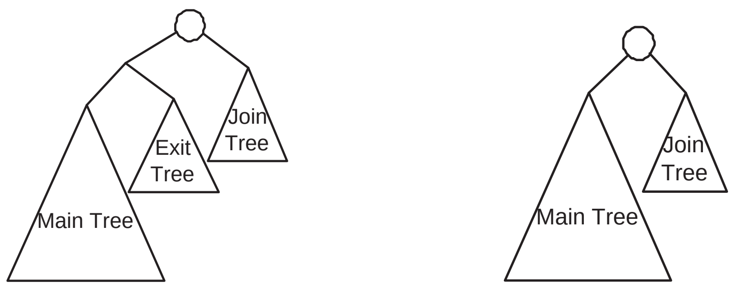
5.1.3 Communication and Computation Efficiency

The communication efficiency of a contributory key agreement refers to the number of messages sent for a key update during a join or leave event. The underlying assumption is that sending each message incurs about the same communication cost. In practice, the size of each message could be different. However, the main cost in sending a rekeying message is the cost in software and hardware to go through the protocol stack and form a packet, along with the cost in networks while routing and transmitting the packet. Similar to the case of time efficiency, we choose the average number of messages per user join or departure as the performance metric for communication efficiency.

In a DH-based contributory group key agreement, the computation of modular exponentiation dominates the total computation cost. Therefore we use the average number of exponentiations per join or departure event as the performance metric for the computation efficiency.

5.2 Join-Exit Tree (JET) Algorithms

In this section, we present a new logical key tree topology and the associated algorithms to achieve better time efficiency in contributory key agreement. As



(a) join, exit, and main tree

(b) join and main tree

Figure 5.2: Topology for the proposed join-exit tree.

shown in Fig. 5.2(a), the proposed logical key tree consists of three parts: the *join tree*, the *exit tree*, and the *main tree*. The proposed key tree is a binary tree built upon the two-party DH protocol. We refer to the key tree in Fig. 5.2(a) as a *join-exit tree* and a key tree without special structures as a *simple key tree*. The prior works have shown that, if a user joins the group at a location closer to the tree root, fewer number of keys need to be updated, thus the join time will be shorter. Similar reasoning applies to user departures. So the join tree and exit trees should be much smaller than the main tree. We define the *join tree capacity* and the *exit tree capacity*, denoted by C_J and C_E, as the maximum number of users that can be accommodated in the join and exit tree, respectively. The number of users in the join tree and the main tree are denoted by N_J and N_M, respectively.

In the proposed scheme, a joining user will first be added to the join tree. Later on, when the join tree reaches its capacity, all users in the join tree will be relocated together into the main tree. In addition, when users' departure time is known, users who are most likely to leave in the near future will be moved in batch from the main tree to the exit tree. The design rationale of the join and exit trees resembles that of memory hierarchy in computer design [43]. Furthermore, the capacities of the join and exit trees can change over time, resulting in a dynamic key tree structure. For example, when there is no user in the exit tree, the key tree reduces to a main tree and join tree topology, as shown in Fig. 5.2(b).

5.2.1 The Join Tree Algorithm

The join tree algorithm consists of four parts: the join tree activation, the insertion strategy, the relocation strategy, and the join tree capacity update. When the group has only a few members, the join tree is not activated. As the group size increases and exceeds a threshold we activate the join tree and choose an initial join tree capacity. Such a threshold condition is referred to as the *activation condition* for the join tree. After the activation, any user joining the group is first inserted to a node in the join tree. The insertion node is chosen according to the *insertion strategy.* When the join tree is full, the members in the join tree are merged into the leaf nodes of the main tree. Such a process is called the *batch relocation.* Since the number of users in the main tree is changed after the batch relocation, the join tree capacity is updated according to a rule that relates the join tree capacity to the main tree user number. According to this rule, the *optimal join tree capacity* in the sense of time efficiency can be computed. We explain these four parts in details below.

User Insertion in the Join Tree

When the join tree is empty and a new user wants to join, the root of the current key tree is chosen as the insertion node. The insertion is done by treating the entire existing group as one logical user, and performing a two-party DH between this logical user and the new user. This process is illustrated in Fig. 5.3, where

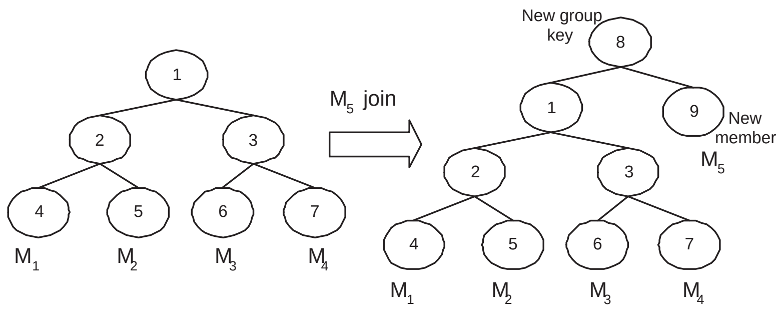


Figure 5.3: User join at the join tree root. Note that the new user M_5 becomes the root of the join tree.

the new user M_5 becomes node 9, the root of the join tree. Member M_5 is paired up with the original root of the key tree (node 1) to perform a DH key exchange and the new group key is established as node 8. When the join tree is not empty, the insertion node is determined by Algorithm 4, where $usernumber(x)$ returns the number of users under a given node x in the key tree. After the insertion node is found, the new member node performs a two-party DH key exchange with the insertion node. Then the keys on the path from the insertion node to the tree root are updated through a series of DH key exchange. Fig.5.4 illustrates the growth of the join tree from one user to eight users using the insertion strategy.

Algorithm 4 Finding the insertion node

```
$x \leftarrow$ *join-tree-root*
**while** $usernumber(x) \neq 2^k$ for some integer $k$ **do**
    $x \leftarrow rightchild(x)$
**end while**
*insertion-node* $\leftarrow x$
```

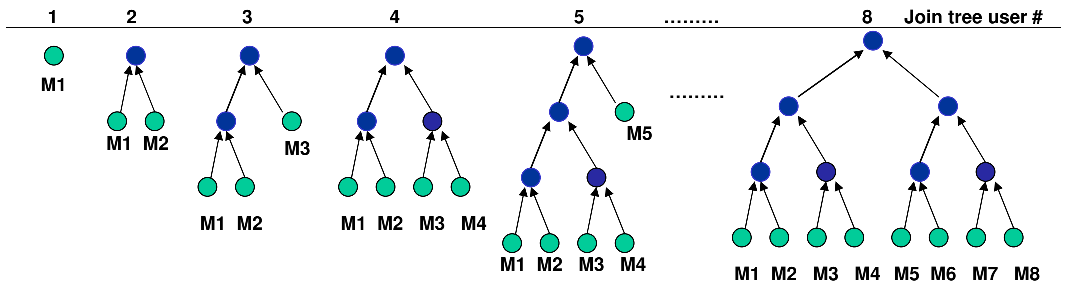


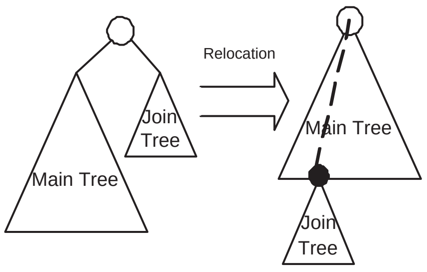
Figure 5.4: Sequential user join strategy (only the join tree is shown).

The Batch Relocation

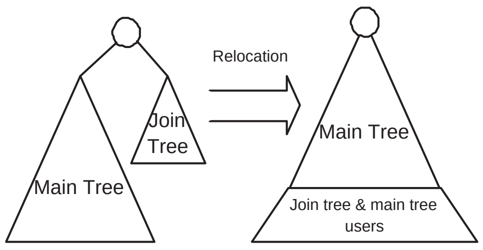
We present two relocation methods that differ in whether the subgroup keys in the join tree are preserved. In the first method, all users in the join tree are viewed as a logical user during relocation, and this logical user is inserted into the shortest-depth leaf node of the main tree. Thus, the subgroup keys among the users in the join tree are preserved. This process is shown in Fig.5.5(a). Then all keys along the path from the insertion node to the tree root are updated, which is indicated by the dash line in Fig. 5.5(a). The reason to choose the shortest branch leaf node in the main tree as the insertion node is to guarantee that the relocation time is at most the log of the main tree size ($\lceil \log N_M \rceil$), because the shortest branch must be smaller or equal to the average length of the branches, which is $\lceil \log N_M \rceil$ [1]. The only exception comes when the main tree is a complete balanced tree, the relocation time is $\log N_M + 1$, because one more level of the key tree must be created to accommodate the new logical user.

In the second relocation method, we find the N_J shortest-depth leaf nodes in the main tree as the insertion nodes for N_J join tree user. These insertion

[1]Throughout this chapter, log stands for base-2 logarithm and ln stands for natural logarithm.



(a) Method 1



(b) Method 2

Figure 5.5: Relocation methods for the join tree.

nodes are found so that the unbalance-ness of the key tree can be alleviated by the relocation process. Then we relocate the join tree users simultaneously to the insertion nodes. The keys on the branches from all original join tree users to the tree root are updated in parallel and finally a new group key is obtained. This process is illustrated in Fig.5.5(b). To analyze the time complexity, we note that this relocation may fill up the empty nodes at the shortest-depth leaf nodes of the main tree. The maximum depth of any relocation path would not exceed $\lceil \log(N_M + N_J) \rceil$. Since the join tree is much smaller than the main tree, the relocation time is upper bounded by $\lceil \log N_M \rceil + 1$.

Although the two relocation methods have similar time complexity, the first method will generally produce a skewed main tree. Since users may leave from a branch longer than the average depth of the key tree, an unbalanced key tree may cause the user departure time to be longer than the case when a balanced key tree is used. The second relocation method helps maintain the balance of the key tree, which reduces the expected cost of leave events [54]. We shall choose the second relocation method in this work because it takes into consideration both the join and leave time cost.

Table 5.1: Latency of Sequential User Join

k	1	2	3	4	5	6	7	8	9	10	...
$r(k)$	1	2	2	3	2	3	3	4	2	3	...

The Optimal Join Tree Capacity

Using the proposed insertion strategy, the user join latency for the k-th user in the join tree is measured as $r(k)$ rounds, which is listed in Table 5.1. We observe a special property of the sequence $r(k)$, namely,

$$r(2^p + q) = 1 + r(q), \quad p \geq 0,\ 0 < q \leq 2^p, \tag{5.4}$$

where p is a non-negative integer, and q a positive integer. For the user join latency $r(k)$ in (5.4), the following inequality holds for any positive integer n, and equality is achieved when n is of power of 2:

$$\frac{1}{n}\sum_{k=1}^{n} r(k) \leq \frac{1}{2}\log n + 1. \tag{5.5}$$

The proof is presented in Appendix 5.7.1.

Consider the average join time for x users joining the group starting form an empty join tree. These x users are inserted into the join tree one by one, then they are relocated together into the main tree. From previous analysis we can see that, when the main tree has N_M users, the average join tree relocation time is $\log N_M$, where we relax the integer value of the tree height to a continuous value to simplify analysis. Taking into account the relocation time, the average join time for these x users is

$$T_{join} = \frac{1}{x}(\sum_{k=1}^{x} r(k) + \log N_M). \tag{5.6}$$

Using (5.5), we obtain

$$T_{join} \leq \frac{1}{2}\log x + \frac{1}{x}\log N_M + 1. \tag{5.7}$$

Since it is not easy to minimize T_{join} directly, we try to minimize its upper bound over x. The optimal join tree capacity C_J that minimizes the upper bound is given by

$$\begin{aligned} C_J &= \arg\min_{x>0}\{\frac{1}{2}\log x + \frac{1}{x}\log N_M + 1\} \\ &= 2\ln N_M \end{aligned} \tag{5.8}$$

The above analysis shows that, for a given number of main tree users N_M and the insertion rule specified by Algorithm 4, the optimal join tree capacity C_J is $2\ln N_M$. Since between two consecutive join tree relocations, the main tree size is fixed at N_M, the join tree capacity should also be fixed during this time at $C_J \approx 2\ln N_M$ and the average join time is upper bounded by

$$T_{join} \leq \frac{1}{2}\log\log N_M + \frac{3}{2} + \frac{1}{2\ln 2} - \frac{1}{2}\log\log e. \tag{5.9}$$

This upper bound indicates that on average, a user needs to spend only $O(\log(\log n))$ rounds for a rekeying operation in user join, where n is the group size. We note that this asymptotic performance is not affected by the variation of the relocation time, because the relocation time of around $\log N_M$ rounds is averaged over $\log N_M$ join events, contributing approximately only one round to the average join cost. This validates the use of the approximate average relocation time $\log N_M$ in the above analysis.

For the joining users, since they can start to communicate once they are inserted into the join tree, their waiting time do not include the relocation time of $\log N_M$ rounds. We refer to the waiting time for the joining users as *user join latency.* We can see that the average user join latency, L_{join}, is also upper bounded as

$$L_{join} \leq \frac{1}{2}\log(\log N_M) - \frac{1}{2}\log\log e + \frac{3}{2}.$$

The Join Tree Activation

To decide whether to activate the join tree, we compare the average join time with and without employing the join tree. For a key tree structure with join tree, adding each user in the join tree incurs at most a time cost of $\log C_J$ rounds. Consider the average user join time for C_J users when the join tree changes from empty to full, followed by a batch relocation of $\log N_M$ rounds. The average join time for these C_J users satisfies

$$T_{join} \leq \log C_J + (\log N_M)/C_J. \tag{5.10}$$

If a simple key tree with only a main tree is used, the average join time would be at least $\log N_M$. Consequently, a reduction in time cost can be obtained by using the join tree when the following inequality holds,

$$\log C_J + (\log N_M)/C_J \leq \log N_M,$$

or equivalently,

$$\log N_M \geq \frac{C_J}{C_J - 1}\log C_J. \tag{5.11}$$

We can see that when the number of users in the group is large enough, a join tree should be activated to reduce the average join time. In Appendix 5.7.2, we show that when $C_J = 2\ln N_M$, the inequality (5.11) is satisfied for any $N_M > 8$. Thus we have found a threshold group size $TH_{join} = 8$. When the group size is smaller than or equal to 8, a simple key tree is used. Otherwise, the join tree is activated.

5.2.2 The Exit Tree Algorithm

In some group applications, users can estimate the duration of their staying time according to their own schedule. Such information can help reduce the time cost of rekeying operations in user departure. In the following analysis, we assume that we can obtain accurate information about users' duration of stay. In later sections, the cases of inaccurate or unavailable staying time will be discussed.

Similar to the join tree algorithm, the exit tree algorithm consists of four parts, namely, the activation condition, the batch movement, the user insertion in the exit tree, and the optimization of the exit tree capacity.

The Batch Movement

The *batch movement* refers to the operations to move the users that are likely to leave in the near future from the main tree to the exit tree. The group communications is not interrupted since the old group key can still be used before the batch movement is completed.

A batch movement takes place when there is a user leaving from the exit tree and a batch movement condition is satisfied. Denoting the number of users in the exit tree after the last batch movement as U_p, and the current number of users in the exit tree as U_c, we propose a batch movement condition as

$$U_c \leq \rho U_p, \tag{5.12}$$

where $\rho \in [0, 1)$ is the *exit tree residual rate* (residual rate for short), a predetermined parameter to control the timing of batch movement. In a batch movement, the first B users who are most likely to leave soon are moved to the exit tree, where B is referred to as the *batch movement size.* Starting from an empty exit tree ($U_p = 0$), the number of users in the exit tree after the k-th batch movement will be upper bounded by $\sum_{i=0}^{k-1} \rho^i B$. As k goes to infinity, the number of users in the exit tree converges to the upper bound $B/(1-\rho)$. Therefore the exit tree capacity C_E is related to the batch movement size by

$$C_E = B/(1-\rho). \tag{5.13}$$

We propose to use a priority queue [21] to keep the departure time of all the users in the main tree. This queue is referred to as the *leaving queue*. The users' departure time is obtained from their arrival time and their estimated staying time. The leaving queue will be update under two circumstances. First, after a batch relocation of the join tree, the departure information of the join tree users are added to the leaving queue. Second, after the batch movement of the exit tree, the departure information of the moved users are removed from the leaving queue.

User Insertion in the Exit Tree

The insertion locations for the users being moved into the exit tree are chosen to maintain the balance of the exit tree. For each user insertion, the leaf node with the minimum depth in the exit tree is chosen as the insertion node.

Optimal Exit Tree Capacity

Here we derive the optimal exit tree capacity that minimizes an upper bound of the average leaving time. Suppose that b users are moved together into the exit tree. A batch movement of these b users will incur a time cost of $(\log N_M + 2)$, where $\log N_M$ is the average height of the main tree, and the addition of 2 refers to the additional two levels above the main tree due to the use of the join tree and the exit tree (refer to Fig. 5.2(a)). If the exit tree capacity is x, each user leaving from the exit tree will incur at most a time cost of $(\log x + 2)$. Thus the average user leave time for these b users is bounded by

$$T_{leave} \leq \frac{1}{b}(\log N_M + 2) + (\log x + 2). \tag{5.14}$$

Using (5.13), $b = x(1-\rho)$, and minimizing the right hand side of (5.14) we obtain

$$\begin{aligned} C_E &= \arg\min_x \left\{ \frac{1}{(1-\rho)x}(\log N_M + 2) + (\log x + 2) \right\} \\ &= \frac{\ln N_M + 2\ln 2}{(1-\rho)}. \end{aligned} \tag{5.15}$$

When the capacity of the exit tree is computed as in (5.15), the average leave time is bounded by

$$T_{leave} \leq \log(\log N_M + 2) + \delta, \tag{5.16}$$

where $\delta = 2 - \log(1-\rho) + \log e - \log\log e$. Combining (5.15) and (5.13), we have

$$B = \ln N_M + 2\ln 2. \tag{5.17}$$

A few comments should be made to provide more insights from the above analysis. First, the batch movement size B is only determined by the number of users in the main tree, and independent of the residue rate ρ. Second, there are actually only two parameters, B and ρ, in our system, since the exit tree capacity is a function of B and ρ as in (5.13). Third, with perfect departure information, the average leave time is bounded by $O(\log\log n)$, where n is the group size, and the residue rate ρ should be set to 0 to minimize the upper bound in (5.16). However, in practice, the choice of ρ is a tradeoff. When ρ is 0, a batch movement cannot be performed unless the exit tree is completely vacant. If some users inaccurately estimate their departure time and stay in the exit tree for a long period of time, no other users can utilize the exit tree during that period. When ρ is close to 1, batch movements are frequently performed, resulting in a large overhead. Based on experimental heuristics, we suggest setting ρ to around 0.5.

The Activation of Exit Tree

The average leave time using a simple key tree with N_M users is $\log N_M$. Comparing this result with the upper bound in (5.14), a reduction in the average leave time can be obtained if

$$\frac{1}{(1-\rho)C_E}(\log N_M + 2) + (\log C_E + 2) \leq \log N_M. \tag{5.18}$$

Using (5.15), we simplify the above condition as

$$\log N_M \geq \log C_E + \log e + 2. \tag{5.19}$$

Similar to the case of the join tree activation, we can prove that when the exit tree capacity is chosen as in (5.15), the inequality (5.19) is satisfied for any $N_M > 256$. Thus we have found a threshold group size $TH_{leave} = 256$. When the group size is larger than this threshold, activating the exit tree can reduce the average leave time.

5.3 Group Key Agreement Based on Join-Exit Tree

In this section we present a protocol suite of the *Join-Exit Tree (JET) Group Key Agreement*, which consists of a key establishment protocol, a user join protocol, and a user leave protocol. These protocols are based on the algorithms we discussed in the previous section.

5.3.1 Group Key Establishment

Many prior works [114][45] assume that all group members are available before starting the group communications, thus parallel computation can take place to establish a group key. We refer to this situation as *concurrent user join*. In reality, there are situations when members join the group sequentially, and we refer to them as *sequential user join*. The proposed JET scheme treats the key establishment in these two types of situations differently. For concurrent user join, subgroup keys in the key tree are computed in a bottom-up fashion in parallel to obtain the final group key, as in [114]. For sequential user join, we use the join protocol (as discussed below) to handle the sequential key updates. The join tree is activated when the group size exceeds the activation threshold $TH_{join} = 8$, but the exit tree will not be activated during the key establishment stage.

5.3.2 Join Protocol

The key update for a user join event follows the next few steps:

1. Choose an insertion node in the key tree.

(a) Before the join tree is activated, Algorithm 4 is used in the simple key tree to choose the insertion node.

(b) After the join tree is activated, when inserting the new user according to Algorithm 4 will not make the join tree height more than $\lceil \log C_J \rceil$, the insertion strategy in Algorithm 4 is followed. Otherwise, the insertion node will be chosen as the leaf node with the minimum depth in the join tree. (When there are user departures from the join tree, this helps keep the join tree balanced.)

2. The insertion node and the new member perform a two-party DH key exchange. Then all the keys on the path from the insertion node to the root are updated subsequently.

3. Adjust the key tree topology and parameters according to the rules specified as follows:

(a) When the group size is larger than $TH_{join} = 8$, the join tree is activated. When the group size is larger than $TH_{leave} = 256$, the exit tree is activated.

(b) When the join tree becomes full after a join event, users in the join tree are relocated into the main tree using the relocation strategy in Section 5.2. Additionally, the departure information of those users who can report their staying time is stored in the leaving queue.

(c) Update the join and exit tree capacities according to Eqn. (5.8) and Eqn. (5.15), respectively.

5.3.3 Leave Protocol

The exit tree residual rate is set to $\rho = 0.5$. The key update for a user leave event follows the next few steps:

1. Delete the leaving user node and its parent node. Promote the leaving user's sibling node to their parent node's position. Mark the keys on the path from the leaving user's grandparent node to the tree root as to be updated later.

2. When the user is leaving from the main tree and there are also users in the join tree, perform a join tree relocation. Mark the keys to be updated for relocation.

3. Update all the keys marked in step 1 and 2 from the bottom to the top of the key tree.

4. When the user is leaving from the exit tree and the batch movement condition is satisfied, perform a batch movement as specified in Section 5.2.

5. Perform the updates for key tree management as follows:

(a) Remove the leaving user's departure information if it is in the leaving queue.

(b) Compute the new join and exit tree capacities according to Eqn. (5.8) and Eqn. (5.15), respectively. If the newly-computed join/exit tree capacity becomes larger than the current number of users in the join/exit tree, the join/exit tree capacity is updated immediately. Otherwise, no update is done.

(c) When the main tree user number N_M falls below the threshold for join/exit tree activation and the join/exit tree is empty, the join/exit tree is deactivated.

5.4 Experiments and Performance Analysis

In this section, we present three simulations. The first simulation focuses on group key establishment, in which we consider sequential user join. The second and third simulation have both join and departure activities. In each simulation, the performance of our proposed scheme is compared with that of TGDH scheme [54], a typical tree-based contributory key agreement.

5.4.1 Key Establishment for Sequential User Join

For sequential user join, the proposed JET protocol uses a simple key tree for small group size, and activates the join tree when the group size is larger than 8. The exit tree will not be activated. We compare the average join time for sequential user join using the proposed JET and TGDH [54] in Fig. 5.6. It can be seen that JET achieves the same performance as TGDH when the group size is

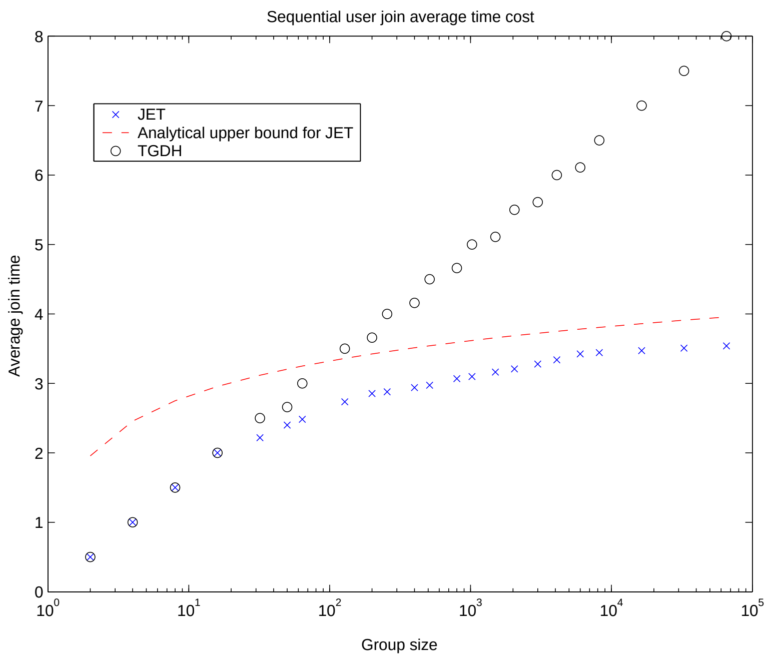


Figure 5.6: Average time cost for sequential user join.

small, and outperforms TGDH when the group size becomes large. Regarding the asymptotic performance, TGDH achieves an average time cost of $O(\log n)$, while the proposed JET scheme achieves $O(\log(\log n))$. The dashed line in Fig. 5.6 shows the theoretical upper bound for the average time cost from (5.9).

5.4.2 Experiment Using MBone User Activity Data

In this simulation, we choose three user activity log files from three Multicast Backbone (MBone) multicast sessions [116] as user activity for our simulation. Two of these three sessions are NASA space shuttle coverage and the other one is CBC News World online test [2].

[2]The sources of these MBone sessions are: (1) NASA-space shuttle STS-80 coverage, video, starting time 11/14/1996, 16:14:09; (2) NASA-space shuttle STS-80 coverage, audio, starting

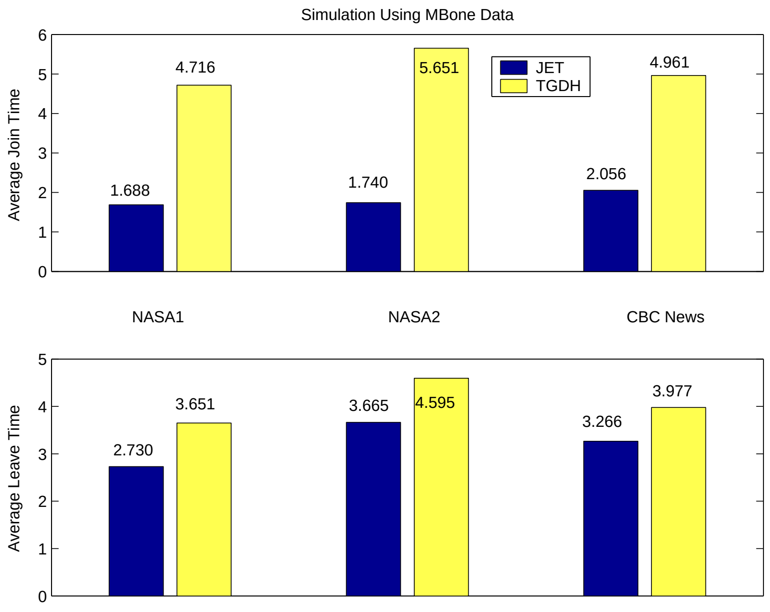


Figure 5.7: Average join and leave time for simulations using MBone data.

Fig. 5.7 shows the experimental results using JET and TGDH scheme, where we can see that JET has about 50% improvement over TGDH in user join, and about 20% improvement in user departure. It is worth noting that the improvement in user departure is not resulted from the use of the exit tree, since all the three sessions have maximum group size below 100 and the exit tree is not activated. From the study of the MBone multicast sessions, Ammeroth *et al.* observed that the MBone multicast group size is usually small (typically 100-200), and users either stay in the group for a short period of time or a very long time [4][3]. Using the proposed JET scheme, the exit tree will not be activated for a small group size. However, when a user stays in the group for only a short period of time, it is highly likely that this user joins and leaves the group in the join tree without

time 12/4/1996, 10:54:49; (3) CBC Newsworld on-line test, audio, starting time 10/29/1996, 12:35:15.

Table 5.2: Statistical Parameters for User Behavior

Duration	0-199	200-499	500-4499	4500-5000
λ_i	7	5	2	1
m_i	2500	500	500	500
Characteristic	long stay	short stay		

getting to the main tree. Thus the use of the join tree reduces both the user join time and the user leave time.

5.4.3 Experiments Using Simulated User Activity Data

In this experiment, we generate user activities according to the probabilistic model suggested in [4]. The duration of simulation is 5000 time units and is divided into four non-overlapping segments, T_1 to T_4. In each time segment T_i, users' arrival time is modelled as a Poisson process with mean arrival rate λ_i and users' staying time follows an exponential distribution with mean value m_i. The values of λ_i and m_i are listed in Table 5.2. The initial group size is 0. The simulated user activities consist of about 12000 join and 10900 leave events. The maximum group size is approximately 2800 and the group size at the end of simulation is about 1100.

In practice, users' accurate staying time will not always be available. To model the inaccuracy in users' *estimated staying time* (EST), we consider three classes of users. The first class of users do not report EST, the second class of users reports accurate EST, and the third class of users reports inaccurate EST. In the third class, the EST for user i is modelled as a random variable with Gaussian distribution $N(\mu_i, \sigma_i^2)$, and the mean value μ_i is the actual staying time [3]. We also

[3]Because of the Gaussian distribution, a user could report a negative staying time. Such a

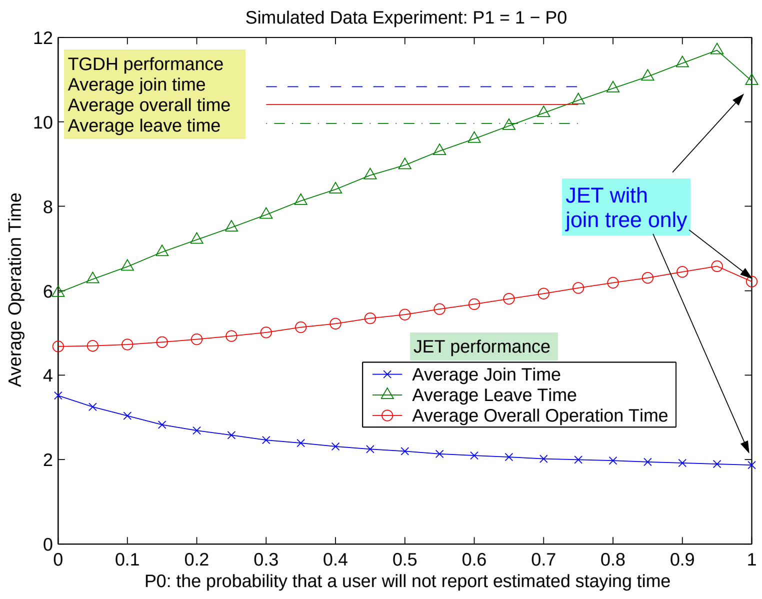


Figure 5.8: Average join, leave and overall time costs for the first experiment using simulated data. A user either do not report EST with probability P_0, or reports accurate EST with probability $P_1 = 1 - P_0$.

assume that in the third class, the ratio of the standard deviation σ_i to the mean μ_i of the EST is constant across the users, and is denoted by $R = \sigma_i/\mu_i$. The probability that a user is in the first, second, and third classes is denoted by P_0, P_1, and $(1 - P_0 - P_1)$, respectively.

In the first experiment, we consider that a user either does not report EST or reports an accurate EST, *i.e.*, $P_1 = 1 - P_0$. By varying the value of P_0, the average join and leave time costs are shown in Fig. 5.8, where the average leave time increases with P_0 almost linearly. The only exception is the data point at $P_0 = 1$, *i.e.*, when no user reports EST. When $P_0 = 1$, the average join, leave, and overall time costs are: $T_{join} = 1.87$, $T_{leave} = 10.97$, and $T = 6.22$. This is the

case is treated as EST unavailable.

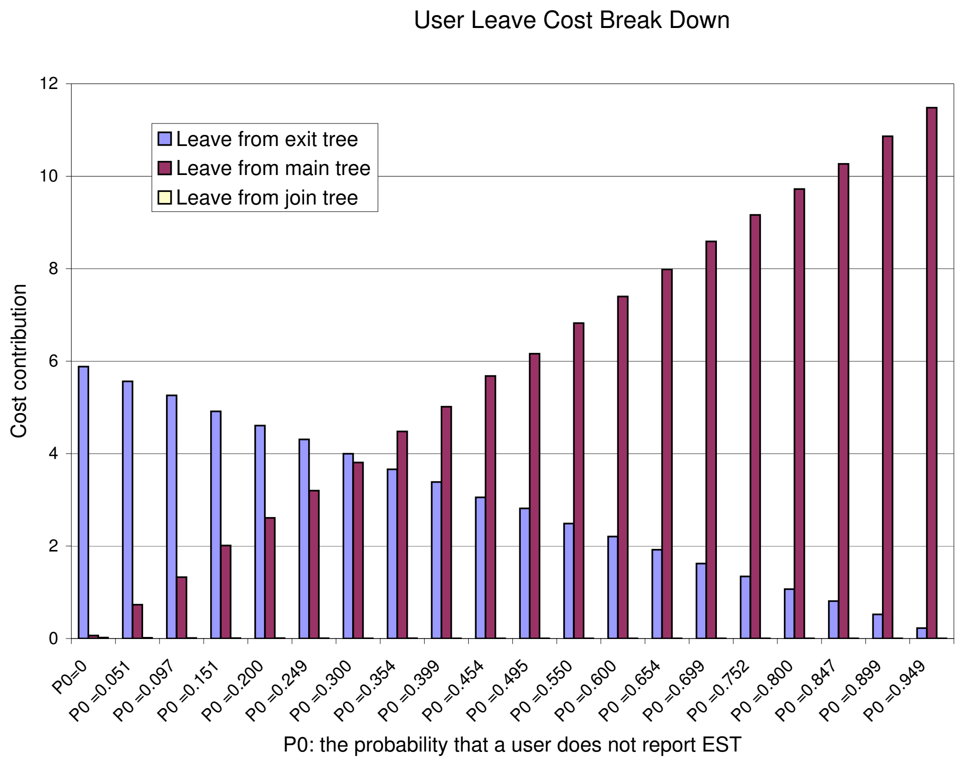


Figure 5.9: A breakdown of the contributions to the user leave time in Fig. 5.8. Shown in the figure is the contribution to the user leave time by users leaving from the exit tree, the main tree, and the join tree. These contributions (in rounds) is plotted against P_0, with $P_1 = 1 - P_0$.

situation when the exit tree is not activated during the group lifetime. From these data, we can see that when the exit tree is not used, the average overall time cost is equal to or lower than the costs when $P_0 \geq 0.8$. This is because activating the exit tree increases the depth of the key tree by one. When P_0 is large, a large portion of users without departure information cannot take advantage of the exit tree, and the overhead of the exit tree structure outweighs its benefit. The benefit of the exit tree is substantial as long as more than 30% (corresponding to $P_0 = 0.7$) or more users report accurate EST. We have also compared the performance of JET with that of TGDH in Fig. 5.8, where the performances of TGDH are shown as horizontal lines because they do not vary with probability P_0. We can see that JET always outperform TGDH in terms of the overall time cost and the join time cost. For user leave time cost, JET will outperform TGDH as long as more than 35% of the users report accurate EST.

The average leave time presented in Fig. 5.8 consists of three parts, namely, the cost of users leaving from the exit tree, from the main tree, and from the join tree, respectively. We illustrate these three parts in Fig. 5.9. In particular, to obtain the first part, we obtain the average leave time for users leaving from the exit tree; then we multiply it with the percentage of user departures from the exit tree with respect to the total number of user departure events. The other two parts can be computed similarly and the average leave cost is the summation of these three parts. We can see that when P_0 is small, the user leave time is dominated by the time cost of the users leaving from the exit tree. As P_0 increases, the user leave time is gradually dominated by the time cost of users leaving from the main tree. In this experiment, most users will stay in the group for a non-trivial period of time, therefore the percentage of users leaving from the join tree is very small.

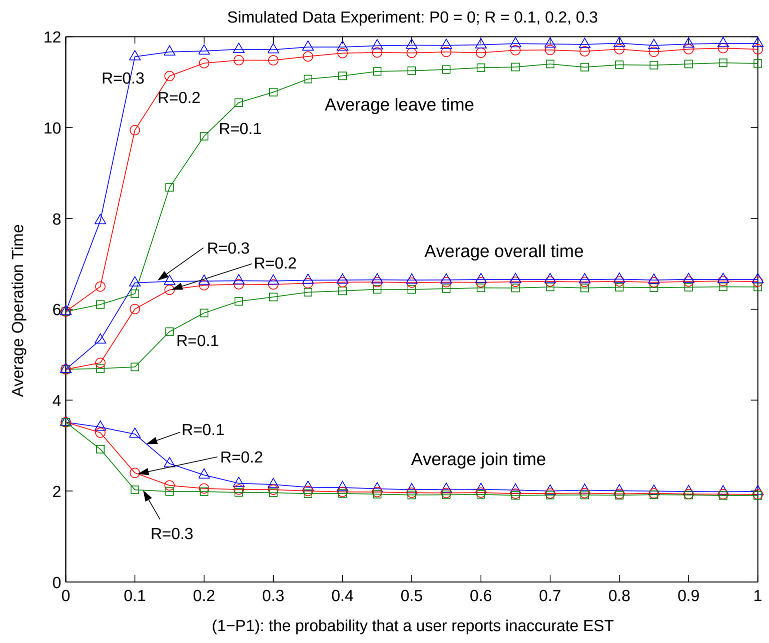


Figure 5.10: Average join, leave and overall time costs for the second experiment using simulated data. A user either reports accurate EST with probability P_1, or reports inaccurate EST with probability $1-P_1$. For inaccurate EST, the deviation parameter $R \in \{0.1, 0.2, 0.3\}$.

In the second experiment, we consider that all users will report EST ($P_0 = 0$). The degree of deviation is at $R \in \{0.1, 0.2, 0.3\}$, and P_1 varies in the range of [0,1]. The average join and leave time under different P_1 values are plotted in Fig. 5.10. We can see that when the proportion of inaccurate estimates $(1 - P_1)$ is small, the proposed JET scheme can achieve good time efficiency in both join and leave events. However, the average leave time is sensitive to the change in R value, especially in the range where $(1 - P_1)$ is small, where the gain obtained by using the exit tree diminishes quickly with the increase of R.

In the third experiment, all users report inaccurate EST, which corresponds to $P_0 = P_1 = 0$. The average join and leave time costs are simulated when the value

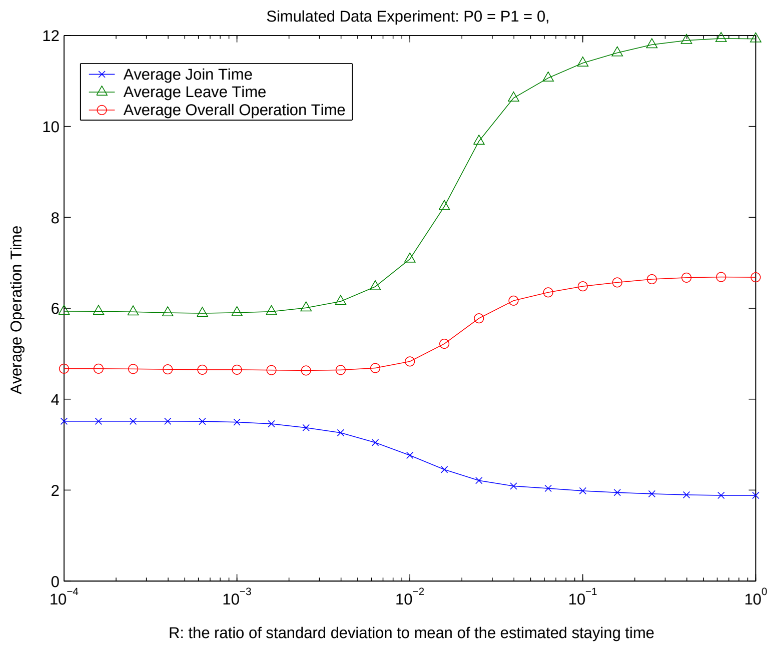


Figure 5.11: Average join, leave and overall time costs for the third experiment using simulated data. All users report inaccurate EST. The deviation parameter $R \in [10^{-4}, 1]$.

of R is in the range of [0,1]. Fig. 5.11 shows that, when the standard deviation is two orders of magnitude smaller than the true staying time ($R \leq 0.01$), the proposed JET scheme can efficiently manage both user join and leave events. We also note from Fig. 5.10 and Fig. 5.11 that, when a large portion of users do not report accurate estimation, the advantage of the exit tree diminishes.

Table 5.3 lists the average and the worst case time costs for JET with both join and exit tree (when $P_0 = 0$, $P_1 = 0.1$ and $R = 0.1$), JET using only the join tree, and TGDH. All the worst case time costs do not change with the simulation parameters (P_0, P_1, and R). Comparing the performance of JET using only the join tree, which does not depend on users' reported EST, with that of JET using

Table 5.3: Simulated Data Experiment: $P_0 = 0,\ P_1 = 0.1,\ R = 0.1$

	average			worst case	
	join	leave	overall	join	leave
JET (join tree only)	1.87	10.97	6.22	13	13
JET (join-exit tree)	3.25	6.34	4.73	14	14
TGDH	10.83	9.96	10.41	12	12

both join and exit tree, we can see that the exit tree indeed provides a reduction in terms of the average overall time cost, even when around 10% of users report inaccurate EST. Comparing these time costs with those of TGDH, we can see that the proposed JET scheme can improve time efficiency in terms of the average time costs, while tolerating a small amount of inaccuracy in EST. However, for a group of size n, the worst case operation time of JET using only the join tree is $\lceil \log n \rceil + 1$, and that of JET using both join and exit tree is $\lceil \log n \rceil + 2$. These worst case time costs are one and two more rounds than that of TGDH, respectively. This is due to the increased depth by the join and exit tree structure, which we refer to as the *structural overhead* of the JET scheme.

Two observations can be made from the above experiments. First, regardless of the accuracy in EST, the join tree scheme can improve the time efficiency for join events. Second, although the inaccuracy in EST comes in different forms (no EST or inaccurate EST), the overall operation time is not very sensitive to the change of experiment parameters P_0, P_1, and R. This is because inaccurate EST leads to user departures from the main tree. When users leave from the main tree, we simultaneously relocate the users from the join tree to the main tree (refer to Section 5.3.3). As such, part of the join tree relocation cost can be amortized by the leave cost. Such an amortized cost can be counted either toward the join time or toward the leave time. In this chapter, we have counted it toward the leave time. Therefore when we see a cost increase in user leave events, we will often see a cost reduction in user join events, which will partially offset the overall cost increase in the overall efficiency.

5.5 Discussions

5.5.1 Extension to Multi-Level Join Tree

The idea of caching the joining and leaving users, as in the design of memory hierarchy, can be extended to multiple levels. Here we illustrate an extension of the one-level join tree to a two-level join tree. We consider a new join tree topology, where a smaller join tree is attached directly to the tree root, a larger join tree is attached one level lower from the tree root with the main tree as its sibling sub-tree. We refer to the smaller join tree as Level-1 join tree and the larger one as Level-2 join tree, respectively. Such a topology can be visualized as in Fig. 5.2(a), where the exit tree is replaced by the Level-2 join tree. We note that the exit tree does not exist in this topology.

We compare the average join time using a two-level join tree with that using a one-level join tree, and try to find the condition under which the two-level join tree has advantage over the one-level join tree. From our analysis, the smallest group size that can benefit from a two-level join tree is around or greater than 180. Compared to the activation group size of 8 for the one-level join tree, the result shows that the two-level join tree would improve the rekeying time efficiency when the group grows larger.

5.5.2 The Implementation of Key Agreement

In this subsection, we discuss two implementation methods for the proposed JET protocol with and without a group coordinator. We show that these two implementations will give the same time cost.

In the first case we consider using a *group controller* in the implementation of JET. In [105], group controller was suggested to be one of the group members who knows all group membership information and facilitates adding and excluding members. However, the group controller does not have the knowledge of the secret keys of other members and therefore would not violate the security requirements of the contributory key management. The joining/leaving user will send a request to the group controller. Then the group controller sends a broadcast message specifying the change in the logical key tree, including the insertion node ID, adjustment of the key tree structure, etc. For each join tree relocation or batch movement to the exit tree, the group controller will also send a broadcast message to specify where each user will be relocated to. The group controller would have a storage overhead proportional to the group size as it needs to store the topology of the key tree. In addition, the group controller has a communication overhead of one broadcast message per join or leave event and one broadcast message for each batch relocation, which takes place infrequently. When the current group controller leaves the group, another member in the group is chosen as the new group controller and the related information is passed from the leaving group controller to the new one. The group controller can also be a non-member entity.

The second implementation does not require a group controller. Instead, each group member stores the topology of the key tree and follows the JET protocol. Thus all users can achieve consistent action for key update. During a join event, the joining user will broadcast a request to the whole group. All members will find the same insertion node according to the insertion strategy, then a *sponsor* is chosen as the rightmost leaf node in the subtree rooted at the insertion node [54]. Since the sponsor knows all the subgroup keys along the path from the insertion node to the root, the sponsor will perform a two-party DH key exchange with the new user, compute the keys on the path from the insertion node to the tree root, and then broadcast the blinded keys on this path. The blinded keys are the results of exponentiation base g raised to the power of the secret keys, which can be public known. Such a procedure is detailed in [54]. When a leave event occurs, the sponsor is chosen as the rightmost leaf node of the subtree rooted at the leaving user's sibling node, and a key update procedure similar to that of a join event takes place. Since all users have the same view of the key tree structure, they can also cooperate in the update operations of the key tree, such as the batch relocation/movement.

An important factor in the rekeying cost is the depth of the joining/leaving node, d. In the first implementation, d rounds of DH key exchange will be performed in key update, which has $2d$ exponentiations and d message transmissions in total. in the second implementation, the sponsor needs to compute d un-blinded keys, d blinded keys, and send d messages. Therefore, the two implementations have the same total computation and communication cost, as well as the same time complexity. But in the second implementation, half of the computation load and almost all communication load are on the sponsor. As for the storage overhead, in the first implementation, only the group controller needs to store the structure of the key tree, whose size is proportional to the group size. In the second implementation, each user needs to store a copy of the key tree structure and the total

Table 5.4: A Comparison of Rekeying Protocol Complexity (group size n)

	Key update time in group lifetime		Rekeying message overhead (# of messages)	
	for one user	for entire system	with multicast	without multicast
JET	$O(n)$	$O(n \log(\log n))$	$O(\log n)$	$O(n)$
TGDH	$O(n \log n)$	$O(n \log n)$	$O(\log n)$	$O(n)$
GDH.2	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n^2)$
	Computation overhead (# of exponentiations)			
	for one user	for all users		
JET	$O(\log(\log n))$	$O(n)$		
TGDH	$O(\log n)$	$O(n)$		
GDH.2	$O(n)$	$O(n^2)$		

storage overhead grows proportional to the square of the group size.

5.5.3 Protocol Complexity

In this subsection, we provide a more comprehensive comparison of the protocol complexity between JET and TGDH. The complexity aspects we consider are the rekeying time cost during a group lifetime, the messaging overhead, and the computation overhead. These results are summarized in Table 5.4.

Time Complexity from Other Perspectives

Our previous discussions have been focused on the time cost from individual user's perspective and on a per event basis. Two additional aspects can help evaluate the overall efficiency of key management from a system perspective. One is the amount of time a user spends on key update during his/her lifetime in the group, and the other is the amount of time the whole group spends on key update during the lifetime of the group.

Consider a sequence of n join events followed by n leave events. We assume that the first user joining the group is also the last one to leave the group. In JET, such a user will spend most of the life time in the main tree for key management purpose. On average, this user will spend 2-round time for key update with each user join event and 3-round time with each user leave event, assuming all users report their staying time accurately. Therefore this user has spent $\Theta(n)$ rounds in total on key update during the life time. Since this user has the longest life time among all users, $O(n)$ is the upper bound for any user's total key update time. For tree-based key agreement using a simple key tree, this first-come-last-leave user will spend $O(n \log n)$ rounds in total on key update.

If we consider a group of n users as a whole, for the same sequence of events described above, the group will spend $O(n \log(\log n))$ rounds in key update using JET. If a key agreement using a simple key tree is employed, the time cost will be $O(n \log n)$. Compared to TGDH, we note that the improvement from a system perspective is not as significant as that from a user's perspective.

Communication Complexity

During member join and leave, a joining/leaving member should send a join/leave request. Afterwards, in the rekeying process, at least two messages will be sent for each DH key exchange. This messaging overhead is the communication cost of the rekeying protocol.

We now discuss the average number of messages for user join and leave events in JET protocol. In the first scenario, we consider that multicast is available. In particular, if a message needs to be sent to m users, sending one multicast message would suffice. In this case, the average number of messages is $O(\log n)$ for both join and leave events. In the second scenario, we consider that multicast is not available. If a message needs to be sent to m users, m duplicate copies of the same message must be sent. In this case the average number of messages is $O(n)$ for both user join and leave event. From Table 5.4 we can see that, the rekeying message overhead in JET is comparable to those in TGDH.

Computation Complexity

In the proposed JET protocol, the total number of exponentiations performed by all users is $O(n)$ during the key update for a join or leave event. Such a measurement captures the overall computation load of the entire group.

During each join or leave event, the number of exponentiations performed by any individual member is less than or equal to two times the number of DH rounds. Therefore for any single user, the average number of exponentiations is also $O(\log(\log n))$ per join/departure event.

5.6 Chapter Summary

In this chapter, we have presented a new contributory key agreement, known as the Join-Exit-Tree Group Key Agreement, for secure group communications. Built upon tree-based group key management, the proposed scheme employs a main tree as well as join and exit subtrees that serve as temporary buffers for joining and leaving users. To achieve time efficiency, we have shown that the optimal subtree capacity is at the log scale of the group size and designed an adaptive algorithm to activate and update join and exit subtrees. As a result, the proposed JET scheme can achieve an average time cost of $O(\log(\log n))$ for user join and leave events in a group of n users, and reduces the total time cost of key update over a system's life time from $O(n \log n)$ by prior works to $O(n \log(\log n))$. In the meantime, the proposed scheme also achieves low communication and computation overhead. Our experimental results on both simulated user activities and the real MBone data have shown that the proposed scheme outperforms the existing tree-based schemes by a large margin in the events of group key establishment, user join, and user departure for large and dynamic groups, without sacrificing the time efficiency for small groups.

5.7 Appendix: Derivations

5.7.1 Derivation for Inequality (5.5)

In this appendix, we prove the inequality (5.5) in Section 5.2:

$$\frac{1}{A} \sum_{k=1}^{A} r(k) \leq \frac{1}{2} \log A + 1, \tag{5.20}$$

where $r(1) = 1$, $r(2^p + q) = 1 + r(q)$, p is a non-negative integer, and $q \in [1, 2^p]$ is a positive integer. The equality holds when A is a power of 2.

We first use induction to show that when $A = 2^p$, $p = 0, 1, 2, ...$, the equality holds.

When $A = 1$, LHS = RHS = 1.

Next, we assume the equality holds for $A = 2^p$, namely,

$$\frac{1}{2^p} \sum_{k=1}^{2^p} r(k) = \frac{1}{2} \log 2^p + 1. \tag{5.21}$$

Consider the case of $A = 2^{p+1}$.

$$\begin{aligned}
LHS &= \frac{1}{2^{p+1}} \sum_{k=1}^{2^{p+1}} r(k) \\
&= \frac{1}{2^{p+1}} \left(\sum_{k=1}^{2^p} r(k) + \sum_{k=1}^{2^p} (r(k)+1) \right) \\
&= \frac{1}{2^{p+1}} \left(2 \cdot (\frac{1}{2} \log 2^p + 1) 2^p + 2^p \right) \qquad (5.22) \\
&= \frac{1}{2} \log 2^{p+1} + 1 = RHS,
\end{aligned}$$

where (5.22) is obtained using the induction assumption (5.21).

We now prove the inequality for any positive integer A. It is obvious to see that inequality is true for $A = 1, 2$. By induction, suppose that the inequality is true for all $1 \leq A < 2^p + q$, and we consider $A = 2^p + q$, where $0 < q \leq 2^p$.

$$\begin{aligned}
LHS &= \frac{1}{A} \sum_{k=1}^{A} r(k) \\
&= \frac{1}{A} \left(\sum_{k=1}^{2^p} r(k) + \sum_{k=1}^{q} (r(k)+1) \right) \\
&\leq \frac{1}{A} [(\frac{1}{2} \log 2^p + 1) 2^p + q(\frac{1}{2} \log q + 1) + q] \qquad (5.23) \\
&= \frac{1}{2} \left\{ \frac{1}{A} (2^p \log 2^p + q \log q + 2q) \right\} + 1, \qquad (5.24)
\end{aligned}$$

where (5.23) is obtained using the induction assumption.

To prove that (5.24) $\leq \frac{1}{2} \log A + 1$ is equivalent to prove

$$\frac{2^p}{A} \log 2^p + \frac{q}{A} \log(4q) \leq \log A. \qquad (5.25)$$

Applying the identity $\log k = \log e \cdot \ln k$ and $\ln k = \int_1^k \frac{1}{x} dx$, (5.25) can be written as an integration form

$$\log e \left\{ \frac{2^p}{A} \int_1^{2^p} \frac{1}{x} dx + \frac{q}{A} \int_1^{4q} \frac{1}{x} dx \right\} \leq \log e \int_1^A \frac{1}{x} dx$$

$$\Leftrightarrow 2^p \int_{2^p}^{A} \frac{1}{x} dx + q \left[\int_{1}^{A} \frac{1}{x} dx - \int_{1}^{4q} \frac{1}{x} dx \right] \geq 0 \tag{5.26}$$

We denote $B = 2^p$ and fix p (hence B is fixed). Thus $A = B + q$. It is straightforward to see that (5.26) holds when $B + q \geq 4q$, or $1 \leq q \leq \frac{B}{3}$.

When $B/3 \leq q \leq B$, (5.26) is equivalent to

$$\frac{2^p}{A} \int_{2^p}^{A} \frac{1}{x} dx - \frac{q}{A} \int_{A}^{4q} \frac{1}{x} dx \geq 0. \tag{5.27}$$

Since q is the only variable in (5.27), let $f(q)$ be the LHS of (5.27), and consider $f(q)$ as a continuous function of q

$$f(q) = \frac{B}{B+q} \int_{B}^{B+q} \frac{1}{x} dx - \frac{q}{B+q} \int_{B+q}^{4q} \frac{1}{x} dx,$$

where $q \in [B/3, B]$. Taking the derivative of $f(q)$, we get

$$\frac{d}{dq} f(q) = -\frac{B}{(B+q)^2} \int_{B}^{4q} \frac{1}{x} dx < 0. \tag{5.28}$$

In previous proof we showed that the equality of (5.20) holds when A is power of 2, i.e. $f(B) = 0$. We also showed that $f(q) > 0$ for $1 \leq q \leq \frac{B}{3}$. Since $f(B/3) > 0$, $f(B) = 0$, $f(q)$ is continuous on $[B/3, B]$ and $f'(q) < 0$, we must have $f(q) > 0$ on $[B/3, B]$. Thus (5.26) also holds for $B/3 \leq q \leq B$. This completes the proof.

5.7.2 Derivation for Inequality (5.11)

In this appendix, we prove the inequality (5.11) holds for any $x > 8$,

$$\log x > \frac{2 \ln x}{2 \ln x - 1} \log(2 \ln x). \tag{5.29}$$

Let $y = \ln x$. We consider the case when the group size is larger than 1, so $x \in (1, +\infty)$, and $y \in (0, +\infty)$. Under such a condition, (5.29) becomes

$$2y - 1 > 2 \ln 2 (1 + \log y). \tag{5.30}$$

Let $g(y) = (2y - 1) - 2\ln 2(1 + \log y)$. Function $g(y)$ havs two zeros at $y_0 = 1/2$ and $y_1 \approx 1.7564$. In addition, $g(y) > 0$ for any $y > y_1$. Therefore (5.29) holds for any $x > e^{y_1} \approx 5.7917$. In our proposed protocol, we choose a larger threshold value 8 as eight users lead to a balanced main tree.

Chapter 6

Conclusions and Future Perspectives

In this dissertation, we have investigated the potential, the mechanism, and the performance of incorporating signal processing techniques, as well as beneficial knowledge from other related fields of study, into different layers of a secure communication system. The fusion of different technical disciplines can take place at physical layer communications for tracing the adversary, at application layer for confidentiality protection and content authentication for multimedia, at system deployment phase of sensor networks for jointly optimizing sensing coverage and secure communication, and at protocol design phase for improving key management efficiency and user satisfaction. Such an interdisciplinary and cross-layer approach for application security is important in meeting the demands of emerging communication paradigms and multimedia applications.

In Chapter 2, we have looked into the unique issues related to multimedia encryption and content authentication in the multimedia communication scenario. We have proposed atomic encryption operations for multimedia that can preserve standard compliance and are friendly to delegate processing, provided quantitative analysis on the security and bit-rate overhead, and presented a systematical study on how to strategically integrate different encryption operations to build a video encryption system. For media authentication, we have adapted the concept of unicity distance to evaluate the security of image hashing, discovered the potential key disclosure problem for popular image hashing schemes, and proposed mitigation solutions.

In chapter 3, we focus on the security issues in cooperative wireless communication systems. We have discovered the threat of signal garbling attack from compromised relay nodes, and propose a countermeasure to trace and pinpoint the adversarial relay. The proposed tracing scheme employs an adaptive signal detection algorithm, coupled with an statistical tracing symbol aggregation scheme, and can pinpoint the malicious relay with very high accuracy and low bandwidth overhead.

In Chapter 4, we have investigated the impact of sensor deployment on the performance of sensing coverage and secure connectivity in sensor networks. We have provided analysis for static sensor deployment scenario, and proposed two sensor location adjustment algorithms for mobility-assisted sensor deployment. Our results show that by jointly optimizing sensing coverage and secure communication connectivity, we can achieve a better and more balanced performance in both performance categories.

In Chapter 5, we have studied the scalability of contributory key management and proposed a new scheme, the Join-Exit Tree (JET) key management scheme, for time-efficient key management. To improve time efficiency, optimization and scheduling techniques, together with cryptographic primitives, are integrated into the proposed key management protocol. The time efficiency for user join and departure event is significantly improved from $O(\log n)$ to $O(\log(\log n))$, while achieving high communication and computation efficiency.

Throughout this dissertation, we have shown that the various security issues in a multi-layer communication system can be better solved by instilling signal processing concept into the design of application security. Such a design paradigm can have a broad impact and wide applications, including in other related areas such as confidentiality preserving search and retrieval for multimedia content, digital forensics for communication and multimedia, as well as high performance and trusted computing. The possible future extensions of this dissertation include the following aspects.

In the multimedia security area, the content encryption and authentication techniques introduced in this paper can be jointly employed, and combined with feature domain obfuscation techniques to build a privacy-preserving search and retrieval engine for multimedia data. To ensure the scalability of such system, the system can employ database techniques such as hashing and indexing to achieve efficiency and scalability. Such an media search and retrieval will have the following benefits. Since the data stored in the database are encrypted, when the database is compromised, the attacker cannot observe the clear-text content stored in the database. The search and retrieval can be performed by comparing the hash distance without the decryption of content, which is computation-efficient and time-efficient. With obfuscation and randomization techniques on the hash vector, the content owner can share the searching capability with some authorized users, without revealing the content of the media to unauthorized users.

In the wireless communication security area, our adversary-tracing scheme in cooperative wireless communications focuses on one-hop relay situation. It will be fruitful to extend such a framework to multi-hop relay, and combine network-layer routing information into the adversary tracing scheme. One security problem similar to the adversary-tracing scenario is the security of channel estimation. Usually channel estimation is achieved through the sender sending pre-determined pilot symbols to the receiver. When the sender node is compromised and sends garbled pilot symbols to confuse the receiver, the receiver should be able to detect such malicious behavior. In our preliminary study, we have seen that the statistical behavior of the estimated channel condition from garbled pilot symbols deviates from that of the normal estimation result. Therefore the receiver can use multiple garbled pilot symbols as a "self-reference" to detect channel estimation attack in a probabilistic manner. We envision such an approach to be especially effective under slow-fading channel condition.

BIBLIOGRAPHY

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38:393 – 422, November 2002.

[2] S. M. Alamouti. A simple transmit diversity technique for wireless communications. *IEEE Journal on Selected Areas in Communications*, 16(8):1451–1458, Oct. 1998.

[3] K. C. Almeroth. A long-term analysis of growth and usage patterns in the multicast backbone (MBone). In *Proceedings of the IEEE INFOCOM'00*, volume 2, pages 824–833, March 2000.

[4] K. C. Almeroth and M. H. Ammar. Multicast group behavior in the Internet's multicast backbone (MBone). *IEEE Communications Magazine*, pages 124–129, June 1997.

[5] R. Anderson, J. Backhouse, and *et al.* Open letter to the health select committee. `http://www.dawba.net/healthselectcommittee20020327.pdf`, April 2006.

[6] S. Banerjee and B. Bhattacharjee. Scalable secure group communication over IP multicast. *IEEE Journal on Selected Areas in Communications*, 20(8):1511–1527, Oct. 2002.

[7] J. Baumgartner. Deciphering the ca conundrum. *Communication Engineering and Design*, 2003.

[8] K. Becker and U. Wille. Communication complexity of group key distribution. In *Proceedings of the 5th ACM conference on Computer and Communications Security*, pages 1–6. ACM Press, 1998.

[9] M. Bellare. Practice-oriented provable security. In *Proc. of First International Workshop on Information Security(ISW 97)*, 1997.

[10] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption. In *Proc. of the 38th Symposium on Foundations of Computer Science*, 1997.

[11] M. Burmester and Y. Desmedt. A secure and efficient conference key distribution system. In *Proceedings of the EUROCRYPT'94*, pages 275–286. LCNS 950, 1994.

[12] H. Cai, B. Zeng, G. Shen, and S. Li. Error-resilient unequal protection of fine granularity scalable video bitstreams. In *Proceedings of IEEE ICC 2004*, 2004.

[13] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pinkas. Multicast security: A taxonomy and some efficient constructions. In *Proceedings of the IEEE INFOCOM'99*, pages 708–716, 1999.

[14] J. Cannons and P. Moulin. Design and statistical analysis of a hash-aided image watermarking system. *IEEE Transactions on Image Processing*, 13(10):1393–1408, Oct. 2004.

[15] H. Chan and A. Perrig. Security and privacy in sensor networks. *IEEE Computer Magazine*, 36(10), Oct. 2003.

[16] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. In *Proc. of the 2003 IEEE Symposium on Security and Privacy*, May 2003.

[17] I. Chang, R. Engel, D. Kandlur, D. Pendarakis, and D. Saha. Key management for secure internet multicast using boolean function minimization techniques. In *Proceedings of the IEEE INFOCOM'99*, volume 2, pages 689–698, 1999.

[18] C.-Y. Chong and S. P. Kumar. Sensor networks: evolution, opportunities, and challenges. *Proceedings of IEEE*, 91(8):1247–1256, Aug. 2003.

[19] Federal Trade Commission. Identity theft survey report. `http://www.consumer.gov/idtheft/pdf`, Sept. 2003.

[20] J. H. Conway and N. J. A. Sloane. *Sphere Packings, Lattices and Groups.* Springer, 3rd edition, 1991.

[21] T. H. Corman, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms.* The MIT Press and McGraw-Hill Book Company, second edition, 2001.

[22] J. Cortes, S. Martinez, T. Karatas, and F. Bullo. Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation*, 20(2):243–255, Apr. 2004.

[23] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* John Wiley & Sons, second edition, 1991.

[24] I. J. Cox and J-P. M. G. Linnartz. Public watermarks and resistance to tampering. In *Proceedings of International Conference on Image Processing*, 1997.

[25] R.V. Cox, D.E. Bock, K.B. Bauer, J.D. Johnston, and J.H.Snyder. The analog voice privacy system. *AT&T Technical Journal*, Jan-Feb 1987.

[26] J. Daemen and V. Rijmen. Aes proposal: Rijndael. *http://csrc.nist.gov/CryptoToolkit/aes/rijndael/Rijndael-ammended.pdf.*

[27] P. Dent, G. E. Bottomley, and T. Croft. Jakes fading model revisited. *IEEE Electronic Letter*, 29(13):1162 – 1163, June 1993.

[28] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, November 1976.

[29] L. R. Dondeti and S. Mukherjee. DISEC: a distributed framework for scalable secure many-to-many communication. In *Proceedings of the 5th IEEE Symposium on Computers and Communications*, pages 693–698, 2000.

[30] W. Du, J. Deng, Y. S. Han, S. Chen, and P. K. Varshney. A key management scheme for wireless sensor networks using deployment knowledge. In *Proc. of the 2004 IEEE INFOCOM*, Mar. 2004.

[31] W. Du, J. Deng, Y. S. Han, and P. K. Varshney. A pairwise key predistribution scheme for wireless sensor networks. In *Proc. of the 2003 ACM CCS*, Oct. 2003.

[32] T. Ekman. Analysis of the ls estimation error on a rayleigh fading channel. In *Proc. of IEEE Vehicular Technology Conference - VTC'01*, pages 372–376, 2001.

[33] S. E. Eldridge and C. D. Walter. Hardware implementation of montgomery's modular multiplication algorithm. *IEEE Transactions on Computers*, 42(6):693–699, June 1993.

[34] L. Eschenauer and V. D. Gligor. A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM conference on computer and communications security*, 2002.

[35] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar. Next century challenges: scalable coordination in sensor networks. In *Proceedings of ACM Mobicom'99*, 1999.

[36] D. J. Field, A. Hayes, and R. F. Hess. Contour integration by the human visual system: evidence for a local 'association field'. *Vision Research*, 33(2), Jan. 1993.

[37] J. Fridrich and M. Goljan. Robust hash functions for digital watermarking. In *Proceedings of International Conference on Information Technology: Coding and Computing*, pages 178–183, March 2000.

[38] A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression.* Springer, 1991.

[39] V. Gligor. Security of emergent properties in ad-hoc networks. In *Lecture Notes in Computer Science*, 2005.

[40] H. Hacigümüs, B. R. Iyer, and S. Mehrotra. Efficient execution of aggregation queries over encrypted relational databases. *Lecture Notes in Computer Science 2973*, pages 125–136, 2004.

[41] O. Harmanci and M. K. Mihcak. Temporal synchronization of watermarked video using image hashing. In *Proceedings SPIE - Security and Watermarking of Multimedia Contents*, volume 5681, 2005.

[42] H. Harney and C. Muckenhirn. Group key management protocol (GKMP) architecture. RFC 2094, July 1997.

[43] J. L. Hennessy and D. A. Patterson. *Computer Architecture: a Quantitative Approach.* Morgan Kaufmann publishers, second edition, 1996.

[44] A. Howard, M. J. Matarić, and G. S. Sukhatme. Mobile sensor network deployment using potential fields: a distributed, scalable solution to the area coverage problem. In *Proc. of DARS'02*, June 2002.

[45] I. Ingemarsson, D. T. Tang, and C. K. Wong. A conference key distribution system. *IEEE Transactions on Information Theory*, IT-28(5):714–720, September 1982.

[46] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva. Directed diffusion for wireless sensor networking. *IEEE/ACM Tran. on Networking*, 11(1):2–16, November 2003.

[47] A.K. Jain. *Fundamentals of Digital Image Processing.* Prentice Hall, 1998.

[48] M. Janani, A. Hedayat, T. E. Hunter, and A. Nosratinia. Coded cooperation in wireless communications: Space-time transmission and iterative decoding. *IEEE Trans. on Signal Processing*, 52(2):362 – 371, Feb. 2004.

[49] M. Johnson, P. Ishwar, V. Prabhakaran, D. Schonberg, and K. Ramchandran. On compressing encrypted data. *IEEE Transactions on Signal Processing*, 52(10):2992–3006, 2004.

[50] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein. Energy efficient computing for wildlife tracking: design rradeoffs and early experiences with zebranet. In *Proc. of ACM ASPLOS'02*, Oct. 2002.

[51] P. Judge and M. Ammar. Gothic: A group access control architecture for secure multicast and anycast. In *Proceedings of the IEEE INFOCOM'02*, pages 1547–1556, 2002.

[52] C. Kaufman, R. Perlman, and M. Speciner. *Network Security: Private Communication in a Public World.* Prentice Hall, 2003.

[53] R. Kershner. The number of circles covering a set. *American Journal of Mathematics*, 60:665–671, 1939.

[54] Y. Kim, A. Perrig, and G. Tsudik. Simple and fault-tolerant key agreement for dynamic collaborative groups. In *Proceedings of the 7th ACM Conference on Computer and Communications Security*, pages 235–244. ACM Press, 2000.

[55] D.E. Knuth. *The Art of Computer Programming.* Addison-Wesley, 3rd edition, 1997.

[56] J. N. Laneman, D. N. C. Tse, and G. W. Wornell. Cooperative diversity in wireless networks: Efficient protocols and outage behavior. *IEEE Transactions on Information Theory*, 50(12):3062 – 3080, Dec. 2004.

[57] J. N. Laneman and G. W. Wornell. Distributed space-time-coded protocols for exploiting cooperative diversity in wireless networks. *IEEE Transactions on Information Theory*, 49(10):2415 – 2425, Oct. 2003.

[58] W. Li. Overview of fine granularity scalability in mpeg-4 video standard. *IEEE Trans. on Circuits & Systems for Video Technology*, 11(3):301–317, March 2001.

[59] Y. Li. Pilot-symobl-aided channel estimation for OFDM in wireless systems. *IEEE Trans. on Vehicular Technology*, 49(4):1207 – 1215, July 2000.

[60] K. Lieska, E. Laitinen, and J. Lahteenmaki. Radio coverage optimization with genetic algorithms. In *Proceedings of the IEEE Inter. Symp. on Personal, Indoor and Mobile Radio Communications*, pages 318 – 322, Sep. 1998.

[61] G. Lin and G. Noubir. On link layer denial of service in data wireless lans. *Wiley Journal on Wireless Communications and Mobile Computing*, August 2004.

[62] S. Lin, M. T. Ozsu, V. Oria, and R. Ng. An extendible hash for multi-precision similarity querying of image databases. In *Proceedings of 27th VLDB Conference*, 2001.

[63] D. Liu and P. Ning. Establishing pairwise keys in distributed sensor networks. In *Proc. of the 2003 ACM CCS*, Oct. 2003.

[64] D. Liu and P. Ning. Location-based pairwise key establishment for static sensor networks. In *Proc. of the ACM Workshop on Security of Ad Hoc and Sensor Networks*, Oct. 2003.

[65] T. Lookabaugh and D.C. Sicker. Selective encryption for consumer applications. *IEEE Communications Magazine*, 42(5):124–129, May 2004.

[66] Y. Mao and M. K. Mihcak. Collusion-resistant desynchronization for digital video fingerprinting. In *Proc. of the IEEE International Conference on Image Processing (ICIP)*, Sept. 2005.

[67] Y. Mao and M. K. Mihcak. Digital video desynchronization for collusion-resistant fingerprinting. *IEEE Tran. on Image Processing (under revision)*, June 2006.

[68] Y. Mao, Y. Sun, M. Wu, and K. J. R. Liu. Dynamic join-exit amortization and scheduling for time-efficient group key agreement. In *Proceedings of the IEEE INFOCOM 2004*, volume 4, pages 2617 – 2627.

[69] Y. Mao, Y. Sun, M. Wu, and K. J. R. Liu. Jet: Dynamic join-exit-tree amortization and scheduling for contributory key management. *to appear in IEEE/ACM Transactions on Networking*, Dec. 2006.

[70] Y. Mao and M. Wu. Security evaluation for communication-friendly multimedia encryption. In *Proc. of the IEEE International Conference on Image Processing (ICIP)*, Oct. 2004.

[71] Y. Mao and M. Wu. Coordinated sensor deployment for secure communications and sensing coverage. In *Proc. of ACM CCS/SASN Workshop*, Nov. 2005.

[72] Y. Mao and M. Wu. A joint signal processing and cryptographic approach to multimedia encryption. *IEEE Tran. on Image Processing*, 15(7):2061 – 2076, July 2006.

[73] Y. Mao and M. Wu. Security issues in cooperative communications: Tracing adversarial relay. In *Proc. of IEEE ICASSP*, May 2006.

[74] S. Megerian, F. Koushanfar, M. Potkonjak, and M. B. Srivastava. Coverage problems in wireless ad-hoc sensor networks. In *Proc. of the 2001 IEEE INFOCOM*, 2001.

[75] S. Megerian, F. Koushanfar, G. Qu, and M. Potkonjak. Exposure in wireless ad-hoc sensor networks. In *Proc. of the 2001 ACM MobiCom*, 2001.

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

[77] P. Michiardi and R. Molva. Core: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In *Proc. IFIPCommun. Multimedia Security Conf.*, pages 107–121, Sep. 2002.

[78] M. K. Mihcak and R. Venkatesan. A tool for robust audio information hiding: A perceptual audio hashing algorithm. In *Proceedings of 4th Intl. Information Hiding Workshop*, April 2001.

[79] M. K. Mihçak and R. Venkatesan. New iterative geometric methods for robust perceptual image hashing. In *Proceedings of ACM Workshop on Security and Privacy in Digital Rights Management*, 2001.

[80] S. Mittra. Iolus: a framework for scalable secure multicasting. In *Proceedings of the ACM SIGCOMM'97*, pages 277–288. ACM Press, 1997.

[81] R. Molva and A. Pannetrat. Scalable multicast security in dynamic groups. In *Proceedings of the 6th ACM conference on Computer and Communications Security*, pages 101–112, 1999.

[82] M. J. Moyer, J. R. Rao, and P. Rohatgi. A survey of security issues in multicast communications. *IEEE Network*, pages 12–23, Nov./Dec. 1999.

[83] G. Noubir. On connectivity in ad hoc networks under jamming using directional antennas and mobility. In *Proc. of International Conference on Wired /Wireless Internet Communications.*

[84] A.V. Oppenheim and J.S. Lim. The importance of phase in signals. *Proc. of the IEEE*, 69(5), May 1981.

[85] A. Pal, K. Shanmugasundaram, and N. Memon. Automated reassembly of fragmented images. In *Proc. of International Conference on Multimedia and Expo*, 2003.

[86] MPEG-21 Part-4. *Intellectual Property Management and Protection (working document).* 2001.

[87] S. Paul. *Multicast on the Internet and its applications.* Kluwer academic Publishers, 1998.

[88] A. Perrig, D. Song, and J. D. Tygar. ELK, a new protocol for efficient large-group key distribution. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 247–262, 2001.

[89] D. Pescovitz. Robugs: smart dust has legs. *http://www.coe.berkeley.edu/labnotes/0903/pister.html.*

[90] L.-M. Po and W.-C. Ma. A novel four step search algorithm for fast block motion estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, 6(3):313–317, June 1996.

[91] S. Poduri and G. S. Sukhatme. Constrained coverage for mobile sensor networks. In *Proc. of IEEE Inter. Conf. on Robotics and Automation*, Apr. 2004.

[92] J. G. Proakis. *Digital Communications.* McGraw Hill, fourth edition, 2001.

[93] A. Puri and T. Chen. *Multimedia Systems, Standards, and Networks.* Marcel Dekker, first edition, 2000.

[94] L. Qiao and K. Nahrstedt. Comparison of mpeg encryption algorithms. *Inter. Journal on Computers & Graphics*, 22(3), 1998.

[95] H. M. Radha, M. van der Schaar, and Y. Chen. The mpeg-4 fine-grained scalable video coding method for multimedia streaming over ip. *IEEE Transactions on Multimedia*, 3(1):53–68, Mar 2001.

[96] T. S. Rappaport. *Wireless Communications: Principles and Practice.* Prentice Hall, 1996.

[97] T. S. Rappaport. *Wireless Communications: Principles and Practice.* Prentice Hall, 1996.

[98] O. Rousseaux, G. Leus, P. Stoica, and M. Moonen. Gaussian maximum-likelihood channel estimation with short training sequences. *IEEE Trans. on Wirelss Communications*, 4(6):2945 – 2955, Nov. 2005.

[99] A. Sendonaris, E. Erkip, and B. Aazhang. User cooperation diversity – part I: System description. *IEEE Trans. on Communications*, 51(11):1927 – 1938, Nov. 2003.

[100] A. Sendonaris, E. Erkip, and B. Aazhang. User cooperation diversity – part II: Implementation aspects and performance analysis. *IEEE Trans. on Communications*, 51(11):1939 – 1948, Nov. 2003.

[101] C. E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 1949.

[102] J.M. Shapiro. Embedded image coding using zerotrees of wavelet coefficients. *IEEE Transactions on Signal Processing*, 41(12):3445–3462, 1993.

[103] G. Song and Y. Li. Cross-layer optimization for OFDM wireless networks-part i: theoretical framework. *IEEE Transactions on Wireless Communications*, 4(2):614–624, Mar. 2005.

[104] M. Steiner, G. Tsudik, and M. Waidner. Diffie-hellman key distribution extended to group communication. In *Proceedings of the 3rd ACM conference on Computer and Communications Security*, pages 31–37. ACM Press, 1996.

[105] M. Steiner, G. Tsudik, and M. Waidner. CLIQUES: a new approach to group key agreement. In *Proceedings of the 18th International Conference on Distributed Computing Systems*, pages 380–387, 1998.

[106] B. Sun, W. Trappe, Y. Sun, and K. J .R. Liu. A time-efficient contributory key agreeement scheme for secure group communications. In *Proceedings of the IEEE International Conference on Communications*, pages 1159–1163, 2002.

[107] Y. Sun, W. Trappe, and K. J. R. Liu. A scalable multicast key management scheme for heterogeneous wireless networks. *IEEE/ACM Transactions on Networking*, 12(4):653–666.

[108] A. Swaminathan, Y. Mao, and M. Wu. Image hashing resilient to geometric and filtering operations. In *Proc. of Multimedia Signal Processing Workshop*, Oct. 2004.

[109] A. Swaminathan, Y. Mao, and M. Wu. Security of feature extraction in image hashing. In *Proc. of IEEE ICASSP*, Mar. 2005.

[110] A. Swaminathan, Y. Mao, and M. Wu. Robust and secure image hasing. *IEEE Transactions on Information Forensics and Security*, 1(2):215 – 230, June 2006.

[111] A. S. Tanenbaum. *Computer Networks.* Prentice Hall, 2003.

[112] L. Tang. Methods for encrypting and decrypting mpeg video data efficiently. In *Proceedings of 4th ACM Inter. Conf. on Multimedia*, pages 219–229, Nov. 1996.

[113] V. Tarokh, H. Jafarkhani, and A. R. Calderbank. Spacetime block codes from orthogonal designs. *IEEE Transactions on Information Theory*, 45(5):1456 – 1467, July 1999.

[114] W. Trappe, Y. Wang, and K. J. R. Liu. Resource-aware conference key establishment for heterogeneous networks. *IEEE/ACM Transactions on Networking*, 13(1):134–146, Feb. 2005.

[115] W. Trappe and L. C. Washington. *Introduction to Cryptography with Coding Theory.* Prentice Hall, first edition, 2001.

[116] MBone user activity data. ftp://ftp.cc.gatech.edu/people/kevin/release-data, March 2003.

[117] R. Venkatesan, S. M. Koon, M. H. Jakubowski, and P. Moulin. Robust image hashing. In *Proceedings IEEE International Conference on Image Processing (ICIP)*, pages 664 –666, Sep. 2000.

[118] M. Waldvogel, G. Caronni, D. Sun, N. Weiler, and B. Plattner. The VersayKey framework: Versatile group key management. *IEEE Journal on Selected Areas in Communications*, pages 1614–1631, Sept. 1999.

[119] D. Wallner, E. Harder, and R. Agee. Key management for multicast: Issues and architecture. Internet-Draft draft-wallner-key-arch-00.txt, June 1997.

[120] G. Wang, G. Cao, and T. L. Porta. Movement-assisted sensor deployment. In *Proc. of the 2004 IEEE INFOCOM*, Mar. 2004.

[121] Y. Wang, S. Wenger, J. Wen, and A. Katasggelos. Error resilient video coding techniques. *IEEE Signal Processing Magazine*, July 2000.

[122] Y. Wang, Q-F. Zhu, and L. Shaw. Maximally smooth image recovery in transform coding. *IEEE Transactions on Communications*, 41(10):1544–1551, Oct. 1993.

[123] Z. Wang and A.C. Bovik. A universal image quality index. *IEEE Signal Processing Letters*, 9(3):81–84, March 2002.

[124] Z. Wang, L. Lu, and A.C. Bovik. Video quality assessment based on structural distortion measurement. *Signal Processing: Image Communications*, 19(1), Jan. 2004.

[125] S. Wee and J. Apostolopoulos. Secure scalable streaming enabling transcoding without decryption. In *Proc. of IEEE Inter. Conf. on Image Processing*, Oct. 2001.

[126] J. Wen, M. Muttrell, and M. Severa. Access control of standard video bitstreams. In *Proc. of Inter. Conf. on Media Future*, May 2001.

[127] J. Wen, M. Severa, W. Zeng, M.H. Luttrell, and W. Jin. A format-compliant configurable encryption framework for access control of video. *IEEE Trans. on Circuits & Systems for Video Technology*, 12(6):545–557, June 2002.

[128] R. Williams. *The Geometrical Foundations of Natural Structure: a Book of Design.* Dover Publications, 1979.

[129] C. K. Wong, M. Gouda, and S. S. Lam. Secure group communications using key graphs. *IEEE/ACM Transactions on Networking*, 8(1):16–30, Feb 2000.

[130] C-P. Wu and C.-C. Kuo. Efficient multimedia encryption via entropy codec design. In *Proc. of SPIE Inter. Symposium on Electronic Imaging*, Jan. 2001.

[131] M. Wu, R. Joyce, H-S. Wong, L. Guan, , and S-Y. Kung. Dynamic resource allocation via video content and short-term traffic statistics. *IEEE Trans. on Multimedia*, Jan. 2001.

[132] M. Wu and Y. Mao. Communication-friendly encryption of multimedia. In *Proc. of IEEE Multimedia Signal Processing Workshop (MMSP'02)*, Dec. 2002.

[133] T-L. Wu and S.F. Wu. Selective encryption and watermarking of mpeg video. In *Proc. of Inter. Conf. on Image Science, Systems, and Technology*, 1997.

[134] W. Xu, W. Trappe, Y. Zhang, and T. Wood. The feasibility of lauching and detecting jamming attacks in wireless networks. In *Proc. of ACM MobiHoc'05*, May. 2005.

[135] W. Xu, T. Wood, W. Trappe, and Y. Zhang. Channel surfing and spatial retreats: Defenses against wireless denial of service. In *Proc. of ACM WiSe'04*, Oct. 2004.

[136] N. Yeadon, F. Garcia, D. Hutchison, and D. Shepherd. Continuous media filters for heterogeneous internetworking. In *Proceedings of SPIE, Multimedia Computing and Networking (MMCN'96)*, Jan. 1996.

[137] B.-L. Yeo and B. Liu. Rapid scene analysis on compressed video. *IEEE Trans. on Circuits & Systems for Video Technology*, 5(6):533–544, Dec. 1995.

[138] P. Yin, A. Vetro, B. Liu, and H. Sun. Drift compensation for reduced spatial resolution transcoding. *IEEE Trans. on Circuits & Systems for Video Technology*, 12(11):1009–1020, Nov. 2002.

[139] W. Yu and K. J. R. Liu. Attack-resistant cooperation stimulation in autonomous ad hoc networks. *IEEE Journal on Selected Areas in Communications*, 23(12):2260 – 2271, Dec. 2005.

[140] C. Yuan, B. Zhu, Y. Wang, S. Li, and Y. Zhong. Efficient and fully scalable encryption for mpeg-4 fgs. In *Proc. of IEEE Inter. Symp. on Circuits and Systems*, May 2003.

[141] W. Zeng and S. Lei. Efficient frequency domain video scrambling for content access control. In *Proc. of ACM Multimedia*, Nov. 1999.

[142] S. Zhong, J. Chen, and Y. R. Yang. Sprite: A simple, cheat-proof, creditbased system for mobile ad-hoc networks. In *Proc. of IEEE INFOCOM*, pages 1987–1997, 2003.

[143] Y. Zhou, Y. Zhang, and Y. Fang. Llk: a link-layer key establishment scheme for wireless sensor networks. In *Proc. of the IEEE Wireless Communications and Networking Conference*, Mar. 2005.

[144] S. Zhu, S. Setia, and S. Jajodia. Performance optimizations for group key management schemes. In *Proceedings of the 23rd International Conference on Distributed Computing Systems*, pages 163–171, 2003.

[145] Y. Zou and K. Chakrabarty. Sensor deployment and target localization based on virtual forces. In *Proc. of the 2003 IEEE INFOCOM*, April 2003.