Identifying, Modeling, and Mitigating Attacks in Wireless Ad-Hoc and Sensor Networks

Patrick Tague

A dissertation submitted in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy

University of Washington

2009

Program Authorized to Offer Degree: Electrical Engineering

University of Washington
Graduate School

This is to certify that I have examined this copy of a doctoral dissertation by

Patrick Tague

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Chair of the Supervisory Committee:

_____

Radha Poovendran

Reading Committee:

_____

Mingyan Li

_____

Radha Poovendran

_____

James Ritcey

Date: _____

In presenting this dissertation in partial fulfillment of the requirements for the doctoral degree at the University of Washington, I agree that the Library shall make its copies freely available for inspection. I further agree that extensive copying of this dissertation is allowable only for scholarly purposes, consistent with "fair use" as prescribed in the U.S. Copyright Law. Requests for copying or reproduction of this dissertation may be referred to Proquest Information and Learning, 300 North Zeeb Road, Ann Arbor, MI 48106-1346, 1-800-521-0600, to whom the author has granted "the right to reproduce and sell (a) copies of the manuscript in microform and/or (b) printed copies of the manuscript made from microform."

Signature_____

Date_____

University of Washington

Abstract

Identifying, Modeling, and Mitigating Attacks
in Wireless Ad-Hoc and Sensor Networks

Patrick Tague

Chair of the Supervisory Committee:
Associate Professor Radha Poovendran
Electrical Engineering

Robust network operation and the ability to provide user and data security while under attack are desirable qualities of network protocols. However, these qualities require a fundamental understanding of network protocol vulnerabilities and characterization of the space of possible attacks. Hence, understanding attacks and their impact is a necessary prerequisite to the design of secure network protocols. In this dissertation, we investigate the problems of modeling attacks on network protocols and performance and design of networking protocols that are robust to attack.

We investigate secure communication link establishment in ad-hoc and sensor networks through symmetric cryptographic key assignment. We propose a key assignment framework which balances design trade-offs between worst-case network connectivity and impact of *node capture attacks* in which an adversary physically compromises nodes and extracts keys from memory.

We study an adversary's use of leaked information to increase the impact of node capture attacks. We formulate minimum cost node capture attacks targeting link and data security as NP-hard integer programming problems. We present an efficient node selection heuristic using network information flow to evaluate attack impact.

We address a jamming attack focused on control channels in which an adversary recovers secret spreading sequences via node capture. We show that random assignment of redundant

sequences restricts the impact of the jamming attack and leads to graceful degradation of service.

We investigate the design of dynamic routing protocols that compensate for the effects of jamming on network traffic flow. We show that source nodes can acquire statistics about the jamming impact over multiple routing paths, allowing for dynamical adjustment of the multiple-path traffic allocation. We map this traffic allocation problem to that of financial portfolio selection and present a constrained optimization problem for traffic allocation using this mapping.

Finally, we investigate the problem of quantifying the impact of jamming attacks on network traffic flows. We model jamming attacks by an adversarial network and show that the use of network flow information allows the adversary to effectively balance the workload. We formulate cross-layer jamming attacks as non-convex optimization problems and present convex approximations and a distributed algorithm for jamming attacks.

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would first like to thank my advisor Radha Poovendran for his direction, advice, and support throughout my Ph.D. studies at the University of Washington. I am deeply indebted to Radha for showing me the excitement that can be found in collaborative academic research and for allowing me to work independently. His guidance has helped to mold me into a successful researcher, teacher, and mentor and has equipped me with the tools necessary to be successful in the future of my academic career. I would also like to thank the members of my Ph.D. supervisory committee: Yoshi Kohno, Mingyan Li, Mehran Mesbahi, and Jim Ritcey.

I would like to extend my thanks to Mingyan Li for her collaboration at UW on the problem of mitigating control channel jamming under node capture and at Boeing on the numerous problems of modeling vulnerabilities in aircraft networking applications. I would also like to extend my thanks to Jim Ritcey for his insight and collaboration on the problem of jamming-aware traffic allocation using portfolio theory. I would like to thank Jooyoung Lee at ETRI Korea for his collaboration on the initial work of modeling node capture attacks. I would like to thank Brian Matt at the JHU APL for his numerous discussions and collaboration on the problem of securing communication over unreliable channels. I would like to thank Guevara Noubir at NEU for his collaboration on the problem of modeling cross-layer jamming attacks. I would like to thank Jason Rogers at NRL for collaborating over a long Spring Break at UW on the problem of evaluating joint routing and security vulnerabilities, and I would like to thank George Dinolt at NPS and Ed Zieglar at NSA for initializing the collaboration. Finally, I would like to acknowledge the generous support by the following funding sources: ONR YIP, N00014-04-1-0479; ARO PECASE, W911NF-05-1-0491; ARL CTA, DAAD19-01-2-0011; ARO MURI, W911NF-07-1-0287; and an NSA/DoD IASP Fellowship.

I would like to collectively acknowledge the current and former members of the Network Security Lab at UW. I would like to thank the members of the previous incarnation of the NSL, including Intae Kang, Loukas Lazos, Mingyan Li, Javier Salido, and Radhakrishna Sampigethaya, who warmly welcomed me into the group and provided unending support during the early years of my Ph.D. studies. I would also like to thank the current instance of the NSL, including Basel Alomair, Andrew Clark, Sidharth Nabar, David Slater, and Jeff Vander Stoep, who worked with me during the later part of my time at UW. I would like to thank all of my friends, both at UW and not, for their patience and support and for the entertainment they provided.

Finally, I would like to thank my family for the inspiration and unconditional support that has allowed me to become the person that I am. More than anyone else, I would like to thank my wife Natalie Linnell for being at my side throughout this difficult journey and for promising to be there forever.

Chapter 1

INTRODUCTION

Continuing advances in communications and hardware technology has lead to the ability to manufacture low-cost wireless embedded devices that can be deployed in ad-hoc networks without relying on pre-existing infrastructure. In such networks, data is transmitted throughout the network using multi-hop routing, with source and destination nodes in the network depending on intermediate nodes to relay traffic. Wireless ad-hoc networking allows for readily available access to a wealth of information without an infrastructure-based network, suggesting that ad-hoc networks will soon be ubiquitously deployed for personal, social, commercial, industrial, and military applications. Examples of ad-hoc network applications include home and office networking, surveillance, inventory and product tracking, disaster recovery and rescue, medical patient monitoring, and tactical military applications.

With the benefit of wireless ad-hoc networking in terms of flexibility and ease of deployment come many challenges in network security. Wireless ad-hoc networks are exposed to a variety of security threats in that adversaries may disrupt or halt network operaton, compromise the continuous flow of valid information, and violate the privacy of network users and their data. In particular, due to the extensive use of the wireless medium in ad-hoc networks, message communciations are vulnerable to passive attacks such as eavesdropping and active attacks such as message insertion or jamming. Such attacks allow an adversary to infer network operation, recover user data, and interfere with the correctness and efficiency of protocols. In addition, due to the fact that network nodes operate in an unattended manner, an adversary can physically attack or *capture* network nodes and extract information from their memories, modify hardware and software configurations, and even create clones. Physical attacks often allow for efficient recovery of secret information and access into the network as a valid user without the computational overhead of cryptanalysis.

In order to provide robust network operation as well as user and data security in the

presence of adversaries, it is necessary to design attack-resilient network protocols. However, this requires a fundamental understanding of network protocol vulnerabilities and characterization of the space of possible attacks. Hence, understanding attacks and their impact on the network is a necessary prerequisite to the design of secure network protocols.

1.1 Investigated Problems

In this dissertation, we investigate several problems of interest related to modeling attacks on wireless ad-hoc networks and designing defense mechanisms. We first study the problem of resource-efficient and secure key management in ad-hoc and sensor networks. We then study the problem of modeling and understanding the impact of node capture attacks on the security of key management solutions. We then investigate the problem of mitigating the effects of jamming by an adversary that captures nodes and extracts private spread spectrum hopping sequences from their memory. We next study the ability for a wireless network routing protocol to proactively adjust traffic allocation to compensate for the impact of jamming attacks. Finally, we study the problem of modeling and understanding the impact of efficient cross-layer jamming attacks by an adversarial network.

1.1.1 Efficient and Secure Key Management for Ad-Hoc and Sensor Networks

A resource-efficient method for establishing secure communication links in ad-hoc and sensor networks is through the assignment of symmetric cryptographic keys to network nodes. Such keys are usually assigned offline prior to network deployment to ensure secure communication once nodes are deployed. Neighboring nodes can establish secure links only if they share any symmetric keys. The key assignment process must thus compensate for the uncertainty in the network topology prior to deployment by assigning a sufficient number of shared keys to each node to guarantee network connectivity using only secure links. However, if a single key is assigned to too many nodes, any links secured using the key are vulnerable to attack by a malicious node or an adversary that physically compromises a node in a *node capture attack*. Hence, there are inherent trade-offs between network connectivity and resilience to attack that are a direct result of the key assignment process.

In this dissertation, we focus on random key assignment, characterizing the random key assignment process by showing that the protocol designer can probabilistically control the number of times each key is assigned. We propose a key assignment framework in which a chosen probability distribution on the number of nodes holding each key can be realized using sampling. We show that the average-case network connectivity and resilience to attack are a function only of the average μ of the designed distribution. In particular, the average probability of connectivity increases with μ, and the average resilience to attack decreases with μ, illustrating the design trade-offs. We also show that the worst-case performance in terms of network connectivity and resilience to attack can be improved by imposing constraints on the tails of the designed distribution.

1.1.2 Formulating Node Capture Attacks using Leaked Network Protocol Information

As previously mentioned, the unattended operation of many ad-hoc and sensor networks may allow an adversary to mount a node capture attack by physically compromising network nodes and extracting stored information from their memory. Node capture attacks are of particular interest with respect to the security of key management protocols, as node capture effectively bypasses the computational overhead of cryptanalysis. Existing work has focused on the analysis of node capture attacks when the adversary targets nodes independently at random. However, an adversary can obtain a significant amount of information leaked during the secure link establishment protocol, either by eavesdropping or impersonating a network node.

In this dissertation, we show that a sophisticated adversary can exploit this leaked information by capturing those nodes which lead to the compromise of the largest number of secure links in the network. We show that such intelligent node capture attacks can be formulated as Linear Integer Programming (LIP) problems and that finding the optimal node capture strategy is NP-hard. We investigate the use of efficient heuristics for solving this LIP problem and illustrate the feasibility of a variety of attack strategies. In addition, we show that further information leakage occurs during the execution of a routing protocol. The additional routing information can improve the effectiveness of node capture attacks,

as the security of data routed through a network can be compromised at various points in the routing topology. We show that attacks using routing information can be formulated as Non-Linear Integer Programming (NLIP) problems. We present an efficient heuristic for the selection of nodes to capture which computes the value of each node toward the adversarys attack goal using properties of network information flow.

1.1.3 Mitigating Efficient Jamming Attacks by Compromised Users

The use of dedicated communication channels to transmit control traffic introduces a single point of failure for a denial-of-service (DoS) attack. An adversary aware of the network protocol operation can jam relevant control channel traffic and indirectly prevent data communication. For example, jamming the Request-to-Send and Clear-to-Send messages in a wireless handshake protocol prevents the sender and receiver from initiating data exchange. This reliance of data communication on control channels allows a jamming adversary to launch a DoS attack which is several orders of magnitude more energy-efficient than jamming the data channel. Typical jamming resistant techniques such as spread spectrum provide resilience to jamming only to the extent that the shared spreading code or frequency hopping sequence remains secret. However, an adversary mounting a node capture attack effectively becomes aware of the secret hopping sequence.

In this dissertation, we propose the use of random key assignment for distribution of spread spectrum hopping sequences in order to mitigate the impact of control channel jamming under a node capture attack. We show that the use of random key assignment restricts the effect of the jamming attack to impact only a subset of network users that increases in size with the number of captured nodes, leading to graceful degradation of service. Furthermore, since users hold distinct sets of hopping sequences with high probability, a network authority that can detect control channel jamming can identify and revoke compromised users from the network. We show that this identification problem can be formulated as a set estimation problem and analyze the performance of the estimation problem in terms of the false alarm and miss rates.

1.1.4 Designing Jamming-Aware Multiple-Path Routing Algorithms

The effects of jamming at the physical layer, including packet decoding errors and dropped packets, lead to a reduction in the throughput achieved by network routing protocols. The ability for a routing protocol to compensate for the effects of jamming is complicated by the non-deterministic and dynamic effects of the jamming attack, primarily due to mobility of the jammers and attack performance details that are unknown from the network perspective. Existing solutions for robust network throughput in the presence of jammers rely on the reaction of network protocols to detection of jamming. For example, if a jammer is detected in a particular area, the routing protocol can be instructed to route around the jammed region.

In this dissertation, we propose the use of a jamming-aware source routing protocol in which each data source dynamically adjusts the allocation of traffic to multiple routing paths based on statistical information about the jamming attack. As the basis of the allocation problem, we investigate the ability of intermediate network nodes to characterize the jamming impact and relay this information to the corresponding source nodes. We formulate the traffic allocation problem across multiple routing paths as a lossy network flow optimization problem, mapping to a financial asset allocation problem using portfolio selection theory. We formulate both a centralized optimization problem and distributed algorithm based on optimization decomposition. We demonstrate that the financial asset interpretation allows the data sources to balance the expected data throughput under jamming with the uncertainty in the jamming attack and the corresponding achievable data rates.

1.1.5 Evaluating the Impact of Efficient Cross-Layer Jamming Attacks

An adversary with a network of jammers can optimize the jamming attack by combining higher-layer network information with intelligent transmission power regulation, thereby balancing the jamming workload across the adversarial network. Jammers can intentionally reduce their probability of success to achieve resource savings and rely on neighboring jammers to share the workload, given sufficient coverage of the network. Hence, the ad-

versary can optimize a global utility function such as the expected reduction in network throughput, total energy expenditure, or the lifetime of the adversarial network through appropriate assignment of jamming workload and transmission power levels.

In this dissertation, we quantify the effect of these cooperative jamming attacks on network performance and idenfity crucial concepts which may then be incorporated into network protocol design. We formulate jamming attacks as constrained optimization problems that jointly optimize the assignment of jamming workload and the jamming transmission power levels for an adversarial network deployed over the target network area. We propose a variety of metrics to evaluate the effect of jamming attacks both in terms of the adversarial network and the target network, noting that these metrics can double as objective functions for optimizing the jamming attacks. We introduce efficient convex and linear optimization problems which approximate the optimal attacks, enabling efficient computation of jamming solutions, and present a cooperative distributed jamming algorithm. We then compare the performance of various jamming attack formulations in terms of the ability to reduce network throughput for a given energy budget.

1.2 Organization of the Dissertation

The remainder of the dissertation is organized as follows. In Chapter 2, we investigate the problem of resource-efficient and secure key management for wireless ad-hoc and sensor networks. In Chapter 3, we model and study the impact of node capture attacks on the security of key management protocols. In Chapter 4, we propose a technique to mitigate control channel jamming by compromised users by mapping to a key management problem. In Chapter 5, we propose an optimization formulation for jamming-aware allocation of traffic over multiple routing paths. In Chapter 6, we present an optimization framework for jamming attack formulation and evaluate the impact of jamming attacks on network throughput. In Chapter 7, we summarize our contributions and outline future research directions.

Chapter 2

A CANONICAL MODEL FOR KEY ASSIGNMENT IN WIRELESS SENSOR NETWORKS

Advances in sensor technology suggest that large-scale wireless sensor networks (WSNs) can provide sensing and distributed processing using low-cost, resource-constrained sensor nodes [5] for commercial, industrial, and military applications such as disaster relief and recovery, medical patient monitoring, smart homes, mechanical system monitoring, and target detection and tracking. As data integrity, authentication, privacy, and confidentiality are often important concerns in such applications, secure communication protocols are required. However, the ad-hoc nature of WSNs require minimal interaction with base stations or a central authority, so trust establishment for secure communication is a critical task [4, 29]. Furthermore, random sensor deployment and the physical communication constraints of sensor nodes make trust establishment a very challenging problem in WSNs.

The resource constraints of sensor nodes are the limiting factor in the type of cryptographic primitives that can be implemented. There have been recent efforts to implement public-key cryptography in wireless sensor networks [27, 33, 35–37]. However, such protocols can not yet be implemented on all sensor nodes. Hence, many of the current solutions to key establishment rely on the use of symmetric key cryptography.

A promising solution for the establishment of secure communication in WSNs using symmetric keys is the use of *key predistribution* [15, 29, 50]. A key predistribution scheme can be described in two primary phases: *key assignment* and *link-key establishment*. In the key assignment phase, executed prior to network deployment, sensor nodes are *seeded* with cryptographic keys (e.g. hashed master keys [46], cryptographic keys [29], or polynomial shares [49]). In the link-key establishment phase, executed after network deployment, neighboring nodes compute *link-keys* as a function of assigned keys in order to establish secure one-hop links. While many existing works in the literature provide novel approaches for the

link-key establishment phase of key predistribution, the scope of key assignment techniques is limited.

2.1 Our Contributions

In this chapter, we present a canonical model for the key assignment phase of key predistribution in WSNs. In the canonical key assignment model, key assignment schemes are characterized in terms of a discrete probability distribution of the number of nodes sharing each assigned key and the algorithm used to perform the key assignment. The canonical model allows the network designer to explicitly control the probability distribution and limit the effects of tail behavior in the probability distribution. We present a sampling framework for randomized key assignment algorithms for use in the canonical model. In the framework, key assignment algorithms are classified according to the selection method used to realize a given probability distribution, and a representative algorithm from each class is illustrated. We demonstrate how the worst-case analysis of any key predistribution scheme can be performed using the canonical model, analysis which has not been possible using techniques in existing literature. We also show that the average case analysis can be performed as in existing works.

In addition to the key assignment model itself, we develop a model for probabilistic network k-connectivity for randomly deployed secure WSNs in which communication is restricted by both radio range and the existence of shared keys. This connectivity model, based on spatial statistics [20] and the asymptotic properties of geometric random graphs [8, 59], can be used along with the canonical model for the purposes of network design. We further illustrate the effect of network extension via node addition using the canonical model.

2.2 Motivation and Problem Statement

Various properties of a key predistribution scheme can be analyzed in terms of the number of nodes sharing each assigned key. Hence, The behavior of a key predistribution scheme is analyzed with respect to the probability that a given key is shared by exactly λ of the N nodes in the WSN.

2.2.1 Motivation

The impact of the number of nodes λ sharing a given key is investigated for the following metrics: the probability that a pair of nodes share at least one key, the probability that no pair of nodes sharing a given key are within radio range, and the potential number of secure links established using a given key.

Intuitively, if the number of nodes λ which share a given key is small, the probability that one of the λ nodes will share the key with a neighboring node will be very small. This statement can be justified by estimating the probability that a neighboring node shares the given key. Since exactly λ of the N nodes in the network hold the given key, the probability that a neighboring node shares the key is approximately $\frac{\lambda}{N}$. Given a node with K keys shared by $\lambda_1, \ldots, \lambda_K$ nodes, the probability that a neighboring node shares at least one key can thus be estimated as

$$\Pr[\text{at least one key shared}] = 1 - \left(1 - \frac{\lambda_1}{N}\right) \times \cdots \times \left(1 - \frac{\lambda_K}{N}\right). \tag{2.1}$$

Furthermore, if λ is small and the area within the radio range of a node is significantly less than the deployment area of the network, the probability that a key shared by λ nodes will not be used to establish a secure link, referred to as the *key wastage* probability, will be large. This statement can be similarly justified by estimating the key wastage probability as follows. Assuming the sensor nodes are randomly distributed over a region $\mathcal{A}$, the probability that a given pair of nodes are not within a distance r is given by

$$n_r = 1 - \frac{\pi r^2}{|\mathcal{A}|}. \tag{2.2}$$

The key wastage probability $w(\lambda)$ can be estimated as

$$w(\lambda) \approx n_r^{\binom{\lambda}{2}} = \left(1 - \frac{\pi r^2}{|\mathcal{A}|}\right)^{\binom{\lambda}{2}}, \tag{2.3}$$

noting that equality does not hold because the $\binom{\lambda}{2}$ events are not independent. Hence, the key wastage probability decreases exponentially in λ, and a key shared by a small number of nodes λ will be unused with high probability.

If the number of nodes λ which share a given key is large, the number of secure links established using the key is potentially large. An adversary with the key can thus compro-

mise a large number of secure links. This statement can be similarly justified by estimating the number of secure links which can be established using the given key. Given λ nodes that share the key, there can be as many as $\binom{\lambda}{2}$ secure links formed using the given key, increasing quadratically in λ.

Quantifying the above metrics as a function of λ also allows for the worst-case analysis with respect to each metric. Let $\mathcal{P}(\lambda)$ denote the probability that a given key is shared by λ nodes and $\mathcal{H}(\lambda) = P\mathcal{P}(\lambda)$ denote the expected number of keys shared by exactly λ nodes, where P is the total number of keys. $\mathcal{P}$ and $\mathcal{H}$ thus denote the *probability distribution* and *expected histogram* of λ, respectively. The *expected* worst-case for each metric can thus be quantified as a function of the expected histogram $\mathcal{H}$.

The expected worst-case probability of sharing keys and key wastage probability can be computed as a function of λ_{min}, defined as the minimum λ such that $\mathcal{H}(\lambda) \geq 1$. The expected worst-case number of compromised links can similarly be computed as a function of λ_{max}, defined as the maximum λ such that $\mathcal{H}(\lambda) \geq 1$. The deviation of each metric due to variation in λ can thus be quantified by comparing the values at λ_{min} and λ_{max} to that at the average value μ of the distribution $\mathcal{P}$.

As an example, the above metrics are evaluated for the random key predistribution scheme of [29]. In this scheme, each node is assigned a random subset of K keys from a pool of $P \gg K$ keys. When a subset of K keys is selected for one node, a particular key is selected with probability $\frac{K}{P}$, which can be modeled as a Bernoulli random variable. Hence, the probability distribution $\mathcal{P}(\lambda)$ is the binomial distribution $\mathcal{B}(N, \frac{K}{P})$ such that $\mathcal{P}(\lambda)$ is given by

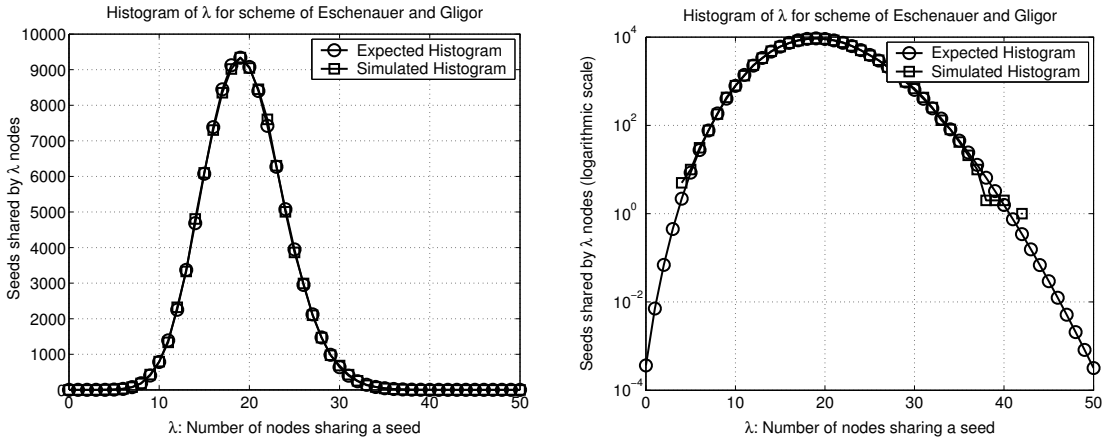
$$\mathcal{P}(\lambda) = \binom{N}{\lambda} \left(\frac{K}{P}\right)^{\lambda} \left(1 - \frac{K}{P}\right)^{N-\lambda} \tag{2.4}$$

with average value $\mu = \frac{NK}{P}$, and the values of the histogram $\mathcal{H}$ are given by

$$\mathcal{H}(\lambda) = P\binom{N}{\lambda} \left(\frac{K}{P}\right)^{\lambda} \left(1 - \frac{K}{P}\right)^{N-\lambda}. \tag{2.5}$$

The following example illustrates the effect of this binomial distribution on the metrics of interest.

Example 2.1. *Let a WSN of $N = 10,000$ nodes be assigned keys according to the key predistribution scheme of [29] with $K = 200$ and $P = 102,881$, where P is chosen to*



(a) (b)

Figure 2.1: The expected histogram $\mathcal{H}(\lambda)$, representing the number of keys shared by exactly λ nodes, is illustrated for Example 2.1 with vertical axis in (a) linear scale and (b) logarithmic scale.

guarantee network connectivity with probability 0.999 *for an average of* $d = 50$ *nodes within radio range. The average number of nodes sharing a given key is* $\mu = \frac{NK}{P} = \frac{10{,}000 \times 200}{102{,}881} \approx 20$. *The expected histogram* $\mathcal{H}$ *and the simulated histogram are provided in Figure 2.1. For the given parameters, the condition* $\mathcal{H}(\lambda) \geq 1$ *is satisfied for all* λ *between* $\lambda_{min} = 4$ *and* $\lambda_{max} = 40$.

The variation in the probability of sharing keys is quantified by computing the probability given in (2.1) for $\lambda_1, \ldots, \lambda_K$ *all equal to the values* λ_{min}, μ, *and* λ_{max}, *yielding* 0.0769, 0.3224, *and* 0.5514, *respectively. The expected worst-case probability of sharing keys can alternatively be defined as a function of the* K *smallest values* $\lambda_{min}^{(1)}, \ldots, \lambda_{min}^{(K)}$ *which occur according to the expected histogram* $\mathcal{H}$.

The variation in the key wastage probability is quantified by computing the probability given in (2.3). Since the network is randomly deployed, the quantity $\frac{\pi r^2}{|\mathcal{A}|}$ *is approximately equal to* $\frac{d}{N} = 0.005$. *Hence, the key wastage probability for the values* λ_{min}, μ, *and* λ_{max} *is equal to* 0.9704, 0.3858, *and* 0.0200, *respectively.*

The variation in the number of potential compromised links is similarly computed for the values λ_{min}*,* μ*, and* λ_{max}*, yielding* 6*,* 190*, and* 780 *links, respectively.*

2.2.2 Problem Statement

Example 2.1 shows that the use of random key predistribution [29] induces a binomial distribution $\mathcal{B}(N, \frac{K}{P})$ on the number of nodes which share each key. As demonstrated, the induced distribution can lead to undesirable tail-effects related to the keys which are shared by very few or very many nodes in the WSN. The natural question which arises is whether key predistribution schemes can be designed to induce other distributions which do not suffer from the undesirable tail-effects. Moreover, the secondary question which arises is whether it is possible to design universal algorithms for key assignment which can be used to realize a wide variety of distributions, leading to a general class of application-dependent key predistribution schemes. To the best of our knowledge, there are no existing key predistribution schemes which can address these questions. In fact, any scheme derived from random key predistribution [29] results in the same binomial distribution and tail-effects as in Example 2.1.

Hence, we aim to characterize the distribution on the number of nodes sharing each key and the algorithms which can be used to assign keys to nodes in the WSN. The goal of this characterization is to decouple the distribution from the algorithm used to assign keys, leading to a class of algorithms which can be used to realize a wide variety of distributions which avoid undesirable tail-effects, thus addressing both of the questions of interest.

2.3 Network and Security Models

In this section, we state our models and assumptions about the capabilities of adversaries and the deployment of the sensor network.

2.3.1 Adversarial Model

We assume that adversaries are able to eavesdrop and record transmissions throughout the WSN. Furthermore, we assume that adversaries are able to physically capture sensor nodes

and access all information stored within them. We are primarily concerned with adversaries attempting to capture a sufficient number of nodes to compromise a given fraction of the secure links in the WSN. Hence, we do not consider attacks on other network protocols (e.g. node replication, sleep deprivation attacks, wormhole attacks, etc.). We assume that the adversary can capture sensor nodes in any part of the network, and we further assume, as in many recently published works (e.g. [29, 50]) that the captured nodes are chosen randomly and independently.

2.3.2 Network Model

Each sensor is assumed to be equipped with an omni-directional radio with fixed communication range r.[1] Furthermore, a pair of nodes that are within distance r can establish a secure link only if sufficient assigned keys are shared between them. The wireless network is made up of N sensor nodes deployed randomly (uniformly) over a region $\mathcal{A} \subseteq \mathbb{R}^2$, and the resulting location of node i is given by $x_i \in \mathcal{A}$ for $i = 1, \ldots, N$. The connectivity of the resulting secure WSN is determined with respect to Definition 2.1 as follows.

Definition 2.1. *The* connectivity $\kappa(G)$ *of a graph G is defined as the minimum number of vertices which leave a disconnected graph when removed. A graph G with $\kappa(G) \geq k$ is said to be* k-connected.

A geometric random graph [8, 59] as given by Definition 2.2 below is used to model the physical radio restrictions on the nodes of the sensor network. Furthermore, the shared-key relation between sensor nodes is modeled using a logical graph as given by Definition 2.3. The combination of the geometric random graph and the logical graph yields a graph theoretical model for the secure WSN in the form of the restricted network graph as given by Definition 2.4.

Definition 2.2. *A* (Euclidean) geometric random graph $G_g(N, \mathcal{A}, r)$ *is the result of random distribution of N vertices in the region $\mathcal{A}$ such that a pair of vertices i and j are adjacent if and only if the (Euclidean) distance between them is no more than r.*

[1] Due to the use of spatial statistics, the area covered by the radio range of a node need not be circular. Hence, this assumption is only necessary to guarantee bi-directional communication between sensor nodes.

Definition 2.3. *A* logical graph $G_L(N, \mathcal{R})$ *models a logical relationship between each pair of sensors such that a pair of nodes i and j are adjacent if and only if the pairwise relation $\mathcal{R}$ is satisfied.*

Definition 2.4. *The* restricted network graph $G(N, \mathcal{A}, r, \mathcal{R})$ *represents a WSN of N nodes deployed over a region $\mathcal{A}$ such that sensors i and j can communicate if and only if they are within distance r and the relation $\mathcal{R}$ is satisfied. The graph G is given by the edge-wise intersection of a geometric random graph $G_g(N, \mathcal{A}, r)$ and a logical graph $G_L(N, \mathcal{R})$.*

We provide the following results relating to the node degree and the connectivity of the restricted network graph. Theorem 2.1 provides a probabilistic connectivity model which can be used to provide parameters to yield sufficient network connectivity with a desired probability.

Lemma 2.1. *Given a node u with degree D in the logical graph $G_L(N, \mathcal{R})$, the probability $Pr[d_u \geq k]$ that u has degree at least k in the graph $G(N, \mathcal{A}, r, \mathcal{R})$ is given by*

$$Pr[d_u \geq k] = 1 - e^{-\rho \frac{D+1}{N} \pi r^2} \sum_{i=0}^{k-1} \frac{(\rho \frac{D+1}{N} \pi r^2)^i}{i!}.$$

Proof. The vertex density of the geometric random graph $G_g(N, \mathcal{A}, r)$ is given by $\rho = \frac{N}{|\mathcal{A}|}$. The vertices are distributed according to a two-dimensional Poisson point process with rate ρ, so the probability distribution of the number of nodes within distance r of a node is a Poisson distribution [20]. Hence, the probability that the degree d_g of a node is at least k in $G_g(N, \mathcal{A}, r)$ is given by

$$Pr[d_g \geq k] = 1 - e^{-\rho \pi r^2} \sum_{i=0}^{k-1} \frac{(\rho \pi r^2)^i}{i!}. \tag{2.6}$$

Given that a vertex u has degree D in $G_L(N, \mathcal{R})$, d_u is at least k in $G(N, \mathcal{A}, r, \mathcal{R})$ if and only if at least k of the D neighbors in $G_L(N, \mathcal{R})$ are within distance r of u. Since the neighbors of u in $G_L(N, \mathcal{R})$ are determined independently of the neighbors of u in $G_g(N, \mathcal{A}, r)$, the neighbors of u in $G(N, \mathcal{A}, r, \mathcal{R})$ are uniformly distributed in the region $\mathcal{A}$. Hence, the neighbors of u in $G_L(N, \mathcal{R})$ form a geometric random graph $G_g^u(D+1, \mathcal{A}, r)$, represented by a Poisson point process with rate $\frac{D+1}{|\mathcal{A}|} = \rho \frac{D+1}{N}$. Hence, replacing ρ by $\rho \frac{D+1}{N}$ in (2.6) completes the proof. □

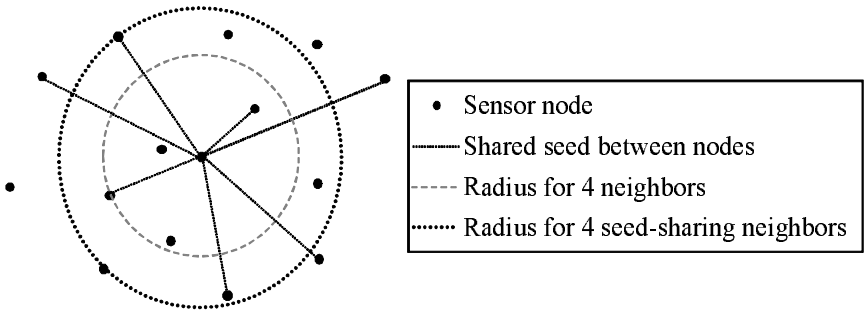


Figure 2.2: The radio range of a node in the WSN required for a connected network increases when considering only neighboring nodes which share keys.

As suggested in the proof of Lemma 2.1, a decrease in the density of a geometric random graph requires an increase in the radio range r in order to guarantee that the degree d_u of a node u in the graph $G_L(N, \mathcal{R})$ is sufficiently high. This increase in radio range is illustrated in Figure 2.2. In what follows, we prove that the probability given by Lemma 2.1 is independent for every pair of nodes.

Lemma 2.2. *In a geometric random graph $G_g(N, \mathcal{A}, r)$, the probability that each of a pair of nodes has degree at least k is independent, i.e. for nodes u and v*

$$Pr[d_u \geq k, d_v \geq k] = Pr[d_u \geq k]Pr[d_v \geq k].$$

Proof. Let $d_{u\setminus v}$ denote the number of nodes in the region $R_{u\setminus v}$ that is within radius r of node u but not within radius r of node v. Similarly, let $d_{u,v}$ denote the number of nodes in the region $R_{u,v}$ that is within radius r of both u and v. The joint probability $Pr[d_u \geq k, d_v \geq k]$ can be decomposed as

$$\begin{aligned} Pr[d_u \geq k, d_v \geq k] &= \sum_{i \geq k} Pr[d_u \geq k | d_v = i] Pr[d_v = i] \\ &= \sum_{i \geq k} \left(1 - \sum_{j < k} Pr[d_u = j | d_v = i] \right) Pr[d_v = i]. \end{aligned} \tag{2.7}$$

Noting that $i > j$ in (2.7), the probability $Pr[d_u = j | d_v = i]$ can be expressed as

$$Pr[d_u = j | d_v = i] = \sum_{n=0}^{j} Pr[d_u = j | d_v = i, d_{u,v} = n] Pr[d_{u,v} = n] \tag{2.8}$$

$$= \sum_{n=0}^{j} Pr[d_{u\setminus v} = j - n | d_v = i, d_{u,v} = n] Pr[d_{u,v} = n] \tag{2.9}$$

$$= \sum_{n=0}^{j} Pr[d_{u\setminus v} = j - n] Pr[d_{u,v} = n] \tag{2.10}$$

$$= \sum_{n=0}^{j} e^{-\rho |R_{u\setminus v}|} \frac{\left(\rho |R_{u\setminus v}|\right)^{j-n}}{(j-n)!} e^{-\rho |R_{u,v}|} \frac{\left(\rho |R_{u,v}|\right)^{n}}{n!} \tag{2.11}$$

$$= e^{-\rho \pi r^2} \frac{\rho^j}{j!} \sum_{n=0}^{j} \binom{j}{n} \left(\pi r^2 - |R_{u,v}|\right)^{j-n} |R_{u,v}|^n \tag{2.12}$$

$$= e^{-\rho \pi r^2} \frac{(\rho \pi r^2)^j}{j!} = Pr[d_u = j]. \tag{2.13}$$

Under the spatial Poisson point process model, the number of points which appear in disjoint regions of $\mathcal{A}$ are independently distributed. Hence, in the above formulation, (2.10) follows from the fact that the region $R_{u\setminus v}$ is disjoint from both the region $R_{u,v}$ and the region within radio range r of node v. The Poisson process model further allows substitution of the identically distributed probabilities in (2.11). Equation (2.12) follows by substituting $|R_{u\setminus v}| = \pi r^2 - |R_{u,v}|$ and collecting terms, and (2.13) is obtained by applying the binomial theorem and again using the properties of the Poisson point process. Substituting (2.13) into (2.7) completes the proof. □

Theorem 2.1. *The restricted network graph $G(N, \mathcal{A}, r, \mathcal{R})$ resulting from the edge-wise intersection of a logical graph $G_L(N, \mathcal{R})$ with average node degree D and a geometric random graph $G_g(N, \mathcal{A}, r)$ with node density $\rho = \frac{N}{|\mathcal{A}|}$ is k-connected with probability $P_G(k)$ given by*

$$P_G(k) = \left(1 - e^{-\rho \frac{D+1}{N} \pi r^2} \sum_{i=0}^{k-1} \frac{(\rho \frac{D+1}{N} \pi r^2)^i}{i!}\right)^N.$$

Proof. Applying Lemma 2.2 to each geometric random graph on $(D+1)$ nodes with density $\rho = \frac{D+1}{|\mathcal{A}|}$ as in Lemma 2.1, the minimum node degree d_{min} in the graph $G(N, \mathcal{A}, r, \mathcal{R})$ is given by

$$Pr[d_{min} \geq k] = Pr[d_1 \geq k, \ldots, d_N \geq k] = Pr[d \geq k]^N. \tag{2.14}$$

As r increases, a geometric random graph becomes k-connected, asymptotically, as soon as the minimum vertex degree is k with high probability [59]. Hence, the probability of connectivity is given by $P_G(k) = Pr[d_{min} \geq k] = Pr[d \geq k]^N$. □

Theorem 2.1 provides the model for probabilistic k-connectivity used throughout this chapter. Several works on key predistribution have used a connectivity model based on the assumption that the underlying logical graph is given by a random graph with independent edge probability p. In Corollary 2.1, we show that this random graph model can be approximated by a special case of the model given by Theorem 2.1.

Corollary 2.1. *If $G_L(N, \mathcal{R})$ is a random graph with independent edge probability p, the probability $P_G(1)$ given by Theorem 2.1 can be approximated by the result given in [29].*

Proof. The average vertex degree in a random graph on N vertices with independent edge probability p is given by $D = p(N-1)$, so Theorem 2.1 yields a connectivity probability of

$$P_G(1) = \left(1 - e^{-\rho \frac{p(N-1)+1}{N} \pi r^2}\right)^N \approx e^{-Ne^{-\rho \frac{p(N-1)+1}{N} \pi r^2}} \approx e^{-Ne^{-\rho p \pi r^2}}, \tag{2.15}$$

from the approximation $1 - x \approx e^{-x}$ for $|x| \ll 1$ and noting that $\frac{p(N-1)+1}{N} \approx p$ for $N \gg 1$. The probability of connectivity stated in [29] using the random graph approach can be expressed as

$$P_c = e^{-Ne^{-\frac{N}{N-1} p(\rho \pi r^2 - 1)}} \approx e^{-Ne^{-\rho p \pi r^2}} \tag{2.16}$$

by noting that $\frac{N}{N-1} p(\rho \pi r^2 - 1) \approx p \rho \pi r^2$ since $\frac{N}{N-1} \approx 1$ and $p \ll 1$. Hence, the connectivity probabilities $P_G(1)$ and P_c are approximately equal for all practical purposes. □

2.4 Key Assignment for Key Predistribution

In this section, we provide a canonical key assignment model for key predistribution. We discuss the assignment of keys to nodes in a WSN and the properties of such key assignment in terms of a bipartite graph process. Based on the graph theoretic interpretation of key assignment, we derive the canonical key assignment model and discuss the properties of the model. Based on the graph theoretical interpretation, we propose a sampling framework for key assignment in the canonical model which decomposes the space of key assignment

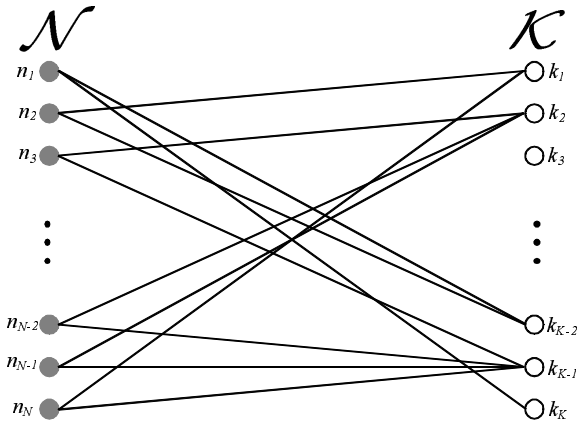


Figure 2.3: Bipartite graph g representing the assignment of keys to nodes in the WSN.

algorithms into four classes. Finally, we propose a key assignment algorithm for each of the four classes.

2.4.1 Proposed Approach

The assignment of keys to the nodes of a WSN can be seen as a process on a bipartite graph g with vertex set $V(g) = \mathcal{N} \cup \mathcal{K}$ where the set $\mathcal{N}$ represents the set of N nodes and the set $\mathcal{K}$ represents the set of P keys. An edge (n, k) in the edge-set $E(g) \subseteq \mathcal{N} \times \mathcal{K}$ represents the assignment of the key k to the node n. Figure 2.3 illustrates the use of a bipartite graph g for the assignment of keys to nodes in the WSN.

For such a bipartite graph g, we can describe the edge-set $E(g)$ in terms of the degree $deg(n)$ of each vertex $n \in \mathcal{N}$ and the degree $deg(k)$ of each vertex $k \in \mathcal{K}$. Similarly, the assignment of keys to nodes can be described in terms of the number of keys assigned to each node and the number of nodes which share each key. We assume that every node receives exactly K keys, corresponding to $deg(n) = K$ for all $n \in \mathcal{N}$, so the number of edges in g is $|E(g)| = NK$. Hence, we can describe key assignment in terms of the degrees $deg(k)$ for $k \in \mathcal{K}$ which result from the assignment of keys to nodes in the WSN. Specifically, we are interested in the probability $Pr\left[deg(k) = \lambda\right]$ that a key k is assigned to exactly λ nodes in the network.

If a desired probability distribution $Pr\left[deg(k) = \lambda\right], \lambda = 0, \ldots, N$, on the set $\mathcal{K}$ is given,

a graph algorithm is required in order to construct the graph g such that the distribution is realized. However, due to the restriction that every vertex in $\mathcal{N}$ must have degree K, such algorithms may not exist for all values of N and K. An example which illustrates this fact for combinatorial design based key predistribution schemes is discussed in [13].

The graph theoretical interpretation of key assignment in WSNs is the basis of our canonical key assignment model. The canonical model is stated formally by the following set of definitions in terms of the bipartite graph g. Table 2.1 summarizes the notation for the canonical key assignment model in WSNs in terms of the graph theoretical interpretation.

2.4.2 Canonical Key Assignment Model

The canonical key assignment model is primarily concerned with the probability distribution on the degrees of the nodes in $\mathcal{K}$, corresponding to the number of nodes which share each key. The set of nodes sharing each key and the probability distribution on the set sizes are defined formally in Definition 2.5 and Definition 2.6.

Definition 2.5. *The set* $S(k) = \{n \in \mathcal{N} : (n,k) \in E(g)\}$ *of nodes which are assigned the key* $k \in \mathcal{K}$ *is the* assignment set *of key* k*.*

Definition 2.6. *The discrete probability function* $\mathcal{P}(\lambda) = Pr\left[|S(k)| = \lambda\right]$ *specifying the probability that an assignment set* S *contains exactly* λ *nodes is the* assignment distribution*. The* support *of an assignment distribution* $\mathcal{P}$ *is given by* $\Lambda = \{\lambda : \mathcal{P}(\lambda) > 0\} \subseteq \{0, \ldots, N\}$*.*

Given a desired assignment distribution, an algorithm must exist which can realize the given distribution on the set $\mathcal{K}$. Such an algorithm is defined formally in Definition 2.7. The degree of imperfection of a key assignment algorithm is defined formally in Definition 2.9.

Definition 2.7. *The* key assignment algorithm $\mathscr{A}$ *is used to realize an assignment distribution* $\mathcal{P}$*, equivalently to construct a bipartite graph* g *with degree distribution* $\mathcal{P}$ *on* $\mathcal{K}$*.*

Definition 2.8. *A* key assignment scheme *is given by the pair* $(\mathcal{P}, \mathscr{A})$ *of an assignment distribution and a key assignment algorithm.*

Definition 2.9. *A* boundary set *resulting from a key assignment scheme* $(\mathcal{P}, \mathscr{A})$ *is an assignment set* $S(k)$ *of size* $\lambda \notin \Lambda$*. The* boundary distance *of such a boundary set is given*

Table 2.1: The notation used in Chapter 2 for the canonical key assignment model in WSNs is summarized in terms of the graph theoretical interpretation.

	Bipartite Graph Process	Canonical Model
g	bipartite graph	key assignment in WSN
$V(g)$	vertex set of g	set of nodes and keys
$\mathcal{N}$	vertex partition set of $V(g)$	set of sensor nodes
N	number of vertices in $\mathcal{N}$	number of nodes in WSN
$\mathcal{K}$	vertex partition set of $V(g)$	set of keys
P	number of vertices in $\mathcal{K}$	number of keys assigned in WSN
$E(g)$	edge set of g, $E(g) \subseteq \mathcal{N} \times \mathcal{K}$	key assignment to nodes
$deg(n) = K$	degree K of each vertex $n \in \mathcal{N}$	K keys assigned to every node
$S(k)$	$\{n : (n,k) \in E(g)\}$	assignment set for key k
$deg(k) = \lvert S(k)\rvert$	degree of vertex $k \in \mathcal{K}$	number of nodes in assignment set $S(k)$
$\mathcal{P}$	distribution of $deg(k), k \in \mathcal{K}$	assignment distribution
Λ	$\{deg(k) : k \in \mathcal{K}\}$	support of assignment distribution $\mathcal{P}$
μ	average vertex degree in $\mathcal{K}$	mean of distribution $\mathcal{P}$
$\mathscr{A}$	algorithm to construct g	key assignment algorithm
$(\mathcal{P}, \mathscr{A})$	-	key assignment scheme
$d(\lambda, \Lambda)$	-	boundary distance, $\min\{\lvert\lambda - \nu\rvert : \nu \in \Lambda\}$

by $d(\lambda, \Lambda) = \min\{|\lambda - \nu| : \nu \in \Lambda\}$. *Boundary sets result from either the algorithm* $\mathscr{A}$ *or the fact that there are only a finite number of keys* $k \in \mathcal{K}$ *with degree* $deg(k)$ *distributed according to* $\mathcal{P}$*, referred to hereafter as the* finite sampling effect.

We give a canonical key assignment model in WSN in terms of the above definitions. A key assignment scheme $(\mathcal{P}, \mathscr{A})$ can be characterized entirely by the assignment distribution $\mathcal{P}$ and the key assignment algorithm $\mathscr{A}$. The performance of a key assignment scheme $(\mathcal{P}, \mathscr{A})$ can be described in terms of the assignment distribution $\mathcal{P}$, the given set of network parameters, and the boundary sets which result from the algorithm $\mathscr{A}$ and the finite sampling effects. The desired outcome for a key assignment scheme $(\mathcal{P}, \mathscr{A})$ is a realization of the assignment distribution $\mathcal{P}$ with no boundary sets. In other words, the histogram representing the values $|\{k \in \mathcal{K} : deg(k) = \lambda\}|$ should be approximately equal to the scaled

assignment distribution $P \cdot \mathcal{P}(\lambda)$ for all $\lambda \in \Lambda$, and every node degree $deg(k), k \in \mathcal{K}$ will be a member of Λ.

As illustrated by Example 2.1, the network connectivity and resilience to node capture for a key predistribution scheme depend on the assignment distribution $\mathcal{P}$. Hence, in order to discuss desirable properties and design an assignment distribution for a given application, the effects of the assignment distribution on network connectivity and resilience to node capture must first be investigated. This detailed analysis is presented in Section 2.5, and the design of assignment distributions is thereafter discussed in Section 2.6.

As discussed in Section 2.2.2, we are interested in designing universal key assignment algorithms which can be used to realize a wide variety of assignment distributions, depending on application requirements. In order to address this problem, we propose a sampling framework for key assignment algorithms. In the sampling framework, an algorithm can realize a given assignment distribution with minimal occurrence of boundary sets through repeated sampling of the assignment distribution. Such a sampling framework ensures that the analytical characteristics of the key assignment scheme depend only on the assignment distribution as desired. Hence, in what follows, the sampling framework for key assignment algorithms is discussed in detail.

2.4.3 Sampling Framework for Key Assignment Algorithms

In this section, we propose a sampling framework for key assignment algorithms. In the framework, the assignment distribution is repeatedly sampled and assignment sets are constructed as a function of the samples of the assignment distribution. We consider algorithms based on random selection using the fundamental combinatorial methods of selection with and without replacement. Furthermore, we consider algorithms of two types. The first type selects an assignment set from $\mathcal{N}$ for each key subject to the constraint that $deg(n) = K$ for all $n \in \mathcal{N}$. The second type selects a subset of K keys from $\mathcal{K}$ for each node subject to the constraint that the values of $deg(k)$ for $k \in \mathcal{K}$ are distributed according to the assignment distribution $\mathcal{P}$. Hence, the sampling framework consists of four classes of algorithms.

We provide an example from each of the four classes of key assignment algorithms in the

Table 2.2: The four classes of key assignment algorithms in the sampling framework are based on whether the algorithm is based on selection with or without replacement and whether the algorithm selects subsets of $\mathcal{K}$ or subsets of $\mathcal{N}$.

	Selection with replacement	Selection without replacement
Subsets of $\mathcal{K}$	KSR	KSNR
Subsets of $\mathcal{N}$	NSR	NSNR

sampling framework, each of which is named for the corresponding class. The **K**ey **S**election with **R**eplacement (KSR) and **K**ey **S**election with **N**o **R**eplacement (KSNR) algorithms are examples from the classes of selection with and without replacement, respectively, of subsets of $\mathcal{K}$. The **N**ode **S**election with **R**eplacement (NSR) and **N**ode **S**election with **N**o **R**eplacement (NSNR) algorithms are examples from the classes of selection with and without replacement, respectively, of subsets of $\mathcal{N}$. Table 2.2 illustrates the four classes of key assignment algorithms in the sampling framework and classifies each of the four algorithms. In what follows, each algorithm is described in detail, and code and an illustration are provided for each of the four algorithms. In the code for each algorithm, $select(X, y)$ denotes uniform random selection of a subset of y elements from the set X, and $sample(\mathcal{P})$ denotes the generation of a sample from an assignment distribution $\mathcal{P}$.

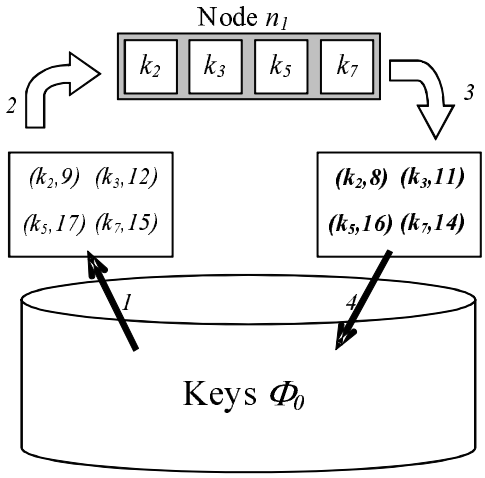
Key Selection with Replacement (KSR)

The KSR algorithm performs *selection with replacement* from a set Φ containing pairs (k, λ) where $k \in \mathcal{K}$ and $\lambda \in \Lambda$ is a sample of the assignment distribution $\mathcal{P}$. The number of keys $P = |\mathcal{K}| = |\Phi|$ must be sufficient to provide a total of NK edges in the graph g. Hence, we require $\sum_{(k,\lambda)\in\Phi} \lambda \geq NK$. Once Φ is constructed, keys are assigned to each node using random selection with replacement. For each of the N nodes, a random selection of K elements of Φ are selected, and the key k of each selected pair (k, λ) is assigned to the node. The value λ in each selected pair (k, λ) is decremented, and the pair is replaced back into Φ if $\lambda > 0$. Thus, as the algorithm proceeds, $|\Phi|$ decreases. Near the termination of the algorithm, it is possible that $\sum_{(k,\lambda)\in\Phi} \lambda = K$ but $|\Phi| < K$, leading to a case where no set of

KSR$(N, K, \mathcal{P})$

$\Phi \leftarrow \emptyset, j \leftarrow 1$
while $\sum_{(k,\lambda)\in\Phi} \lambda < N \cdot K$
  $\Phi \leftarrow \Phi \cup \{(k_j, sample(\mathcal{P}))\}$
  $j \leftarrow j + 1$
end while
for $n \in \mathcal{N}$
  $\Phi_0 \leftarrow \{(k, \lambda) \in \Phi : \lambda > 0\}$
  $E \leftarrow select\,(\Phi_0, \min(K, |\Phi_0|))$
  if $|E| < K$
    $F \leftarrow select\,(\Phi \setminus E, K - |E|)$
    $E \leftarrow E \cup F$
  end if
  assign $\{k : (k, \lambda) \in E\}$ to n
  $(k, \lambda) \leftarrow (k, \lambda - 1)$ for $(k, \lambda) \in E$
end for

(a)



(b)

Figure 2.4: The KSR key assignment algorithm is illustrated (a) in pseudo-code and (b) graphically with numbered steps: 1- select key subset, 2 - assign keys to node, 3 - decrement λ for each key, 4 - replace keys.

K unique keys can be assigned to a remaining node. Hence, if $|\Phi| = K_0 < K$, the $(K - K_0)$ remaining keys must be selected from those which have already been removed from Φ. If any of the K_0 keys were initially assigned a sample value of $\lambda_{min} = \min\{\lambda \in \Lambda\}$, these keys will correspond to boundary sets of size $\lambda_{min} - 1$. Furthermore, if any of the $(K - K_0)$ keys selected from those which were already removed from Φ were initially assigned a sample value of $\lambda_{max} = \max\{\lambda \in \Lambda\}$, these keys will correspond to boundary sets of size $\lambda_{max} + 1$. Pseudo-code for the KSR algorithm is provided in Figure 2.4(a), and a graphic illustration is provided in Figure 2.4(b).

Key Selection with No Replacement (KSNR)

The KSNR algorithm performs *selection without replacement* from a set Φ containing pairs (k, λ) where $k \in \mathcal{K}$ and $\lambda \in \Lambda$ is a sample of the assignment distribution $\mathcal{P}$. The number of keys $P = |\mathcal{K}| = |\Phi|$ must be sufficient to provide a total of NK edges in the graph g. Hence, we require $\sum_{(k,\lambda)\in\Phi} \lambda \geq NK$. Once Φ is constructed, keys are assigned to each

node using random selection without replacement in a total of $\lambda_{max} = \max\{\lambda \in \Lambda\}$ rounds. In a single round, which continues as long as Φ is non-empty, a random subset of K pairs (k, λ) in Φ is selected without replacement for each subsequent node, and the value λ in each selected pair is decremented. Pairs (k, λ) such that $\lambda = 0$ are permanently removed from Φ for all subsequent rounds, so the initial size of Φ can decrease in every subsequent round. In a given round, if K is not a factor of $|\Phi|$, there will be $K_0 < K$ keys remaining for the last node of the round. These K_0 keys can be combined with a random selection of $(K - K_0)$ keys which have not been permanently removed from Φ. The $(K - K_0)$ selected pairs will then be excluded from the subsequent round of the algorithm. If this occurs in the K^{th} round, any of the $(K - K_0)$ selected keys which were initially assigned a sample value of $\lambda_{max} = \max\{\lambda \in \Lambda\}$ will yield a boundary set of size $\lambda_{max} + 1$. Pseudo-code for the KSNR algorithm is provided in Figure 2.5(a), and a graphic illustration is provided in Figure 2.5(b).

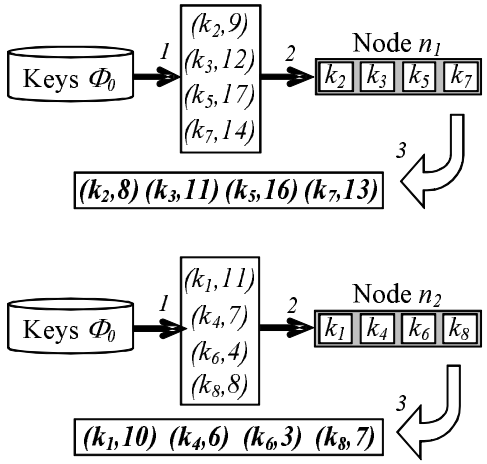
Node Selection with Replacement (NSR)

The NSR algorithm performs *selection with replacement* from a set Φ containing pairs (n, c) where $n \in \mathcal{N}$ and $c \geq 0$ counts the number of keys assigned to node n. For each key, a sample λ is generated from the assignment distribution $\mathcal{P}$, and a set of λ pairs (n, c) are selected from Φ. The assignment set for the given key is composed of the n entries in the λ selected pairs. Each time a pair (n, c) is selected, the counter c is incremented, and the pair is replaced back into Φ only if $c < K$. Hence, $|\Phi|$ decreases as the algorithm proceeds. As soon as $|\Phi| < \lambda_{max} = \max\{\lambda \in \Lambda\}$, it is possible for the sampled value of λ to be less than $|\Phi|$, so the entire set Φ is selected. If $|\Phi| < \lambda_{min} = \min\{\lambda \in \Lambda\}$, this will lead to boundary sets which vary in size between 1 and $\lambda_{min} - 1$. In simulation, a majority of the boundary sets which occur have size much smaller than $\lambda_{min} - 1$. Pseudo-code for the NSR algorithm is provided in Figure 2.6(a), and a graphic illustration is provided in Figure 2.6(b).

KSNR$(N, K, \mathcal{P})$

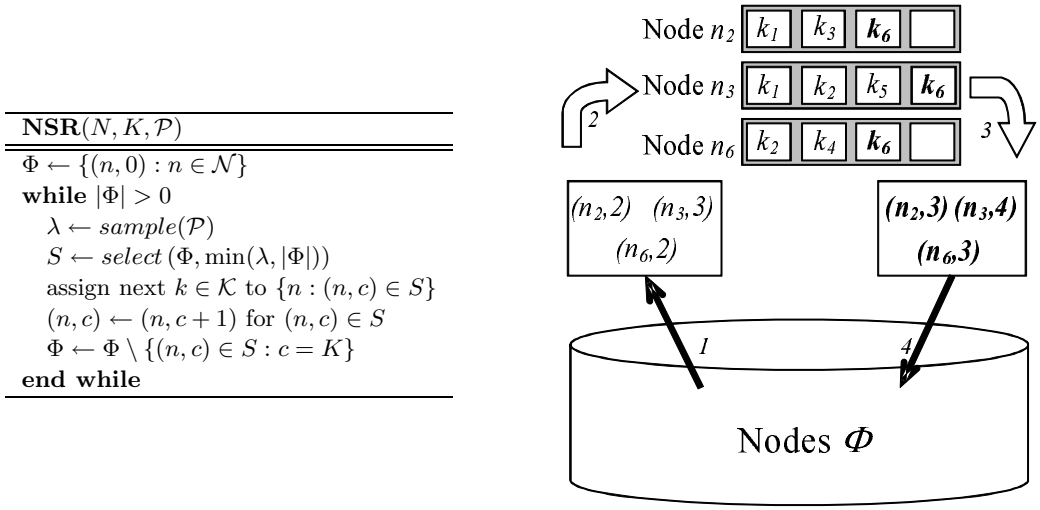
$\Phi, \Phi_1 \leftarrow \emptyset, j \leftarrow 1$
while $\sum_{(k,\lambda)\in\Phi} \lambda < N \cdot K$
  $\Phi \leftarrow \Phi \cup \{(k_j, sample(\mathcal{P}))\}$
  $j \leftarrow j + 1$
end while
$\lambda_{max} = \max\{\lambda : (k, \lambda) \in \Phi\}$
for i from 1 to λ_{max}
  $\Phi_0 \leftarrow \Phi \setminus \Phi_1$
  while $|\Phi_0| \geq K$
    $E \leftarrow select(\Phi_0, K)$
    $\Phi_0 \leftarrow \Phi_0 \setminus E$
    assign $\{k : (k, \lambda) \in E\}$ to next $n \in \mathcal{N}$
    $(k, \lambda) \leftarrow (k, \lambda - 1)$ for $(k, \lambda) \in E$
  end while
  $\Phi \leftarrow \Phi \setminus \{(k, \lambda) : \lambda = 0\}$
  if $|\Phi_0| > 0$
    $\Phi_1 \leftarrow select\,(\Phi \setminus \Phi_0, K - |\Phi_0|)$
    assign $\{k : (k, \lambda) \in \Phi_0 \cup \Phi_1\}$ to next $n \in \mathcal{N}$
    $(k, \lambda) \leftarrow (k, \lambda - 1)$ for $(k, \lambda) \in \Phi_0 \cup \Phi_1$
  end if
end for

(a)



(b)

Figure 2.5: The KSNR key assignment algorithm is illustrated (a) in pseudo-code and (b) graphically with numbered steps: 1 - select subset of keys, 2 - assign keys to node, 3 - decrement λ for each key.



(a) (b)

Figure 2.6: The NSR key assignment algorithm is illustrated (a) in pseudo-code and (b) graphically with numbered steps: 1 - select subset of nodes, 2 - assign key to nodes, 3 - increment c for each node, 4 - replace nodes.

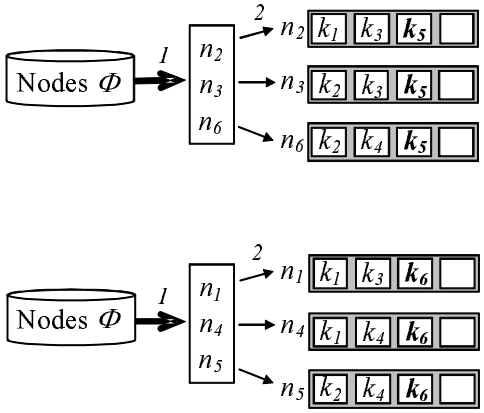
Node Selection with No Replacement (NSNR)

The NSNR algorithm performs *selection without replacement* from the set Φ, initially equal to $\mathcal{N}$. Assignment sets are generated using random selection without replacement in a total of K rounds. In a single round, which continues as long as Φ is non-empty, a sample λ is generated from the assignment distribution $\mathcal{P}$, a set of λ nodes in Φ is selected for each subsequent key, and the key is assigned to the selected nodes. If the sample λ is such that $|\Phi| < \lambda$, the key is assigned to the $|\Phi|$ remaining nodes and a random selection of $(\lambda - |\Phi|)$ other nodes which are then removed from the subsequent round. In the K^{th} round, since we do not want to assign $(K + 1)$ keys to any node, the final key may be assigned to less than $\lambda_{min} = \min\{\lambda \in \Lambda\}$ nodes, resulting in a single boundary set of size between 1 and $\lambda_{min} - 1$. We note that if $\Lambda = \{\lambda\}$ and λ is a factor of N, the NSNR algorithm will not yield boundary sets, and the result of the algorithm is equivalent to a deterministic key assignment algorithm similar to those of [13, 45]. Pseudo-code for the NSNR algorithm is provided in Figure 2.7(a), and a graphic illustration is provided in Figure 2.7(b).

NSNR$(N, K, \mathcal{P})$

$\Phi_1 \leftarrow \emptyset$
for i from 1 to K
  $\Phi \leftarrow \mathcal{N} \setminus \Phi_1$
  while $|\Phi| > 0$
    $\lambda \leftarrow sample(\mathcal{P})$
    $S \leftarrow select\,(\Phi, \min(\lambda, |\Phi|))$
    if $|S| < \lambda$ and $i < K$
      $\Phi_1 \leftarrow select\,(\mathcal{N} \setminus S, \lambda - |S|)$
      $S \leftarrow S \cup \Phi_1$
    end if
    assign next $k \in \mathcal{K}$ to $\{n \in S\}$
    $\Phi \leftarrow \Phi \setminus S$
  end while
end for

(a)



(b)

Figure 2.7: The NSNR key assignment algorithm is illustrated (a) in pseudo-code and (b) graphically with numbered steps: 1 - select subset of nodes, 2 - assign key to nodes.

The algorithms proposed herein yield a result that is essentially similar. The primary differences are the behavior of the boundary sets which result and their computational cost. Though these sets occur non-deterministically, their general behavior can be characterized. Furthermore, there tends to be a trade-off between the computational cost of an algorithm and the resulting boundary distance, in that the boundary distance can be decreased at the expense of increased computation. Hence, the choice of algorithm may depend on the desired boundary distance tolerance and the allowable computational cost.

2.5 Analysis of Key Assignment Schemes

In this section, we provide general analysis for a key assignment scheme $(\mathcal{P}, \mathscr{A})$ assuming the impact of any boundary sets is negligible. We compute the probability that a pair of nodes share a given number of keys, the probability of network connectivity, and the resilience to node capture. Each of the quantities is provided in such a way that the worst case with respect to the assignment distribution $\mathcal{P}$ can be easily determined. Furthermore, the average case is computed for each quantity with respect to the assignment distribution

$\mathcal{P}$. The average case is most helpful in determining sufficiency of network parameters, while the worst case is most helpful in determining whether a given set of parameters will result in undesirable tail-effects as discussed in Section 2.2.1.

2.5.1 Probability of Sharing Keys

The average and worst-case analysis of a key predistribution scheme can be performed with respect to the probability that a pair of nodes share any number of keys. In addition to performance analysis, this probability is important for various applications based on local connectivity properties. For example, the q-composite scheme of [16] requires a pair of nodes to share at least q keys for some $q \geq 1$. We compute the probability $p_s(i)$ that a pair of nodes share exactly i keys as a function of the assignment set sizes λ corresponding to the keys in each node. We then compute the average probability taken over the assignment distribution $\mathcal{P}$.

Lemma 2.3. *A node u containing a key k, such that $\lambda = |S(k)|$ is known, will share k with a node v with probability $p_\lambda = \frac{\lambda-1}{N-1}$.*

Proof. Given a node u containing k, exactly $(\lambda - 1)$ of the remaining $(N-1)$ nodes contain k. Hence, the probability that v is one of these $(\lambda - 1)$ nodes is $\frac{\lambda-1}{N-1}$. □

Theorem 2.2. *A node u containing keys $k_1, \ldots, k_K$, such that $\lambda_j = |S(k_j)|$ for $j = 1, \ldots, K$ are known, will share exactly i keys with a node v with probability $p_s(i, \lambda_1, \ldots, \lambda_K)$ given by*

$$p_s(i, \lambda_1, \ldots, \lambda_K) = \frac{1}{i!(K-i)!} \sum_{\pi} \left(\prod_{j=1}^{i} \frac{\lambda_{\pi_j} - 1}{N-1} \times \prod_{j=i+1}^{K} \frac{N - \lambda_{\pi_j}}{N-1} \right)$$

where the summation is over all permutations $\pi = (\pi_1, \ldots, \pi_K)$ of $(1, \ldots, K)$.

Proof. The event that v shares k_j with u can be modeled as a Bernoulli trial with success probability p_{λ_j} given by Lemma 2.3. Since the assignment sets are chosen independently, the K events are independent. Hence, the number of events i which occur is given by the sum of the K independent Bernoulli random variables. The probability that exactly i of the K events occur is given by the sum over all possible choices of i of the K events. For a

given choice of i events, the contribution to the overall probability is the product of p_{λ_j} for the i events which occur multiplied by the product of $1-p_{\lambda_j}$ for the $(K-i)$ events which do not occur. The term $\frac{1}{i!(K-i)!}$ is added to compensate for the $i!(K-i)!$ permutations which result in the same choice of i events. □

Theorem 2.3. *A node u will share exactly i keys with a node v with probability $p_s(i)$ given by*

$$p_s(i) = \binom{K}{i} \left(\frac{\mu - 1}{N-1}\right)^i \left(\frac{N-\mu}{N-1}\right)^{K-i}$$

where μ is the average assignment set size according to the assignment distribution $\mathcal{P}$.

Proof. The probability $p_s(i)$ can be computed by taking the expected value of the probability $p_s(i, \lambda_1, \ldots, \lambda_K)$ given in Theorem 2.2 with respect to the set of samples $\lambda_1, \ldots, \lambda_K$. Hence, letting $\mathcal{E}[\cdot]$ represent this expected value, $p_s(i)$ is given by

$$p_s(i) = \mathcal{E}\left[\frac{1}{i!(K-i)!} \sum_{\pi} \left(\prod_{j=1}^{i} \frac{\lambda_{\pi_j} - 1}{N-1} \prod_{j=i+1}^{K} \frac{N - \lambda_{\pi_j}}{N-1}\right)\right]. \tag{2.17}$$

Since the samples λ_j are independent, this is equivalent to taking the expected value with respect to each λ_j. Moving the expected value within the summation and using the independence of the λ_j yields

$$p_s(i) = \frac{1}{i!(K-i)!} \sum_{\pi} \left(\prod_{j=1}^{i} \frac{\mathcal{E}_{\pi_j}[\lambda_{\pi_j}] - 1}{N-1} \times \prod_{j=i+1}^{K} \frac{N - \mathcal{E}_{\pi_j}[\lambda_{\pi_j}]}{N-1}\right). \tag{2.18}$$

Identical distribution of the λ_j suggests that each $\mathcal{E}_{\pi_j}[\lambda_{\pi_j}]$ is equal to the mean μ of the assignment distribution $\mathcal{P}$. The product terms are thus independent of the index j, and the summands are independent of the permutation π, so the sum-of-products form is replaced by a single product of powers with coefficient $\frac{K!}{i!(K-i)!}$. Replacing this coefficient with $\binom{K}{i}$ completes the proof. □

Theorem 2.2 and Theorem 2.3 are useful in respectively determining the worst-case and average probability of sharing keys. Theorem 2.2 is particularly applicable to the worst-case analysis in that it can be used to compute the worst-case probability of sharing keys regardless of how the worst case is defined. For example, the designer of the key predistribution

scheme can design an assignment distribution based on a given tolerance to one minimal λ value by bounding the probability $1-p_s(0, \lambda_{min}, \mu, \ldots, \mu)$. The designer can similarly design the key predistribution scheme based on the expected worst-case probability by bounding the probability $1 - p_s(0, \lambda_{min}^{(1)}, \ldots, \lambda_{min}^{(K)})$ where $\lambda_{min}^{(1)}, \ldots, \lambda_{min}^{(K)}$ are order statistics similar to those discussed in Section 2.2.1.

2.5.2 Network Connectivity

The probability of connectivity of the secure WSN is given by Theorem 2.1 in Section 2.3.2 as a function of the expected node degree D in the logical graph $G_L(N, \mathcal{R})$. For simplicity, we assume the relation $\mathcal{R}$ is true if and only if the given pair of nodes share at least one key. Similar results can be derived for the modified relations of schemes such as the q-composite scheme [16]

We note that there are two forms of randomness present in a key assignment algorithm. The number of nodes λ is sampled randomly from the assignment distribution $\mathcal{P}$, and the assignment set of λ nodes is selected randomly. We first compute the expected degree $d(u)$ of a node u assuming the sizes $\lambda_1, \ldots, \lambda_K$ of the K assignment sets corresponding to the keys stored in node u are fixed and known. This computation is performed using a combinatorial occupancy problem in which each pair (u, v), for $v \in \mathcal{N} \setminus \{u\}$, is represented by a bin and a shared key between nodes u and v is represented by a ball in the bin representing the pair (u, v). The assignment of a key k_j to node u and $(\lambda_j - 1)$ of the $(N-1)$ other nodes thus corresponds to placing one ball in each of $(\lambda_j - 1)$ of the $(N-1)$ bins. This occupancy problem is illustrated in Figure 2.8. The degree $d(u)$ of node u in the graph $G_L(N, \mathcal{R})$ is given by the number of bins (u, v) which contain at least one ball. The expected node degree D is computed by taking the expected value of the node degree $d(u)$ over all possible values of $\lambda_1, \ldots, \lambda_K$ according to a given assignment distribution $\mathcal{P}$.

Lemma 2.4. *A node u with keys $k_1, \ldots, k_K$, such that $\lambda_j = |S(k_j)|$ for $j = 1, \ldots, K$ are known, will not share a key with $e(u)$ nodes according to the probability $Pr[e(u) \geq E]$ given by*

$$Pr[e(u) \geq E] = \sum_{m=E}^{N-1} (-1)^{m-E} \binom{m-1}{E-1} \binom{N-1}{m} \prod_{j=1}^{K} \frac{\binom{N-1-m}{\lambda_j - 1}}{\binom{N-1}{\lambda_j - 1}}.$$

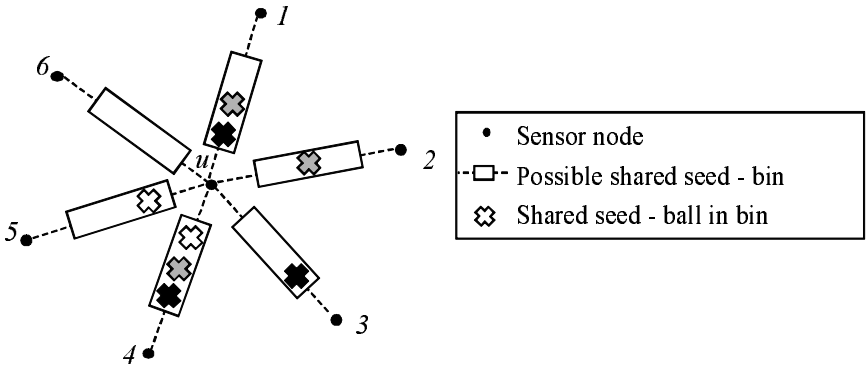


Figure 2.8: Key assignment to nodes in the WSN is represented by a combinatorial occupancy problem where each pair of nodes (u, v) is represented by a bin, and a shared key between nodes u and v is indicated by a ball in the bin (u, v).

Proof. Placing $(\lambda_j - 1)$ balls in $(N-1)$ bins such that a given set of m bins remain empty can be done in exactly $\binom{N-1-m}{\lambda_j - 1}$ ways. Thus, the number of ways to assign K keys in such a way that a particular set of m bins remains empty is given by the product $\prod_{j=1}^{K} \binom{N-1-m}{\lambda_j - 1}$. The number of ways to select the m bins to remain empty is $\binom{N-1}{m}$. By the Inclusion-Exclusion Principle [43], the number of ways $M(E)$ that K subsets of bins can be chosen such that at least E bins remain empty is given by

$$M(E) = \sum_{m=E}^{N-1} (-1)^{m-E} \binom{m-1}{E-1} \binom{N-1}{m} \prod_{j=1}^{K} \binom{N-1-m}{\lambda_j - 1}. \tag{2.19}$$

Dividing $M(E)$ by the total number of ways to choose the K subsets given by $M(0)$ yields the probability that at least E bins remain empty. □

Theorem 2.4. *A node u with keys $k_1, \ldots, k_K$, such that $\lambda_j = |S(k_j)|$ for $j = 1, \ldots, K$ are known, will have expected degree $\mathcal{E}[d(u)]$ in the logical graph $G_L(N, \mathcal{R})$ given by*

$$\mathcal{E}[d(u)] = (N-1)\left(1 - \prod_{j=1}^{K} \frac{\lambda_j - 1}{N-1}\right).$$

Proof. The expected number of empty bins $\mathcal{E}[e(u)]$ can be computed using the fact that

$$\mathcal{E}[e(u)] = \sum_{E=1}^{N-1} Pr[e(u) \geq E] \tag{2.20}$$

since $e(u)$ is a non-negative discrete random variable [32]. Substituting the result of Lemma 2.4 into (2.20) provides an expression for $\mathcal{E}[e(u)]$. The expected degree $\mathcal{E}[d(u)]$ is then given by

$$\mathcal{E}[d(u)] = N - 1 - \mathcal{E}[e(u)] \tag{2.21}$$

because each non-empty bin corresponds to an edge in the graph $G_L(N, \mathcal{R})$. Replacing $\mathcal{E}[e(u)]$ with the result from Lemma 2.4 yields

$$\mathcal{E}[d(u)] = N - 1 - \sum_{E=1}^{N-1} \sum_{m=E}^{N-1} (-1)^{m-E} \binom{N-1}{m} \binom{m-1}{E-1} \prod_{j=1}^{K} \frac{\binom{N-1-m}{\lambda_j - 1}}{\binom{N-1}{\lambda_j - 1}}. \tag{2.22}$$

The order of summation can be reversed by changing the limits of summation to sum over $m = 1, \ldots, N-1$ and $E = 1, \ldots, m$. Terms which are independent of E can then be moved outside of the inner summation, yielding

$$\mathcal{E}[d(u)] = N - 1 - \sum_{m=1}^{N-1} \binom{N-1}{m} \prod_{j=1}^{K} \frac{\binom{N-1-m}{\lambda_j - 1}}{\binom{N-1}{\lambda_j - 1}} \sum_{E=1}^{m} (-1)^{m-E} \binom{m-1}{E-1}. \tag{2.23}$$

The binomial theorem suggests that

$$\sum_{E=1}^{m} (-1)^{m-E} \binom{m-1}{E-1} = \sum_{E=0}^{m-1} (-1)^{m-1-E} \binom{m-1}{E} = 0^{m-1}. \tag{2.24}$$

Since $0^0 = 1$, the only non-zero term of the summation is when $m = 1$. Hence the expected degree of node u is given by

$$\mathcal{E}[d(u)] = N - 1 - \binom{N-1}{1} \prod_{j=1}^{K} \frac{\binom{N-2}{\lambda_j - 1}}{\binom{N-1}{\lambda_j - 1}} = (N-1) \left(1 - \prod_{j=1}^{K} \frac{N - \lambda_j}{N-1} \right). \tag{2.25}$$

□

Theorem 2.5. *The expected node degree D in the logical graph $G_L(N, \mathcal{R})$ is given by*

$$D = (N-1) \left(1 - \left(\frac{N-\mu}{N-1} \right)^K \right).$$

Proof. The expected node degree D is computed by taking the expected value of $\mathcal{E}[d(u)]$ given by Theorem 2.4 with respect to each of the set of random variables $\lambda_1, \ldots, \lambda_K$. Denoting this expected value by $\mathcal{E}[\cdot]$ yields

$$D = \mathcal{E} \left[(N-1) \left(1 - \prod_{j=1}^{K} \frac{N - \lambda_j}{N-1} \right) \right]. \tag{2.26}$$

Since the samples $\lambda_1, \ldots, \lambda_K$ are independent, this is equivalent to taking the expected value with respect to each of the random variables λ_j, denoted by $\mathcal{E}_j[\cdot]$. This independence yields

$$D = (N-1)\left(1 - \prod_{j=1}^{K} \frac{N - \mathcal{E}_j[\lambda_j]}{N-1}\right). \tag{2.27}$$

Identical distribution of the λ_j suggests that $\mathcal{E}_j[\lambda_j]$ can be replaced by the mean μ of the assignment distribution $\mathcal{P}$ completing the proof. □

The result of Theorem 2.5 can then be used in conjunction with Theorem 2.1 to yield the probability $P_G(k)$ that the restricted network graph $G(N, \mathcal{A}, r, \mathcal{R})$ is k-connected. Hence, given the number N of sensors in the network, key storage K, desired connectivity k, deployment density ρ, and radio range r, the mean μ of the assignment distribution $\mathcal{P}$ can be chosen to guarantee k-connectivity with the desired probability.

2.5.3 Resilience to Attacks

Since every key is assigned to multiple nodes, a key may be used to establish many secure links throughout the network. Thus, an adversary who randomly captures nodes may be able to decrypt secure communication links between uncaptured nodes, referred to as *link compromise*. The average probability of link compromise $f(x)$ due to the capture of x nodes often depends on the underlying structure of the key predistribution scheme. Hence, for generality, our primary security metric is the probability $p(m, x)$ that exactly m of the x captured nodes contain a given key. Similar to the results of Section 2.5.1, we first compute the results when the assignment set size λ of the given key is fixed and known, and then compute the average probability as a function of the assignment distribution $\mathcal{P}$.

Lemma 2.5. *Given uncaptured nodes u and v which share a key k such that $\lambda = |S(k)|$ is known, the probability $p(m, x, \lambda)$ that exactly m of the x captured nodes contain s is given by*

$$p(m, x, \lambda) = \sum_{I} \left(\prod_{j=1}^{m} \frac{\lambda - j - 1}{N - I_j - 1} \times \prod_{i \notin I} \frac{N - \lambda - i + m_i + 1}{N - i - 1} \right)$$

where the summation is taken over all vectors $I = (I_1, \ldots, I_m)$ *such that* $1 \leq I_1 < \ldots < I_m \leq x$ *and* $m_i = \max\{h : I_h < i\}$.

Proof. Each successive node capture can be modeled as a Bernoulli trial which is successful if an additional copy of the key k is contained in the captured node. The success probability of the x^{th} trial, however, depends on the number of previously successful trials because the maximum number of successful trials is fixed at λ. Hence, the Bernoulli trials are not independent. Letting $I = (I_1, \ldots, I_m)$ represent the indices of the m successful trials out of the x attempts. In trial i, given that m_i nodes containing k have been captured, the probability that one of the $\lambda - 2 - m_i$ nodes containing the key k was selected randomly from the $(N-2)-(i-1)$ nodes remaining in the network is given by $\frac{\lambda-2-m_i}{N-i-1}$. The number of previously captured nodes m_i is given by the number of indices I_h in I with $h < i$, i.e. $m_i = \max\{h : I_h < i\}$. The contribution $p(m, x, \lambda, I)$ for a given vector I is thus equal to the product of the success probabilities for the m trials $I_1, \ldots, I_m$ and the failure probabilities for the $(x - m)$ remaining trials given by

$$p(m, x, \lambda, I) = \prod_{i \in I} \frac{\lambda - 2 - m_i}{N - i - 1} \times \prod_{i \notin I} \frac{N - \lambda - i + m_i + 1}{N - i - 1}. \tag{2.28}$$

For $I_j \in I$, the value of m_{I_j} is simply given by the number of prior successes $(j-1)$. Hence, the contribution $p(m, x, \lambda, I)$ for a given vector I is given by

$$p(m, x, \lambda, I) = \prod_{j=1}^{m} \frac{\lambda - j - 1}{N - I_j - 1} \times \prod_{i \notin I} \frac{N - \lambda - i + m_i + 1}{N - i - 1}. \tag{2.29}$$

The final result is obtained by summing over all possible I. □

Lemma 2.6. *Given uncaptured nodes* u *and* v *which share a key* k *such that* $\lambda = |S(k)|$ *is known, if* $x \ll N - 2$ *and* $m \ll \lambda - 2$*, the probability* $p(m, x, \lambda)$ *that exactly* m *of the* x *captured nodes contain* k *can be approximated as*

$$p(m, x, \lambda) \approx \binom{x}{m} \left(\frac{\lambda - 2}{N - 2}\right)^m \left(\frac{N - \lambda}{N - 2}\right)^{x-m}.$$

Proof. If $x \ll N$ and $m \ll \lambda - 2$ and $m_i = \max\{h : I_h < i\}$ as in Lemma 2.5, then the

approximations

$$\frac{(\lambda-2)-(m_i-1)}{(N-2)-(x-1)} \approx \frac{\lambda-2}{N-2} \tag{2.30}$$

$$\frac{(N-2)-(\lambda-2-m_i)-(x-1)}{(N-2)-(x-1)} \approx \frac{(N-2)-(\lambda-2)}{N-2} = \frac{N-\lambda}{N-2} \tag{2.31}$$

can be substituted into the result of Lemma 2.5 yielding

$$p(m,x,\lambda) = \sum_{I} \left(\prod_{j=1}^{m} \frac{\lambda-2}{N-2} \times \prod_{i \notin I} \frac{N-\lambda}{N-2} \right). \tag{2.32}$$

Each product term is independent of the indices i and j, so the result reduces to

$$p(m,x,\lambda) = \sum_{I} \left(\frac{\lambda-2}{N-2} \right)^{m} \left(\frac{N-\lambda}{N-2} \right)^{x-m}. \tag{2.33}$$

Furthermore, the summand is independent of the index I, so the summation over I can be replaced by the summand multiplied by $\binom{x}{m}$ corresponding to the number of possible vectors I. □

Theorem 2.6. *Given uncaptured nodes u and v which share a key k, the probability $p(m,x)$ that exactly m of the x captured nodes contain s can be approximated as*

$$p(m,x) \approx \binom{x}{m} \left(\frac{\mu-2}{N-2} \right)^{m} \left(\frac{N-\mu}{N-2} \right)^{x-m}$$

where μ is the mean of a given assignment distribution $\mathcal{P}$.

Proof. This result is an approximation to the result of Lemma 2.6 obtained by replacing the λ by the mean μ of the random variable λ with respect to the assignment distribution $\mathcal{P}$. □

The approximations in Lemma 2.6 and Theorem 2.6 are useful in respectively approximating the worst-case and average probability of link compromise. The average probability of link compromise $f(x)$ is dependent on the application and the link-key establishment protocol, though it is typically a function of $p(m,x)$ approximated by Theorem 2.6. Since $p(m,x)$ depends only on the mean μ of the assignment distribution $\mathcal{P}$, the network size N, and the number of captured nodes x, it can be a useful metric in designing the assignment distribution $\mathcal{P}$.

The probability of link compromise $f(x, \lambda)$ for a link secured by a key shared by λ nodes is also application- and protocol-dependent, though it is typically a function of $p(m, x, \lambda)$ approximated by Lemma 2.6. The worst-case probability of link compromise can thus be computed as $f(x, \lambda_{max})$ where λ_{max} is similar to that discussed in Section 2.2.1. Since $p(m, x, \lambda_{max})$ depends only on the maximum value λ_{max} in the support of the assignment distribution $\mathcal{P}$, the network size N, and the number of captured nodes x, it can be a useful metric in designing the assignment distribution $\mathcal{P}$.

2.6 Assignment Distributions

Due to the fact that the algorithms in the sampling framework presented in Section 2.4.3 can realize a given assignment distribution $\mathcal{P}$ with negligible occurrence of boundary sets, the assignment distributions can be designed independently of the key assignment algorithms. Hence, assignment distributions can be designed with respect to the analytical results in Section 2.5 in terms of average and worst-case network connectivity and resilience to node capture. However, the finite sampling effects described in Definition 2.9 must still be considered in the design of an assignment distribution.

In general, the design of an assignment distribution depends highly on the application requirements and link-key establishment scheme. Furthermore, the average network connectivity and resilience to node capture depend only on the mean μ of the assignment distribution $\mathcal{P}$. Hence, the optimal assignment distribution is *application specific*.

The design of an assignment distribution $\mathcal{P}$ can be broken into two primary steps. The first step is to determine the support Λ of the assignment distribution, and the second step is to determine the probability mass $\mathcal{P}(\lambda)$ for every $\lambda \in \Lambda$.

2.6.1 Assignment Distribution Support

In order to compensate for finite sampling effects, the size of the support Λ of the assignment distribution $\mathcal{P}$ should be larger than 1. In contrast, however, the size of Λ should be as small as possible to avoid the undesirable tail-effects discussed in Section 2.2.1. Though not a requirement, we assume the support Λ is a contiguous subset of $\{0, \ldots, N\}$, i.e. if $\lambda_1, \lambda_2 \in \Lambda$ then $\lambda \in \Lambda$ for all $\lambda \in \{0, \ldots, N\}$ such that $\lambda_1 \leq \lambda \leq \lambda_2$. Furthermore, Λ should contain the

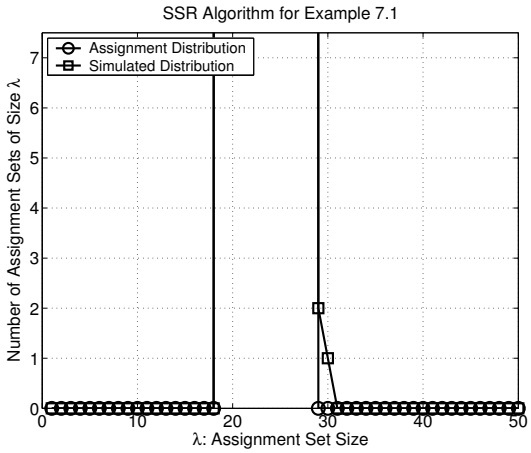
values nearest to the average value μ of the assignment distribution $\mathcal{P}$ required for sufficient network connectivity as given by Theorem 2.1 and Theorem 2.5. Hence, the design of the support Λ is equivalent to determination of $\lambda_{min} = \min\{\lambda \in \Lambda\}$ and $\lambda_{max} = \max\{\lambda \in \Lambda\}$ such that $\lambda_{min} \leq \mu \leq \lambda_{max}$.

In order to determine the value of λ_{min}, we consider the worst-case probability of sharing keys $p_s(i, \lambda_{min}, \ldots, \lambda_{min})$ as given by Theorem 2.2. Similarly, to determine the value of λ_{max}, we consider the worst-case resilience to node capture in terms of the probability $p(m, x, \lambda_{max})$ as given by Lemma 2.5 and approximated by Lemma 2.6. Furthermore, we must consider the finite sampling effects which arise due to the choice of λ_{min}, λ_{max}, and the key assignment algorithms. For the KSR and KSNR algorithms, only boundary sets with distance 1 can occur, so the finite sampling effects can be seen as negligible. However, for the NSR and NSNR algorithms, boundary sets with distance between 1 and $\lambda_{min} - 1$ may occur. Hence, for the NSR and NSNR algorithms, we are interested in minimizing the boundary distance of the resulting boundary sets. Through simulation, we note that as the value $|\Lambda| = \lambda_{max} - \lambda_{min} + 1$ decreases, the distance of boundary sets due to finite sampling effects tends to increase. Thus, to avoid boundary sets with large distance, λ_{max} should be increased and λ_{min} should be decreased. Hence, there exists a trade-off between improving the worst-case probability of sharing keys, improving the worst-case resilience to node capture, and minimizing the boundary distance of boundary sets which occur due to finite sampling effects. Therefore, determining the optimal values of λ_{min} and λ_{max} is application dependent.

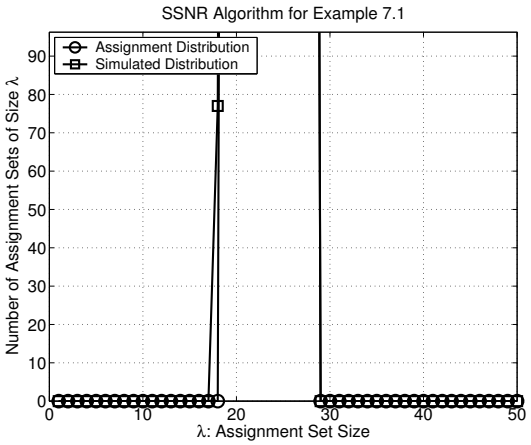
2.6.2 Probability Mass on Λ

Once the support Λ of the assignment distribution $\mathcal{P}$ is determined, the probability mass $\mathcal{P}(\lambda)$ for each $\lambda \in \Lambda$ must be determined. However, if $|\Lambda| > 1$, there are an uncountably infinite number of possible assignment distributions for given values of μ, λ_{min}, and λ_{max}, leading to a high degree of freedom in determining the assignment distribution $\mathcal{P}$.

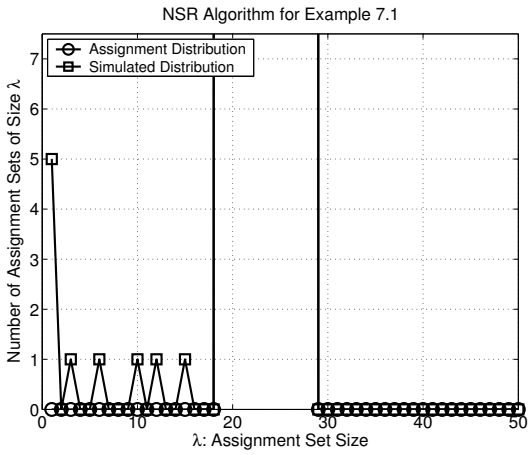
As worst-case probability of sharing keys and the worst-case resilience to node capture are best mitigated by an assignment distribution with trivial support, i.e. $|\Lambda| = 1$, we



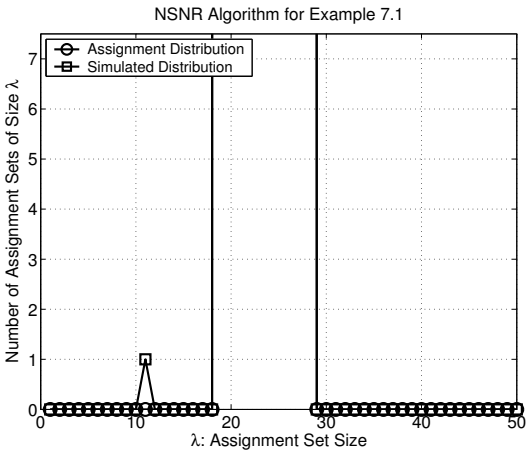
(a)



(b)



(c)



(d)

Figure 2.9: Occurrence of boundary sets for Example 2.2 using (a) KSR algorithm (b) KSNR algorithm (c) NSR algorithm (d) NSNR algorithm.

approximate this performance by placing more probability mass on the values of λ nearest to μ, resulting in an assignment distribution which is peaked near μ and decreases as $|\mu - \lambda|$ increases.

2.6.3 Illustration of Assignment Distribution Design

We provide the following example to illustrate the design of an assignment distribution for a given link-key establishment scheme and set of network parameters.

Example 2.2. *Let a WSN of $N = 5,000$ nodes with $K = 100$ keys per node and a radio range of $r = 40$ m be deployed over a region $\mathcal{A}$ of area $|\mathcal{A}| = 0.5$ km^2 such that 2-connectivity is desired with probability 0.99. We assume that any nodes sharing at least one key can establish a link-key as a function of the shared keys and a link can be compromised as soon as the keys used to compute the link-key are captured. Furthermore, we assume that the value λ_{min} must be such that the worst-case probability of sharing keys is within 20% of the average probability of sharing keys, and the value λ_{max} must be such that the worst-case probability of link compromise is within 20% of the average probability of link compromise for $x = 50$ captured nodes.*

Theorem 2.1 yields a minimum average vertex degree of $D = 1,813$ in the logical graph $G_L(N, \mathcal{R})$. Theorem 2.5 yields a minimum average assignment set size of $\mu \geq 23.47$. The average probability of sharing at least 1 key is given by Theorem 2.3 as $(1 - p_s(0)) = 0.3627$. Hence, the value of λ_{min} must result in $(1 - p_s(0, \lambda_{min}, \ldots, \lambda_{min})) \geq 0.3$. Theorem 2.2 yields $\lambda_{min} \geq 19$. The value of λ_{max} must result in $1 - p(0, 50, \lambda_{max}) \leq 0.25$. Lemma 2.6 yields $\lambda_{max} \leq 30$. Hence, we choose the support $\Lambda = \{19, \ldots, 28\}$ and the symmetric probability mass function $\mathcal{P}$ given by

$$\mathcal{P}(\lambda) = \begin{cases} \frac{\lambda - 18}{30}, & \lambda \in \{19, \ldots, 23\} \\ \frac{29 - \lambda}{30}, & \lambda \in \{24, \ldots, 28\} \\ 0, & else \end{cases} \tag{2.34}$$

resulting in average assignment set size $\mu = 23.5 \geq 23.47$. Figure 2.9 displays the boundary sets which occur as a result of finite sampling effects for the assignment distribution given in (2.34) when each of the four algorithms in Section 2.4.3 is used.

2.7 Deployment of Additional Nodes in the WSN

In many applications, it may be necessary to deploy additional sensor nodes to replace those which have a depleted energy supply or to increase the coverage of an existing WSN. If the link-key establishment method is such that addition of nodes to the WSN does not require a prohibitive amount of communication overhead, the incorporation of the additional sensor nodes into the secure WSN can be described in terms of the canonical model. However, if a sufficient number of nodes are to be deployed into an existing WSN, the key assignment scheme for the subsequent deployment might be very different from that of the original deployment. We investigate such scenarios using the canonical model of key assignment schemes assuming that N nodes have been deployed using the key assignment scheme $(\mathcal{P}, \mathscr{A})$ and M additional nodes are to be deployed into the existing WSN. We provide a general approach for deployment of additional nodes and give an example which yields a well-known result.

In deploying M additional nodes into an existing WSN, it is highly desirable for the $N + M$ nodes to act as a single secure WSN, so the M additional nodes must be assigned keys which can be used to establish link-keys with any of the $N + M$ nodes. Furthermore, if M is sufficiently large, a subset of the keys assigned to the M additional nodes can be fresh. The exact proportion of fresh and existing keys used in key assignment for the additional nodes is application dependent, though it is computed as a function of N and M. For simplicity, we assume that a fraction f of the keys assigned to the M additional nodes are fresh, and the remaining fraction $(1 - f)$ of the keys are chosen randomly from the set of existing keys.

The key assignment scheme $(\mathcal{P}', \mathscr{A}')$ used to assign keys to the M additional nodes can be designed as a function of the key assignment scheme $(\mathcal{P}, \mathscr{A})$, the parameters N, M, and f, the total total number of keys P assigned to the N nodes in the existing WSN, and the total number of keys P' to be assigned to the M additional nodes. The overall assignment distribution $\mathcal{Q}$ for the network of $N + M$ nodes assigned keys using assignment distributions $\mathcal{P}$ and $\mathcal{P}'$ is given by the following theorem.

Theorem 2.7. *Given $N + M$ nodes such that N nodes are assigned a total of P keys*

using the assignment distribution $\mathcal{P}$ and M nodes are assigned a total of P' keys using the assignment distribution $\mathcal{P}'$ where a fraction f of the P' keys are fresh, the overall assignment distribution $\mathcal{Q}$ is given by

$$\mathcal{Q}(\lambda) = \frac{P-(1-f)P'}{P+fP'}\mathcal{P}(\lambda) + \frac{(1-f)P'}{P+fP'}(\mathcal{P}*\mathcal{P}')(\lambda) + \frac{fP'}{P+fP'}\mathcal{P}'(\lambda)$$

where $\mathcal{P}*\mathcal{P}'$ is the discrete convolution of the assignment distributions $\mathcal{P}$ and $\mathcal{P}'$ given by

$$(\mathcal{P}*\mathcal{P}')(\lambda) = \sum_{\nu\in\Lambda}\mathcal{P}(\nu)\mathcal{P}'(\lambda-\nu).$$

Proof. The probability that a key k is assigned only to the M additional nodes is equal to the number of fresh keys divided by total keys, $\frac{fP'}{P+fP'}$. The probability that k is assigned only to the N existing nodes is equal to the number of existing keys divided by total keys, $\frac{P-(1-f)P'}{P+fP'}$. The probability that k is assigned to both existing and additional nodes is then $\frac{(1-f)P'}{P+fP'}$. The probability that k is assigned to λ_1 existing nodes and λ_2 additional nodes can then be written in terms of $\mathcal{P}$, $\mathcal{P}'$, and $\mathcal{P}*\mathcal{P}'$ using the above probabilities as weights. □

The parameters of the assignment distribution $\mathcal{P}'$ can be chosen in a similar way to the methods described in Section 2.6 and the relationship between the expected value μ of $\mathcal{P}$, μ' of $\mathcal{P}'$, and γ of $\mathcal{Q}$ given by

$$\gamma = \frac{P-(1-f)P'}{P+fP'}\mu + \frac{(1-f)P'}{P+fP'}\left(\mu+\mu'\right) + \frac{fP'}{P+fP'}\mu' = \frac{P}{P+fP'}\mu + \frac{P'}{P+fP'}\mu'. \quad (2.35)$$

We note that, if $0 < f < 1$, the support of the distribution $\mathcal{Q}$ is necessarily larger than the support of either distribution $\mathcal{P}$ or $\mathcal{P}'$. Hence, the addition of nodes to the network with $0 < f < 1$ causes the assignment distribution to spread. This tends to increase the value of λ_{max} of the assignment distribution $\mathcal{Q}$ compared to that of either $\mathcal{P}$ or $\mathcal{P}'$, thus increasing the worst-case probability estimated by Lemma 2.6.

In both Theorem 2.7 and (2.35), the number of keys P' assigned to the M additional nodes may be unknown prior to key assignment, but it can be approximated by its expected value $P' = \frac{MK}{\mu'}$. We consider the following examples of deployment of additional nodes into a WSN.

Example 2.3. *Consider an existing network of N nodes, each of which is assigned K randomly selected keys from a set of P keys using the random key predistribution scheme of [29]. The assignment distribution $\mathcal{P}$ is thus given by the binomial distribution $\mathcal{B}(N, \frac{K}{P})$ with mean $\mu = \frac{NK}{P}$. In order to replace depleted nodes and reinforce the WSN, M additional nodes with K keys per node are to be added to the existing network. To paraphrase [29], since a sufficient number of the K keys stored in each node are not used to establish link-keys in the existing network, the same set of P keys can be used in the random key predistribution scheme for the M additional nodes. Hence, $P' = P$ and the distribution $\mathcal{P}'$ is given by the binomial distribution $\mathcal{B}(M, \frac{K}{P})$ with mean $\mu' = \frac{MK}{P}$. Since the same P keys are used, and the trial probability for the binomial distributions of $\mathcal{P}$ and $\mathcal{P}'$ are the same, this situation is equivalent to deploying a network of $N + M$ nodes with assignment distribution $\mathcal{Q}$ given by the binomial distribution $\mathcal{B}(N + M, \frac{K}{P})$ with mean $\gamma = \mu + \mu' = \frac{(N+M)K}{P}$. This result corresponds to the result of Theorem 2.7 with the given distributions $\mathcal{P}$ and $\mathcal{P}'$, $P = P'$, and $f = 0$.*

The result of Theorem 2.7 is far more general, however, than is illustrated by Example 2.3. The following example demonstrates the generality of Theorem 2.7.

Example 2.4. *Similar to Example 2.3, consider an existing network of N nodes, each of which is assigned K randomly selected keys from a set of P keys using the random key predistribution scheme of [29] with assignment distribution $\mathcal{P}$ given by the binomial distribution $\mathcal{B}(N, \frac{K}{P})$. The additional M nodes are assigned K keys each from a total of $P' = P$ keys, such that a given fraction f of the P' keys are fresh, and a fraction $(1 - f)$ are randomly selected from the initial set of P keys. For this example, we assume the assignment distribution $\mathcal{P}'$ is given by the a symmetric peaked distribution similar to that of (2.34) with $\Lambda = \{\lambda \in \{0, \ldots, N\} : |\lambda - \mu'| \leq 5\}$ where μ' is a given value for the mean of the assignment distribution $\mathcal{P}'$. Hence, the overall assignment distribution $\mathcal{Q}$ corresponding to the $N + M$ nodes is given by Theorem 2.7. The mean of the distribution $\mathcal{Q}$ is given by (2.35) as*

$$\gamma = \frac{P}{P + fP'}\mu + \frac{P'}{P + fP'}\mu' = \frac{1}{1 + f}(\mu + \mu'). \tag{2.36}$$

Since $\frac{1}{2} \leq \frac{1}{1+f} \leq 1$, the maximum value of γ is $\mu + \mu'$, and the minimum value of γ is $\frac{\mu+\mu'}{2}$. Hence, choosing $f = 0$ (as in Example 2.3) maximizes the connectivity of the resulting

network, as given by Theorem 2.1, but also maximizes the probability $p(m, x)$ approximated by Theorem 2.6. Since the original network parameters were specified to guarantee network connectivity, the maximum value of γ given by $f = 0$ is a secondary concern to the significant reduction in resilience to node capture. Hence, choosing $f \gg 0$ in this example can maintain the connectivity of the network without sacrificing resilience to node capture. Furthermore, the choice of $\mathcal{P}'$ with support of size at most $|\Lambda| = 11$ yields an overall support set of size at most $N + 11$. If $M > 11$, this choice of $\mathcal{P}'$ results in a significant improvement in the worst-case probability of sharing keys as given by Theorem 2.2 and the worst-case resilience to node capture given by Lemma 2.5 or Lemma 2.6 compared to the resulting assignment distribution in Example 2.3.

2.8 Summary of Contributions

We proposed a canonical key assignment model for key predistribution schemes in WSNs based on the probability that a key is shared by a given number of nodes and the algorithm for assignment of keys to nodes. We proposed a sampling framework for key assignment algorithms for use in the canonical model and a model for probabilistic connectivity of secure WSNs restricted by radio range and the existence of shared keys. We analyzed key predistribution schemes in the canonical model in terms of network connectivity and resilience to node capture, reflecting the worst-case probabilities for each metric. We demonstrated the design of new key predistribution schemes using the canonical model while paying particular attention to the worst-case seed-sharing probability and resilience to node capture. Finally, we presented an approach to analyze the effect of adding nodes to an existing secure WSN. This approach enables the design of assignment distributions that can tightly match the security requirements of a given application.

Chapter 3

MODELING NODE CAPTURE ATTACKS IN WIRELESS AD-HOC NETWORKS

Assurance of secure applications and services in wireless networks relies on the properties of *confidentiality* and *integrity*, respectively defined as the ability to keep data secret from unauthorized entities and the ability to verify that data has not been maliciously or accidentally altered [53]. Eschenauer and Gligor recently demonstrated in [29] that these properties can be efficiently compromised by physically capturing network nodes and extracting cryptographic keys from their memories. These *node capture attacks* are possible in most wireless networks due to the unattended operation of wireless nodes and the prohibitive cost of tamper-resistant hardware in portable devices [29]. Recent literature [16, 26, 29, 50] has focused on random node capture attacks. However, an intelligent adversary can improve the efficiency of a node capture attack using publicly available information learned by eavesdropping on insecure message exchanges throughout the network.

The recovery of cryptographic keys from node memories and the fact that keys tend to be re-used for efficient key management leads to an effective wire-tapping attack [78]. Such an attack can be used to compromise the security of single-hop wireless links. However, messages in a wireless network traverse multiple links and paths between a source and destination node, and a message may be compromised by traversing a single insecure link. Hence, the overall security of routed messages depends on the assignment of keys to nodes in the network, the wireless network routing protocol, the physical network topology, and the relative positions of the source and destination nodes in the network. Moreover, the fact that a message is transmitted over numerous links between a source and destination node implies that the overall confidentiality and integrity of the routed message may only be as secure as the least secure link, implying that vulnerabilities arise due to the topology of secure links in the wireless network. Hence, the impact of a node capture attack is a

function of both the cryptographic protocol which provides link security and the routing protocol which determines the set of links traversed by a given message.

In this chapter, we introduce a class of metrics to measure the effective security offered in a wireless network as a function of the routing topology and the link security provided by the key assignment protocol. This joint protocol analysis allows a network analyst or an adversary to evaluate the vulnerability of network traffic and isolate weakly secured connections. We approach the problem from an adversarial perspective and show how an intelligent adversary can mount a node capture attack using vulnerability evaluation to focus the attack on the nodes which contribute maximally to the compromise of network traffic. The necessary resource expenditure associated with the node capture attack implies that the optimal attack with minimum resource expenditure corresponds to a minimum cost set of nodes, in contrast to wiretapping attacks in routing or secure network coding [12, 41] which seek a minimum cost set of links. We demonstrate that joint consideration of the information from routing and key assignment protocols leads to a significant reduction in resource expenditure in comparison to consideration of information from either protocol separately.

3.1 Our Contributions

We aim to provide a formal characterization of node capture attacks. We define a collection of vulnerability metrics and formulate the minimum cost node capture attack problem as a nonlinear integer program using the defined vulnerability metrics. We further show that node capture attacks attempting to compromise secure links independent of the routing topology can be reduced to a linear integer program formulation. We present the GNAVE algorithm, a **G**reedy **N**ode capture **A**pproximation using **V**ulnerability **E**valuation, to approximate the minimum cost node capture attack using any of the vulnerability metrics of interest.

We present a collection of vulnerability metrics for differing attack strategies based only on key assignment and jointly on key assignment and routing. We demonstrate that, although certain information can be hidden from the adversary through the use of privacy-preserving protocols, statistical methods can be employed by the adversary to effectively

mitigate this attempt at attack defense. We provide a detailed simulation study to demonstrate and compare the impact of node capture attacks using the GNAVE algorithm with various strategies in wireless networks with examples of both classical routing and network coding protocols.

3.2 Models and Notation

In this section, we state the assumed wireless network, key assignment, and adversary models. We summarize the notation used throughout this chapter in Table 3.1.

3.2.1 Network Model

The topology of the wireless network with a set of nodes $\mathcal{N}$ is represented by the directed *network graph* $G = (\mathcal{N}, L)$. The link set L contains all ordered pairs of one-hop communicating neighbors, equivalent to an asymmetric relation [19], such that (i, j) is in L for $i \neq j$ if and only if node i can reliably send messages to node j without intermediate relay nodes. The link set L is dependent on parameters such as node location and configuration and properties of the radios, transmission medium, and MAC layer protocols.

We denote the subsets of $\mathcal{N}$ of message source and destination nodes in the network as $\mathcal{S}$ and $\mathcal{D}$, respectively. The set of source-destination pairs is denoted $\mathcal{T} \subseteq \mathcal{S} \times \mathcal{D}$ and is constructed based on the routing protocol decisions. For a given source-destination pair $(s, d) \in \mathcal{T}$, the routing protocol will construct one or more directed routing *paths* through G, where a path is defined as a set of sequential links in L. We define the *route* $\mathcal{R}_{sd}$ as the set of all paths traversed by any message from s to d, and we let f_π denote the fraction of traffic from s to d that traverses the given path $\pi \in \mathcal{R}_{sd}$. The route $\mathcal{R}_{sd}$ can be represented graphically by the *route subgraph* G_{sd} of G consisting of nodes and directed links traversed by at least one routing path $\pi \in \mathcal{R}_{sd}$ from s to d.

We define the following classes of routing protocols, partitioning the space of routing protocols based on the dependence of messages routed along different (not necessarily disjoint) paths, as follows.

Definition 3.1. *The class of* independent path routing *protocols consists of any protocol*

which uses one or more paths to route separate messages such that messages traversing different paths are independently coded and secured.

The class of independent path routing protocols contains, for example, protocols using a single, fixed path such as AODV [66] or DSR [44] as well as protocols using multiple paths such as GBR [68] or GEAR [81]. The route $\mathcal{R}_{sd}$ under independent path routing is equivalent to the superposition of $|\mathcal{R}_{sd}|$ single-path routes, where each single-path route $\{\pi\}$ for $\pi \in \mathcal{R}_{sd}$ is weighted by the corresponding traffic fraction f_π.

Definition 3.2. *The class of* dependent path routing *protocols consists of any protocol which uses multiple paths in which packets traversing separate paths are jointly coded, fragmented, or secured.*

The class of dependent path routing protocols contains, for example, protocols based on threshold secret sharing [69] and network coding [12, 38, 41] in which a set of coded packets must be jointly decoded in order to recover the original set of messages.

3.2.2 Key Assignment Model

We assume the existence of a secure key assignment mechanism as follows. Let $\mathcal{K}$ be a set of symmetric cryptographic keys and $\mathcal{L}$ be a corresponding set of publicly available key labels. Each node $i \in \mathcal{N}$ is assigned a subset $\mathcal{K}_i \subseteq \mathcal{K}$ and the corresponding subset $\mathcal{L}_i \subseteq \mathcal{L}$. We denote the subset of keys shared by nodes i and j as $\mathcal{K}_{ij} = \mathcal{K}_i \cap \mathcal{K}_j$ and allow communication between i and j if and only if $\mathcal{K}_{ij} \neq \varnothing$[1]. We assume that nodes i and j use the entire set $\mathcal{K}_{ij}$ of shared keys to secure the link (i, j), so the strength of the link security is directly related to the number of shared keys. We assume that nodes i and j compute the intersection $\mathcal{L}_{ij} = \mathcal{L}_i \cap \mathcal{L}_j$ in order to determine the set of shared keys $\mathcal{K}_{ij}$ using a protocol from one of the following classes.

Definition 3.3. *The class of* public label exchange *protocols consists of any protocol which provides necessary information for any node $j \in \mathcal{N}$ to compute the set $\mathcal{L}_i$ of key labels for any node $i \in \mathcal{N}$.*

[1]This requirement can be strengthened as in [16] to require $|\mathcal{K}_{ij}| \geq q$ for a fixed integer $q \geq 1$, though we do not explicitly address this requirement.

Table 3.1: We provide a summary of the notation used in Chapter 3 for the problem of modeling node capture attacks.

Symbol	Definition
$\mathcal{N}$	Set of N wireless nodes
L	Set of ordered pairs of one-hop neighbor nodes
G	Network graph $(\mathcal{N}, L)$
$\mathcal{K}, \mathcal{L}$	Set of keys, labels
$\mathcal{K}_i, \mathcal{L}_i$	Set of keys, labels assigned to node $i \in \mathcal{N}$
$\mathcal{K}_{ij}, \mathcal{L}_{ij}$	Set of keys, labels shared by nodes i and j
$\mathcal{S}, \mathcal{D}$	Set of source, destination nodes
$\mathcal{T}$	Subset of $\mathcal{S} \times \mathcal{D}$ of source-destination pairs
$\mathcal{R}_{sd}$	Set of paths forming the route from s to d
G_{sd}	Route subgraph of G corresponding to $\mathcal{R}_{sd}$
f_π	Fraction of $\mathcal{R}_{sd}$ traffic traversing π
$\mathcal{K}_{sd}^E$	Set of keys securing the end-to-end link (s, d)
$\mathcal{T}_A$	Adversary's target subset of $\mathcal{T}$
$\mathcal{C}$	Subset of $\mathcal{N}$ of captured nodes
$\mathcal{K}_\mathcal{C}, L_\mathcal{C}$	Set of compromised keys, links when $\mathcal{C}$ captured
w_i	Weight or cost of capturing node $i \in \mathcal{N}$
ρ_{sd}	Weight representing adversary's route preference
$V_{sd}(\mathcal{C})$	Route vulnerability of $\mathcal{R}_{sd}$ when $\mathcal{C}$ captured
$\nu_i(\mathcal{C})$	Incremental value of node i when $\mathcal{C}$ captured
$R_\mathcal{C}(i, j)$	Link resistance of (i, j) when $\mathcal{C}$ captured
$R_\mathcal{C}(\mathcal{R}_{sd})$	Route resistance of $\mathcal{R}_{sd}$ when $\mathcal{C}$ captured

The class of public label exchange protocols contains such protocols as the public broadcast of $\mathcal{L}_i$ by each node $i \in \mathcal{N}$ as in [29] or the use of a public identity-based function to compute $\mathcal{L}_i$ as a function of i as in [62].

Definition 3.4. *The class of* privacy-preserving set intersection *protocols consists of any protocol which provides necessary information for any node $j \in \mathcal{N}$ to only compute the set $\mathcal{L}_{ij}$ of keys labels shared with any node $i \in \mathcal{N}$ without giving any information to j about the remaining key labels in $\mathcal{L}_i \setminus \mathcal{L}_j$.*

The class of privacy-preserving set intersection protocols contains such protocols as the challenge-response protocol proposed in [29] in which each node $i \in \mathcal{N}$ computes a random nonce α and broadcasts α and the challenge $E_k(\alpha)$ for each $k \in \mathcal{K}_i$.

In addition to the link security provided by the set of shared keys $\mathcal{K}_{ij}$ for each link (i, j), we consider the incorporation of an additional *end-to-end* security mechanism for each route $\mathcal{R}_{sd}$ which depends only on the source s and destination d. If it is physically possible and allowed by policy, the source node s can compute the set $\mathcal{K}_{sd}$ of keys shared with the destination node d and additionally secure messages in the route $\mathcal{R}_{sd}$ using the shared keys $\mathcal{K}_{sd}$. We denote the set of keys securing the end-to-end connection between s and d as $\mathcal{K}_{sd}^E$, noting that $\mathcal{K}_{sd}^E = \mathcal{K}_{sd}$ if s and d are able and allowed to use end-to-end security and $\mathcal{K}_{sd}^E = \varnothing$ otherwise. We include the additional end-to-end secure link (s, d) in the route subgraph G_{sd} with the corresponding link security depending only on $\mathcal{K}_{sd}^E$.

3.2.3 Adversarial Model

We consider a polynomial-time adversary with the ability and resources to eavesdrop on and record messages throughout the network, capture nodes, and extract cryptographic keys from the memory of captured nodes. We assume that the adversary has knowledge of the key assignment and routing protocols, including protocol parameters, and can participate actively in any network protocols by assuming the roles of captured, replicated, or fabricated nodes. We further assume that the route subgraph G_{sd} for each $(s, d) \in \mathcal{T}$ is available to the adversary or is computable using traffic analysis and estimation [21].

The primary goal of the adversary is to compromise the confidentiality and integrity

of all messages routed between a *target set* of source-destination pairs denoted $\mathcal{T}_A \subseteq \mathcal{T}$ by extracting cryptographic keys from the memory of captured nodes $\mathcal{C} \subseteq \mathcal{N}$ with minimum resource expenditure. The adversary thus captures nodes intelligently by associating an individual weight or *cost* w_i with the resource expenditure required to capture each node $i \in \mathcal{N}$. We do not address further attacks on network protocols and services that can be performed as a result of message compromise.

3.3 Route Vulnerability Metrics under Node Capture Attacks

In this section, we define a class of route vulnerability metrics (RVM) to quantify the effective security of traffic traversing a given route $\mathcal{R}_{sd}$. Using the RVM definition, we formulate the minimum cost node capture attack problem as a nonlinear integer programming minimization problem. Since determining the optimal node capture attack is likely infeasible, we propose the GNAVE algorithm using a greedy heuristic to iteratively capture nodes which maximize the increase in route vulnerability.

3.3.1 Route Vulnerability Metric (RVM)

In order to evaluate the effect of a node capture attack on the effective security of traffic traversing a route $\mathcal{R}_{sd}$, we formally define link, path, and route compromise due to the capture of a subset $\mathcal{C} \subseteq \mathcal{N}$ of network nodes. We denote the set of keys recovered by the adversary in capturing the subset $\mathcal{C}$ as $\mathcal{K}_\mathcal{C} = \bigcup_{i \in \mathcal{N}} \mathcal{K}_i$. If a message traverses a link which is secured by keys in $\mathcal{K}_\mathcal{C}$, the security of the message is compromised. The compromise of individual links in the network, with respect to the network and routing models in Section 3.2, is defined as follows.

Definition 3.5. *The link $(i,j) \in L$ or $(s,d) \in \mathcal{T}$ is compromised if and only if $\mathcal{K}_{ij} \subseteq \mathcal{K}_\mathcal{C}$ or $\mathcal{K}_{sd}^E \subseteq \mathcal{K}_\mathcal{C}$, respectively, and the set of all compromised links is denoted $L_\mathcal{C} \subseteq L \cup \mathcal{T}$.*

Using Definition 3.5, we further define the compromise of paths and message routes as follows.

Definition 3.6. *The path $\pi \in \mathcal{R}_{sd}$ is compromised if and only if $(s,d) \in L_\mathcal{C}$ and there is at least one link (i,j) in π for which $(i,j) \in L_\mathcal{C}$.*

Note that the inclusion of the end-to-end link (s, d) in the requirement for path compromise indicates that any message traversing a compromised path can be eavesdropped or modified by the adversary.

Definition 3.7. *The route $\mathcal{R}_{sd}$ for $(s, d) \in \mathcal{T}$ is compromised if and only if every path $\pi \in \mathcal{R}_{sd}$ is compromised.*

Using Definition 3.7, an adversary can compute the fraction of target routes compromised due to the capture of a set of nodes $\mathcal{C}$. However, this evaluation does not provide the adversary with a method for selection of the set $\mathcal{C}$. Furthermore, the fraction of compromised target routes does not provide any indication of the contribution of nodes in $\mathcal{C}$ toward the future compromise of additional routes, as the compromise of a route is a binary event.

To adequately capture the progression toward the compromise of additional routes, we introduce the metric of route vulnerability $V_{sd}(\mathcal{C})$ as defined by the following RVM class.

Definition 3.8. *The* route vulnerability $V_{sd}(\mathcal{C})$ *of the route $\mathcal{R}_{sd}$ due to the capture of nodes in $\mathcal{C}$ is defined as any of the class of functions mapping into the unit interval $[0, 1]$ such that*

1. *$V_{sd}(\varnothing) = 0$, where $\varnothing$ is the empty set,*

2. *$V_{sd}(\mathcal{C}) = 1$ if and only if $\mathcal{R}_{sd}$ is compromised when $\mathcal{C}$ is captured, and*

3. *$0 < V_{sd}(\mathcal{C}) < 1$ if and only if $\mathcal{R}_{sd}$ is not compromised when $\mathcal{C}$ is captured but $\mathcal{C}$ contributes to the weakening of the security of at least one link in the route $\mathcal{R}_{sd}$.*

The class of RVMs thus relaxes the binary notion of route compromise to a continuous measure of the progress of the attack and allows for comparison of partial compromise by different sets $\mathcal{C}_1$ and $\mathcal{C}_2$ of captured nodes.

3.3.2 Node Capture Attack Formulation

For any RVM realization satisfying the conditions of Definition 3.8, we devise a node capture strategy that maximizes the progression toward the goal of compromising all routes $\mathcal{R}_{sd}$ for

Problem: Minimum Cost Node Capture Attack	
Given:	$\mathcal{L}_i, w_i$ for $i \in \mathcal{N}$, $\mathcal{R}_{sd}$ for $(s,d) \in \mathcal{T}_A$
Find:	$\mathcal{C} \subseteq \mathcal{N}$
such that	$\sum_{i \in \mathcal{C}} w_i$ is minimized
and	$V_{sd}(\mathcal{C}) = 1$ for all $(s,d) \in \mathcal{T}_A$.

Figure 3.1: The minimum cost node capture attack is formulated as a constrained optimization problem.

$(s,d) \in \mathcal{T}_A$. The choice of subset $\mathcal{C}$ requiring the minimum resource expenditure is thus given by the minimum cost node capture problem in Figure 3.1.

In general, based on Definition 3.6 of path compromise, the metric $V_{sd}(\mathcal{C})$ is nonlinear in the entries of $\mathcal{C}$. Hence, the minimum cost node capture attack above is a nonlinear integer programming minimization problem, known to be NP-hard [19, 24]. We thus propose the use of a greedy heuristic that iteratively adds nodes to $\mathcal{C}$ based on maximizing the increase in route vulnerability $V_{sd}(\mathcal{C})$ at each step. The heuristic is thus similar to a known greedy heuristic for set covering [18] and linear integer programming [24]. However, due to the nonlinearity in $V_{sd}(\mathcal{C})$, the worst-case performance of the greedy heuristic cannot be analyzed using the ratio-bound analysis in [18, 19, 24] and is left as an open problem.

To maximize the route vulnerability $V_{sd}(\mathcal{C})$ with minimum resource expenditure, it is beneficial to the adversary to attempt to maximize the vulnerability resulting from the capture of each individual node using the information recovered from previously captured nodes. The contribution of a node i is thus given by the increase in route vulnerability $V_{sd}(\mathcal{C} \cup \{i\}) - V_{sd}(\mathcal{C})$ due to the addition of i to $\mathcal{C}$. Allowing for an additional weight ρ_{sd} to indicate the adversary's preference to compromise the route $\mathcal{R}_{sd}$ over other routes, the value of each node i is defined as follows.

Definition 3.9. *The individual* incremental node value *of adding node $i \in \mathcal{N}$ to $\mathcal{C}$ is defined as*

$$\nu_i(\mathcal{C}) = \sum_{(s,d) \in \mathcal{T}_A} \rho_{sd} \left(V_{sd}(\mathcal{C} \cup \{i\}) - V_{sd}(\mathcal{C}) \right)$$

GNAVE Algorithm

Given: $\mathcal{L}_i, w_i$ for $i \in \mathcal{N}$, $\mathcal{R}_{sd}$ for $(s,d) \in \mathcal{T}_A$

$\mathcal{C} \leftarrow \varnothing$

while there exists $(s,d) \in \mathcal{T}_A$ with $V_{sd}(\mathcal{C}) < 1$ **do**

$i^* \leftarrow \arg\max_{i \in \mathcal{N}} \nu_i(\mathcal{C})/w_i$

$\mathcal{C} \leftarrow \mathcal{C} \cup \{i^*\}$

end while

Figure 3.2: We present the algorithmic form of the GNAVE algorithm to approximate the minimum cost node capture attack in Figure 3.1.

for any route vulnerability function $V_{sd}(\mathcal{C})$ satisfying the conditions in Definition 3.8.

To maximize the cost-effectiveness of the node capture attack at each iteration, the adversary chooses to capture the node with maximum incremental value per unit cost $\nu_i(\mathcal{C})/w_i$. Based on this greedy approach, we propose the GNAVE algorithm for **G**reedy **N**ode capture **A**pproximation using **V**ulnerability **E**valuation as given in Figure 3.2.

We note that the GNAVE algorithm being greedy implies that the attack performance depends only on the order of the weighted node values $\nu_i(\mathcal{C})/w_i$ for the nodes $\mathcal{N} \setminus \mathcal{C}$. In order to illustrate the effect of node capture attacks using the GNAVE algorithm, we next provide candidate realizations of the RVM $V_{sd}(\mathcal{C})$.

3.4 RVM Realization

In this section, we propose an RVM realization satisfying the conditions in Definition 3.8, noting that there is a high degree of freedom in the given conditions. We present an RVM realization for each of the routing protocol classes discussed in Section 3.2.1, hereafter denoting the route vulnerability for independent and dependent path routing protocols as $V^I_{sd}(\mathcal{C})$ and $V^D_{sd}(\mathcal{C})$, respectively. The definitions presented in this section are derived using the following necessary and sufficient condition for the compromise of a route $\mathcal{R}_{sd}$ with respect to the edge cuts [19] of the route subgraph G_{sd}.

Theorem 3.1. *The route $\mathcal{R}_{sd}$ is compromised if and only if the set $L_{\mathcal{C}}$ of compromised links contains at least one (s,d) edge cut of the route subgraph G_{sd} as a subset.*

Proof. To prove the forward implication, suppose that $\mathcal{R}_{sd}$ is compromised. By Definitions 3.6 and 3.7, there is at least one compromised link (i_π, j_π) in each path $\pi \in \mathcal{R}_{sd}$ and the end-to-end link (s,d) is compromised. Let $L_{\text{cut}} = \{(i_\pi, j_\pi) : \pi \in \mathcal{R}_{sd}\} \subseteq L_{\mathcal{C}}$. Since each path π traverses at least one edge in L_{cut}, $L_{\text{cut}} \cup \{(s,d)\}$ is an edge cut of G_{sd}

To prove the reverse implication, let L_{cut} be an edge cut of G_{sd}. By the definition of an edge cut, $(s,d) \in L_{\text{cut}}$ and each path π from s to d in G_{sd} traverses at least one link in L_{cut}. Hence, by Definition 3.6, every path π in $\mathcal{R}_{sd}$ is compromised, implying by Definition 3.7 that the route itself is compromised. □

Theorem 3.1 thus implies that the task of compromising each route $(s,d) \in \mathcal{T}_A$ is equivalent to capturing a set of nodes $\mathcal{C}$ leading to the compromise of an edge cut of G_{sd}. We thus formulate an RVM realization using the properties of edge cuts of G_{sd}.

3.4.1 Set Theoretic RVM

We formulate a set theoretic RVM realization $V_{sd}(\mathcal{C})_{\text{SET}}$ by interpreting the properties of edge cuts of G_{sd} set theoretically. From Theorem 3.1, the existence of a compromised edge cut set $L_{\text{cut}} \subseteq L_{\mathcal{C}}$ of the route subgraph G_{sd} implies that the route $\mathcal{R}_{sd}$ is compromised. In terms of the set $\mathcal{K}_{\mathcal{C}}$ of compromised keys, a necessary and sufficient condition for $L_{\mathcal{C}}$ to contain an edge cut set of G_{sd} is

$$\mathcal{K}_{sd} \subseteq \mathcal{K}_{\mathcal{C}} \text{ and } \forall \pi \in \mathcal{R}_{sd}, \exists (i,j) \in \pi, \mathcal{K}_{ij} \subseteq \mathcal{K}_{\mathcal{C}}.$$

Letting $\mathbf{1}(\cdot)$ denote the binary indicator function of a specified event, Theorem 3.1 thus implies that the first two conditions of Definition 3.8 can be satisfied by defining a binary RVM equal to

$$\mathbf{1}\left(\mathcal{K}_{sd} \subseteq \mathcal{K}_{\mathcal{C}}\right) \prod_{\pi \in \mathcal{R}_{sd}} \left(1 - \prod_{(i,j)\in\pi} \left(1 - \mathbf{1}\left(\mathcal{K}_{ij} \subseteq \mathcal{K}_{\mathcal{C}}\right)\right)\right). \tag{3.1}$$

However, this function does not satisfy the third condition of Definition 3.8 as the resulting function does not take continuous values between 0 and 1.

The above formulation provides insight into the route vulnerability, however, suggesting that a valid RVM can be obtained with minor modifications. First, to ensure that any compromised path is accounted for in the vulnerability evaluation, the product over all paths in $\mathcal{R}_{sd}$ can be replaced by a weighted summation over the corresponding paths, including the secure end-to-end link (s, d) as a single-hop path. We denote the relative weight assigned to the secure end-to-end link (s, d) as f_{sd} with the assumption that $f_{sd} > 0$ is allowed to vary arbitrarily when the additional end-to-end secure link is used and that $f_{sd} = 0$ otherwise, thus impacting the choice of captured nodes. We relax the binary condition imposed by the indicator function $\mathbf{1}(\mathcal{K}_{ij} \subseteq \mathcal{K}_{\mathcal{C}})$ by the function $\phi_{ij}(\mathcal{C})$ equal to the fraction of keys in $\mathcal{K}_{ij}$ that are contained in $\mathcal{K}_{\mathcal{C}}$, given by

$$\phi_{ij}(\mathcal{C}) = \begin{cases} \dfrac{|\mathcal{K}_{ij} \cap \mathcal{K}_{\mathcal{C}}|}{|\mathcal{K}_{ij}|}, & \text{if } \mathcal{K}_{ij} \neq \varnothing \\ 1, & \text{otherwise} \end{cases} \tag{3.2}$$

for links in L and

$$\phi_{sd}(\mathcal{C}) = \begin{cases} \dfrac{|\mathcal{K}^E_{sd} \cap \mathcal{K}_{\mathcal{C}}|}{|\mathcal{K}^E_{sd}|}, & \text{if } \mathcal{K}^E_{sd} \neq \varnothing \\ 1, & \text{otherwise} \end{cases} \tag{3.3}$$

for the secure end-to-end link (s, d). Applying this relaxation to the right-hand side of (3.1) thus yields the following RVMs for independent and dependent path routing protocols, which vary only in the weighting of individual paths in $\mathcal{R}_{sd}$.

For independent path routing protocols, the compromise of an individual path $\pi \in \mathcal{R}_{sd}$ is sufficient to allow the adversary to recover a fraction f_π of the traffic from s to d. Applying the continuous relaxation to the right-hand side of (3.1) for each single path route in $\mathcal{R}_{sd}$ and summing over the single path routes with corresponding weights f_π, including the end-to-end link (s, d) with weight f_{sd}, yields the RVM for independent path routing protocols as

$$V^I_{sd}(\mathcal{C})_{\text{SET}} = \frac{f_{sd}\phi_{sd}(\mathcal{C}) + 1}{1 + f_{sd}} - \sum_{\pi \in \mathcal{R}_{sd}} \frac{f_\pi}{1 + f_{sd}} \prod_{(i,j)\in\pi} (1 - \phi_{ij}(\mathcal{C})) \,. \tag{3.4}$$

For dependent path routing protocols, even though the compromise of an individual path does not reveal any information to the adversary, it brings the adversary closer to

compromising the route. Hence, we obtain the corresponding RVM by applying the continuous relaxation to the right-hand side of (3.1) and summing over the equally-weighted single path routes, including the end-to-end link (s, d) with weight f_{sd}, yielding

$$V_{sd}^{D}(\mathcal{C})_{\text{SET}} = \frac{f_{sd}\phi_{sd}(\mathcal{C}) + 1}{1 + f_{sd}} - \frac{1}{|\mathcal{R}_{sd}|(1 + f_{sd})} \sum_{\pi \in \mathcal{R}_{sd}} \prod_{(i,j)\in\pi} (1 - \phi_{ij}(\mathcal{C})) \,. \tag{3.5}$$

The set theoretic formulation of the RVM $V_{sd}(\mathcal{C})_{\text{SET}}$ in this section is derived by explicitly analyzing the necessary condition for the existence of an edge cut of G_{sd}.

3.5 Node Capture Attacks without Routing Information

In this section, we present a special case of node capture attacks and vulnerability metrics when the adversary has not collected (or is unable to collect) information about the routing topology in the network. We show that node capture attacks based only on the key assignment protocol can be modeled using an integer programming framework and give example strategies and the corresponding vulnerability metrics.

3.5.1 Formulation of the Key-Based Node Capture Attack Model

In the absence of routing information, the adversary is unable to determine whether paths or routes are compromised as in Definitions 3.6 and 3.7. Hence, the attack is re-formulated with respect to the compromise of secure links as in Definition 3.5. Furthermore, the route vulnerability metric $V_{sd}(\mathcal{C})$ must be replaced by an alternative vulnerability metric that evaluates the effect of the attack on the security of links instead of routes.

For a key-based node capture attack, we let $\mathcal{Z} = \{z_1, \ldots, z_M\}$ denote the collection of M items of interest to the adversary. For example each z_m can be a key $z_m \in \mathcal{K}$ or a subset of shared keys $z_m = \mathcal{K}_j \cap \mathcal{K}_j \subseteq \mathcal{K}$. In order to plan the attack, the adversary must characterize the relationship between each set $\mathcal{K}_n$ of assigned keys and each target item $z_m \in \mathcal{Z}$. The relationship can be characterized by defining a variable $a_{m,n}$ which is non-zero if and only if the assigned keys $\mathcal{K}_n$ aid in the recovery of the target item z_m. The variables $a_{m,n}$ can be collected into an $M \times N$ constraint matrix $\boldsymbol{A}$ representing the goals of the attack. Letting $\boldsymbol{x}$ be a binary vector of elements x_n where $x_n = 1$ if and only if $n \in \mathcal{C}$,

the product $\boldsymbol{Ax}$ denotes the outcome of the attack. Depending on the key assignment and link-key establishment protocols, the adversary may be required to obtain a certain amount of information about an element z_m before it can be recovered. Hence, we let s_m denote the corresponding quantity such that z_m is recovered if and only if

$$\sum_{n \in \mathcal{N}} a_{m,n} x_n \geq s_m. \tag{3.6}$$

The sufficiency of the attack is thus given by the condition that $\boldsymbol{Ax} \geq \boldsymbol{s}$. We note that the adversary's preference for individual items z_m in $\mathcal{Z}$ can be incorporated by scaling the m^{th} row of $\boldsymbol{A}$ and the value s_m by a constant, with no effect on the inequality in (3.6).

The vulnerability metric $V_{\mathcal{Z}}(\mathcal{C})$ of interest to the adversary is thus a function of the matrix $\boldsymbol{A}$ and the sufficiency vector $\boldsymbol{s}$. Similar to the conditions in Definition 3.8, the vulnerability $V_{\mathcal{Z}}(\mathcal{C})$ will be 1 only if (3.6) is satisfied for all $m = 1, \ldots, M$. We define a candidate vulnerability function as

$$V_{\mathcal{Z}}(\mathcal{C}) = \frac{\| \min(\boldsymbol{Ax}, \boldsymbol{s}) \|_1}{\| \boldsymbol{s} \|_1}, \tag{3.7}$$

where min is the element-wise vector minimum of the two vector arguments and $\| \cdot \|_1$ denote the ℓ_1 (absolute vector sum) norm [39]. Replacing the final condition in the minimum cost attack formulation in Section 3.3.2 with the metric in (3.7) yields the constraint $\boldsymbol{Ax} \geq \boldsymbol{s}$, making the formulation that of a minimum cost integer programming problem [24]. We note that the average $V(x)$ over all sets $\mathcal{C}$ of size x is equivalent to the widely-used measure of the resilience of the key predistribution scheme to a node capture attack [16,25,26,29,45,49,50].

Due to the NP-hardness of the integer programming minimization problem, the GNAVE algorithm presented in Section 3.3.2 can be applied in a similar way to the case without routing information. We note that the greedy algorithm does not require the adversary to explicitly construct the constraint matrix $\boldsymbol{A}$ but only to evaluate the vulnerability function $V_{\mathcal{Z}}(\mathcal{C})$, a task which requires far less information, as will be discussed.

3.5.2 Key-Based Node Capture Attacks

We next present two example node capture attacks based only on the key assignment information using the formulation in Section 3.5.1. We present the *key cover attack* and

the *link cover attack* which are named for their relationships to the well-known set cover problem [19, 24].

Key Cover Attack

The key cover attack is modeled according to the well known set cover problem. In this attack, the collection $\mathcal{Z}$ of items sought by the adversary is equal to the set of keys $\mathcal{K}$. The adversary's primary goal is to capture a set of nodes whose sets $\mathcal{K}_n$ *cover* the set $\mathcal{K}$ and thus can be used to compromise the security of every secure link in the network. In this attack, each element $k \in \mathcal{K}$ is of equal importance to the adversary, so the rows of $\boldsymbol{A}$ and elements of $\boldsymbol{s}$ are equally weighted[2].

A key coverage attack can be formulated using the minimization problem in Section 3.5.1 as follows. Each entry s_m of the vector $\boldsymbol{s}$ is equal to the number of elements derived from k_m which must be obtained to recover the secret k_m. For example, the value s_m can be equal to the threshold of a secret-sharing scheme [9, 10, 25, 26, 49, 50, 69] applied to the elements of $\mathcal{K}$. Each entry $a_{m,n}$ of the binary matrix $\boldsymbol{A}$ is equal to 1 if and only if an element in $\mathcal{K}_n$ was derived from $k_m \in \mathcal{Y}$. Hence, the column sum A_n of the matrix $\boldsymbol{A}$ is equal to the number of elements in $\mathcal{K}_n$ which are unknown to the adversary. To perform a key cover attack using the GNAVE algorithm, the key establishment protocol must allow the adversary to compute the set $\mathcal{L}_n$ for each node $n \in \mathcal{N}$. We first present a result on the adversary's ability to perform the key cover attack using the GNAVE algorithm.

Lemma 3.1. *Given any key establishment protocol such that $|\mathcal{K}_n|$ is computable by the adversary for each $n \in \mathcal{N}$, a key cover attack can be performed deterministically using the GNAVE algorithm.*

Proof. Let L denote the set of indices of elements in $\mathcal{K}$ recovered by the adversary from previously captured nodes. Since the adversary has obtained all of the information stored within each captured node, the intersection set $L \cap \mathcal{L}_n$ is necessarily computable for each $n \in \mathcal{N}$, as the adversary can simply play the role of each captured node in the key establishment protocol. The GNAVE algorithm can then be performed by realizing that the sum of the

[2] A simplified version of this strategy was used to develop a probabilistic attack in [40].

n^{th} column of $\boldsymbol{A}$ is equal to $|\mathcal{K}_n| - |L \cap \mathcal{L}_n|$. Note that the result does not require the number of assigned keys $|\mathcal{K}_n|$ to be fixed for all $n \in \mathcal{N}$. □

The primary implication of Lemma 3.1 is that the use of a privacy-preserving key establishment protocol based on a cryptographic proof-of-knowledge [53] does not prevent the adversary from performing key cover attacks.

Link Cover Attack

The link cover attack is also modeled according to the well known set cover problem. In this attack, each element in the collection $\mathcal{Z}$ of items sought by the adversary is a subset $z_{(i,j)}$ of $\mathcal{K}$ equal to the intersection $\mathcal{K}_i \cap \mathcal{K}_j$. Since the same elements of $\mathcal{K}$ can be used by multiple pairs of nodes in the network, $\mathcal{Z}$ is a multi-set of subsets of $\mathcal{K}$ whose union is not necessarily all of $\mathcal{K}$. In this attack, the adversary's primary goal is to capture a set of nodes whose sets $\mathcal{K}_n$ *cover* the collection of multi-sets $\mathcal{Z}$, corresponding to the compromise of as many secure links in the network as is possible.

A link cover attack can be formulated using the minimization problem in Section 3.3.2 and the GNAVE algorithm as follows. Similar to that of the set coverage strategy, each entry $s_{(i,j)}$ of the vector $\boldsymbol{s}$ is equal to the threshold number of elements derived from $z_{(i,j)}$ which must be obtained to recover the set $z_{(i,j)}$. Each entry $a_{(i,j),n}$ of the binary matrix $\boldsymbol{A}$ is equal to 1 if and only if $\mathcal{K}_i \cap \mathcal{K}_j \subseteq \mathcal{K}_n$. Furthermore, to perform a link cover attack using the GNAVE algorithm, the key establishment protocol must allow the adversary to compute the label set $\mathcal{L}_n$.

If the adversary cannot compute the label set $\mathcal{L}_n$ for each node $n \in \mathcal{N}$, it is impossible to determine the subsets $z_{(i,j)}$ of $\mathcal{K}$ corresponding to each secure link. Furthermore, there is no method for computing or updating the column sums of the matrix $\boldsymbol{A}$. Hence, subset coverage attacks can be prevented by the use of a privacy-preserving key establishment protocol.

3.6 Uncertainty in RVM Parameters due to Privacy-Preserving Set Intersection

In order for an adversary to mount a node capture attack using the GNAVE algorithm, the contribution $V_{sd}(\mathcal{C} \cup \{n\}) - V_{sd}(\mathcal{C})$ to the incremental node value $\nu_n(\mathcal{C})$ of node $n \in \mathcal{N}$ must be computed using Definition 3.9 with an RVM realization that satisfies Definition 3.8. The set theoretic RVM realization in Section 3.4 as well as the link cover attack in Section 3.5 require the adversary to compute the quantities $|\mathcal{K}_{ij}|$ and $|\mathcal{K}_{ij} \cap \mathcal{K}_{\mathcal{C}}|$ for each link (i, j). As shown in Lemma 3.1, the set $\mathcal{K}_i \cap \mathcal{K}_{\mathcal{C}}$ can be computed for any $i \in \mathcal{N}$ by the adversary with captured nodes $\mathcal{C}$. Hence, the quantity $|\mathcal{K}_{ij} \cap \mathcal{K}_{\mathcal{C}}|$ can always be computed using the equality

$$\mathcal{K}_{ij} \cap \mathcal{K}_{\mathcal{C}} = (\mathcal{K}_i \cap \mathcal{K}_{\mathcal{C}}) \cap (\mathcal{K}_j \cap \mathcal{K}_{\mathcal{C}}).$$

However, if the network nodes in $\mathcal{N}$ are using a privacy-preserving set intersection protocol according to Definition 3.4, the quantity $|\mathcal{K}_{ij}|$ cannot be computed deterministically. We thus demonstrate how this required quantity can be estimated probabilistically. In what follows, we assume that each set $\mathcal{K}_i$ is randomly and independently selected from $\mathcal{K}$ as in [29] and that the quantities $k_i = |\mathcal{K}_i|$ and $k = |\mathcal{K}|$ are publicly known.

A probabilistic estimate $\hat{k}_{ij}$ of the quantity $|\mathcal{K}_{ij}|$ can be computed using the probability distribution $\Pr[|\mathcal{K}_{ij}| = k_{ij}]$ using the known parameters $k_{i\mathcal{C}} = |\mathcal{K}_i \cap \mathcal{K}_{\mathcal{C}}|$, $k_{j\mathcal{C}} = |\mathcal{K}_j \cap \mathcal{K}_{\mathcal{C}}|$, and $k_{ij\mathcal{C}} = |\mathcal{K}_{ij} \cap \mathcal{K}_{\mathcal{C}}|$. This probability is exactly the probability that $(k_{ij} - k_{ij\mathcal{C}})$ of the $(k_i - k_{i\mathcal{C}})$ keys in $\mathcal{K}_i$ not known to the adversary are equal to $(k_{ij} - k_{ij\mathcal{C}})$ of the $(k_j - k_{j\mathcal{C}})$ keys in $\mathcal{K}_j$ not known to the adversary. Letting $k_{\mathcal{C}} = |\mathcal{K}_{\mathcal{C}}|$, the desired probability can be computed as

$$\Pr[|\mathcal{K}_{ij}| = k_{ij}] = \binom{k_j - k_{j\mathcal{C}}}{k_{ij} - k_{ij\mathcal{C}}} \left(\frac{k_i - k_{i\mathcal{C}}}{k - k_{\mathcal{C}}}\right)^{k_{ij} - k_{ij\mathcal{C}}} \left(1 - \frac{k_i - k_{i\mathcal{C}}}{k - k_{\mathcal{C}}}\right)^{k_j - k_{j\mathcal{C}} - k_{ij} + k_{ij\mathcal{C}}} \tag{3.8}$$

for $k_{ij} = k_{ij\mathcal{C}}, \ldots, k_j - k_{j\mathcal{C}} + k_{ij\mathcal{C}}$.

We compute the estimate $\hat{k}_{ij}$ as the expected value of $|\mathcal{K}_{ij}|$, conditioned on the fact that $|\mathcal{K}_{ij}| > k_{ij\mathcal{C}}$ since $|\mathcal{K}_{ij}|$ is only unknown if $k_j > k_{j\mathcal{C}}$. The estimate $\hat{k}_{ij}$ is thus computed as the expected value of the random variable with probability distribution $\Pr[|\mathcal{K}_{ij}| =$

$k_{ij}]/\Pr[|\mathcal{K}_{ij}| > k_{ij\mathcal{C}}]$, subject to $|\mathcal{K}_{ij}| > k_{ij\mathcal{C}}$, using (3.8), yielding

$$\hat{k}_{ij} = k_{ij\mathcal{C}} + \frac{(k_i - k_{i\mathcal{C}})(k_j - k_{j\mathcal{C}})}{(k - k_{\mathcal{C}})\left(1 - \left(1 - \frac{k_i - k_{i\mathcal{C}}}{k - k_{\mathcal{C}}}\right)^{k_j - k_{j\mathcal{C}}}\right)}. \tag{3.9}$$

The estimate $\hat{k}_{ij}$ of $|\mathcal{K}_{ij}|$ using (3.9) can then be used to estimate the route vulnerability $V_{sd}(\mathcal{C})$. However, in order to estimate the incremental node value $\nu_n(\mathcal{C})$ of each node $n \in \mathcal{N} \setminus \mathcal{C}$, the route vulnerability $V_{sd}(\mathcal{C} \cup \{n\})$ must also be estimated, where the union $\mathcal{K}_{\mathcal{C}\cup\{n\}}$ cannot be computed deterministically.

We note that for any $i, j, n \in \mathcal{N}$, the number of keys securing the link (i, j) if n is added to $\mathcal{C}$ is given by

$$|\mathcal{K}_{ij} \setminus \mathcal{K}_{\mathcal{C}\cup\{n\}}| = |\mathcal{K}_{ij}| - |\mathcal{K}_{ij} \cap (\mathcal{K}_{\mathcal{C}} \cup \mathcal{K}_n)| = |\mathcal{K}_{ij}| - |\mathcal{K}_{ij} \cap \mathcal{K}_{\mathcal{C}}| - |\mathcal{K}_{ij} \cap \mathcal{K}_n| + |\mathcal{K}_{ij} \cap \mathcal{K}_n \cap \mathcal{K}_{\mathcal{C}}|. \tag{3.10}$$

Since the quantities $k_{ij\mathcal{C}} = |\mathcal{K}_{ij} \cap \mathcal{K}_{\mathcal{C}}|$ and $k_{ijn\mathcal{C}} = |\mathcal{K}_{ij} \cap \mathcal{K}_n \cap \mathcal{K}_{\mathcal{C}}|$ are known, and an estimate $\hat{k}_{ij}$ of $|\mathcal{K}_{ij}|$ has already been computed in (3.9), the remaining quantity to estimate is $|\mathcal{K}_{ij} \cap \mathcal{K}_n|$. We let $Q(k_{ijn})$ denote the probability $\Pr[|\mathcal{K}_{ij} \cap \mathcal{K}_n| = k_{ijn}]$ and $Q(k_{ijn}|k_{ij})$ denote the similar probability conditioned on the event that $|\mathcal{K}_{ij}| = k_{ij}$, computed as

$$Q(k_{ijn}) = \sum_{k_{ij}=k_{ij\mathcal{C}}}^{k_j - k_{j\mathcal{C}} + k_{ij\mathcal{C}}} Q(k_{ijn}|k_{ij}) \Pr[|\mathcal{K}_{ij}| = k_{ij}]. \tag{3.11}$$

Similar to the computation in (3.8), the conditional probability $Q(k_{ijn}|k_{ij})$ is computed as

$$Q(k_{ijn}|k_{ij}) = \binom{k_n - k_{n\mathcal{C}}}{k_{ijn}} \left(\frac{k_{ij} - k_{ij\mathcal{C}}}{k - k_{\mathcal{C}}}\right)^{k_{ijn}} \left(1 - \frac{k_{ij} - k_{ij\mathcal{C}}}{k - k_{\mathcal{C}}}\right)^{k_n - k_{n\mathcal{C}} - k_{ijn}}. \tag{3.12}$$

An estimate $\hat{k}_{ijn}$ of $|\mathcal{K}_{ij} \cap \mathcal{K}_n|$ is computed conditioned on the event that $|\mathcal{K}_{ij}| > k_{ij\mathcal{C}}$ as before. The estimate $\hat{k}_{ijn}$ is thus computed as the expected value of the random variable with probability distribution $Q(k_{ijn})/\Pr[|\mathcal{K}_{ij}| > k_{ij\mathcal{C}}]$, subject to $k_{ij} > k_{ij\mathcal{C}}$, using (3.8),

(3.11), and (3.12) yielding

$$\begin{aligned}\hat{k}_{ijn} &= \sum_{k_{ijn}=0}^{k_n-k_{n\mathcal{C}}} \frac{k_{ijn} \sum_{k_{ij}=k_{ij\mathcal{C}}+1}^{k_j-k_{j\mathcal{C}}+k_{ij\mathcal{C}}} Q(k_{ijn}|k_{ij}) \Pr[|\mathcal{K}_{ij}| = k_{ij}]}{\Pr[|\mathcal{K}_{ij}| > k_{ij\mathcal{C}}]} \\ &= \sum_{k_{ij}=k_{ij\mathcal{C}}+1}^{k_j-k_{j\mathcal{C}}+k_{ij\mathcal{C}}} \frac{\Pr[|\mathcal{K}_{ij}| = k_{ij}]}{\Pr[|\mathcal{K}_{ij}| > k_{ij\mathcal{C}}]} \sum_{k_{ijn}=0}^{k_n-k_{n\mathcal{C}}} k_{ijn} Q(k_{ijn}|k_{ij}) \\ &= \frac{k_n - k_{n\mathcal{C}}}{k - k_{\mathcal{C}}} \sum_{k_{ij}=k_{ij\mathcal{C}}+1}^{k_j-k_{j\mathcal{C}}+k_{ij\mathcal{C}}} \frac{\Pr[|\mathcal{K}_{ij}| = k_{ij}](k_{ij} - k_{ij\mathcal{C}})}{\Pr[|\mathcal{K}_{ij}| > k_{ij\mathcal{C}}]} = \frac{(k_n - k_{n\mathcal{C}})(\hat{k}_{ij} - k_{ij\mathcal{C}})}{k - k_{\mathcal{C}}}, \end{aligned} \tag{3.13}$$

where $\hat{k}_{ij}$ is the estimate given in (3.9).

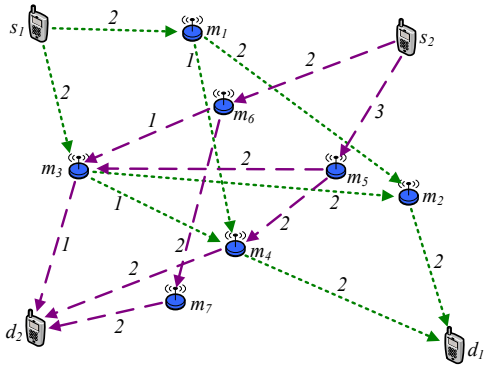
The estimates $\hat{k}_{ij}$ and $\hat{k}_{ijn}$ are then used to estimate the incremental node value $\nu_n(\mathcal{C})$ of each node $n \in \mathcal{N}\backslash\mathcal{C}$ using Definition 3.9 with the corresponding route vulnerability definition in Section 3.4. We note that the contribution of a node toward the compromise of a link, path, or route is deterministic if the captured node is incident to the link, path, or route of interest. Hence, at early stages of the attack, it is likely that captured nodes will be located along paths from source nodes to destination nodes. The adversary will, however, learn significantly more information about the remainder of the network by capturing one node at a time using the GNAVE algorithm with the vulnerability estimates obtained herein.

3.7 Examples and Simulation Study

In this section, we illustrate the application of the route vulnerability metric $V_{sd}(\mathcal{C})$ and the GNAVE algorithm. We first present two small-scale examples using independent and dependent path routing and the set theoretic route vulnerability metric. We then provide simulation results to illustrate the effect of node capture attacks in a large-scale wireless network under various different key assignment and routing models.

3.7.1 Example: Multi-Path Geographic Forwarding

We illustrate a node capture attack using the GNAVE algorithm with the set theoretic route vulnerability metric presented in Section 3.4.1 for a wireless network using multi-path geographic forwarding. In this example, we construct independent path routes using a



(a)

$\mathcal{K}_{s_1} = \{k_2, k_7, k_8, k_{10}\}$
$\mathcal{K}_{s_2} = \{k_3, k_4, k_6, k_{11}\}$
$\mathcal{K}_{d_1} = \{k_1, k_2, k_{10}, k_{11}\}$
$\mathcal{K}_{d_2} = \{k_3, k_8, k_9, k_{11}\}$
$\mathcal{K}_{m_1} = \{k_8, k_9, k_{10}, k_{12}\}$
$\mathcal{K}_{m_2} = \{k_2, k_6, k_9, k_{10}\}$
$\mathcal{K}_{m_3} = \{k_2, k_3, k_6, k_7\}$
$\mathcal{K}_{m_4} = \{k_3, k_5, k_{10}, k_{11}\}$
$\mathcal{K}_{m_5} = \{k_3, k_4, k_5, k_6\}$
$\mathcal{K}_{m_6} = \{k_1, k_6, k_{11}, k_{12}\}$
$\mathcal{K}_{m_7} = \{k_1, k_7, k_8, k_{11}\}$

(b)

Figure 3.3: Sources s_1 and s_2 send messages to destinations d_1 and d_2, respectively, using independent path routing. Each link (i, j) is labeled with the number of shared keys $|\mathcal{K}_{ij}|$. The end-to-end secure links, not illustrated, have $|\mathcal{K}^E_{s_1 d_1}| = |\mathcal{K}^E_{s_2 d_2}| = 2$ shared keys each. The example network is illustrated in (a), and the assigned keys are shown in (b).

multi-path geographic forwarding algorithm in which each node forwards the corresponding message to the two next-hop neighbors nearest the destination node, similar to the idea in GBR [68]. We consider the network topology given in Figure 3.3(a) with source-destination routing pairs $\mathcal{T} = \{(s_1, d_1), (s_2, d_2)\}$. The additional end-to-end security mechanism discussed in Section 3.2.2 is used by each source-destination pair, and keys are assigned to nodes in the network as given in Figure 3.3(b).

To illustrate the security of each link using the assigned keys above, we note that nodes s_1 and m_1 share keys $\mathcal{K}_{s_1 m_1} = \{k_8, k_{10}\}$, so the link (s_1, m_1) is secure as long as $\{k_8, k_{10}\} \nsubseteq \mathcal{K}_{\mathcal{C}}$.

Assuming the messages traversing different paths through the network are independently secured, the route vulnerability of the two routes $\mathcal{R}_{s_1 d_1}$ and $\mathcal{R}_{s_2 d_2}$ can be computed using (3.4) by individually considering the four paths and the end-to-end secure link in each route. The route vulnerability $V^I_{sd}(\mathcal{C})_{\text{SET}}$ and the corresponding node value $\nu_i(\mathcal{C})_{\text{SET}}$ computed using Definition 3.9 are provided in Table 3.2. In computing the node value and considering which nodes can appear in $\mathcal{C}$, we assume that the node capture cost w_i for each source s_j and intermediate node m_j is unity, while that of each destination node is infinity.

Table 3.2: Route vulnerabilities and node values are computed for the set theoretic route vulnerability metrics for the network in Figure 3.3(a), rounding each quantity to the nearest 0.001.

i	s_1	s_2	m_1	m_2	m_3	m_4	m_5	m_6	m_7
$V^I_{s_1d_1}(\{i\})_{\text{SET}}$	1.000	0.400	0.700	0.950	0.600	0.775	0.300	0.300	0.500
$V^I_{s_2d_2}(\{i\})_{\text{SET}}$	0.100	1.000	0.100	0.500	0.783	0.975	0.800	0.767	0.500
$\nu_i(\varnothing)_{\text{SET}}$	1.100	1.400	0.800	1.450	1.383	**1.750**	1.100	1.067	1.000

To demonstrate the computation of quantities in Table 3.2, we consider the source-destination pair (s_1, d_1) in the second column and compute the route vulnerability resulting from the capture of node m_4. The route $\mathcal{R}_{s_1d_1}$ consists of four independent paths,

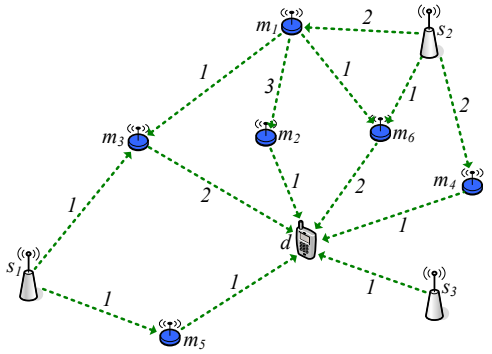
$$\pi_1 = \{(s_1, m_1), (m_1, m_2), (m_2, d_1)\} \quad \pi_2 = \{(s_1, m_1), (m_1, m_4), (m_4, d_1)\}$$
$$\pi_3 = \{(s_1, m_3), (m_3, m_2), (m_2, d_1)\} \quad \pi_4 = \{(s_1, m_3), (m_3, m_4), (m_4, d_1)\},$$

each corresponding to an independent single-path route. We assume that $f_{\pi_i} = f_{sd} = 1/4$.

To compute the set theoretic vulnerability $V^I_{sd}(\{m_4\})_{\text{SET}}$ using Figure 3.3(a), we first compute $\phi_{s_1d_1}(\{m_4\})$ as 1/2, the ϕ values for path π_1 as 1/2, 1/2, and 1/2, the ϕ values for path π_2 as 1/2, 1, and 1, the ϕ values for path π_3 as 0, 0, and 1/2, and the ϕ values for path π_4 as 0, 1, and 1, implying that paths π_2 and π_4 are compromised. From (3.4), the vulnerability is computed as $V^I_{sd}(\{m_4\})_{\text{SET}} = 31/40 = 0.775$, as indicated in Table 3.2. As indicated in Table 3.2, the first node added to $\mathcal{C}$ using the GNAVE algorithm under the set theoretic vulnerability function is node m_4.

3.7.2 Example: Distributed Data Access Using Network Coding

We illustrate a node capture attack using the GNAVE algorithm with the set theoretic route vulnerability metric presented in Section 3.4.1 for a network with three sources sending the same set of messages using network coding. In this example, we construct dependent path routes using a randomized network coding algorithm [38] in which each node forwards a different linear combination of previously received messages in the same message batch along each secure link. We consider the network topology given in Figure 3.4(a) with keys assigned to nodes in the network as given in Figure 3.4(b).



(a)

$\mathcal{K}_{s_1} = \{k_7, k_8, k_{11}, k_{13}\}$
$\mathcal{K}_{s_2} = \{k_3, k_5, k_9, k_{13}\}$
$\mathcal{K}_{s_3} = \{k_2, k_7, k_8, k_{11}\}$
$\mathcal{K}_d = \{k_1, k_2, k_{12}, k_{14}\}$
$\mathcal{K}_{m_1} = \{k_3, k_4, k_5, k_{10}\}$
$\mathcal{K}_{m_2} = \{k_3, k_4, k_5, k_{12}\}$
$\mathcal{K}_{m_3} = \{k_2, k_4, k_{11}, k_{14}\}$
$\mathcal{K}_{m_4} = \{k_3, k_9, k_{10}, k_{12}\}$
$\mathcal{K}_{m_5} = \{k_5, k_6, k_{12}, k_{13}\}$
$\mathcal{K}_{m_6} = \{k_2, k_4, k_{13}, k_{14}\}$

(b)

Figure 3.4: A destination node d receives messages from source nodes s_1, s_2, and s_3, with copies of the same data, using randomized network coding. Each link (i, j) is labeled with the number of shared keys $|\mathcal{K}_{ij}|$. The example network is illustrated in (a), and the assigned keys are shown in (b).

Table 3.3: Node values, equal to the route vulnerabilities, are computed for the set theoretic route vulnerability metric for the network in Figure 3.4(a), rounding each quantity to the nearest 0.001.

i	m_1	m_2	m_3	m_4	m_5	m_6
$\nu_i(\varnothing)_{\text{SET}}$	0.438	0.625	0.667	0.500	0.625	**0.792**

Since network coding is used to construct each transmitted packet as a function of the entire batch of messages, packets traversing different paths are dependent, even though links are independently secured. Furthermore, since the three sources s_1, s_2, and s_3 act as a single information source, we can treat the message traversal through the network as a single dependent route, effectively joining the source nodes s_1, s_2, and s_3 into a single source s. Hence, the route vulnerability of the route $\mathcal{R}_{sd}$ can be computed using (3.5). The route vulnerability $V_{sd}^{D}(\mathcal{C})_{\text{SET}}$ and the corresponding node value $\nu_i(\mathcal{C})_{\text{SET}}$ computed using Definition 3.9 are provided in Table 3.3. In computing the node value and considering which nodes can appear in $\mathcal{C}$, we assume that the node capture cost w_i for each intermediate node m_j is unity, while that of each source s_j and the destination node d is infinity.

To demonstrate the computation of quantities in Table 3.3, we evaluate the route vulner-

ability due to the capture of node m_6, which is the first node added to $\mathcal{C}$ using the GNAVE algorithm under the set theoretic vulnerability functions. For the network in Figure 3.4(a), we note that the route $\mathcal{R}_{sd}$ consists of 8 paths

$$\begin{aligned}
\pi_1 &= \{(s_1, m_3), (m_3, d)\}, & \pi_2 &= \{(s_1, m_5), (m_5, d)\}, \\
\pi_3 &= \{(s_2, m_4), (m_4, d)\}, & \pi_4 &= \{(s_2, m_6), (m_6, d)\}, \\
\pi_5 &= \{(s_2, m_1), (m_1, m_2), (m_2, d)\}, & \pi_6 &= \{(s_2, m_1), (m_1, m_3), (m_3, d)\}, \\
\pi_7 &= \{(s_2, m_1), (m_1, m_6), (m_6, d)\}, & \pi_8 &= \{(s_3, d)\},
\end{aligned}$$

where the end-to-end link (s, d) is already included as the path π_8 joining s_3 to d. By inspection of the collection of paths and the keys assigned to each node, we compute the ϕ values for each path as 0 and 1 for π_1, 1 and 0 for π_2, 0 and 0 for π_3, 1 and 1 for π_4, 0, 1/3, and 0 for π_5, 0, 1, and 1 for π_6, 0, 1, and 1 for π_7, and 1 for π_8. From (3.5) with $f_{sd} = 0$, the vulnerability is computed as $V_{sd}^{D}(\{m_6\})_{\text{SET}} = 19/24 \approx 0.792$, as indicated in Table 3.3.

3.7.3 Simulation Study: Wireless Sensor Network

We provide simulation results to illustrate a node capture attack using the GNAVE algorithm. We compare the performance of the attack to node capture attacks using existing node selection metrics.

The simulation was performed for a wireless sensor network of $|\mathcal{N}| = 500$ sensor nodes deployed randomly over a square region with density to yield an average of 25 neighbors per sensor node. Each node $i \in \mathcal{N}$ was randomly assigned a set of $|\mathcal{K}_i| = 50$ keys using key predistribution as in [29]. A subset of $|\mathcal{S}| = 100$ nodes was randomly selected as the set of source nodes, and a subset of $|\mathcal{D}| = 10$ nodes was randomly selected as the set of destination nodes. For each source $s \in \mathcal{S}$, the three nearest destination nodes in $\mathcal{D}$ were chosen as route pairs $(s, d) \in \mathcal{T}$. Each route $\mathcal{R}_{sd}$ was constructed using geographic forwarding with a hop-count mechanism to avoid routing loops and geographic dead-ends due to holes [81]. For both independent and dependent path routing, each node chose three next-hop neighbors closest to the destination and with a lower or equal hop count. For dependent path routing, we assume that any compromised edge cut is sufficient to compromise the route.

We simulated the node capture attacks using multiple strategies for both independent

and dependent path routing. We simulated secure link establishment using public label exchange without end-to-end security, public label exchange with end-to-end security, and privacy-preserving set intersection without end-to-end security using the estimation techniques in Section 3.6. Node capture attacks on each case were simulated for the following five node capture strategies.

1. Nodes are captured independently at random, serving as the baseline performance for the adversary.

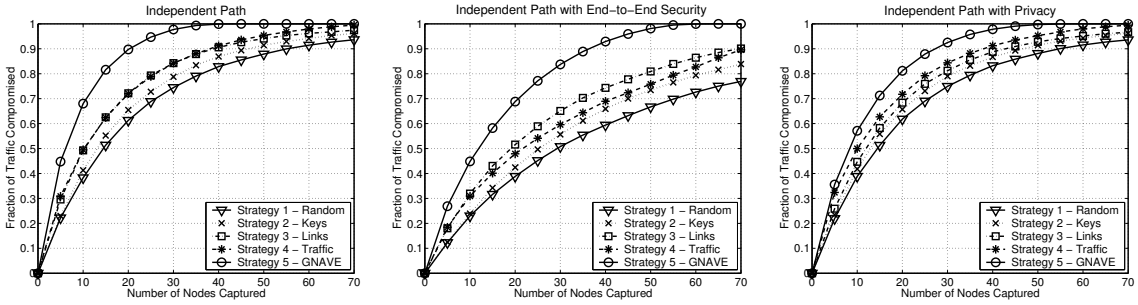
2. Nodes are captured iteratively to maximize the number of compromised keys $|\mathcal{K}_{\mathcal{C}}|$ by choosing the node i with maximum $|\mathcal{K}_i \setminus \mathcal{K}_{\mathcal{C}}|$ at each iteration, independent of the routing protocol. We note that such an attack can be performed deterministically under privacy-preserving protocols.

3. Nodes are captured iteratively to maximize the number of compromised links $|L_{\mathcal{C}}|$ by choosing the node i which compromises the maximum number of additional links, independent of the routing protocol. Under privacy-preserving protocols, this attack uses the estimation techniques in Section 3.6.

4. Nodes are captured iteratively to maximize the amount of network traffic routed through captured nodes, independent of the key assignment protocol.

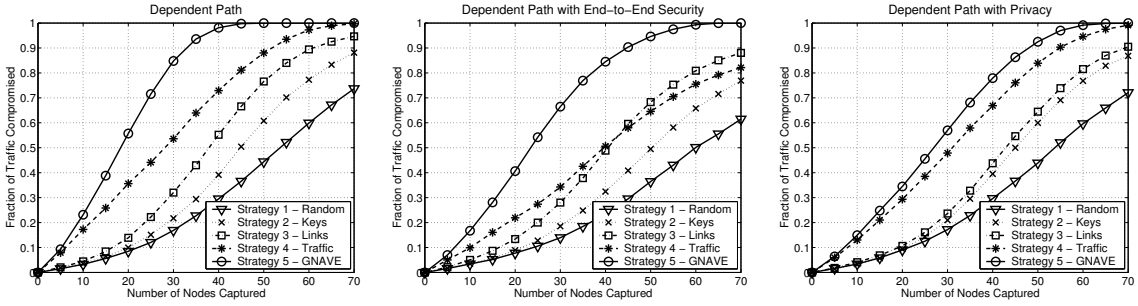
5. Nodes are captured using the GNAVE algorithm and the route vulnerability metric $V_{sd}^{I}(\mathcal{C})$ or $V_{sd}^{D}(\mathcal{C})$, using information from both the routing and key assignment protocols.

Figure 3.5 and Figure 3.6 illustrate the node capture attacks on independent and dependent path routing, respectively. In each figure, we notice that the node capture attack using the GNAVE algorithm outperforms the remaining attacks. The inclusion of the end-to-end shared keys $\mathcal{K}_{sd}$ in Figure 3.5(b) and Figure 3.6(b) show a consistent decrease in the attack performance for all attacks and all routing protocols due to the additional secure end-to-end



(a) (b) (c)

Figure 3.5: Node capture attacks using the five strategies are illustrated for a wireless sensor network of $|\mathcal{N}| = 500$ nodes for independent path routing (a) without end-to-end security, (b) with end-to-end security, and (c) using a privacy-preserving set intersection protocol.



(a) (b) (c)

Figure 3.6: Node capture attacks using the five strategies are illustrated for a wireless sensor network of $|\mathcal{N}| = 500$ nodes for dependent path routing (a) without end-to-end security, (b) with end-to-end security, and (c) using a privacy-preserving set intersection protocol.

link that must be compromised in each route. The addition of privacy-preserving set intersection protocols in Figure 3.5(c) and Figure 3.6(c) illustrate the increased uncertainty in route vulnerability which slightly degrades the performance of the attack using the GNAVE algorithm. In comparing Figure 3.5 and Figure 3.6, we notice that the dependence of messages traversing different paths displays a threshold behavior, reducing the vulnerability of routes for small $|\mathcal{C}|$, but only slightly increasing the number of captured nodes $|\mathcal{C}|$ required to compromise all traffic.

3.8 Summary of Contributions

We investigated the problem of developing new vulnerability metrics that improve the efficiency of node capture attacks when the routing and key assignment protocols used in a wireless network are jointly analyzed. We proposed a class of route vulnerability metrics (RVMs) to evaluate the effect of node capture attacks on secure network traffic and developed an RVM realization using a set theoretic interpretation of the compromise of secure network traffic. We formulated the optimal node capture attack using RVM evaluation as a nonlinear integer programming minimization problem and presented the GNAVE algorithm using a greedy heuristic to approximate the NP-hard problem. We demonstrated a probabilistic approach to estimate the route vulnerability when privacy-preserving set intersection protocols are used to hide information from the adversary. Finally, we illustrated node capture attacks using the GNAVE algorithm and compared the performance of the GNAVE algorithm with previously proposed node capture strategies. In the future, the node capture attack framework proposed in this chapter will assist in the joint design of key assignment and routing protocols for wireless networks that are robust to node capture attacks.

Chapter 4

MITIGATING CONTROL CHANNEL JAMMING USING RANDOM KEY ASSIGNMENT

Efficient communication in mobile networks requires the use of multiple access protocols allowing mobile users to share the wireless medium by separating user data in any combination of time, frequency, signal space, and physical space. The entire class of multiple access can thus be described by the unifying framework of orthogonal frequency division multiple access (OFDMA) [31]. Allocation of access and resources to mobile users must be periodically updated in order to maintain the efficiency of the multiple access protocol when base station group membership, user demands, and wireless channel conditions are dynamic. Hence, there is a necessary overhead involved in the multiple access protocol to handle the resource allocation to users. This overhead often takes the form of control messages exchanged between mobile users and base stations.

In many systems, dedicated channels are established for the exchange of control messages. These control channels can be used for a wide variety of functions, from topological information propagation for network routing to access control in subscription services. In a cellular system such as GSM [63, 67, 72], for example, base stations and mobile users must coordinate over a variety of control channels in order to perform access control, traffic channel allocation, and inter-cell user handoff. Control channels thus serve as a platform on which higher-level protocol functionality is supported and, hence, as critical points of failure that can be targeted by a malicious adversary in a denial of service (DoS) attack [4].

An adversary with knowledge of the underlying channel access protocol can perform a DoS attack against individual users or local neighborhoods in the mobile network by jamming the communication channels. Moreover, if the access protocol uses a fixed predetermined schedule for data and control messages, allowing the adversary to distinguish between channels for data and control messages, a *control channel jamming* attack focusing

only on the control channels can be mounted with energy savings of several orders of magnitude less than that required to jam all communication channels [14]. The use of jamming-resistant communication protocols such as Direct Sequence Spread Spectrum (DSSS) or Frequency Hopping Spread Spectrum (FHSS) [31, 67] introduce pseudo-randomness into the access schedule by keeping the spreading or hopping sequences, respectively, unknown to the adversary. It was noted in [60] that the effect of DSSS and FHSS may be further improved by using cryptographic primitives. Alternative anti-jamming techniques include the use of random channel surfing [80] to randomly hop away from jammed channels and re-synchronize on available channels and the use of wormholes [76] to create a channel for reports or alarms from a jammed region.

The above-mentioned anti-jamming techniques consider jamming attacks by an external adversary and are not intended to mitigate jamming by valid network insiders. A set of malicious colluding users or an adversary who captures or subverts network users in a *node capture attack* [29], potentially inserting replicated or fabricated devices into the system [57], is able to bypass the anti-jamming techniques above by assuming the collective roles of the compromised users in the network. For example, a set of malicious colluding users can use the available DSSS or FHSS sequences to perform an efficient jamming attack that follows the corresponding pseudo-random sequence as though it is a fixed schedule. An access protocol which gives the same information to all network users is thus ineffective against DoS attacks by internal adversaries, as a malicious insider has the ability to perform any task of a valid user. Hence, solutions to prevent or mitigate control channel jamming attacks by malicious insiders must make use of the following properties. First, multiple distinct pseudo-random sequences must exist and be held by different users. Second, the set of distinct sequences should exhibit a degree of *cover-freeness* [28] in that at least one of the sequences of each user should be different from the union of the set of sequences held by malicious colluding users with a non-negligible probability to ensure collusion resistance. Finally, the total number of pseudo-random sequences should scale favorably as the number of users increases, suggesting that there exist trade-offs between the cover-free property and the resource efficiency of the protocol.

We thus approach the problem of designing control channel access schemes which allow

for efficient reception of control messages while maintaining a degree of independence between the hopping sequences held by different users. In this work, we focus our attention on designing schemes which are robust to control channel jamming attacks by malicious colluding insiders or compromised users.

4.1 Our Contributions

In this chapter, we present a framework for control channel access schemes that are robust to control channel jamming. Furthermore, we provide techniques for random allocation of control channels to users which yields *graceful performance degradation as the number of compromised users increases*. In order to do so, we develop a correspondence between the problems of key establishment and control channel access in wireless networks and develop a framework for control channel access schemes providing probabilistic availability of control messages using random key assignment. We propose metrics of resilience and delay to quantify the probabilistic availability of service and the quality of provided service, respectively, under control channel jamming attacks. We evaluate the proposed metrics by extending existing results for resilience to node capture in wireless networks.

We propose techniques for the identification and revocation of compromised users by the service provider or a trusted authority that need not be constantly on-line. We formulate the identification problem as a maximum likelihood estimation problem and provide greedy heuristic algorithms using information available to the service provider. We evaluate the identification algorithm by approximating the false alarm and miss rates under the greedy algorithms. We provide a simulation study to demonstrate trade-offs that exist between robustness to control channel jamming and resource expenditure which result from the use of random key assignment protocols, serving as a foundation for the design of control channel access schemes.

4.2 Model Assumptions

In this section, we state the assumed models for the multiple access protocol and control message structure, adversary, and service provider or trusted authority. We provide a summary of the notation used throughout this work in Table 4.1.

4.2.1 Control Message Access Model

We describe the multiple access protocol in terms of the OFDMA framework [31] with separation of signals over orthogonal carrier signals and in time as follows. We let $\Psi = \{\psi_0, \ldots, \psi_{M-1}\}$ denote the set of M orthogonal carriers used for wireless communication. We assume that time is slotted and that an initial portion of each time slot is dedicated to control messages. Since we are focusing on the availability of control messages in this chapter, we ignore the portion of each time slot dedicated to data. We further partition each time slot t into S sub-slots with duration sufficient to transmit a single control message. Each control channel is thus specified by the time slot t, the sub-slot index $s \in \{0, \ldots, S-1\}$, and the carrier index $j \in \{0, \ldots, M-1\}$ into the set Ψ.

We let $\mathcal{B}$ denote the set of B base stations present in the network. Each base station $b \in \mathcal{B}$ holds the set $\mathcal{K}_t$ of $q_t = |\mathcal{K}_t|$ *control channel identifiers*, each corresponding to a control channel in time slot t. The sub-slot index s and carrier index j corresponding to each identifier $k_t \in \mathcal{K}_t$ are computed using the *control channel locator function* f, assumed to be publicly known. We let $\mathcal{U}$ denote the set of U mobile users in the network and assume that each user $u \in \mathcal{U}$ is within range of at least one base station. Any user $u \in \mathcal{U}$ holding the identifier $k_t \in \mathcal{K}_t$ can locate the corresponding control channel using the function f. We let $\mathcal{K}_{tu}$ denote the subset of $\mathcal{K}_t$ held by user u. We assume that each control message received over the channel identified by k_t carries information relevant to all users in $\mathcal{U}$ holding k_t. This access model is expanded in detail in Section 4.3.2.

4.2.2 Adversarial Model

We consider two types of adversaries. First, when malicious insiders collude under the described control channel access structure, they can jam any control channel which can be located using the control channel identifiers they possess. Second, an adversary who captures or subverts system users and assumes their identities in the network can jam control channels using identifiers acquired from compromised users. We let $\mathcal{C} \subseteq \mathcal{U}$ denote the set of *compromised users*, either colluding insiders or those captured by an adversary. For each time slot t, we let $\mathcal{K}_{t\mathcal{C}}$ denote the subset of $\mathcal{K}_t$ collectively held by the compromised

Table 4.1: We provide a summary of the notation used in Chapter 4 for the problem of mitigating control channel jamming.

Symbol	Definition
$\mathcal{U}, \mathcal{B}$	Set of U users, B base stations
p	Number of time slots in the reuse period
$\mathcal{K}_t$	Set of channel identifiers, or keys, for time slot t
q_t	Number of control channels in time slot t, $\lvert\mathcal{K}_t\rvert$
$\mathcal{K}_{tu}$	Subset of $\mathcal{K}_t$ assigned to user u
m_t	Number of keys per user in time slot t, $\lvert\mathcal{K}_{tu}\rvert$
$\mathcal{C}, c$	Set and number of compromised users, $c = \lvert\mathcal{C}\rvert$
$\mathcal{K}_{t\mathcal{C}}$	Subset of $\mathcal{K}_t$ held by $\mathcal{C}$
$\mathcal{J}_t$	Subset of $\mathcal{K}_{t\mathcal{C}}$ corresponding to jammed channels
θ_t	Probability that each key $k \in \mathcal{K}_{t\mathcal{C}}$ is added to $\mathcal{J}_t$
$r_t(c)$	Slot resilience for time slot t
$r(c)$	Resilience to control channel jamming
$d_t(c)$	Initial-slot delay for time slot t
$d(c)$	Delay due to control channel jamming
$\widehat{\mathcal{C}}$	Estimate of set $\mathcal{C}$ of compromised users
$\mathcal{F}(c)$	False alarm rate in the estimate $\widehat{\mathcal{C}}$ of $\mathcal{C}$
$\mathcal{M}(c)$	Miss rate in the estimate $\widehat{\mathcal{C}}$ of $\mathcal{C}$

users, i.e. $\mathcal{K}_{t\mathcal{C}} = \bigcup_{u \in \mathcal{C}} \mathcal{K}_{tu}$. Furthermore, we let $\mathcal{J}_t$ denote the subset of $\mathcal{K}_{t\mathcal{C}}$ corresponding to control channels in time slot t that are jammed by compromised users.

The case that insiders expose their identifiers to each other and collaboratively choose the subset $\mathcal{J}_t$ is equivalent to that of a adversary in control of multiple compromised users. Alternatively, malicious insiders may independently choose contributions to the overall subset $\mathcal{J}_t$, suggesting that the probability that each key k_t is included in $\mathcal{J}_t$ may increase with the number of compromised users $|\mathcal{C}|$. We let θ_t denote the probability that each identifier $k_t \in \mathcal{K}_{t\mathcal{C}}$ is included in $\mathcal{J}_t$, noting that θ_t may be a function of $|\mathcal{C}|$. In this work, we assume that identifiers in $\mathcal{K}_{t\mathcal{C}}$ are added to $\mathcal{J}_t$ independently with probability θ_t[1].

4.2.3 Trusted Authority

We assume that a trusted authority[2] (TA) is responsible for the assignment and update of control channel identifiers to users in $\mathcal{U}$ and the identification and revocation of compromised users in the network. We assume that the TA keeps a record of the sets $\mathcal{K}_{tu}$ for each user $u \in \mathcal{U}$ and time slot t and can detect jammed control channels without error, thus recovering the set $\mathcal{J}_t$ for each t. By comparing the collections of sets $\mathcal{K}_{tu}$ and $\mathcal{J}_t$ for various time slots, the TA can determine a set $\widehat{\mathcal{C}}$ of suspected jammers to eliminate from the network. For each time slot t, the set of identifiers $\mathcal{K}_{t\widehat{\mathcal{C}}} \subseteq \mathcal{K}_t$ are removed from $\mathcal{K}_t$ and replaced with fresh identifiers. Any user $u \in \mathcal{U} \setminus \widehat{\mathcal{C}}$ holding an identifier in $\mathcal{K}_{t\widehat{\mathcal{C}}}$ is assigned the corresponding fresh identifiers. We assume that a mechanism for secure key refresh exists and do not further address the key refreshing protocol in this dissertation. We note that, unless the estimation $\widehat{\mathcal{C}}$ of $\mathcal{C}$ is perfect, which is unlikely given the intelligent adversary model, it is possible that valid users in $\mathcal{U} \setminus \mathcal{C}$ will be eliminated from the network or that compromised users in $\mathcal{C}$ who participate in control channel jamming may not appear in $\widehat{\mathcal{C}}$ as suspected jammers.

Our approach does not require constant presence of the TA to oversee the network. In fact, the TA may only be available occasionally to perform the identification and elimination

[1]This assumption is information-theoretically minimal in that the entropy of the set $\mathcal{J}_t$ is maximized for a given θ_t [51].

[2]The TA can be a service provider or a subset of the base stations in $\mathcal{B}$, for example.

steps by recording jamming evidence $\mathcal{J}_t$, computing the set $\widehat{\mathcal{C}}$ of suspected jammers, and refreshing the control channel identifiers for the remaining users in $\mathcal{U} \backslash \widehat{\mathcal{C}}$. When the adversary compromises system users over an extended duration of time, the random or deterministic *identification interval* between successive identification steps by the TA impacts the total number of compromised users $|\mathcal{C}|$ for a given identification step and the ability for the TA to identify those users with the estimate $\widehat{\mathcal{C}}$.

4.3 Random Key Assignment Framework for Control Channel Access

In this section, we develop a correspondence between the problems of control channel access and symmetric key assignment. We show that efficient and robust control channel access can be provided using random key assignment, yielding a framework for probabilistic control channel access schemes.

4.3.1 Problem Mapping

We provide a one-to-one mapping between control channel access for multiple users in a single time slot and the assignment of symmetric keys to network nodes for use in cryptographic protocols. The mapping is formalized by constructing a bipartite graph [23] which uniquely maps between control channel access schemes and symmetric key assignment schemes.

For a given time slot t, let $G_t = (\mathcal{U} \cup \mathcal{B}, \mathcal{K}_t, E_t)$ be a bipartite graph with left vertex set $\mathcal{U} \cup \mathcal{B}$, right vertex set $\mathcal{K}_t$, and edge set $\mathcal{E}_t \subseteq (\mathcal{U} \cup \mathcal{B}) \times \mathcal{K}_t$. The edge (u, k_t) for $u \in \mathcal{U}$ is in E_t if and only if $k_t \in \mathcal{K}_{tu}$, so u can compute the corresponding channel location using the locator function f. A user $u \in \mathcal{U}$ can receive control messages from a base station $b \in \mathcal{B}$ if and only if $(b, k_t) \in E_t$ and $(u, k_t) \in E_t$. Any compromised user $w \in \mathcal{C}$ that also holds the identifier k_t can compute the channel location using the locator function f, so the channel can be jammed if and only if there is at least one such user $w \in \mathcal{C}$ such that $(w, k_t) \in E_t$.

The bipartite graph G_t constructed above is next used to uniquely construct a symmetric key assignment scheme used to establish secure communication in a wireless network. Let $\mathcal{K}_t$ be a set of symmetric cryptographic keys [53], and let $\mathcal{N} = \mathcal{U} \cup \mathcal{B}$ represent the set of network nodes. For each $k_t \in \mathcal{K}_t$, assign the key k_t to node $n \in \mathcal{N}$ if and only if $(n, k_t) \in E_t$. A pair of nodes $n_1, n_2 \in \mathcal{N}$ can communicate securely if and only if there exists at least

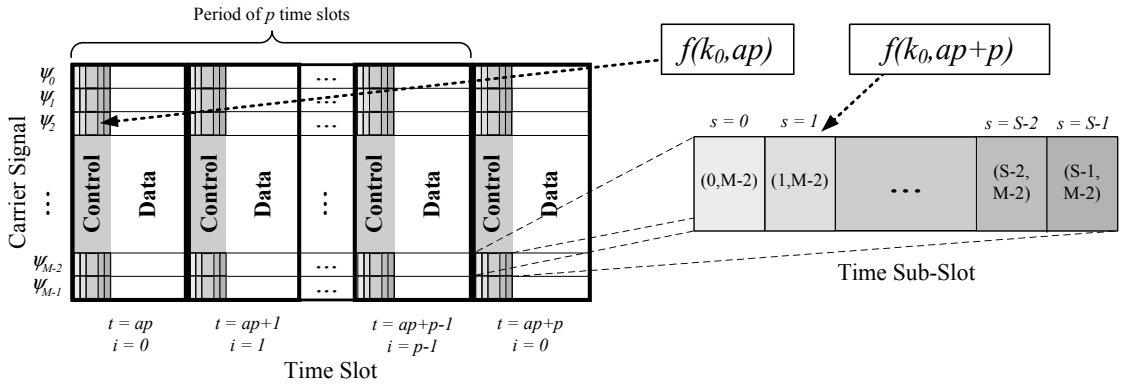


Figure 4.1: A control channel access scheme using random key assignment allows for pseudo-random relocation of control channels over time, preventing an adversary from learning via correlation. Each user and base station with a control channel identifier k_i for $i \equiv t \pmod{p}$ locates the corresponding control channel in time slot t as $(s, j) = f(k_i, t)$, where s is a sub-slot index in slot t and j is an index into the set Ψ of carrier signals.

one key $k_t \in \mathcal{K}_t$ such that both $(n_1, k_t) \in E_t$ and $(n_2, k_t) \in E_t$. If the key k_t is held by any compromised node $w \in \mathcal{C}$, the communication between n_1 and n_2 is insecure against attacks by the adversary (e.g. eavesdropping on encrypted messages). Hence, the secure link is compromised if and only if there is one such user $w \in \mathcal{C}$ such that $(w, k_t) \in E_t$.

The bipartite graph G_t thus provides a one-to-one correspondence between control channel access schemes and symmetric key assignment schemes[3]. Hence, key assignment solutions that provide secure communication which is robust to node capture attacks can be used to design control channel access schemes which are resilient to control channel jamming attacks by compromised users.

4.3.2 Random Assignment of Control Channel Keys

Using the mapping in Section 4.3.1, we make use of the symmetric key assignment model in Chapter 2 to provide a framework for probabilistic control channel access using random

[3]We note that, by assumption, each left node $b \in \mathcal{B}$ is joined to all right nodes $k_t \in \mathcal{K}_t$, though the mapping holds regardless of this assumption, suggesting a natural extension of the problem in which each base station b is assigned a subset of $\mathcal{K}_t$ instead of the entire set. Alternatively, the mapping allows for modeling of the case in which base stations are not present and the users organize in an ad-hoc manner. These extensions are not addressed in this dissertation.

key assignment. The proposed framework can then be used to design control channel access schemes which are robust to jamming by compromised users. For the remainder of this chapter, we use the term *control channel key* interchangeably with control channel identifier.

As previously discussed, a control channel access scheme is only robust to control channel jamming by compromised users if a user holds keys that are not held by any compromised user with high probability. Due to fact that any users in $\mathcal{U}$ can be compromised, it is necessary to impose a degree of disparity between the sets $\mathcal{K}_{tu}$ of assigned keys. This necessary disparity is seen by noting that if all sets $\mathcal{K}_{tu}$ are equal, for example when all users share a single global key, a single compromised user can jam all control messages. However, increasing the diversity of keys assigned to different users implies that the total number of keys q_t for each time slot t must increase. Increasing the number of keys q_t in time slot t further implies that the key storage and the number of control messages transmitted by each base station in time slot t increase. Hence, inherent trade-offs exist between the robustness to control channel jamming and the efficiency of the protocol in terms of storage and communication overhead. In order to balance the trade-offs between robustness and efficiency, we provide a control channel key assignment framework in generality and explicitly discuss the trade-offs in Section 4.6.

The random assignment of control channel keys to system users is described as follows, using the random key assignment framework in Chapter 2. For each user $u \in \mathcal{U}$ and time slot t, the subset $\mathcal{K}_{tu}$ of m_t keys[4] is randomly selected from $\mathcal{K}_t$ and assigned to u, independent of other users in $\mathcal{U}$.

The choice of random key assignment is motivated by the following observations. First, without prior assumptions on the maximum number of compromised users, choosing the parameters of a deterministic key assignment scheme [14] may not be possible. Random key assignment allows for a high degree of flexibility in parameter choice. Second, the imposed structure of deterministic key assignment schemes may allow the adversary to learn information about the assignment of keys to users other than those in $\mathcal{C}$, as shown in Chapter 2. In random key assignment, each user is assigned keys independently, so the

[4]The number of assigned keys per time slot can vary among users as m_{tu}, though we do not address this extension in this work.

adversary cannot learn any information about the assignment of keys to users other than those in $\mathcal{C}$.

To maintain finite key storage for each user and base station and to prevent frequent re-assignment of keys to all users in the network, we adopt the periodic reuse of keys in time slots such that in any time slot t, control channels are located using the keys in the assigned subset $\mathcal{K}_{iu} \subseteq \mathcal{K}_i$ for $i \equiv t \pmod{p}$. The *reuse period* p is thus a parameter in the design of the control channel access scheme. An additional benefit of the finiteness constraint on the number of distinct time slots is that we can construct the sets $\mathcal{K}_i$ for $i = 0, \ldots, p-1$ to be pairwise disjoint. Furthermore, we assume that the p subsets $\mathcal{K}_{iu}$ for each user u are independently selected, implying that the probabilistic availability of control messages in each time slot is independent.

A consequence of the periodic reuse of keys from each subset $\mathcal{K}_i$ is that control channels will appear at the same location every p time slots if the locator function f depends only on the key k_i. In this case, the adversary may be able to learn the locations of control channels by correlating transmission patterns in corresponding time slots. To prevent transmission correlation, the locator function f must take an additional parameter to vary the control channel location in subsequent periods. In a given time slot t such that $i \equiv t \pmod{p}$, the sub-slot index s and carrier index j are thus given by $(s, j) = f(k_i, t)$. To ensure that distinct control channel keys map to distinct ordered pairs (s, j) with high probability, the locator function f can be implemented using a cryptographic hash function [14, 53]. Figure 4.1 illustrates the control channel access scheme.

A control channel access scheme using random key assignment is primarily dependent on the key reuse period p, the number of control channels q_i in each time slot $i = 0, \ldots, p-1$, and the number of control channel keys m_i assigned to each user in $\mathcal{U}$ for use in each time slot $i = 0, \ldots, p-1$. To provide a basis for the design problem, the following sections evaluate the robustness to control channel jamming and the ability to identify and eliminate compromised users from the system.

4.4 Availability of Control Messages Under Control Channel Jamming

In order to evaluate the effect of control channel jamming by compromised users, we define and evaluate metrics to quantify the probabilistic availability of control messages. We note that users in the proposed control channel access scheme as outlined in Section 4.3 do not exchange any information about the assigned keys $\mathcal{K}_{iu}$, so the adversary cannot obtain any deterministic information about the key assignment. Intelligent node capture attacks using the techniques proposed in Chapter 3 using such information are thus impossible. Hence, the selection of the subset $\mathcal{C} \subseteq \mathcal{U}$ of compromised users is independent of the key assignment, implying that the compromised users are randomly selected by the adversary. We thus define the following metrics to measure the availability of control messages as a function of the number $c = |\mathcal{C}|$ of compromised users, noting that the proposed metrics are computed for the average case.

Definition 4.1. *The* slot resilience *to control channel jamming by $c = |\mathcal{C}|$ compromised users is the probability $r_i(c)$ that a user in $\mathcal{U} \setminus \mathcal{C}$ is able to receive at least one control message in time slot i.*

Definition 4.2. *The* resilience *to control channel jamming by $c = |\mathcal{C}|$ compromised users is the probability $r(c)$ that a user in $\mathcal{U} \setminus \mathcal{C}$ is able to receive at least one control message.*

Definition 4.3. *The* initial slot delay *due to control channel jamming by $c = |\mathcal{C}|$ compromised users is the average number of time slots $d_i(c)$ that a user in $\mathcal{U} \setminus \mathcal{C}$ must wait to receive a control message when the initial access attempt is made during time slot i. As the delay is infinite for users with no control channel availability, this metric considers only those users able to receive control messages.*

Definition 4.4. *The* delay *due to control channel jamming by $c = |\mathcal{C}|$ compromised users is the average number of time slots $d(c)$ that a user in $\mathcal{U} \setminus \mathcal{C}$ must wait to receive a control message, considering only those users able to receive control messages.*

In the following sequence of results, we evaluate the resilience and delay metrics using the properties of random control channel key assignment in Section 4.3.2. We first derive

an approximation for the slot resilience $r_i(c)$. We then prove that the resilience $r(c)$, initial slot delay $d_i(c)$, and delay $d(c)$ can be expressed as a function of the slot resilience $r_i(c)$. The following lemma provides a necessary component in the evaluation of the slot resilience $r_i(c)$.

Lemma 4.1. *When $|\mathcal{C}| = c$, the probability $p_{i,c}(s)$ that $|\mathcal{K}_{iu} \cap \mathcal{J}_i| = s$ for any user $u \in \mathcal{U} \setminus \mathcal{C}$ is approximated as*

$$p_{i,c}(s) \approx \binom{m_i}{s} (\theta_i z_{i,c})^s (1 - \theta_i z_{i,c})^{m_i - s}$$

where $z_{i,c} \approx 1 - \left(1 - \frac{m_i}{q_i}\right)^c$.

Proof. Let $p_{c,u}$ denote the probability for a user $u \in \mathcal{U} \setminus \mathcal{C}$ that a particular key $k_i \in \mathcal{K}_{iu}$ is in $\mathcal{J}_i$. Since $\mathcal{J}_i \subseteq \mathcal{K}_{i\mathcal{C}}$, $p_{c,u}$ can be expressed as the product of probabilities $\Pr[k_i \in \mathcal{J}_i | k_i \in \mathcal{K}_{i\mathcal{C}}, k_i \in \mathcal{K}_{iu}]$ and $\Pr[k_i \in \mathcal{K}_{i\mathcal{C}} | k_i \in \mathcal{K}_{iu}]$. The former probability is equal to θ_i by definition. The latter is the probability that at least one of the c compromised users shares the key k_i with user u. Letting $\lambda(k_i)$ denote the number of users in $\mathcal{U}$ holding the key $k_i \in \mathcal{K}$, this probability is approximated by Lemma 2.6 as $1 - \left(\frac{U - \lambda(k_i)}{U-1}\right)^c$, yielding

$$p_{c,u} \approx \theta_i \left(1 - \left(\frac{U - \lambda(k_i)}{U - 1}\right)^c\right). \tag{4.1}$$

Similar to the result of Theorem 2.6, the average p_c of $p_{c,u}$ over all users $u \in \mathcal{U} \setminus \mathcal{C}$ can be approximated by replacing $\lambda(k_i)$ by its expected value μ_i, yielding $p_c \approx \theta_i z_{i,c}$ where

$$z_{i,c} = 1 - \left(\frac{U - \mu_i}{U - 1}\right)^c. \tag{4.2}$$

The probability $z_{i,c}$ can be approximated independent of U by noting that $\mu_i = U m_i / q_i$ when keys are assigned randomly, noting that the approximation holds with equality in the limit of large U. Since the keys in $\mathcal{K}_i$ are assigned independently, the probability $p_{i,c}(s)$ satisfies a binomial distribution corresponding to m_i independent trials with success probability p_c. □

The following theorem provides an approximation for the slot resilience $r_i(c)$ using the previous result.

Theorem 4.1. *The slot resilience $r_i(c)$ is approximated as*

$$r_i(c) \approx 1 - \theta_i^{m_i} \left(1 - \left(1 - \frac{m_i}{q_i}\right)^c\right)^{m_i}.$$

Proof. By Definition 4.1, the slot resilience $r_i(c)$ is equal to the probability that $|\mathcal{K}_{iu} \cap \mathcal{J}_i| \neq m_i$ for a user $u \in \mathcal{U} \setminus \mathcal{C}$. Hence, $r_i(c) = 1 - p_{i,c}(m_i)$, where $p_{i,c}(s)$ is given in Lemma 4.1. □

We next show how the resilience $r(c)$ can be computed as a function of the slot resilience $r_i(c)$ for $i = 0, \ldots, p-1$.

Theorem 4.2. *The resilience $r(c)$ can be computed from the slot resilience $r_i(c)$ for $i = 0, \ldots, p-1$ as*

$$r(c) = 1 - \prod_{i=0}^{p-1} (1 - r_i(c)).$$

Proof. A user $u \in \mathcal{U} \setminus \mathcal{C}$ can receive a control message in slot i if and only if $\mathcal{K}_{iu} \nsubseteq \mathcal{J}_i$, an event which occurs with probability $r_i(c)$, by Definition 4.1. Similarly, a user $u \in \mathcal{U} \setminus \mathcal{C}$ can receive at least one control message in at least one slot if and only if $\mathcal{K}_{iu} \nsubseteq \mathcal{J}_i$ for at least one time slot i, an event which occurs with probability $r(c)$, by Definition 4.2. The probability $1 - r(c)$ that no channel is available in any slot is given by the product of probabilities $(1 - r_i(c))$ for $i = 0, \ldots, p-1$. Independence of key assignment for different time slots yields the desired result. □

We next show how the initial slot delay $d_i(c)$ and delay $d(c)$ can be expressed as a function of the slot resilience $r_i(c)$. We note that the delay $d(c)$ can be expressed as a weighted sum of the initial slot delays $d_i(c)$ for $i = 0, \ldots, p-1$, where the weight multiplying each $d_i(c)$ is the probability that the initial access attempt is made at time slot i. The probability distribution of delay $d(c)$ can thus be computed as a function of the probability distribution of initial slot delay $d_i(c)$ for $i = 0, \ldots, p-1$. We provide the following result to evaluate the latter probability distribution as a function of the slot resilience $r_i(c)$.

Theorem 4.3. *The probability distribution of $d_i(c)$ is given by*

$$\Pr[d_i(c) = \delta] = \gamma_i r_{i+\delta \bmod p}(c) \prod_{d=0}^{\delta-1} (1 - r_{i+d \bmod p}(c))$$

where γ_i is a normalization constant to ensure the summation over $\delta = 0, \ldots, p-1$ equals 1.

Proof. A user must wait δ time slots starting at slot i if and only if there is no control channel available in the first δ time slots beginning at (and including) i and there is a control channel available in slot $(i+\delta \bmod p)$. In each time slot $(i+d \bmod p)$, the probability that no control channel is available is the complement $(1 - r_{i+d \bmod p}(c))$ of the corresponding slot resilience. Independence of key assignment for different time slots yields the desired result. □

In the special case of equal key assignment and jamming parameters for all time slots, the resilience $r(c)$ and the distribution of the delay $d(c)$ can be greatly simplified. The following results illustrate these simplifications.

Theorem 4.4. *When $m_i = m$, $q_i = q$, and $\theta_i = \theta$ for all $i = 0, \ldots, p-1$, the resilience $r(c)$ is approximated as*

$$r(c) \approx 1 - \theta^{mp} \left(1 - \left(1 - \frac{m}{q}\right)^c\right)^{mp}.$$

Proof. The result follows directly from Theorem 4.2 and Theorem 4.1. □

Theorem 4.5. *When $m_i = m$, $q_i = q$, and $\theta_i = \theta$ for all $i = 0, \ldots, p-1$, the probability distribution of delay $d(c)$ is given by*

$$\Pr[d(c) = \delta] = \frac{r_0(c)}{r(c)} \left(1 - r_0(c)\right)^\delta$$

for $\delta = 0, \ldots, p-1$.

Proof. In the given special case, the slot resilience $r_i(c) = r_0(c)$ for all $i = 0, \ldots, p-1$. Theorem 4.3 thus yields the probability distribution of $d_i(c)$ as

$$\Pr[d_i(c) = \delta] = \gamma_i r_0(c)(1 - r_0(c))^\delta \tag{4.3}$$

for $\delta = 0, \ldots, p-1$. The normalization constant $\gamma_i = 1/r(c)$ is obtained by evaluating the finite geometric sum and using the result of Theorem 4.2. The probability distribution of $d_i(c)$ in (4.3) is independent of i, so the distribution of $d(c)$ given by any normalized weighted sum of the $d_i(c)$ for $i = 0, \ldots, p-1$ is equal to the distribution of $d_i(c)$. □

To complete the analysis of the delay metric, we compute the expected value $\bar{d}(c)$ of the delay $d(c)$ for the special case approached in Theorem 4.5. We note that similar techniques can be applied in computing the expected delay in the general case.

Theorem 4.6. *When $m_i = m$, $q_i = q$, and $\theta_i = \theta$ for all $i = 0, \ldots, p-1$, the expected delay $\bar{d}(c)$ is given by*

$$\bar{d}(c) = p - 1 + \frac{1}{r_0(c)} - \frac{p}{r(c)}.$$

Proof. The expected value $\bar{d}(c)$ of $d(c)$ is obtained from the result of Theorem 4.5 using the properties of finite geometric random variables [32]. □

The results obtained in this section can thus be used in the design of control channel key assignment schemes, in particular to balance trade-offs between robustness to control channel jamming and efficiency in terms of key storage and control overhead. These trade-offs in the design process are further discussed in Section 4.6, after investigating the ability for the TA to identify and eliminate compromised users in the network.

4.5 Identification of Compromised Users

In this section, we formulate a statistical estimation problem for the identification of compromised users by the TA, constructing a set $\widehat{\mathcal{C}}$ of suspected jammers to eliminate from the network with no knowledge of the number of compromised users $c = |\mathcal{C}|$. Due to the complexity of the resulting identification problem, we propose two algorithms, collectively referred to as GUIDE (Greedy User IDEntification), based on a greedy heuristic which ranks users according to the likelihood of being a compromised user. Finally, we approximate the estimation error resulting from the GUIDE algorithms.

Throughout this section, we denote the parameter vector $(\mathcal{K}_{0u}, \ldots, \mathcal{K}_{(p-1)u})$ as $\mathcal{K}_u$, $(\mathcal{K}_{0\mathcal{C}}, \ldots, \mathcal{K}_{(p-1)\mathcal{C}})$ as $\mathcal{K}_{\mathcal{C}}$, $(\mathcal{J}_0, \ldots, \mathcal{J}_{p-1})$ as $\mathcal{J}$, and $(\theta_0, \ldots, \theta_{p-1})$ as Θ. Furthermore, we extend the use of logical relations and set cardinalities to parameter vectors in a natural way. For example, we say that $\mathcal{J} \subseteq \mathcal{K}_{\mathcal{C}}$ if and only if $\mathcal{J}_i \subseteq \mathcal{K}_{i\mathcal{C}}$ for $i = 0, \ldots, p-1$, and we let $|\mathcal{J}| = \sum_{i=0}^{p-1} |\mathcal{J}_i|$.

The information available to the TA and adversary during the attack and identification process are illustrated in Figure 4.2. The parameter $\mathcal{K}_u$ is known to the TA for all $u \in \mathcal{U}$

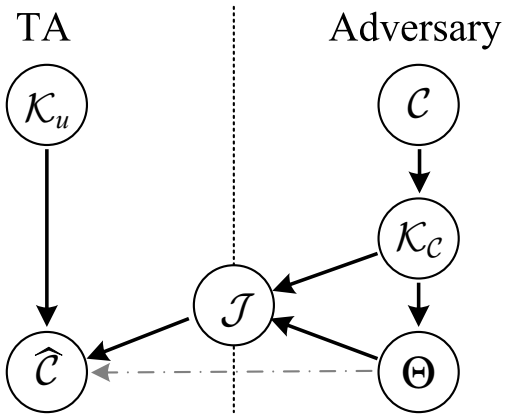


Figure 4.2: The information available to the TA and adversary during the attack and identification process is illustrated. The TA has knowledge of the parameters $\mathcal{K}_u$ and $\mathcal{J}$ and uses this available information to construct an estimate $\widehat{\mathcal{C}}$ of $\mathcal{C}$. The adversary has knowledge of the parameters $\mathcal{C}$, $\mathcal{K}_\mathcal{C}$, Θ, and $\mathcal{J}$. The dotted line from Θ to $\widehat{\mathcal{C}}$ indicates that the TA may or may not know Θ.

and to the adversary only for $u \in \mathcal{C}$. The evidence $\mathcal{J} \subseteq \mathcal{K}_\mathcal{C}$ is known to the TA and the adversary, but $\mathcal{K}_\mathcal{C}$ is known only to the adversary. The problem of identifying the set $\mathcal{C}$ of jammers is first formulated as a statistical estimation problem in which the TA constructs an estimate $\widehat{\mathcal{C}}$ of $\mathcal{C}$ as a function of the known parameters $\mathcal{J}$ and Θ. When $\Theta = 1$, then $\mathcal{J} = \mathcal{K}_\mathcal{C}$ and the uncertainty in the identification process is greatly reduced. However, it is quite likely that Θ will not be known to the TA, so we vary the heuristic for the case of unknown Θ.

4.5.1 Identification with $\Theta = 1$

When $\Theta = 1$, the evidence $\mathcal{J}$ allows the TA to deterministically know $\mathcal{K}_\mathcal{C}$. Hence, the only information the adversary has that the TA does not have is the set $\mathcal{C}$. The TA can thus infer that any user $u \in \mathcal{U}$ holding a key $k_i \notin \mathcal{K}_{i\mathcal{C}}$ for any i cannot be a compromised user. Hence, any user u such that $\mathcal{K}_u \subseteq \mathcal{K}_\mathcal{C}$ is identified as a compromised user, though it is possible that users are falsely identified. This case was addressed in [73] for a similar random key assignment model and in [14] for certain deterministic key assignment schemes.

4.5.2 Estimation of Compromised User Set $\mathcal{C}$

We formulate the jammer identification problem using statistical estimation by defining the probability $\Pr[\mathcal{C}|\mathcal{J},\Theta]$ that $\mathcal{C}$ is the set of compromised users responsible for jamming the control channels indicated by the parameters $\mathcal{J}$ and Θ. The estimate which maximizes the probability $\Pr[\mathcal{C}|\mathcal{J},\Theta]$ is defined as follows.

Definition 4.5. *The* maximum a posteriori (MAP) estimate *[51] $\widehat{\mathcal{C}}$ of the set $\mathcal{C}$ of compromised users is given by*

$$\widehat{\mathcal{C}} = \arg\max_{\mathcal{C}\subseteq\mathcal{U}} \Pr[\mathcal{C}|\mathcal{J},\Theta].$$

An alternate statistical estimation problem can be formulated by defining the likelihood function $\Pr[\mathcal{J}|\mathcal{C},\Theta]$ that the evidence $\mathcal{J}$ is the outcome of jamming by a given set of compromised users $\mathcal{C}$ and parameter Θ. The estimate which maximizes the likelihood function is defined as follows.

Definition 4.6. *The* maximum likelihood (ML) estimate *[51] $\widehat{\mathcal{C}}$ of the set $\mathcal{C}$ of compromised users is given by*

$$\widehat{\mathcal{C}} = \arg\max_{\mathcal{C}\subseteq\mathcal{U}} \Pr[\mathcal{J}|\mathcal{C},\Theta].$$

The primary difference between the MAP and ML estimates is the availability of prior information about the set $\mathcal{C}$ being estimated, as can be shown using Bayes' Theorem [51]. Since there is no prior information available to the TA about $\mathcal{C}$ and all users are equally likely to be compromised, the MAP and ML estimates are equivalent [51]. The problem of estimating $\widehat{\mathcal{C}}$ can thus be formulated with respect to the likelihood function $\Pr[\mathcal{J}|\mathcal{C},\Theta]$ characterized by the following results.

Theorem 4.7. *The likelihood function* $\Pr[\mathcal{J}|\mathcal{C},\Theta]$ *is given by*

$$\Pr[\mathcal{J}|\mathcal{C},\Theta] = \begin{cases} \prod_{i=0}^{p-1} \theta_i^{|\mathcal{J}_i|}(1-\theta_i)^{|\mathcal{K}_{i\mathcal{C}}|-|\mathcal{J}_i|}, & \text{if } \mathcal{J} \subseteq \mathcal{K}_{\mathcal{C}} \\ 0, & \text{else} \end{cases}.$$

Proof. If $\mathcal{J} \nsubseteq \mathcal{K}_{\mathcal{C}}$, then jamming by compromised users $\mathcal{C}$ could not lead to evidence $\mathcal{J}$. Hence, the likelihood function is non-zero only if $\mathcal{J} \subseteq \mathcal{K}_{\mathcal{C}}$. By assumption in Section 4.2.2,

the sets $\mathcal{J}_i$ for $i = 0, \ldots, p-1$ are selected independently, simplifying the likelihood function as

$$\Pr[\mathcal{J}|\mathcal{C}, \Theta] = \prod_{i=0}^{p-1} \Pr[\mathcal{J}_i|\mathcal{C}, \theta_i]. \tag{4.4}$$

The dependence of $\mathcal{J}_i$ on $\mathcal{C}$ is in the form of the set $\mathcal{K}_{i\mathcal{C}}$ of keys assigned to compromised users. The likelihood $\Pr[\mathcal{J}_i|\mathcal{C}, \theta_i]$ is thus equal to the probability that independent selection of each element of $\mathcal{K}_{i\mathcal{C}}$ with probability θ_i yields $\mathcal{J}_i$. The likelihood function $\Pr[\mathcal{J}_i|\mathcal{C}, \theta_i]$ is thus given by

$$\Pr[\mathcal{J}_i|\mathcal{C}, \theta_i] = \begin{cases} \theta_i^{|\mathcal{J}_i|}(1-\theta_i)^{|\mathcal{K}_{i\mathcal{C}}|-|\mathcal{J}_i|}, & \text{if } \mathcal{J}_i \subseteq \mathcal{K}_{i\mathcal{C}} \\ 0, & \text{else} \end{cases}. \tag{4.5}$$

□

In the case that Θ is a constant vector, i.e. $\theta_i = \theta$ for $i = 0, \ldots, p-1$, the likelihood $\Pr[\mathcal{J}|\mathcal{C}, \Theta]$ can be simplified as follows.

Theorem 4.8. *If $\theta_i = \theta$ for $i = 0, \ldots, p-1$, the ML estimate $\widehat{\mathcal{C}}$ of $\mathcal{C}$ is independent of Θ and given by*

$$\widehat{\mathcal{C}} = \underset{\substack{\mathcal{C}\subseteq\mathcal{U}, \\ \mathcal{J}\subseteq\mathcal{K}_{\mathcal{C}}}}{\arg\min} |\mathcal{K}_{\mathcal{C}}|.$$

Proof. The result of Theorem 4.7 applied to Definition 4.6 with $\theta_i = \theta$ yields the estimate

$$\begin{aligned} \widehat{\mathcal{C}} &= \underset{\substack{\mathcal{C}\subseteq\mathcal{U}, \\ \mathcal{J}\subseteq\mathcal{K}_{\mathcal{C}}}}{\arg\max} \prod_{i=0}^{p-1} \theta^{|\mathcal{J}_i|}(1-\theta)^{|\mathcal{K}_{i\mathcal{C}}|-|\mathcal{J}_i|} \\ &= \underset{\substack{\mathcal{C}\subseteq\mathcal{U}, \\ \mathcal{J}\subseteq\mathcal{K}_{\mathcal{C}}}}{\arg\max} \left(\frac{\theta}{1-\theta}\right)^{|\mathcal{J}|} (1-\theta)^{|\mathcal{K}_{\mathcal{C}}|}. \end{aligned} \tag{4.6}$$

θ and $\mathcal{J}$ are fixed parameters in the estimation, so the first term in (4.6) is constant and can be eliminated from the problem. Since $0 \leq (1-\theta) < 1$ and $|\mathcal{K}_{\mathcal{C}}|$ is non-negative, the maximum is achieved when the exponent is minimized, yielding the desired result independent of Θ. □

We note that, even in the simplified case in Theorem 4.8, the computation of the ML estimate $\widehat{\mathcal{C}}$ of $\mathcal{C}$ according to Definition 4.6 requires an exhaustive search through the space

of $2^U - 1$ subsets of $\mathcal{U}$, as every subset $\mathcal{C} \subseteq \mathcal{U}$ must be considered in computing the arg max and arg min functions. Hence, the computation of an estimate $\widehat{\mathcal{C}}$ using maximum likelihood estimation is likely computationally infeasible, unlike maximum likelihood estimation of continuous parameters (e.g. Gaussian noise). We thus shift our attention to the use of heuristics to estimate $\mathcal{C}$.

4.5.3 Greedy Identification of Jammers - Θ Known

Instead of basing the identification of jammers on the probability $\Pr[\mathcal{C}|\mathcal{J}, \Theta]$ over subsets of $\mathcal{U}$, we base the identification on the probability $\Gamma(u|\mathcal{J}, \Theta)$ that a user $u \in \mathcal{U}$ is a compromised user in $\mathcal{C}$. This heuristic reduces the set estimation problem to a set membership estimation problem. We refer to the identification algorithm as GUIDE, for the **G**reedy **U**ser **IDE**ntification algorithm. We first address a version of the algorithm, referred to as GUIDE-Θ, for use when the parameter Θ is known to the TA.

Using the probability $\Gamma(u|\mathcal{J}, \Theta)$, we construct an estimate $\widehat{\mathcal{C}}$ of $\mathcal{C}$ using a greedy algorithm. The greedy algorithm GUIDE-Θ constructs $\widehat{\mathcal{C}}$ by adding users $u \in \mathcal{U}$ to $\widehat{\mathcal{C}}$ in decreasing order of probability $\Gamma(u|\mathcal{J}, \Theta)$ until $\widehat{\mathcal{C}}$ satisfies the condition $\mathcal{J} \subseteq \mathcal{K}_{\widehat{\mathcal{C}}}$, as given in Figure 4.3. We note that ties can be broken arbitrarily in the arg max function. Further, we note that instead of breaking ties, the GUIDE algorithm can be modified to select the subset $\mathcal{U}^* \subseteq \mathcal{U}$ of all users with the maximum value of $\Gamma(u|\mathcal{J}, \Theta)$. This technique and its implications are not addressed further in this dissertation.

The probability $\Gamma(u|\mathcal{J}, \Theta)$ for each $u \in \mathcal{U}$ is computed independent of other users in $\mathcal{U}$. In order to compute the desired probability, we define the vector random variable $S_u = (S_{0u}, \ldots, S_{(p-1)u})$ where $S_{iu} = |\mathcal{K}_{iu} \cap \mathcal{J}_i|$ for fixed $\mathcal{K}_{iu}$ and unknown or random $\mathcal{J}_i$. For fixed parameter θ_i, we let $P_{\mathcal{C},i}(s_{iu})$ denote the probability that $S_{iu} = s_{iu}$ given that $u \in \mathcal{C}$ and $P_{\mathcal{U}\setminus\mathcal{C},i}(s_{iu})$ denote the similar probability for $u \in \mathcal{U} \setminus \mathcal{C}$. We further let $P_{\mathcal{C}}(s_u)$ and $P_{\mathcal{U}\setminus\mathcal{C}}(s_u)$ denote the corresponding probabilities that $S_u = s_u$ for a given vector $s_u = (s_{0u}, \ldots, s_{(p-1)u})$ and fixed Θ.

GUIDE-Θ: Greedy Estimate of $\mathcal{C}$ given $\mathcal{J}, \Theta$

$\widehat{\mathcal{C}} \leftarrow \varnothing$

while $\mathcal{J} \nsubseteq \mathcal{K}_{\widehat{\mathcal{C}}}$ **do**

$u^* \leftarrow \arg\max_{u \in \mathcal{U} \setminus \widehat{\mathcal{C}}} \Gamma(u|\mathcal{J}, \Theta)$

$\widehat{\mathcal{C}} \leftarrow \widehat{\mathcal{C}} \cup \{u^*\}$

end while

Figure 4.3: The algorithm GUIDE-Θ constructs a greedy estimate $\widehat{\mathcal{C}}$ of the set $\mathcal{C}$ of compromised users using the jamming evidence $\mathcal{J}$ and parameter Θ.

Lemma 4.2. *The probability* $\Gamma(u|\mathcal{J}, \Theta)$ *can be expressed as*

$$\Gamma(u|\mathcal{J}, \Theta) = \frac{\prod_{i=0}^{p-1} P_{\mathcal{C},i}(s_{iu})}{\prod_{i=0}^{p-1} P_{\mathcal{C},i}(s_{iu}) + \prod_{i=0}^{p-1} P_{\mathcal{U}\setminus\mathcal{C},i}(s_{iu})}.$$

Proof. The keys in $\mathcal{J}_i$ for each time slot i that influence the probability $\Gamma(u|\mathcal{J}, \Theta)$ are only those keys in $\mathcal{K}_{iu} \cap \mathcal{J}_i$ held by u. Since keys are assigned independently and randomly, identification of compromised nodes depends only on the number of keys s_{iu} and not on the specific keys used, so $\Gamma(u|\mathcal{J}, \Theta) = \Gamma(u|s_u, \Theta)$. Using Bayes' Theorem [51] and noting that the event that $u \in \mathcal{C}$ is independent of the parameter Θ, we can express the probability $\Gamma(u|s_u, \Theta)$ as

$$\Gamma(u|s_u, \Theta) = \frac{P_{\mathcal{C}}(s_u) \Pr[u \in \mathcal{C}]}{P_{\mathcal{C}}(s_u) \Pr[u \in \mathcal{C}] + P_{\mathcal{U}\setminus\mathcal{C}}(s_u) \Pr[u \notin \mathcal{C}]}. \tag{4.7}$$

Since the TA has no prior information about $\mathcal{C}$, every user is equally likely to be compromised, and each possible non-empty set $\mathcal{C}$ is equally likely. This implies that the events $u \in \mathcal{C}$ and $u \notin \mathcal{C}$ are equally likely, so the corresponding factors in (4.7) cancel. Independence of the key assignment in different time slots implies that the probabilities $P_{\mathcal{C}}(s_u)$ and

$P_{\mathcal{U}\setminus\mathcal{C}}(s_u)$ can be factored as

$$P_{\mathcal{C}}(s_u) = \prod_{i=0}^{p-1} P_{\mathcal{C},i}(s_{iu}) \tag{4.8}$$

$$P_{\mathcal{U}\setminus\mathcal{C}}(s_u) = \prod_{i=0}^{p-1} P_{\mathcal{U}\setminus\mathcal{C},i}(s_{iu}), \tag{4.9}$$

yielding the desired result. □

To complete the evaluation necessary to perform the GUIDE-Θ algorithm to construct the estimate $\widehat{\mathcal{C}}$, we provide the following lemma to evaluate the probabilities $P_{\mathcal{C},i}(s)$ and $P_{\mathcal{U}\setminus\mathcal{C},i}(s)$.

Lemma 4.3. *The probabilities $P_{\mathcal{C},i}(s)$ and $P_{\mathcal{U}\setminus\mathcal{C},i}(s)$ are given by*

$$\begin{aligned} P_{\mathcal{C},i}(s) =& \binom{m_i}{s} \theta_i^s (1-\theta_i)^{m_i - s}, \\ P_{\mathcal{U}\setminus\mathcal{C},i}(s) =& 2^{1-U} \sum_{c=0}^{U-1} \binom{U-1}{c} p_{i,c}(s), \end{aligned}$$

where $p_{i,c}(s)$ is approximated by Lemma 4.1.

Proof. When $u \in \mathcal{C}$ is a compromised user, each key in $\mathcal{K}_{iu}$ appears in the set $\mathcal{J}_i$, and hence in the set $\mathcal{K}_{iu} \cap \mathcal{J}_i$, independently with probability θ_i, yielding

$$P_{\mathcal{C},i}(s) = \binom{m_i}{s} \theta_i^s (1-\theta_i)^{m_i - s}. \tag{4.10}$$

When $u \notin \mathcal{C}$, the desired probability is the probability that $|\mathcal{K}_{iu} \cap \mathcal{J}_i| = s$ given $u \in \mathcal{U} \setminus \mathcal{C}$. Conditioning this probability on the event that $|\mathcal{C}| = c$ yields the probability $p_{i,c}(s)$ approximated by Lemma 4.1. Since the TA has no prior information about $\mathcal{C}$, all non-empty subsets of $\mathcal{U} \setminus \{u\}$ are assumed to be equally likely, so $\Pr[|\mathcal{C}| = c | u \notin \mathcal{C}] = 2^{1-U}\binom{U-1}{c}$. □

4.5.4 Greedy Identification of Jammers - Θ Unknown

When the parameter Θ is unknown to the TA, the GUIDE-Θ algorithm cannot be used, as the probability $\Gamma(u|\mathcal{J}, \Theta)$ cannot be computed. Though it may be possible to construct an estimate $\widehat{\Theta}$ of Θ, we instead suggest replacing the probability $\Gamma(u|\mathcal{J}, \Theta)$ by the alternate

selection metric $\kappa_u = \sum_{i=0}^{p-1} s_{iu}$ for each user $u \in \mathcal{U}$. This choice of selection metric is intuitive as users with larger portions of the set of jamming evidence $\mathcal{J}$ should be more likely to appear as compromised users in $\widehat{\mathcal{C}}$. The following result qualifies this replacement as the selection metric.

Theorem 4.9. *The addition of users to $\widehat{\mathcal{C}}$ according to the variables κ_u for $u \in \mathcal{U}$ approximates the addition of users to $\widehat{\mathcal{C}}$ according to the probabilities $\Gamma(u|\mathcal{J}, \Theta)$. Furthermore, if $q_i = q$, $m_i = m$, $\theta_i = \theta$, and c is known, the ordering of $\mathcal{U}$ for the two sets of quantities is identical up to permutation of equal-valued users, suggesting that the quality of the approximation is inversely related to the parameter variation among the p key assignment schemes.*

Proof. From Lemma 4.2 and Lemma 4.3 with $s_{iu} = |\mathcal{K}_{iu} \cap \mathcal{J}_i|$ and $|\mathcal{C}| = c$ known, $\Gamma(u|\mathcal{J}, \Theta)$ is given by

$$\begin{aligned}\Gamma(u|\mathcal{J}, \Theta) &= \left(1 + \prod_{i=0}^{p-1} z_{i,c}^{s_{iu}} \left(\frac{1 - \theta_i z_{i,c}}{1 - \theta_i}\right)^{m_i - s_{iu}}\right)^{-1} \\ &= \left(1 + \alpha \prod_{i=0}^{p-1} \beta_i^{s_{iu}}\right)^{-1} \end{aligned} \tag{4.11}$$

where α and β_i are given by

$$\alpha = \prod_{i=0}^{p-1} \left(\frac{1 - \theta_i z_{i,c}}{1 - \theta_i}\right)^{m_i}, \tag{4.12}$$

$$\beta_i = \frac{1 - \theta_i}{z_{i,c}^{-1} - \theta_i}. \tag{4.13}$$

When $\beta_i = \beta$ for all $i = 0, \ldots, p-1$, as in the special case of $m_i = m$, $q_i = q$, and $\theta_i = \theta$ for all $i = 0, \ldots, p-1$, (4.11) can be simplified to

$$\Gamma(u|\mathcal{J}, \Theta) = \left(1 + \alpha \beta^{-\kappa_u}\right)^{-1}. \tag{4.14}$$

The expression in (4.14) is a monotone increasing function of κ_u. Hence, in this case, the orderings of $\mathcal{U}$ for the probabilities $\Gamma(u|\mathcal{J}, \Theta)$ and κ_u are identical up to permutation of equal-valued elements. □

GUIDE-κ: Greedy Estimate of $\mathcal{C}$ given $\mathcal{J}$

$\widehat{\mathcal{C}} \leftarrow \varnothing$

while $\mathcal{J} \notin \mathcal{K}_{\widehat{\mathcal{C}}}$ **do**

$u^* \leftarrow \arg\max_{u \in \mathcal{U} \setminus \widehat{\mathcal{C}}} \kappa_u$

$\widehat{\mathcal{C}} \leftarrow \widehat{\mathcal{C}} \cup \{u^*\}$

end while

Figure 4.4: The algorithm GUIDE-κ constructs a greedy estimate $\widehat{\mathcal{C}}$ of the set $\mathcal{C}$ of compromised users using the jamming evidence $\mathcal{J}$ and can be used when Θ is unknown to the TA.

The result of Theorem 4.9 suggests that an alternative to the GUIDE-Θ algorithm in Figure 4.3 is given by the GUIDE-κ algorithm in Figure 4.4. Moreover, the simplified GUIDE-κ algorithm in Figure 4.4 may be a suitable alternative to GUIDE-Θ even if Θ is known, as the required computation is greatly reduced.

4.5.5 Error in Identification of Compromised Users

In order to evaluate the heuristic estimation problem formulated for the identification of compromised users by the TA, we provide the following metrics of estimation error.

Definition 4.7. *The* false alarm rate $\mathcal{F}(c)$ *is the average fraction of jamming suspects in* $\widehat{\mathcal{C}}$ *which are not compromised users in* $\mathcal{C}$ *when* $|\mathcal{C}| = c$.

Definition 4.8. *The* miss rate $\mathcal{M}(c)$ *is the average fraction of compromised users in* $\mathcal{C}$ *which do not appear as jamming suspects in* $\widehat{\mathcal{C}}$ *when* $|\mathcal{C}| = c$.

The false alarm and miss rates in Definitions 4.7 and 4.8 are approximated using the following sequence of results. For added clarity, the estimation error is approximated with respect to the GUIDE-κ algorithm in Figure 4.4 using the quantities κ_u instead of the GUIDE-Θ algorithm in Figure 4.3 using the probabilities $\Gamma(u|\mathcal{J}, \Theta)$. We partition the set of random variables κ_u to distinguish between the compromised users in $\mathcal{C}$ and the remaining users in $\mathcal{U} \setminus \mathcal{C}$. The n^{th} largest κ_u values of users in $\mathcal{C}$, $\mathcal{U} \setminus \mathcal{C}$, and $\mathcal{U}$ are respectively denoted

$\kappa_{\mathcal{C}}^{(n)}$, $\kappa_{\mathcal{U}\setminus\mathcal{C}}^{(n)}$, and $\kappa_{\mathcal{U}}^{(n)}$. For clarity, let $\kappa_{\mathcal{U}\setminus\mathcal{C}}^{(n)} = \kappa_{\mathcal{U}}^{(n)} = \kappa_{\mathcal{C}}^{(n)} = \infty$ for $n = 0$, $\kappa_{\mathcal{C}}^{(n)} = 0$ for $n > c$, $\kappa_{\mathcal{U}\setminus\mathcal{C}}^{(n)} = 0$ for $n > U - c$, and $\kappa_{\mathcal{U}}^{(n)} = 0$ for $n > U$. The following lemma characterizes the distributions of the random variables κ_u and $\kappa^{(n)}$, and a proof is provided in Section 4.8.

Lemma 4.4. *For $A \in \{\mathcal{C},\mathcal{U},\mathcal{U}\setminus\mathcal{C}\}$, the probability $\Phi_A^{(n)}(\kappa|c) = \Pr\left[\kappa_A^{(n)} \geq \kappa \;\middle|\; |\mathcal{C}| = c\right]$ is computed as*

$$\Phi_A^{(n)}(\kappa|c) = \sum_{j=n}^{|A|} \binom{|A|}{j} \Phi_A(\kappa|c)^j \left(1 - \Phi_A(\kappa|c)\right)^{|A|-j},$$

from the probabilities $\Phi_A(\kappa|c) = \Pr\left[\kappa_A \geq \kappa \mid |\mathcal{C}| = c\right]$ given by

$$\begin{aligned}
\Phi_{\mathcal{C}}(\kappa|c) &= \sum_{k\geq\kappa} \left(P_{\mathcal{C},0} * \cdots * P_{\mathcal{C},p-1}\right)(k) \\
\Phi_{\mathcal{U}\setminus\mathcal{C}}(\kappa|c) &= \sum_{k\geq\kappa} \left(p_{0,c} * \cdots * p_{p-1,c}\right)(k) \\
\Phi_{\mathcal{U}}(\kappa|c) &= \frac{c}{U}\Phi_{\mathcal{C}}(\kappa|c) + \frac{U-c}{U}\Phi_{\mathcal{U}\setminus\mathcal{C}}(\kappa|c)
\end{aligned}$$

where $p_{i,c}$ is the probability distribution given by Lemma 4.1, $P_{\mathcal{C},i}$ is the probability distribution given by Lemma 4.3, and $*$ is the convolution operator for discrete probability distributions.

We note that when $|\widehat{\mathcal{C}}| = \hat{c}$ users appear as jamming suspects with $|\mathcal{C}| = c$ compromised users, the number of falsely accused users $F = |\widehat{\mathcal{C}} \setminus \mathcal{C}|$ and missed compromised users $M = |\mathcal{C} \setminus \widehat{\mathcal{C}}|$ satisfy $\hat{c} = c - M + F$. Hence, the false alarm and miss rates for fixed c are approximated by estimating the distribution of $\hat{c}$ given c and the distribution of F given $\hat{c}$ and c. The probability $p(\hat{c}|c) = \Pr\left[|\widehat{\mathcal{C}}| = \hat{c} \;\middle|\; |\mathcal{C}| = c\right]$ is first estimated using the following result, a proof of which can be found in Section 4.8.

Lemma 4.5. *The probability distribution $p(\hat{c}|c)$ of $|\widehat{\mathcal{C}}|$ given $|\mathcal{C}|$ is approximated as*

$$p(\hat{c}|c) \approx \sum_J P_{\mathcal{J}}(J,c) \frac{Q_{J,c}(J,\hat{c}) - Q_{J,c}(J,\hat{c}-1)}{1 - Q_{J,c}(J,\hat{c}-1)}$$

where $Q_{J,c}(L,\hat{c})$ and $P_{\mathcal{J}}(J,c)$ are defined recursively as

$$\begin{aligned}
Q_{J,c}(L,\hat{c}) =& \sum_{\kappa}\left(\Phi_{\mathcal{U}}^{(\hat{c})}(\kappa|c)-\Phi_{\mathcal{U}}^{(\hat{c})}(\kappa+1|c)\right)\\
&\times\sum_{n=0}^{\kappa}\nu_n Q_{J,c}(L-n,\hat{c}-1),\\
\nu_n =& \binom{\kappa}{n}\left(1-\frac{L-n}{J}\right)^{n}\left(\frac{L-n}{J}\right)^{\kappa-n},\\
P_{\mathcal{J}}(J,c) =& \left(P_{\mathcal{J}}^{0}(\cdot,c)\ast\cdots\ast P_{\mathcal{J}}^{p-1}(\cdot,c)\right)(J),\\
P_{\mathcal{J}}^{i}(J_i,c) =& \sum_{k=J_i}^{q_i}\binom{k}{J_i}\theta_i^{J_i}(1-\theta_i)^{k-J_i}P_{\mathcal{K}}^{i}(k,c),\\
P_{\mathcal{K}}^{i}(k,c) =& \sum_{n=0}^{m_i}\tau_n P_{\mathcal{K}}^{i}(k-n,c-1),\\
\tau_n =& \binom{m_i}{n}\left(1-\frac{k-n}{q_i}\right)^{n}\left(\frac{k-n}{q_i}\right)^{m_i-n},
\end{aligned}$$

where $\ast$ is the convolution operator for discrete probability distributions.

The probability distribution $p(F|\hat{c},c)=\Pr\left[|\widehat{\mathcal{C}}\setminus\mathcal{C}|=F \;\middle|\; |\widehat{\mathcal{C}}|=\hat{c},|\mathcal{C}|=c\right]$, characterizing the behavior of both F and M, is estimated using the following result, a proof of which can be found in Section 4.8.

Lemma 4.6. *The probability distribution $p(F|\hat{c},c)$ of the number of falsely accused users given $|\widehat{\mathcal{C}}|$ and $|\mathcal{C}|$ is approximated as*

$$\begin{aligned}
p(F|\hat{c},c)\approx& \sum_{\kappa_1,\kappa_2}\min\left(1,\frac{\Phi_{\mathcal{C}}^{(\hat{c}-F)}(\kappa_1|c)}{\Phi_{\mathcal{C}}^{(\hat{c}-F)}(\kappa_2|c)}\right)\\
&\times\min\left(1,\frac{\Phi_{\mathcal{U}\setminus\mathcal{C}}^{(F)}(\kappa_2|c)}{\Phi_{\mathcal{U}\setminus\mathcal{C}}^{(F)}(\kappa_1|c)}\right)\\
&\times\left(\Phi_{\mathcal{C}}^{(\hat{c}-F+1)}(\kappa_2|c)-\Phi_{\mathcal{C}}^{(\hat{c}-F+1)}(\kappa_2+1|c)\right)\\
&\times\left(\Phi_{\mathcal{U}\setminus\mathcal{C}}^{(F+1)}(\kappa_1|c)-\Phi_{\mathcal{U}\setminus\mathcal{C}}^{(F+1)}(\kappa_1+1|c)\right).
\end{aligned}$$

Theorem 4.10. *The false alarm rate $\mathcal{F}(c)$ is given by*

$$\mathcal{F}(c)=\sum_{\hat{c}=1}^{U}\frac{p(\hat{c}|c)}{\hat{c}}\sum_{F=0}^{\hat{c}}Fp(F|\hat{c},c),$$

where $p(\hat{c}|c)$ is approximated by Lemma 4.5 and $p(F|\hat{c},c)$ is approximated by Lemma 4.6.

Proof. Letting $\mathcal{E}[x]$ denote the expected value of x, Definition 4.7 suggests that

$$\mathcal{F}(c) = \mathcal{E}\left[\frac{|\widehat{\mathcal{C}} \setminus \mathcal{C}|}{|\widehat{\mathcal{C}}|} \;\middle|\; |\mathcal{C}| = c\right] \tag{4.15}$$

$$= \sum_{\hat{c}=1}^{U} p(\hat{c}|c)\mathcal{E}\left[\frac{|\widehat{\mathcal{C}} \setminus \mathcal{C}|}{|\widehat{\mathcal{C}}|} \;\middle|\; |\widehat{\mathcal{C}}| = \hat{c}, |\mathcal{C}| = c\right] \tag{4.16}$$

$$= \sum_{\hat{c}=1}^{U} \frac{p(\hat{c}|c)}{\hat{c}} \sum_{F=0}^{\hat{c}} F p(F|\hat{c}, c). \tag{4.17}$$

□

Theorem 4.11. *The miss rate $\mathcal{M}(c)$ is given by*

$$\mathcal{M}(c) = \frac{1}{c} \sum_{\hat{c}=1}^{U} p(\hat{c}|c) \sum_{F=0}^{\hat{c}} (c + F - \hat{c}) p(F|\hat{c}, c),$$

where $p(\hat{c}|c)$ is approximated by Lemma 4.5 and $p(F|\hat{c}, c)$ is approximated by Lemma 4.6.

Proof. This result follows from Theorem 4.10 and the relationship $\hat{c} = c - M + F$. □

4.6 Numerical Illustration and Design

In this section, we provide simulation results to illustrate design trade-offs, providing a basis for parameter selection in design of the system. We evaluate the metrics derived in Section 4.4 and 4.5 and discuss the effect of varying individual design parameter. We simulate the long-term performance of the system as a function of the identification interval of the TA as defined in Section 4.2.3.

4.6.1 Simulation Setup

We simulate a network of $U = 250$ users with varying parameter values of p, m_i, and q_i with the jamming probability $\theta_i = 0.9$. For each set of parameters p, m_i, and q_i, we randomly assign p sets of m_i control channel keys to each user from the p sets of q_i keys. For each value of c, the subset $\mathcal{C}$ is randomly selected from the set of users $\mathcal{U}$, and the subsets $\mathcal{J}_i$ of keys used for jamming are selected randomly using the parameter θ_i. For each subset $\mathcal{C}$ of size c, the resilience $r(c)$ is computed as the fraction of the $|\mathcal{U} \setminus \mathcal{C}|$ remaining users that can

access at least one control channel. Similarly, the average delay $\bar{d}(c)$, false alarm rate $\mathcal{F}(c)$, and miss rate $\mathcal{M}(c)$ are computed using the GUIDE-Θ algorithm based on the assigned keys, jammed control channels, and compromised users. Each data point in our simulation reflects an average over 100 simulated network and random key assignment instances. The results of the simulation study are illustrated in Figure 4.5 for four parameter sets. The solid and dashed lines in each plot represent the analytical results derived in Sections 4.4 and 4.5, and the symbol-marked points represent the results of the simulation study. As can be seen from Figure 4.5, the analytical results for the resilience $r(c)$ and the average delay $\bar{d}(c)$ coincide. While the analytical and simulation results for the false alarm rate $\mathcal{F}(c)$ and the miss rate $\mathcal{M}(c)$ disagree at individual values of c, the analytical results provide a reasonable approximation of the error behavior that can be expected.

4.6.2 Trade-offs in Key Assignment Parameters

We next identify and discuss design trade-offs in key assignment parameters by investigating the impact of individual parameters using the proposed evaluation metrics. We compare resource trade-offs with respect to the required key storage $\sum_{i=0}^{p-1} m_i$ for users in $\mathcal{U}$ and $\sum_{i=0}^{p-1} q_i$ for base stations in $\mathcal{B}$. We note that since each key corresponds to a unique control channel, the communication overhead for base stations is proportional to the base station key storage.

Time Slots per Reuse Period p

We first investigate the impact of varying the reuse period p. Intuitively, increasing the number of assigned key sets p increases the disparity of assigned keys between valid and compromised users, increasing the overall robustness to attacks. In terms of resilience, illustrated in Figure 4.5(a), the results of Theorems 4.1 and 4.4 validate this intuition, as the complement $(1 - r(c))$ of the resilience is a product of p probability terms. However, we note that key storage at each user and base station increases linearly with p. In addition, the average delay increases linearly with p, as seen in Theorem 4.6 and Figure 4.5(b). As seen in Figure 4.5(c) and 4.5(d), increasing p slightly improves the false alarm and miss

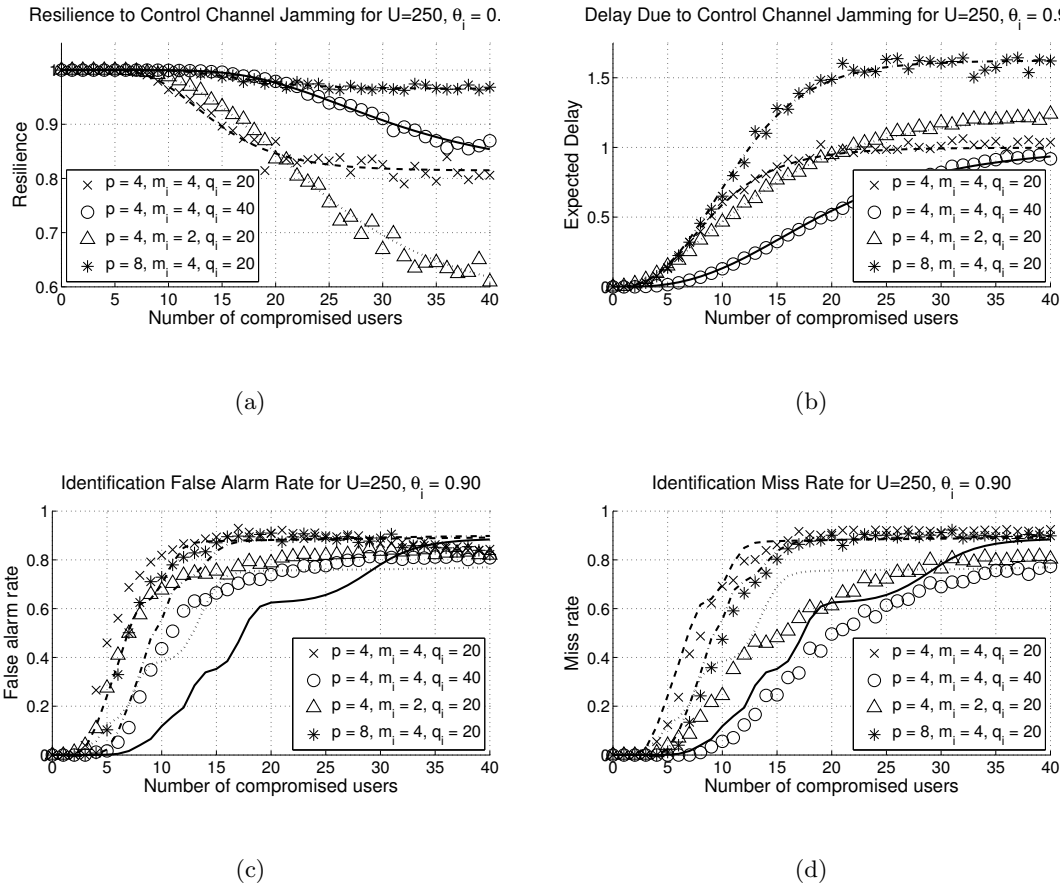


Figure 4.5: Variations in the (a) resilience $r(c)$, (b) expected delay $\bar{d}(c)$, (c) false alarm rate $\mathcal{F}(c)$, and (d) miss rate $\mathcal{M}(c)$ are illustrated for a network of $U = 250$ users with varying parameter values of p, m_i, and q_i and a jamming parameter of $\theta_i = 0.9$ using GUIDE-Θ. Solid and dashed lines represent analytical results derived in Sections 4.4 and 4.5, and symbol-marked points represent the simulated results averaged over 100 simulated network instances.

rates. The effect of varying p is illustrated by comparing the results in Figure 4.5 for $p = 4$, $m_i = 4$, and $q_i = 20$ to those of $p = 8$, $m_i = 4$, and $q_i = 20$.

Control Channels per Time Slot q_i

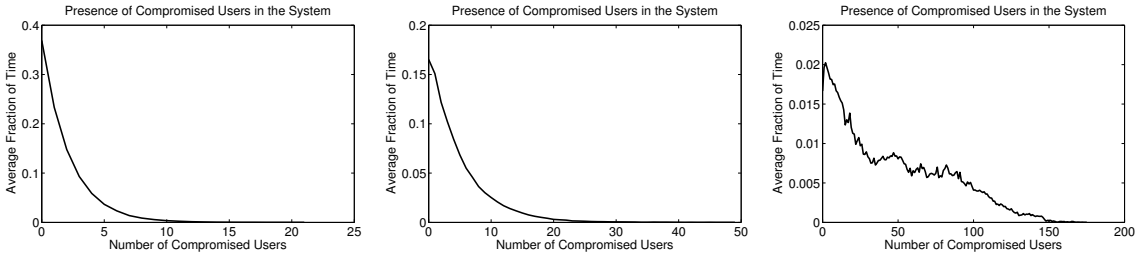
We next investigate the impact of varying the number of control channels q_i in each time slot i. Increasing the number of channels implies that users share fewer keys on average, leading to an improvement in robustness to attacks. The results of Theorems 4.1 and 4.4 illustrate an inverse dependence on each parameter q_i, suggesting that resilience and delay improve as q_i increases, as seen in Figure 4.5(a) and 4.5(b). Similarly, the identification capabilities of the TA improve with increasing q_i for small c because users are less likely to share keys, as seen in Figure 4.5(c) and 4.5(d). However, key storage at each base station increases linearly with q_i. The effect of varying q_i is illustrated by comparing the results in Figure 4.5 for $p = 4$, $m_i = 4$, and $q_i = 20$ to those of $p = 4$, $m_i = 4$, and $q_i = 40$.

Control Channel Keys per User per Time Slot m_i

We next investigate the impact of varying the number of control channel keys m_i assigned to each user in time slot i. Increasing m_i implies that users share more keys on average. However, the parameter m_i also appears as an exponential term in the slot resilience given by Theorem 4.1. Hence, the effect of varying m_i depends on the values of the parameters p and q_i, suggesting that there exist various trade-offs in varying m_i. The effect of varying m_i is illustrated by comparing the results in Figure 4.5 for $p = 4$, $m_i = 2$, and $q_i = 20$ to those of $p = 4$, $m_i = 4$, and $q_i = 20$.

Control Channels per Time Slot q_i and per User m_i

We note that a constant increase in both m_i and q_i allows the designer to take advantage of the increased exponent m_i in the slot resilience given by Theorem 4.1 without increasing the ratio m_i/q_i. Hence, increasing key storage at users and base stations by a constant leads to an improvement in resilience. The effect of jointly varying q_i and m_i is illustrated by comparing the results in Figure 4.5 for $p = 4$, $m_i = 2$, and $q_i = 20$ to those of $p = 4$,



(a) (b) (c)

Figure 4.6: Identification of compromised users is simulated using the GUIDE-κ algorithm with an adversary compromising users over an extended duration. Each figure plots the normalized histogram of the fraction of 1000 key reuse periods with a given number of compromised users present in the network. The system parameters are chosen as $p = 4$, $m_i = 4$, and $q_i = 20$, and the jamming parameter is chosen as $\theta = 0.9$. The average identification interval is chosen as (a) 5, (b), 10, and (c) 20 key reuse periods.

$m_i = 4$, and $q_i = 40$.

4.6.3 Trade-offs in Identification Interval of the TA

As illustrated in Figure 4.5(c) and Figure 4.5(d), the false alarm rate $\mathcal{F}(c)$ and miss rate $\mathcal{M}(c)$ are increasing functions of c when c is small. The performance of the identification process is thus improved if the TA can identify and eliminate the compromised users in the network when there are relatively few of them. Hence, by reducing the length of the identification interval and sampling the system more frequently, the TA can achieve decreased false alarm and miss rates. However, this performance gain comes with a necessary increase in computation and communication overhead for the TA due to the increased rate of collection of jamming evidence $\mathcal{J}$, execution of the GUIDE algorithm, and update of fresh control channel keys to remaining users.

Figure 4.6 illustrates the effect of the identification interval on the time-varying number of compromised users in the system using the following simulation setup. A new user is added to the network whenever a user is revoked, so the total number of users U remains constant. After each key reuse period of p time slots, a user in $\mathcal{U} \setminus \mathcal{C}$ is randomly selected

and added to $\mathcal{C}$ with probability 0.4. The identification rate determines the frequency with which the TA collects jamming evidence $\mathcal{J}$ and executes the GUIDE algorithm. Figure 4.6 plots the normalized histogram of the number of compromised users after each key reuse period to illustrate the long-term average of the number of compromised users in the system. The average identification interval, equal to the inverse of the identification rate, is chosen as 5 periods in Figure 4.6(a), 10 periods in Figure 4.6(b), and 20 periods in Figure 4.6(c). As illustrated, the number of compromised users tends to increase as the identification interval increases, eventually leading to cascading system failure where a majority of the users in the network are compromised.

4.7 Summary of Contributions

In this chapter, we addressed the mitigation of control channel jamming by malicious colluding insiders and compromised system users as well as the identification of compromised users without prior knowledge of the number of compromised users in the system. We mapped the problem of control channel access that is robust to jamming by compromised users to the problem of secure key establishment under node capture attacks. Based on the mapping, we proposed a framework for control channel access schemes using random key assignment. We proposed and evaluated metrics for resilience and delay which quantify the availability of control messages under control channel jamming attacks and demonstrated that the use of random key assignment provides graceful degradation in availability as the number of compromised users increases. We formulated the identification of compromised users in the system as a maximum likelihood estimation problem and proposed the GUIDE algorithms using greedy heuristics for jammer identification. We provided an analytical approximation to evaluate the false alarm and miss rates in the identification of compromised users resulting from the GUIDE algorithms. We discussed design trade-offs in the key assignment parameters and the identification interval used by the TA. In future work, we will investigate modifications to the adversary's jamming strategy and the effect on the availability of control messages and the ability to identify compromised users.

4.8 Appendix

4.8.1 Proof of Lemma 4.4

Proof. For each $A \in \{\mathcal{C}, \mathcal{U}, \mathcal{U} \setminus \mathcal{C}\}$, the derivation of $\Phi_A^{(n)}(\kappa|c)$ from $\Phi_A(\kappa|c)$ follows directly using order statistics [22] by noticing that at least n of the $|A|$ independent events are successful, and the success of each event occurs with probability $\Phi_A(\kappa|c)$. When $u \in \mathcal{C}$, the probability $\Phi_{\mathcal{C}}(\kappa|c)$ is the summation over $k \geq \kappa$ of the probability that $\kappa_u = k$. Since $\kappa_u = \sum_{i=0}^{p-1} s_{iu}$, the probability that $\kappa_u = k$ is given by the p-fold convolution evaluated at k of the probability distributions $P_{\mathcal{C},i}$ given by Lemma 4.3. When $u \in \mathcal{U} \setminus \mathcal{C}$, the corresponding probability is similarly given by the p-fold convolution evaluated at k of probability distributions $p_{i,c}$ given by Lemma 4.1. The probability $\Phi_{\mathcal{U}}(\kappa|c)$ is computed using the law of total probability and the fact that $\Pr[u \in \mathcal{C}] = c/U$ and $\Pr[u \in \mathcal{U} \setminus \mathcal{C}] = (U-c)/U$. □

4.8.2 Proof of Lemma 4.5

Proof. The probability $p(\hat{c}|c)$ is computed as the probability that the greedy algorithm stops after adding $\hat{c}$ users to $\widehat{\mathcal{C}}$ when $|\mathcal{C}| = c$. This is thus equivalent to the probability that $|\mathcal{K}_{\widehat{\mathcal{C}}} \cap \mathcal{J}| = |\mathcal{J}|$ when $|\mathcal{C}| = c$ given that $|\mathcal{K}_{\widehat{\mathcal{C}}\setminus\{u\}} \cap \mathcal{J}| < |\mathcal{J}|$ and $u \in \mathcal{U}$ is the $\hat{c}^{th}$ user added to $\widehat{\mathcal{C}}$. We condition on the event that $|\mathcal{J}| = \sum_{i=0}^{p-1} |\mathcal{J}_i| = J$. Letting $P_{\mathcal{J}}^i(J_i, c)$ denote the probability that $|\mathcal{J}_i| = J_i$, the probability $P_{\mathcal{J}}(J, c)$ that $|\mathcal{J}| = J$ is equal to the p-fold convolution evaluated at J of the probability distributions $P_{\mathcal{J}}^i(\cdot, c)$. The probability $P_{\mathcal{J}}^i(J_i, c)$ is equal to the summation over all $k \geq J_i$ of the probability that $|\mathcal{K}_{i\mathcal{C}}| = k$ multiplied by the probability that J_i of the k compromised keys are used for jamming, equal to $\binom{k}{J_i}\theta_i^{J_i}(1-\theta_i)^{k-J_i}$. The probability $P_{\mathcal{K}}^i(k, c)$ that $|\mathcal{K}_{i\mathcal{C}}| = k$ is computed recursively by counting the number of new keys recovered from each compromised user. Given that the first $(c-1)$ compromised users had $(k-n)$ of the q_i keys in $\mathcal{K}_i$, the probability that the m_i keys held by the c^{th} compromised user contain n new keys is $\binom{m_i}{n}\left(1 - \frac{k-n}{q_i}\right)^n \left(\frac{k-n}{q_i}\right)^{m_i - n}$.

The remaining probability of interest, denoted $p(\hat{c}|c, J)$, is thus the probability that $|\mathcal{K}_{\widehat{\mathcal{C}}} \cap \mathcal{J}| = J$ when $|\mathcal{C}| = c$ given that $|\mathcal{K}_{\widehat{\mathcal{C}}\setminus\{u\}} \cap \mathcal{J}| < J$ and $|\mathcal{J}| = J$. To compute this probability, we define $Q_{J,c}(L, \hat{c})$ as the probability that $|\mathcal{K}_{\widehat{\mathcal{C}}} \cap \mathcal{J}| = L$ given $|\mathcal{J}| = J$, $|\mathcal{C}| = c$,

and $|\widehat{\mathcal{C}}| = \hat{c}$. If we next condition on the event that $\kappa^{(\hat{c})} = \kappa$, summing over κ with weight $\Phi_{\mathcal{U}}^{(\hat{c})}(\kappa|c) - \Phi_{\mathcal{U}}^{(\hat{c})}(\kappa+1|c)$, this probability can be computed recursively in a similar way to that of $P_{\mathcal{K}}^{i}(k,c)$ above. Given that the first $(\hat{c}-1)$ identified users had $(L-n)$ of the J keys in $\mathcal{J}$, the probability that the κ keys held by the $\hat{c}^{th}$ identified user contain n new keys is $\binom{\kappa}{n}\left(1-\frac{L-n}{J}\right)^{n}\left(\frac{L-n}{J}\right)^{\kappa-n}$.

Since the desired probability $p(\hat{c}|c,J)$ is conditional on the event that $|\mathcal{K}_{\widehat{\mathcal{C}}\setminus\{u\}} \cap \mathcal{J}| < J$, and the probability $Q_{J,c}(J,\hat{c})$ is not, we use conditional probability to show the relationship as

$$Q_{J,c}(J,\hat{c}) = p(\hat{c}|c,J)\left(1 - Q_{J,c}(J,\hat{c}-1)\right) + Q_{J,c}(J,\hat{c}-1), \tag{4.18}$$

which is rearranged to yield the desired result. □

4.8.3 Proof of Lemma 4.6

Proof. Given $|\mathcal{C}| = c$ and $|\widehat{\mathcal{C}}| = \hat{c}$, the event that F users in $\mathcal{U}\setminus\mathcal{C}$ appear in $\widehat{\mathcal{C}}$ is exactly the intersection of the events

$$\kappa_{\mathcal{C}}^{(\hat{c}-F)} \geq \kappa_{\mathcal{U}\setminus\mathcal{C}}^{(F+1)} \text{ and } \kappa_{\mathcal{U}\setminus\mathcal{C}}^{(F)} \geq \kappa_{\mathcal{C}}^{(\hat{c}-F+1)}. \tag{4.19}$$

The probability $p(F|\hat{c},c)$ is thus the probability that both of these events occur. The desired probability is evaluated by conditioning on the independent events $\kappa_{\mathcal{U}\setminus\mathcal{C}}^{(F+1)} = \kappa_1$ and $\kappa_{\mathcal{C}}^{(\hat{c}-F+1)} = \kappa_2$. Combining these conditions with the events in (4.19) and the inequalities $\kappa_A^{(n)} \geq \kappa_A^{(n+1)}$ for $A \in \{\mathcal{C}, \mathcal{U}\setminus\mathcal{C}\}$ yields the probability $p(F|\hat{c},c)$ as

$$\begin{aligned} p(F|\hat{c},c) = \sum_{\kappa_1,\kappa_2} & \Pr\left[\kappa_{\mathcal{C}}^{(\hat{c}-F)} \geq \kappa_1 \mid \kappa_{\mathcal{C}}^{(\hat{c}-F)} \geq \kappa_2\right] \\ & \times \Pr\left[\kappa_{\mathcal{U}\setminus\mathcal{C}}^{(F)} \geq \kappa_2 \mid \kappa_{\mathcal{U}\setminus\mathcal{C}}^{(F)} \geq \kappa_1\right] \\ & \times \Phi_{\mathcal{U}\setminus\mathcal{C}}^{(F+1)}(\kappa_1|c)\Phi_{\mathcal{C}}^{(\hat{c}-F+1)}(\kappa_2|c), \end{aligned} \tag{4.20}$$

noting that all of the probabilities involved are conditional probabilities given $|\mathcal{C}| = c$. The first probability term in (4.20) is 1 when $\kappa_2 \geq \kappa_1$ and $\Phi_{\mathcal{C}}^{(\hat{c}-F)}(\kappa_1|c)/\Phi_{\mathcal{C}}^{(\hat{c}-F)}(\kappa_2|c)$ otherwise. Similarly, the second probability term in (4.20) is 1 when $\kappa_1 \geq \kappa_2$ and $\Phi_{\mathcal{U}\setminus\mathcal{C}}^{(F)}(\kappa_2|c)/\Phi_{\mathcal{U}\setminus\mathcal{C}}^{(F)}(\kappa_1|c)$ otherwise. □

Chapter 5

JAMMING-AWARE TRAFFIC ALLOCATION FOR MULTIPLE-PATH ROUTING

Jamming point-to-point transmissions in a wireless mesh network [3] or underwater acoustic network [71] can have debilitating effects on data transport through the network. The effects of jamming at the physical layer resonate through the protocol stack, providing an effective denial-of-service (DoS) attack [4] on end-to-end data communication. The simplest methods to defend a network against jamming attacks comprise physical layer solutions such as spread-spectrum or beamforming, forcing the jammers to expend a greater resource to reach the same goal. However, recent work has demonstrated that intelligent jammers can incorporate cross-layer protocol information into jamming attacks, reducing resource expenditure by several orders of magnitude by targeting certain link layer and MAC implementations [6, 74, 77] as well as link layer error detection and correction protocols [48]. Hence, more sophisticated anti-jamming methods and defensive measures must be incorporated into higher-layer protocols, for example channel surfing [79] or routing around jammed regions of the network [77].

The majority of anti-jamming techniques make use of diversity. For example, anti-jamming protocols may employ multiple frequency bands, different MAC channels, or multiple routing paths. Such diversity techniques help to curb the effects of the jamming attack by requiring the jammer to act on multiple resources simultaneously. In this dissertation, we consider the anti-jamming diversity based on the use of multiple routing paths. Using multiple-path variants of source routing protocols such as Dynamic Source Routing (DSR) [44] or Ad-Hoc On-Demand Distance Vector (AODV) [66], for example the MP-DSR protocol [47], each source node can request several routing paths to the destination node for concurrent use. To make effective use of this routing diversity, however, each source node must be able to make an intelligent allocation of traffic across the available paths while

considering the potential effect of jamming on the resulting data throughput.

In order to characterize the effect of jamming on throughput, each source must collect information on the impact of the jamming attack in various parts of the network. However, the extent of jamming at each network node depends on a number of unknown parameters, including the strategy used by the individual jammers and the relative location of the jammers with respect to each transmitter-receiver pair. Hence, *the impact of jamming is probabilistic from the perspective of the network*[1], and the characterization of the jamming impact is further complicated by the fact that the jammers' strategies may be dynamic and *the jammers themselves may be mobile.*

In order to capture the non-deterministic and dynamic effects of the jamming attack, we model the packet error rate at each network node as a random process. At a given time, the randomness in the packet error rate is due to the uncertainty in the jamming parameters, while the time-variability in the packet error rate is due to the jamming dynamics and mobility. Since the effect of jamming at each node is probabilistic, the end-to-end throughput achieved by each source-destination pair will also be non-deterministic and, hence, must be studied using a stochastic framework.

5.1 Our Contributions

In this chapter, we investigate the ability of network nodes to characterize the jamming impact and the ability of multiple source nodes to compensate for jamming in the allocation of traffic across multiple routing paths. We formulate the problem of allocating traffic across multiple routing paths in the presence of jamming as a lossy network flow optimization problem. We map the optimization problem to that of asset allocation using portfolio selection theory [11,52]. We formulate the centralized traffic allocation problem for multiple source nodes as a convex optimization problem.

We show that the multi-source multiple-path optimal traffic allocation can be computed at the source nodes using a distributed algorithm based on decomposition in network utility maximization (NUM) [55]. We propose methods which allow individual network nodes to

[1]We assume that the network does not make use of a jamming detection, localization, or tracking infrastructure

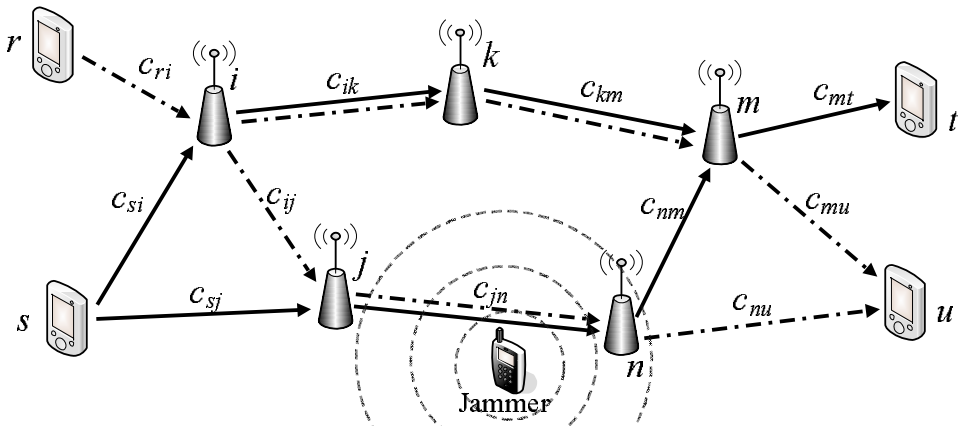


Figure 5.1: An example network with sources $\mathcal{S} = \{r, s\}$ is illustrated. Each unicast link $(i, j) \in \mathcal{E}$ is labeled with the corresponding link capacity.

locally characterize the jamming impact and aggregate this information for the source nodes. We demonstrate that the use of portfolio selection theory allows the data sources to balance the expected data throughput with the uncertainty in achievable traffic rates.

5.2 System Model and Assumptions

The wireless network of interest can be represented by a directed graph $G = (\mathcal{N}, \mathcal{E})$. The vertex set $\mathcal{N}$ represents the network nodes, and an ordered pair (i, j) of nodes is in the edge set $\mathcal{E}$ if and only if node j can receive packets directly from node i. We assume that all communication is unicast over the directed edges in $\mathcal{E}$, i.e. each packet transmitted by node $i \in \mathcal{N}$ is intended for a unique node $j \in \mathcal{N}$ with $(i, j) \in \mathcal{E}$. The maximum achievable data rate, or capacity, of each unicast link $(i, j) \in \mathcal{E}$ in the absence of jamming is denoted by the pre-determined constant rate c_{ij} in units of packets per second[2]. We further assume that jamming is the only factor leading to packet loss, in that network congestion and transmission errors are managed by the underlying network protocols.

Each source node s in a subset $\mathcal{S} \subseteq \mathcal{N}$ generates data for a single destination node

[2]We assume that this capacity is an available constant which corresponds to the maximum packet rate for reliable transport over each wireless link. We do not address the analysis or estimation of this link capacity parameter.

$d_s \in \mathcal{N}$. We assume that each source node s constructs multiple routing paths to d_s using a route request process similar to those of the DSR [44] or AODV [66] protocols. We let $\mathcal{P}_s = \{p_{s1}, \ldots, p_{sL_s}\}$ denote the collection of L_s loop-free routing paths for source s, noting that these paths need not be disjoint as in MP-DSR [47]. Representing each path $p_{s\ell}$ by a subset of directed link set $\mathcal{E}$, the sub-network of interest to source s is given by the directed subgraph

$$G_s = \left(\mathcal{N}_s = \bigcup_{\ell=1}^{L_s} \{j : (i,j) \in p_{s\ell}\}, \mathcal{E}_s = \bigcup_{\ell=1}^{L_s} p_{s\ell} \right)$$

of the graph G.

Figure 5.1 illustrates an example network with sources $\mathcal{S} = \{r, s\}$. The subgraph G_r consists of the two routing paths

$$p_{r1} = \{(r,i), (i,k), (k,m), (m,u)\}$$
$$p_{r2} = \{(r,i), (i,j), (j,n), (n,u)\},$$

and the subgraph G_s consists of the two routing paths

$$p_{s1} = \{(s,i), (i,k), (k,m), (m,t)\}$$
$$p_{s2} = \{(s,j), (j,n), (n,m), (m,t)\}.$$

In this chapter, we assume that the source nodes in $\mathcal{S}$ have no prior knowledge about the jamming attack being performed. That is, we make no assumption about the jammer's goals, method of attack, or mobility patterns. We assume that the number of jammers and their locations are unknown to the network nodes. Instead of relying on direct knowledge of the jammers, we suppose that the network nodes characterize the jamming impact in terms of the empirical packet delivery rate. Network nodes can then relay the relevant information to the source nodes in order to assist in optimal traffic allocation. Each time a new routing path is requested or an existing routing path is updated, the responding nodes along the path will relay the necessary parameters to the source node as part of the reply message for the routing path. Using the information from the routing reply, each source node s is thus provided with additional information about the jamming impact on the individual nodes.

5.3 Characterizing the Impact of Jamming

In this section, we propose techniques for the network nodes to estimate and characterize the impact of jamming and for a source node to incorporate these estimates into its traffic allocation. In order for a source node s to incorporate the jamming impact in the traffic allocation problem, the effect of jamming on transmissions over each link $(i, j) \in \mathcal{E}_s$ must be estimated and relayed to s. However, to capture the jammer mobility and the dynamic effects of the jamming attack, the local estimates need to be continually updated. We begin with an example to illustrate the possible effects of jammer mobility on the traffic allocation problem and motivate the use of continually updated local estimates.

5.3.1 Illustrating the Effect of Jammer Mobility on Network Throughput

Figure 5.2 illustrates a single-source network with three routing paths

$$p_1 = \{(s, x), (x, b), (b, d)\},\ p_2 = \{(s, y), (y, b), (b, d)\},\ \text{and } p_3 = \{(s, z), (z, b), (b, d)\}.$$

The label on each edge (i, j) is the link capacity c_{ij} indicating the maximum number of packets per second ($pkts/s$) which can be transported over the wireless link. In this example, we assume that the source is generating data at a rate of 300 $pkts/s$. In the absence of jamming, the source can continuously send 100 $pkts/s$ over each of the three paths, yielding a throughput rate equal to the source generation rate of 300 $pkts/s$. If a jammer near node x is transmitting at high power, the probability of successful packet reception, referred to as the *packet success rate*, over the link (s, x) drops to nearly zero, and the traffic flow to node d reduces to 200 $pkts/s$. If the source node becomes aware of this effect, the allocation of traffic can be changed to 150 $pkts/s$ on each of paths p_2 and p_3, thus recovering from the jamming attack at node x. However, this one-time re-allocation by the source node s does not adapt to the potential mobility of the jammer. If the jammer moves to node y, the packet success rate over (s, x) returns to one and that over (s, y) drops to zero, reducing the throughput to node d to 150 $pkts/s$, which is less than the 200 $pkts/s$ that would be achieved using the original allocation of 100 $pkts/s$ over each of the three paths. Hence, each node must relay an estimate of its packet success rate to the source node s and the

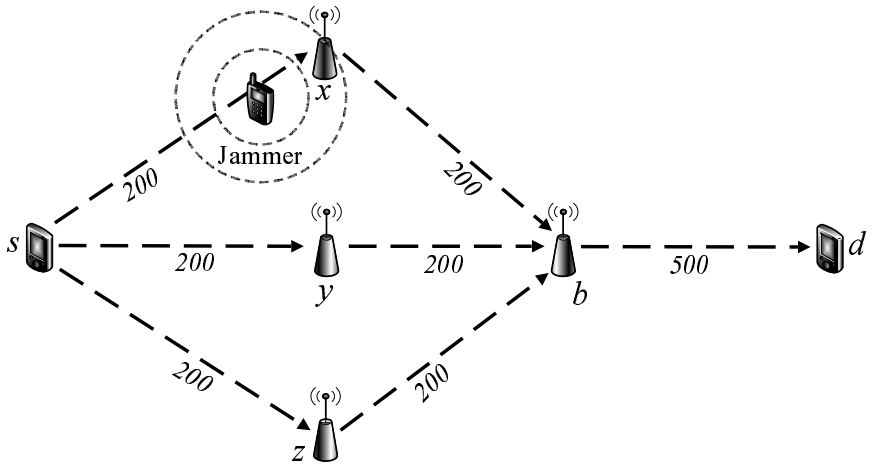


Figure 5.2: An example network that illustrates a single-source network with three routing paths. Each unicast link (i, j) is labeled with the corresponding link capacity c_{ij} in units of packets per second. The proximity of the jammer to nodes x and y impedes packet delivery over the corresponding paths, and the jammer mobility affects the allocation of traffic to the three paths as a function of time.

source must use this information to reallocate traffic in a timely fashion if the effect of the attack is to be mitigated. The relay of information from the nodes can be done periodically or at the instants when the packet success rates change significantly. These updates must be performed at a rate comparable to the rate of the jammer movement to provide an effective defense against the mobile jamming attack.

Next, suppose the jammer continually changes position between nodes x and y, causing the packet success rates over links (s, x) and (s, y) to oscillate between zero and one. This behavior introduces a high degree of variability into the observed packet success rates, leading to a less certain estimate of the future success rates over the links (s, x) and (s, y). However, since the packet success rate over link (s, z) has historically been more steady, it may be a more reliable option. Hence, the source s can choose to fill p_3 to its capacity and partition the remaining 100 $pkts/s$ equally over p_1 and p_2. This solution takes into account the historic variability in the packet success rates due to jamming mobility. In the following section, we build on this example, providing a set of parameters to be estimated by network nodes and methods for the sources to aggregate this information and characterize

the available paths on the basis of expected throughput.

5.3.2 Estimating Local Packet Success Rates

We let $x_{ij}(t)$ denote the packet success rate over link $(i,j) \in \mathcal{E}$ at time t, noting that $x_{ij}(t)$ can be computed analytically as a function of the transmitted signal power of node i, the signal power of the jammers, their relative distances from node j, and the path loss behavior of the wireless medium. In reality, however, the locations of mobile jammers are often unknown, and, hence, the use of such an analytical model is not applicable. Due to the uncertainty in the jamming impact, we model the packet success rate $x_{ij}(t)$ as a random process and allow the network nodes to collect empirical data in order to characterize the process. We suppose that each node j maintains an estimate $\mu_{ij}(t)$ of the packet success rate $x_{ij}(t)$ as well as a variance parameter $\sigma_{ij}^2(t)$ to characterize both the uncertainty in the estimate and the variability in the process[3] $x_{ij}(t)$.

We propose the use of a recursive update mechanism allowing each node j to periodically update the estimate $\mu_{ij}(t)$ as a function of time. As illustrated in Figure 5.3, we suppose that each node j updates the estimate $\mu_{ij}(t)$ after each *update period* of T seconds and relays the estimate to each relevant source node s after each *update relay period* of $T_s \gg T$ seconds. The shorter update period of T seconds allows each node j to characterize the variation in $x_{ij}(t)$ over the update relay period of T_s seconds, a key factor in determining the variance $\sigma_{ij}^2(t)$.

Similar to the approach in [79] which uses the packet delivery ratio (PDR) to detect jamming attacks, we use the observed PDR to compute the estimate $\mu_{ij}(t)$. During the update period represented by the time interval $[t-T,t]$, each node j can record the number $r_{ij}([t-T,t])$ of packets received over link (i,j) and the number $v_{ij}([t-T,t]) \leq r_{ij}([t-T,t])$ of valid packets which pass an error detection check[4]. The PDR over link (i,j) for the

[3] At a time instant t, the estimate $\mu_{ij}(t)$ and estimation variance $\sigma_{ij}^2(t)$ define a random variable describing the current view of the packet success rate. This random variable can be appropriately modeled as a beta random variable [30], though the results of this chapter do not require such an assumption.

[4] In the case of jamming attacks which prevent the receiving node j from detecting transmissions by node i, additional header information can be periodically exchanged between nodes i and j to achieve the convey the total number of transmissions, yielding the same overall effect.

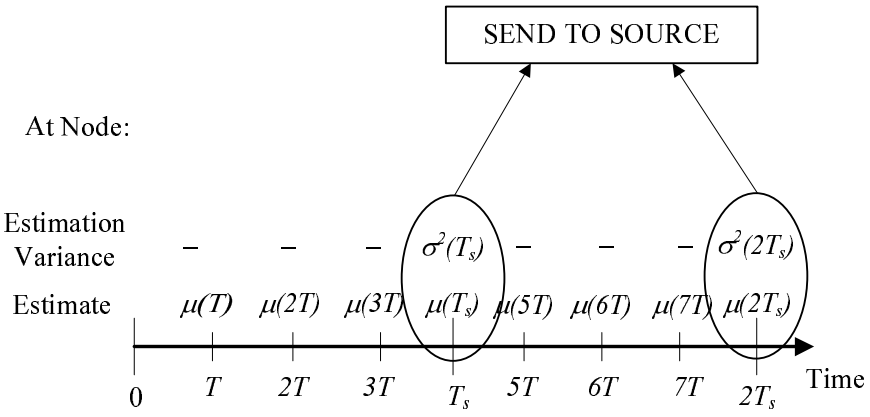


Figure 5.3: The estimation update process is illustrated for a single link. The estimate $\mu_{ij}(t)$ is updated every T seconds, and the estimation variance $\sigma^2_{ij}(t)$ is computed only every T_s seconds. Both values are relayed to relevant source nodes every T_s seconds.

update period $[t-T, t]$, denoted $PDR_{ij}([t-T, t])$, is thus equal to the ratio

$$PDR_{ij}([t-T,t]) = \frac{v_{ij}([t-T,t])}{r_{ij}([t-T,t])}. \tag{5.1}$$

This PDR can be used to update the estimate $\mu_{ij}(t)$ at the end of the update period. In order to prevent significant variation in the estimate $\mu_{ij}(t)$ and to include memory of the jamming attack history, we suggest using an exponential weighted moving average (EWMA) [64] to update the estimate $\mu_{ij}(t)$ as a function of the previous estimate $\mu_{ij}(t-T)$ as

$$\mu_{ij}(t) = \alpha\mu_{ij}(t-T) + (1-\alpha)PDR_{ij}([t-T,t]), \tag{5.2}$$

where $\alpha \in [0,1]$ is a constant weight indicating the relative preference between current and historic samples.

We use a similar EWMA process to update the variance $\sigma^2_{ij}(t)$ at the end of each update relay period of T_s seconds. Since this variance is intended to capture the variation in the packet success rate over the last T_s seconds, we consider the sample variance $V_{ij}([t-T_s, t])$ of the set of packet delivery ratios computed using (5.1) during the interval $[t-T_s, t]$ as

$$\begin{aligned} V_{ij}([t-T_s,t]) =& Var\left\{PDR_{ij}([t-kT, t-kT+T]) : \right. \\ & \left. k = 0, \ldots, \lceil T_s/T \rceil - 1\right\}. \end{aligned} \tag{5.3}$$

The estimation variance $\sigma_{ij}^2(t)$ is thus defined as a function of the previous variance $\sigma_{ij}^2(t-T_s)$ as

$$\sigma_{ij}^2(t) = \beta\sigma_{ij}^2(t - T_s) + (1 - \beta)V_{ij}([t - T_s, t]), \tag{5.4}$$

where $\beta \in [0, 1]$ is a constant weight similar to α in (5.2).

The EWMA method is widely used in sequential estimation processes, including estimation of the round-trip time (RTT) in TCP [58]. We note that the parameters α in (5.2) and β in (5.4) allow for design of the degree of historical content included in the parameter estimate updates, and these parameters can themselves be functions $\alpha(t)$ and $\beta(t)$ of time. For example, decreasing the parameter α allows the mean $\mu_{ij}(t)$ to change more rapidly with the PDR due to jammer mobility, and decreasing the parameter β allows the variance $\sigma_{ij}^2(t)$ to give more preference to variation in the most recent update relay period over historical variations. We further note that the update period T and update relay period T_s between subsequent updates of the parameter estimates have significant influence on the quality of the estimate. In particular, if the update period T_s is too large, the relayed estimates $\mu_{ij}(t)$ and $\sigma_{ij}(t)$ will be outdated before the subsequent update at time $t + T_s$. Furthermore, if the update period T at each node is too large, the dynamics of the jamming attack may be averaged out over the large number of samples $r_{ij}([t - T, t])$. The update periods T and T_s must thus be short enough to capture the dynamics of the jamming attack. However, decreasing the update period T_s between successive updates to the source node necessarily increases the communication overhead of the network. Hence, there exists a trade-off between performance and overhead in the choice of the update period T_s. We note that the design of the update relay period T_s depends on assumed path-loss and jammer mobility models. The application-specific tuning of the update relay period T_s is not further addressed in this dissertation.

Using the above formulation, each time a new routing path is requested or an existing routing path is updated, the nodes along the path will include the estimates $\mu_{ij}(t)$ and $\sigma_{ij}^2(t)$ as part of the reply message. In what follows, we show how the source node s uses these estimates to compute the end-to-end packet success rates over each path.

5.3.3 Estimating End-to-End Packet Success Rates

Given the packet success rate estimates $\mu_{ij}(t)$ and $\sigma_{ij}^2(t)$ for the links (i,j) in a routing path $p_{s\ell}$, the source s needs to estimate the effective end-to-end packet success rate to determine the optimal traffic allocation. Assuming the total time required to transport packets from each source s to the corresponding destination d_s is negligible compared to the update relay period T_s, we drop the time index and address the end-to-end packet success rates in terms of the estimates μ_{ij} and σ_{ij}^2. The end-to-end packet success rate $y_{s\ell}$ for path $p_{s\ell}$ can be expressed as the product

$$y_{s\ell} = \prod_{(i,j)\in p_{s\ell}} x_{ij}, \tag{5.5}$$

which is itself a random variable[5] due to the randomness in each x_{ij}. We let $\gamma_{s\ell}$ denote the expected value of $y_{s\ell}$ and $\omega_{s\ell m}$ denote the covariance of $y_{s\ell}$ and y_{sm} for paths $p_{s\ell}, p_{sm} \in \mathcal{P}_s$. Due to the difficulty in inferring correlation between estimated random variables, we let the source node s assume the packet success rates x_{ij} as mutually independent, even though they are likely correlated. Under this independence assumption, the mean $\gamma_{s\ell}$ of $y_{s\ell}$ given in (5.5) is equal to the product of estimates μ_{ij} as

$$\gamma_{s\ell} = \prod_{(i,j)\in p_{s\ell}} \mu_{ij}, \tag{5.6}$$

and the covariance $\omega_{s\ell m} = E[y_{s\ell}y_{sm}] - E[y_{s\ell}]E[y_{sm}]$ is similarly given by

$$\omega_{s\ell m} = \prod_{(i,j)\in p_{s\ell}\oplus p_{sm}} \mu_{ij} \prod_{(i,j)\in p_{s\ell}\cap p_{sm}} \left(\sigma_{ij}^2 + \mu_{ij}^2\right) - \gamma_{s\ell}\gamma_{sm}. \tag{5.7}$$

In (5.7), $\oplus$ denotes the exclusive-OR set operator such that an element is in $A \oplus B$ if it is in either A or B but not both. The covariance formula in (5.7) reflects the fact that the end-to-end packet success rates $y_{s\ell}$ and y_{sm} of paths $p_{s\ell}$ and p_{sm} with shared links are correlated even when the rates x_{ij} are independent. We note that the variance $\omega_{s\ell}^2$ of the end-to-end rate $y_{s\ell}$ can be computed using (5.7) with $\ell = m$.

Let $\boldsymbol{\gamma}_s$ denote the $L_s \times 1$ vector of estimated end-to-end packet success rates $\gamma_{s\ell}$ computed using (5.6), and let $\boldsymbol{\Omega}_s$ denote the $L_s \times L_s$ covariance matrix with (ℓ, m) entry $\omega_{s\ell m}$ computed

[5] If the x_{ij} are modeled as beta random variables, the product $y_{s\ell}$ is well-approximated by a beta random variable [42].

using (5.7). The estimate pair $(\boldsymbol{\gamma}_s, \boldsymbol{\Omega}_s)$ provides the sufficient statistical characterization of the end-to-end packet success rates for source s to allocate traffic to the paths in $\mathcal{P}_s$. Furthermore, the off-diagonal elements in $\boldsymbol{\Omega}_s$ denote the extent of mutual overlap between the paths in $\mathcal{P}_s$.

5.4 Optimal Jamming-Aware Traffic Allocation

In this section, we present an optimization framework for jamming-aware traffic allocation to multiple routing paths in $\mathcal{P}_s$ for each source node $s \in \mathcal{S}$. We develop a set of constraints imposed on traffic allocation solutions and then formulate a utility function for optimal traffic allocation by mapping the problem to that of portfolio selection in finance. Letting $\phi_{s\ell}$ denote the traffic rate allocated to path $p_{s\ell}$ by the source node s, the problem of interest is thus for each source s to determine the optimal $L_s \times 1$ rate allocation vector $\boldsymbol{\phi}_s$ subject to network flow capacity constraints using the available statistics $\boldsymbol{\gamma}_s$ and $\boldsymbol{\Omega}_s$ of the end-to-end packet success rates under jamming.

5.4.1 Traffic Allocation Constraints

In order to define a set of constraints for the multiple-path traffic allocation problem, we must consider the source data rate constraints, the link capacity constraints, and the reduction of traffic flow due to jamming at intermediate nodes. The traffic rate allocation vector $\boldsymbol{\phi}_s$ is trivially constrained to the non-negative orthant, i.e. $\boldsymbol{\phi}_s \geq \mathbf{0}$, as traffic rates are non-negative. Assuming data generation at source s is limited to a maximum data rate R_s, the rate allocation vector is also constrained as $\mathbf{1}^T \boldsymbol{\phi}_s \leq R_s$. These non-negativity and data rate constraints define the convex space $\boldsymbol{\Phi}_s$ of feasible allocation vectors $\boldsymbol{\phi}_s$ characterizing rate allocation solutions for source s.

Due to jamming at nodes along the path, the traffic rate is potentially reduced at each receiving node as packets are lost. Hence, while the initial rate of $\phi_{s\ell}$ is allocated to the path, the residual traffic rate forwarded by node i along the path $p_{s\ell}$ may be less than $\phi_{s\ell}$. Letting $p_{s\ell}^{(i)}$ denote the sub-path of $p_{s\ell}$ from source s to the intermediate node i, the residual traffic rate forwarded by node i is given by $y_{s\ell}^{(i)} \phi_{s\ell}$, where $y_{s\ell}^{(i)}$ is computed using (5.5) with $p_{s\ell}$ replaced by the sub-path $p_{s\ell}^{(i)}$.

The capacity constraint on the total traffic traversing a link (i,j) thus imposes the stochastic constraint

$$\sum_{s\in\mathcal{S}}\sum_{\ell:(i,j)\in p_{s\ell}}\phi_{s\ell}y_{s\ell}^{(i)}\leq c_{ij} \tag{5.8}$$

on the feasible allocation vectors $\boldsymbol{\phi}_s$. To compensate for the randomness in the capacity constraint in (5.8), we replace the residual packet success rate $y_{s\ell}^{(i)}$ with a function of its expected value and variance. The mean $\gamma_{s\ell}^{(i)}$ and variance $(\omega_{s\ell}^{(i)})^2$ of $y_{s\ell}^{(i)}$ can be computed using (5.6) and (5.7), respectively, with $p_{s\ell}$ replaced by the sub-path $p_{s\ell}^{(i)}$. We thus replace $y_{s\ell}^{(i)}$ in (5.8) with the statistic $\gamma_{s\ell}^{(i)}+\delta\omega_{s\ell}^{(i)}$, where $\delta\geq 0$ is a constant which can be tuned based on tolerance to delay resulting from capacity violations[6]. We let $\boldsymbol{W}_s$ denote the $|\mathcal{E}|\times L_s$ *weighted link-path incidence matrix* for source s with rows indexed by links (i,j) and columns indexed by paths $p_{s\ell}$. The element $w((i,j),p_{s\ell})$ in row (i,j) and column $p_{s\ell}$ of $\boldsymbol{W}_s$ is thus given by

$$w\left((i,j),p_{s\ell}\right)=\begin{cases}\min\left\{1,\gamma_{s\ell}^{(i)}+\delta\omega_{s\ell}^{(i)}\right\}, & \text{if } (i,j)\in p_{s\ell}\\ 0, & \text{otherwise.}\end{cases} \tag{5.9}$$

Letting $\boldsymbol{c}$ denote the $|\mathcal{E}|\times 1$ vector of link capacities c_{ij} for $(i,j)\in\mathcal{E}$, the link capacity constraint in (5.8) including expected packet loss due to jamming can be expressed by the vector inequality

$$\sum_{s\in\mathcal{S}}\boldsymbol{W}_s\boldsymbol{\phi}_s\leq\boldsymbol{c}, \tag{5.10}$$

which is a linear constraint in the variable $\boldsymbol{\phi}_s$. We note that this statistical constraint formulation generalizes the standard network flow capacity constraint corresponding to the case of $x_{ij}=1$ for all $(i,j)\in\mathcal{E}$ in which the incidence matrix $\boldsymbol{W}_s$ is deterministic and binary.

5.4.2 Optimal Traffic Allocation Using Portfolio Selection Theory

In order to determine the optimal allocation of traffic to the paths in $\mathcal{P}_s$, each source s chooses a utility function $U_s(\boldsymbol{\phi}_s)$ that evaluates the total data rate, or throughput, suc-

[6]The case of $\delta=0$ corresponds to the average-case constraint and will lead to increased queueing delay whenever $y_{s\ell}^{(i)}>\gamma_{s\ell}^{(i)}$. Increasing the value of δ improves the robustness to variations around the mean but decreases the amount of traffic which can be allocated to the corresponding path.

Table 5.1: The mapping is illustrated between the financial portfolio selection problem and the multiple-path traffic allocation problem.

Portfolio Selection	**Traffic Allocation**
Funds to be invested	Source data rate R_s
Financial assets	Routing paths $\mathcal{P}_s$
Expected asset return	Expected packet success rate $\gamma_{s\ell}$
Investment portfolio	Traffic allocation $\boldsymbol{\phi}_s$
Portfolio return	Mean throughput $\boldsymbol{\gamma}_s^T \boldsymbol{\phi}_s$
Portfolio risk	Estimation variance $\boldsymbol{\phi}_s^T \boldsymbol{\Omega}_s \boldsymbol{\phi}_s$

cessfully delivered to the destination node d_s. In defining our utility function $U_s(\boldsymbol{\phi}_s)$, we present an analogy between traffic allocation to routing paths and allocation of funds to correlated assets in finance.

In Markowitz's portfolio selection theory [11, 52], an investor is interested in allocating funds to a set of financial assets that have uncertain future performance. The expected performance of each investment at the time of the initial allocation is expressed in terms of return and risk. The return on the asset corresponds to the value of the asset and measures the growth of the investment. The risk of the asset corresponds to the variance in the value of the asset and measures the degree of variation or uncertainty in the investment's growth.

We describe the desired analogy by mapping this allocation of funds to financial assets to the allocation of traffic to routing paths. We relate the expected investment return on the financial portfolio to the estimated end-to-end success rates $\boldsymbol{\gamma}_s$ and the investment risk of the portfolio to the estimated success rate covariance matrix $\boldsymbol{\Omega}_s$. We note that the correlation between related assets in the financial portfolio corresponds to the correlation between non-disjoint routing paths. The analogy between financial portfolio selection and the allocation of traffic to routing paths is summarized in Table 5.1.

As in Markowitz's theory, we define a constant *risk-aversion factor* $k_s \geq 0$ for source $s \in \mathcal{S}$ to indicate the preference for source s to allocate resources to less risky paths with lower throughput variance. This risk-aversion constant weighs the trade-off between expected throughput and estimation variance. We note that each source s can choose a different risk-aversion factor, and a source may vary the risk-aversion factor k_s with time or for different

Optimal Jamming-Aware Traffic Allocation

$$\begin{aligned}
\boldsymbol{\phi}^* = \quad & \arg\max_{\{\boldsymbol{\phi}_s\}} \sum_{s\in\mathcal{S}} \boldsymbol{\gamma}_s^T \boldsymbol{\phi}_s - k_s \boldsymbol{\phi}_s^T \boldsymbol{\Omega}_s \boldsymbol{\phi}_s \\
\text{s.t.} \quad & \sum_{s\in\mathcal{S}} \boldsymbol{W}_s \boldsymbol{\phi}_s \leq \boldsymbol{c} \\
& \mathbf{1}^T \boldsymbol{\phi}_s \leq R_s \text{ for all } s \in \mathcal{S}, \\
& \mathbf{0} \leq \boldsymbol{\phi}_s \text{ for all } s \in \mathcal{S}.
\end{aligned}$$

Figure 5.4: The jamming-aware multiple-path traffic allocation problem is formulated as a convex optimization problem.

types of data. For a given traffic rate allocation vector $\boldsymbol{\phi}_s$, the expected total throughput for source s is equal to the vector inner product $\boldsymbol{\gamma}_s^T \boldsymbol{\phi}_s$. The corresponding variance in the throughput for source s due to the uncertainty in the estimate $\boldsymbol{\gamma}_s$ is equal to the quadratic term $\boldsymbol{\phi}_s^T \boldsymbol{\Omega}_s \boldsymbol{\phi}_s$. Based on the above analogy making use of the mappint in Table 5.1 to portfolio selection theory, we define the utility function $U_s(\boldsymbol{\phi}_s)$ at source s as the weighted sum

$$U_s(\boldsymbol{\phi}_s) = \boldsymbol{\gamma}_s^T \boldsymbol{\phi}_s - k_s \boldsymbol{\phi}_s^T \boldsymbol{\Omega}_s \boldsymbol{\phi}_s. \tag{5.11}$$

Setting the risk-aversion factor k_s to zero indicates that the source s is willing to put up with any amount of uncertainty in the estimate $\boldsymbol{\gamma}_s$ of the end-to-end success rates to maximize the expected throughput. The role of the risk-aversion factor is thus to impose a penalty on the objective function proportional to the uncertainty in the estimation process, potentially narrowing the gap between expected throughput and achieved throughput. The cases of $k_s = 0$ and $k_s > 0$ are compared in detail in Section 5.5.

Combining the utility function in (5.11) with the set of constraints defined in Section 5.4.1 yields the jamming-aware traffic allocation optimization problem given in Figure 5.4 which aims to find the globally optimal traffic allocation over the set $\mathcal{S}$ of sources. Since the use of centralized protocols for source routing may be undesirable due to excessive communication overhead in large-scale wireless networks, we seek a distributed formulation for the optimal traffic allocation problem in Figure 5.4.

5.4.3 Optimal Distributed Traffic Allocation using NUM

In the distributed formulation of the algorithm, each source s determines its own traffic allocation $\boldsymbol{\phi}_s$, ideally with minimal message passing between sources. By inspection, we see that the optimal jamming-aware flow allocation problem in Figure 5.4 is similar to the network utility maximization (NUM) formulation of the basic maximum network flow problem [55]. We thus develop a distributed traffic allocation algorithm using Lagrangian dual decomposition techniques [55] for the NUM problem.

The dual decomposition technique is derived by decoupling the capacity constraint in (5.10) and introducing the *link prices* λ_{ij} corresponding to each link (i, j). Letting $\boldsymbol{\lambda}$ denote the $|\mathcal{E}| \times 1$ vector of link prices λ_{ij}, the Lagrangian $L(\boldsymbol{\phi}, \boldsymbol{\lambda})$ of the optimization problem in Figure 5.4 is given by

$$L(\boldsymbol{\phi}, \boldsymbol{\lambda}) = \sum_{s \in \mathcal{S}} \boldsymbol{\gamma}_s^T \boldsymbol{\phi}_s - k_s \boldsymbol{\phi}_s^T \boldsymbol{\Omega}_s \boldsymbol{\phi}_s + \boldsymbol{\lambda}^T \left(\boldsymbol{c} - \sum_{s \in \mathcal{S}} \boldsymbol{W}_s \boldsymbol{\phi}_s \right). \tag{5.12}$$

The distributed optimization problem is solved iteratively using the Lagrangian dual method as follows. For a given set of link prices $\boldsymbol{\lambda}_n$ at iteration n, each source s solves the local optimization problem

$$\boldsymbol{\phi}_{s,n}^* = \underset{\boldsymbol{\phi}_s \in \boldsymbol{\Phi}_s}{\arg\max} \left(\boldsymbol{\gamma}_s^T - \boldsymbol{\lambda}_n^T \boldsymbol{W}_s \right) \boldsymbol{\phi}_s - k_s \boldsymbol{\phi}_s^T \boldsymbol{\Omega}_s \boldsymbol{\phi}_s. \tag{5.13}$$

The link prices $\boldsymbol{\lambda}_{n+1}$ are then updated using a gradient descent iteration as

$$\boldsymbol{\lambda}_{n+1} = \left(\boldsymbol{\lambda}_n - a \left(\boldsymbol{c} - \sum_{s \in \mathcal{S}} \boldsymbol{W}_s \boldsymbol{\phi}_{s,n}^* \right) \right)^+, \tag{5.14}$$

where $a > 0$ is a constant step size and $(\boldsymbol{v})^+ = \max(\boldsymbol{0}, \boldsymbol{v})$ is the element-wise projection into the non-negative orthant. In order to perform the local update in (5.14), sources must exchange information about the result of the local optimization step. Since updating the link prices $\boldsymbol{\lambda}$ depends only on the expected link usage, sources must only exchange the $|\mathcal{E}| \times 1$ link usage vectors $\boldsymbol{u}_{s,n} = \boldsymbol{W}_s \boldsymbol{\phi}_{s,n}^*$ to ensure that the link prices are consistently updated across all sources. The iterative optimization step can be repeated until the allocation vectors $\boldsymbol{\phi}_s$ converge[7] for all sources $s \in \mathcal{S}$, i.e. when $\|\boldsymbol{\phi}_{s,n}^* - \boldsymbol{\phi}_{s,n-1}^*\| \leq \epsilon$ for all s with a given

[7] In order to prevent premature termination at a local minimum, sources could additionally exchange a flag f_s indicating whether or not local convergence has been attained such that all sources continue to iterate until all convergence flags have been set.

Distributed Jamming-Aware Traffic Allocation

Initialize $n = 1$ with initial link prices $\boldsymbol{\lambda}_1$.

1. Each source s independently computes
$$\boldsymbol{\phi}^*_{s,n} = \arg\max_{\boldsymbol{\phi}_s \in \boldsymbol{\Phi}_s} \left(\boldsymbol{\gamma}_s^T - \boldsymbol{\lambda}_n^T \boldsymbol{W}_s\right) \boldsymbol{\phi}_s - k_s \boldsymbol{\phi}_s^T \boldsymbol{\Omega}_s \boldsymbol{\phi}_s.$$
2. Sources exchange the link usage vectors $\boldsymbol{u}_{s,n} = \boldsymbol{W}_s \boldsymbol{\phi}^*_{s,n}$.
3. Each source locally updates link prices as $\boldsymbol{\lambda}_{n+1} = \left(\boldsymbol{\lambda}_n - a\left(\boldsymbol{c} - \sum_{s \in \mathcal{S}} \boldsymbol{u}_{s,n}\right)\right)^+$.
4. If $\|\boldsymbol{\phi}^*_{s,n} - \boldsymbol{\phi}^*_{s,n-1}\| > \epsilon$ for any s, increment n and go to step 1.

Figure 5.5: A distributed algorithm to solve the jamming-aware multiple-path traffic allocation problem is presented.

$\epsilon > 0$. The above approach yields the distributed algorithm for optimal jamming-aware flow allocation given in Figure 5.5.

Given the centralized optimization problem in Figure 5.4 and the distributed formulation for jamming-aware traffic allocation in Figure 5.5, a set of sources with estimated parameters $\boldsymbol{\gamma}_s$ and $\boldsymbol{\Omega}_s$ describing the effect of a jamming attack can proactively compensate for the presence of jamming on network traffic flow.

5.4.4 Computational Complexity

We note that both the centralized optimization problem in Figure 5.4 and the local optimization step in the distributed algorithm in Figure 5.5 are quadratic programming optimization problems with linear constraints [11]. The computational time required for solving these problems using numerical methods for quadratic programming is a polynomial function of the number of optimization variables and the number of constraints.

In the centralized problem, there are $\sum_{s \in \mathcal{S}} |\mathcal{P}_s|$ optimization variables corresponding to the number of paths available to each of the sources. The number of constraints in the centralized problem is equal to the total number of links $|\bigcup_{s \in \mathcal{S}} \mathcal{E}_s|$, corresponding to the number of link capacity constraints. In the distributed algorithm, each source iteratively solves a local optimization problem, leading to $|\mathcal{S}|$ decoupled optimization problems. Each

of these problems has $|\mathcal{P}_s|$ optimization variables and $|\mathcal{E}_s|$ constraints. Hence, as the number of sources in the network increases, the distributed algorithm may be advantageous in terms of total computation time. In what follows, we provide a detailed performance evaluation of the methods proposed in this chapter.

5.5 Performance Evaluation

In this section, we simulate various aspects of the proposed techniques for estimation of jamming impact and jamming-aware traffic allocation. We first describe the simulation setup, including descriptions of the assumed models for routing path construction, jammer mobility, packet success rates, and estimate updates. We then simulate the process of computing the estimation statistics $\mu_{ij}(t)$ and $\sigma^2_{ij}(t)$ for a single link (i, j). Next, we illustrate the effects of the estimation process on the throughput optimization, both in terms of optimization objective functions and the resulting simulated throughput. Finally, we simulate a small-scale network similar to that in Figure 5.2 while varying network and protocol parameters in order to observe performance trends.

5.5.1 Simulation Setup

The simulation results presented herein are obtained using the following simulation setup. A network of nodes is deployed randomly over an area, and links are formed between pairs of nodes within a fixed communication range. The set $\mathcal{S}$ of source nodes is chosen randomly, and the destination node d_s corresponding to each source $s \in \mathcal{S}$ is randomly chosen from within the connected component containing s. Each routing path in the set $\mathcal{P}_s$ is chosen using a randomized geometric routing algorithm which chooses the next hop toward the destination d_s from the set of neighboring nodes that are closer to d_s in terms of either distance or hop-count. Nodes transmit using fixed power P_t.

We simulate the case of continuous jamming at a fixed power P_j using omnidirectional antennas. The mobility of each jammer j consists of repeatedly choosing a random direction $\theta_j \in [0, 2\pi)$ and a random speed $v_j \in [0, V_{\max}]$ and moving for a random amount of time $\tau_j > 0$ at the chosen direction and speed. At each instant in time, the packet error rate is a function of the transmission powers P_t and P_j, the distance d_{tr} from the transmitter to the

Table 5.2: We provide a summary of the parameters used to simulate jamming-aware multiple-path traffic allocation.

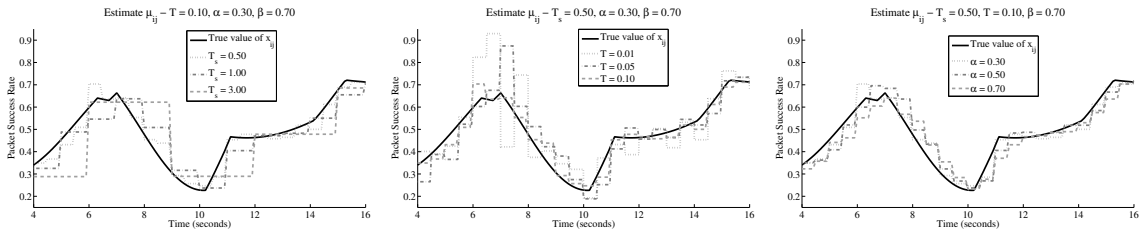
Parameter	Value
Network area	$100\ m \times 100\ m$
Radio range	$20\ m$
Number of sources	$\lvert\mathcal{S}\rvert = 1$
Number of nodes	$\lvert\mathcal{N}\rvert = 100$
Maximum number of paths	$\lvert\mathcal{P}_s\rvert \leq 5$
Transmission power	$P_t = 20\ mW$
Link capacity	$c_{ij} = 1000\ pkts/s$
Jamming transmission power	$P_j = 50\ mW$
Maximum jammer mobility speed	$V_{\max} = 5\ m/s$
Packet error rate parameter	$\xi = 1.1$
Path-loss constant	$\rho = 2.5 \times 10^{-4}$
Path-loss exponent	$\nu = 2.7$
Receiver noise	$N = 1\ nW$
EWMA coefficients	$\alpha = 0.7$, $\beta = 0.3$
Update period	$T = 0.05\ s$
Update relay period	$T_s = 1\ s$

receiver, and the distances d_{jr} from each jammer to the receiver. The packet error rate is set equal to $e^{-\xi s}$ where s is the signal to interference and noise ratio (SINR) $s = S/(I+N)$. The SINR is computed as a function of the received signal power $S = \rho P_t d_{tr}^{-\nu}$ from the transmitter, the received interference power $I = \rho \sum_j P_j d_{jr}^{-\nu}$ from the jammers, and the noise N at the receiver. The constant $\xi > 0$ determines the relationship between the SINR and the packet error rate, and the constants $\rho > 0$ and $\nu \geq 2$ characterize the path-loss of the wireless medium. Table 5.2 summarizes the parameter values used in our simulation study.

We are interested in comparing the performance of several methods of traffic allocation using the given network and jamming models. We define the following cases of interest.

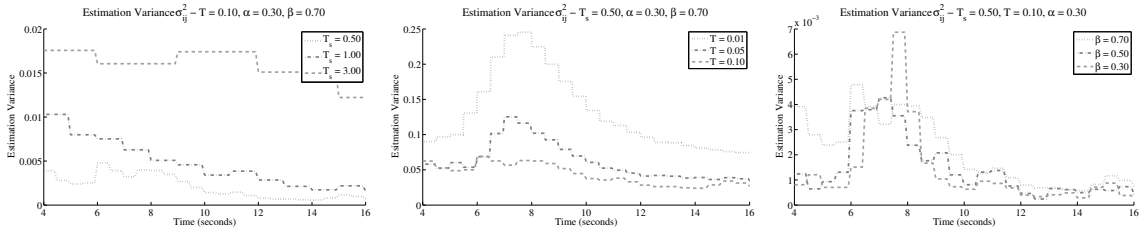
Case I - Ignoring jamming: Each source s chooses the allocation vector $\boldsymbol{\phi}_s$ using the standard maximum-flow formulation corresponding to $\mu_{ij} = 1$ and $\sigma_{ij}^2 = 0$ for all links (i, j). This case is included in order to observe the improvement that can be obtained by incorporating the jamming statistics.

Case II - Maximum throughput: The allocation vectors $\boldsymbol{\phi}_s$ are chosen using the



(a) (b) (c)

Figure 5.6: The estimate $\mu_{ij}(t)$ is simulated and compared to the packet success rate $x_{ij}(t)$ for varying values of the (a) update relay period T_s, (b) update period T, and (c) EWMA coefficient α.



(a) (b) (c)

Figure 5.7: The estimation variance $\sigma_{ij}^2(t)$ is simulated for varying values of the (a) update relay period T_s, (b) update period T, and (c) EWMA coefficient β.

jamming-aware optimization problem in Figure 5.4 with risk-aversion constant $k_s = 0$. This case incorporates the estimates μ_{ij}, updated every T_s seconds, in the sequential allocation problem.

Case III - Minimum risk-return: Similar to Case II with $k_s > 0$. This case incorporates the estimates μ_{ij} and uncertainty parameters σ_{ij}^2 to balance the mean throughput with the estimation variance.

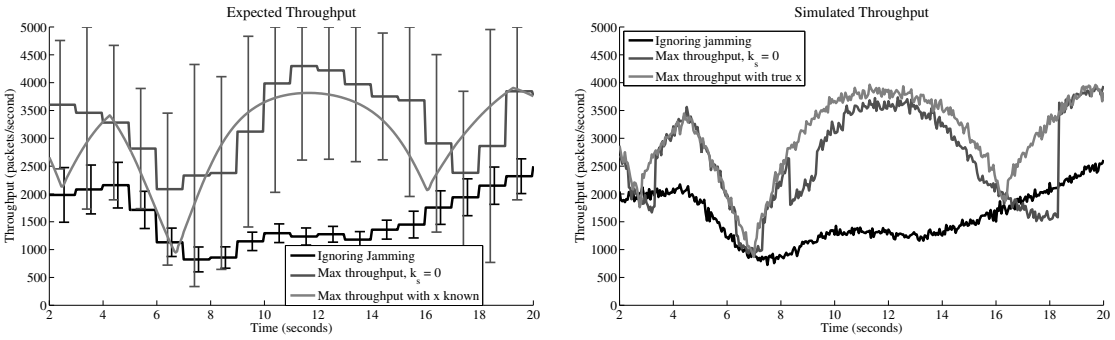
Case IV - Oracle model: Each source s continuously optimizes the allocation vector $\boldsymbol{\phi}_s$ using the true values of the packet success rates x_{ij}. This impractical case is included in order to illustrate the effect of the estimation process.

Our simulations are performed using a packet simulator which generates and allocates packets to paths in a fixed network according to the current value of the allocation vector $\boldsymbol{\phi}_s$. Each trial of the simulation compares several of the above cases using the same jammer mobility patterns over an interval of 20 s.

5.5.2 Simulation of Estimation Process

We first simulate the process of computing the estimate $\mu_{ij}(t)$ and the variance $\sigma_{ij}^2(t)$ over a single link (i, j). Figure 5.6 shows the true packet success rate $x_{ij}(t)$ with the estimate $\mu_{ij}(t)$ for various parameter values, and Figure 5.7 shows the estimation variance $\sigma_{ij}^2(t)$ for various parameter values. By inspection of Figure 5.6, we see that a shorter update relay period T_s and a longer update period T yield a more consistent estimate $\mu_{ij}(t)$ with less variation around the true value of $x_{ij}(t)$. In addition, a smaller value of α allows the estimate $\mu_{ij}(t)$ to reflect rapid changes in $x_{ij}(t)$, while a larger value of α smooths the estimate $\mu_{ij}(t)$ over the sampled PDRs.

Similarly, by inspection of Figure 5.7, we see that a shorter update relay period T_s and a longer update period T yield a lower estimation variance $\sigma_{ij}^2(t)$. In addition, a smaller value of the EWMA coefficient β allows the estimation variance $\sigma_{ij}^2(t)$ to primarily reflect recent variations in the sampled PDRs, while a larger value of β incorporates PDR history to a greater degree.



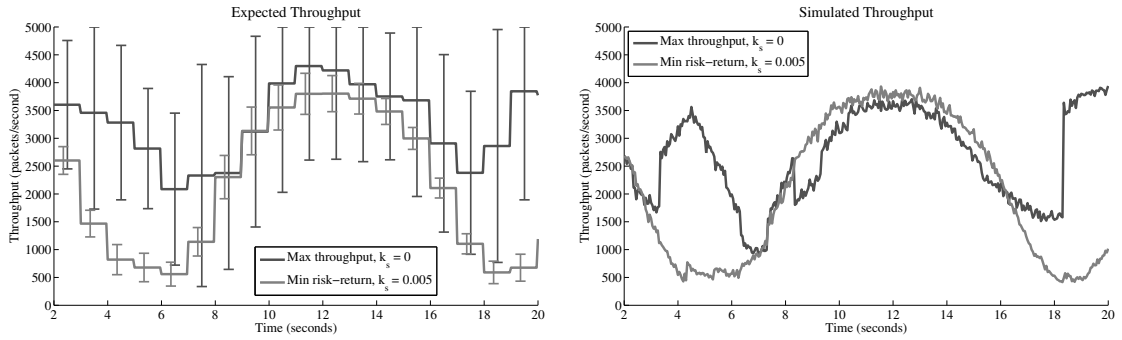
(a) (b)

Figure 5.8: Case I with $\mu_{ij}(t) = 1$ and $\sigma_{ij}^2(t) = 0$ for all (i, j), case II with the estimated $\mu_{ij}(t)$ and $\sigma_{ij}^2(t)$, and case IV with the true packet success rates $x_{ij}(t)$ are compared in terms of the (a) optimal expected throughput $\boldsymbol{\gamma}_s^T \boldsymbol{\phi}_s$ and the (b) actual achieved throughput $\boldsymbol{y}_s^T \boldsymbol{\phi}_s$. The error bars in (a) indicate one standard deviation $\sqrt{\boldsymbol{\phi}_s^T \boldsymbol{\Omega}_s \boldsymbol{\phi}_s}$ above and below the mean, limited by the network capacity of 5000 $pkts/s$.

5.5.3 Network Simulation

We next simulate the jamming-aware traffic allocation using the estimated parameters $\mu_{ij}(t)$ and $\sigma_{ij}^2(t)$ as described in Section 5.5.1. To observe the effects of the jamming-aware formulation and the estimation process, we first compare the optimal expected throughput and the actual achieved throughput for Case I, Case II, and Case IV in Figure 5.8. Figure 5.8(a) illustrates the expected throughput $\boldsymbol{\gamma}_s^T \boldsymbol{\phi}_s$ and throughput variance $\boldsymbol{\phi}_s^T \boldsymbol{\Omega}_s \boldsymbol{\phi}_s$ over time, and Figure 5.8(b) illustrates the resulting throughput $\boldsymbol{y}_s^T \boldsymbol{\phi}_s$ over time. By inspection, we see that both cases II and IV consistently outperform the maximum flow approach of Case I, showing the benefit of incorporating the jamming statistics into the allocation problem. In addition, we see that the effect of the estimation error on the optimal value in Case II is noticeable, but relatively small, compared to that of the oracle model of Case IV, indicating that the estimation process effectively characterizes the uncertainty in the jamming impact.

To observe the effect of the risk-aversion constant k_s, we next compare the optimal expected throughput and the actual achieved throughput for Case II with $k_s = 0$ to that of Case III with $k_s > 0$ in Figure 5.9. Figure 5.9(a) illustrates the expected throughput



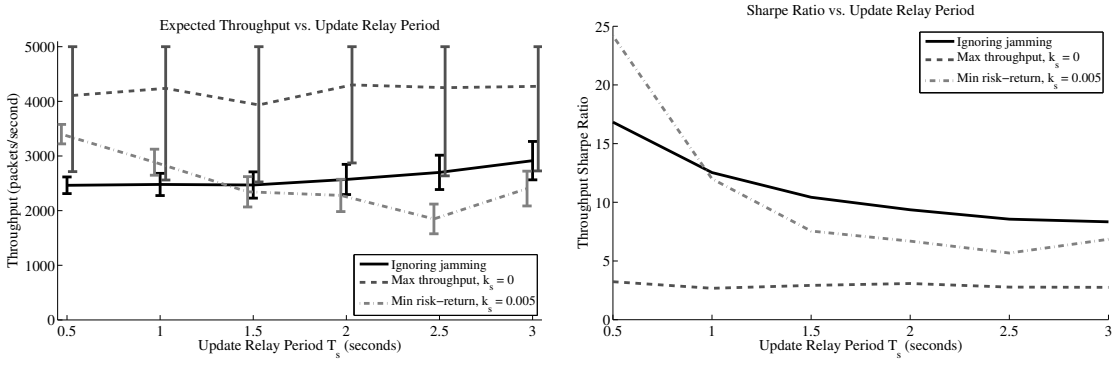
(a) (b)

Figure 5.9: Case II with $k_s = 0$ is compared to case III with $k_s > 0$ using the estimated $\mu_{ij}(t)$ and $\sigma_{ij}^2(t)$ in terms of the (a) expected throughput $\boldsymbol{\gamma}_s^T \boldsymbol{\phi}_s$ and the (b) achieved throughput $\boldsymbol{y}_s^T \boldsymbol{\phi}_s$. The error bars in (a) indicate one standard deviation $\sqrt{\boldsymbol{\phi}_s^T \boldsymbol{\Omega}_s \boldsymbol{\phi}_s}$ above and below the mean, bounded by the network capacity of 5000 $pkts/s$.

$\boldsymbol{\gamma}_s^T \boldsymbol{\phi}_s$ and throughput variance $\boldsymbol{\phi}_s^T \boldsymbol{\Omega}_s \boldsymbol{\phi}_s$ over time, and Figure 5.9(b) illustrates the resulting throughput $\boldsymbol{y}_s^T \boldsymbol{\phi}_s$ over time. By inspection, we see that Case III exhibits a significant reduction in the throughput variance $\sigma_{ij}^2(t)$ compared to that of Case II, resulting in achievable throughput much closer to the expected throughput. This reduction in variance in Case III sometimes comes in trade for a reduction in both expected and achieved throughput compared to that of Case II, as seen over the intervals from 3 to 7 seconds and 16 to 20 seconds in Figure 5.9. However, due to the higher variance in Case II, Case III can sometimes out-perform Case II in terms of achieved throughput, as seen over the interval from 8 to 16 seconds in Figure 5.9(b).

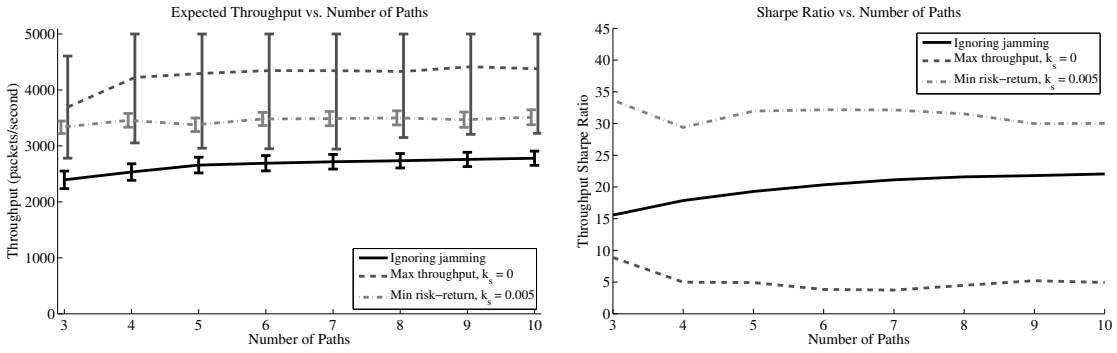
5.5.4 Simulation of Parameter Dependence

We next evaluate the effect of varying network and protocol parameters in order to observe the performance trends using the jamming-aware traffic allocation formulation. In particular, we are interested in the effect of the update relay period T_s and the maximum number of routing paths $|\mathcal{P}_s|$ on the performance of the flow allocation algorithm. In order to compare trials with different update times or numbers of paths, we average the simulated results



(a) (b)

Figure 5.10: The expected throughput is computed for Cases I, II, and III with varying update relay period T_s. In (a), the expected throughput $\boldsymbol{\gamma}_s^T \boldsymbol{\phi}_s$ is illustrated with error bars to indicate one standard deviation $\sqrt{\boldsymbol{\phi}_s^T \boldsymbol{\Omega}_s \boldsymbol{\phi}_s}$ around the mean, limited by the network capacity of 5000 $pkts/s$. In (b), the Sharpe ratio $\boldsymbol{\gamma}_s^T \boldsymbol{\phi}_s / \sqrt{\boldsymbol{\phi}_s^T \boldsymbol{\Omega}_s \boldsymbol{\phi}_s}$ is illustrated.



(a) (b)

Figure 5.11: The expected throughput is computed for Cases I, II, and III with varying number of routing paths $|\mathcal{P}_s|$. In (a), the expected throughput $\boldsymbol{\gamma}_s^T \boldsymbol{\phi}_s$ is illustrated with error bars to indicate one standard deviation $\sqrt{\boldsymbol{\phi}_s^T \boldsymbol{\Omega}_s \boldsymbol{\phi}_s}$ around the mean, limited by the network capacity of 5000 $pkts/s$. In (b), the Sharpe ratio $\boldsymbol{\gamma}_s^T \boldsymbol{\phi}_s / \sqrt{\boldsymbol{\phi}_s^T \boldsymbol{\Omega}_s \boldsymbol{\phi}_s}$ is illustrated.

over each simulated interval of 20 s, yielding a single value for each trial. In addition to comparing the expected throughput for various parameter values, we compute the Sharpe ratio [70], given by the ratio of the expected throughput $\boldsymbol{\gamma}_s^T \boldsymbol{\phi}_s$ to the standard deviation $\sqrt{\boldsymbol{\phi}_s^T \boldsymbol{\Omega}_s \boldsymbol{\phi}_s}$, measuring the throughput-per-unit-risk achievable by the different methods. To ensure that the observed trends are due to the intended parameter variation, we simulate a simple network topology similar to that given in Figure 5.2. Figure 5.10 illustrates the trends in expected throughput, throughput variance, and Sharpe ratio as the update relay period T_s increases. Since increased update times lead to increased variance, as previously seen in Figure 5.7(a), the Sharpe ratio decreases with increasing T_s. Figure 5.11 illustrates similar figures as the number of paths $|\mathcal{P}_s|$ increases, showing that the achieved throughput increases as the routing path diversity increases.

5.6 Summary of Contributions

In this chapter, we studied the problem of traffic allocation in multiple-path routing algorithms in the presence of jammers whose effect can only be characterized statistically. We have presented methods for each network node to probabilistically characterize the local impact of a dynamic jamming attack and for data sources to incorporate this information into the routing algorithm. We formulated multiple-path traffic allocation in multi-source networks as a lossy network flow optimization problem using an objective function based on portfolio selection theory from finance. We showed that this centralized optimization problem can be solved using a distributed algorithm based on decomposition in network utility maximization (NUM). We presented simulation results to illustrate the impact of jamming dynamics and mobility on network throughput and to demonstrate the efficacy of our traffic allocation algorithm. We have thus shown that multiple-path source routing algorithms can optimize the throughput performance by effectively incorporating the empirical jamming impact into the allocation of traffic to the set of paths.

Chapter 6

QUANTIFYING THE IMPACT OF NETWORK FLOW-BASED JAMMING ATTACKS

The open, shared nature of a broadcast communication channel introduces vulnerabilities to wireless networks such as denial-of-service (DoS) attacks [4]. Jamming is a particularly debilitating DoS attack where an adversary transmits interfering signals over the shared medium to block valid communications, for instance by transmitting wide-band noise or high-power narrow-band pulses [61, 75]. Communication systems attempting to perform jamming mitigation typically employ spread-spectrum techniques, forcing the adversary to exhaust significantly more resources by increasing the jamming bandwidth or power required to perform the attack [61,63,75]. Such techniques significantly increase the resource cost required for the jamming attack, leading to effective anti-jamming against resource-constrained jammers.

By incorporating cross-layer information and network communication into the jamming attack, a resource-constrained adversary can significantly increase the efficiency of the attack by targeting specific communication channels, helping to counteract the effect of the anti-jamming systems. Recent work has shown that intelligent jammers can exploit the structure of wireless link layer and MAC protocols [6, 74, 77] and link layer error correction protocols [48] to jamming attacks that require significantly less energy than jamming continuously or randomly.

In this work, we propose to additionally incorporate information from the network layer into the jamming attack, leading to a further reduction in the required energy resources. By noting that network flows traverse multiple links, the jamming adversary can effectively block an entire network flow [2] by jamming only the link where minimal energy is required. Additionally, since the probability of correct packet decoding is a function of the interference power at the receiver, the adversary can adjust its transmission power to moderate the

probability of successfully jamming a packet [75]. This jamming power regulation can be applied independently for each network flow and allows jammers to balance the resource expenditure over multiple flows in trade for a decreased probability of jamming success.

An adversary with a network of jammers can optimize the jamming attack by combining the network-layer information with transmission power regulation, and balancing the jamming workload across the jamming network. Furthermore, to counteract the probabilistic success of jamming packets, the jamming workload can be allocated such that packets unsuccessfully jammed by upstream jammers can be targeting by downstream jammers, given sufficient coverage of the network. Hence, the adversary can optimize a global utility function such as the expected flow rate reduction, energy expenditure, or jamming network lifetime through intelligent assignment of jamming workload and transmission power levels to the jamming networks. We denote this cross-layer DoS attack as a *flow-jamming attack*.

6.1 Our Contributions

In this chapter, we quantify the effect of flow-jamming attacks on network performance, and identify crucial concepts which may then be incorporated into network protocol design. We formulate flow-jamming attacks as constrained optimization problems which jointly optimize over jamming transmission power levels and jamming workload allocation to a network of jammers distributed throughout a wireless network. We propose a variety of metrics to evaluate the effect of flow-jamming attacks on network traffic and the fractional resource expenditure of the jamming adversary, which double as objective functions for optimization. We introduce convex and linear programming relaxations to the optimization framework by decomposing the optimization into two independent optimization problems, thereby enabling efficient computation. We propose a cooperative distributed algorithm for flow-jamming attacks for a decentralized jamming network and compare the performance to the centralized approach.

6.2 Network Model

In this work, we utilize cross-layer information from the network (layer 3), data-link/MAC (layer 2), and physical channel (layer 1). We state our models and assumptions governing

these layers, from the top down. A summary of the notation and metrics defined throughout this chapter is given in Table 6.1.

6.2.1 Network Layer

Let $\mathcal{N}$ denote the set of nodes which make up the wireless network. Traffic flows $f \in \mathcal{F}$ are established throughout the network between source-destination ordered pairs (s, d) via a suitable routing algorithm. For each (s, d) pair, we define an associated traffic flow f with rate r_f packets per second, which we assume is fixed over the duration of interest to the adversary. Without loss of generality, we assume that each flow consists of a single path from the source s to the destination d, as any multiple-path flow can be decomposed into a set of corresponding single-path flows.

6.2.2 Data-Link/MAC Layer

We are primarily interested in characterizing the worst-case effect of flow-jamming on the performance of networking protocols, without coupling its effect with that of typical link-layer errors. Therefore, we assume a number of network idealities, which if removed would only increase the impact of the jamming attack, by requiring additional network transmissions. Firstly, we assume that packetization or framing at the data-link layer [7] occurs without error, i.e. that there are no framing errors at each receiving node. Similarly, we assume that the packet transmissions of all flows in $\mathcal{F}$ do not lead to collisions at the MAC layer. Collision-free scheduling can be achieved, for example, by requiring that neighboring nodes transmit on orthogonal communication channels using any of the class of OFDMA protocols [31], which includes TDMA, FDMA, and CDMA as special cases. A network attempting to mitigate jamming is likely to utilize OFDMA or other spread-spectrum techniques to accomplish this goal.

6.2.3 Physical Layer

In this section, our goal is to derive a physical layer model that can incorporate a broad range of environments and modulation/coding techniques. We are primarily interested in

Table 6.1: We provide a summary of the notation and metrics used in Chapter 6 for the problem of quantifying the impact of cross-layer jamming attacks.

Symbol	Definition
$\mathcal{N}$	Set of wireless network nodes
$\mathcal{F}$	Collection of network flows
r_f	Packet rate of flow $f \in \mathcal{F}$
$\mathcal{J}$	Set of jammers
E_j	Energy budget for jammer $j \in \mathcal{J}$
d_{jf}	Distance from jammer $j \in \mathcal{J}$ to flow $f \in \mathcal{F}$
P_{jf}	Transmission power used by $j \in \mathcal{J}$ targeting $f \in \mathcal{F}$
$q(d_{jf}, P_{jf})$	Packet error rate (PER) in $f \in \mathcal{F}$ due to $j \in \mathcal{J}$
x_{jf}	Jammer-to-flow assignment for $j \in \mathcal{J}$ and $f \in \mathcal{F}$
$\boldsymbol{P}$	Vector of variables P_{jf}
$\boldsymbol{x}$	Vector of variables x_{jf}
$\pi_f(j)$	Order of jammer j along flow f
$\lambda_j(\boldsymbol{x}_j, \boldsymbol{P}_j)$	Resource expenditure of jammer j
$\lambda(\boldsymbol{x}, \boldsymbol{P})$	Resource expenditure metric (6.18)
$I(\boldsymbol{x}, \boldsymbol{P})$	Jamming impact metric (6.17)
$G(\boldsymbol{x}, \boldsymbol{P})$	Jamming gain metric (6.19)
$V(\boldsymbol{x}, \boldsymbol{P})$	Resource variation metric (6.20)
$\Phi(\boldsymbol{x}, \boldsymbol{P})$	Demand penalty function (6.21)

analyzing the physical layer packet error rate (PER), the probability that a packet is received in error, as the primary purpose of a jamming attack is to drastically increase this quantity.

For each transmitter and receiver, we assume the following physical communication model [75]. Given that the transmitter T and receiver R are separated by a distance d and T transmits with constant power P_T, the received signal power P_R at node R is given by

$$P_R = \rho P_T d^{-\alpha}. \tag{6.1}$$

The constant ρ in (6.1) incorporates the antenna gains G_{TR} of T in the direction of R and G_{RT} of R in the direction of T, the transmission wavelength λ, and the constant loss factor L, assumed to be independent of transmit power P_T. The constant α is the path-loss exponent, assumed to be $\alpha \geq 2$, that captures the decay in signal power with distance. Typical values of the path-loss exponent vary in the range $2 \leq \alpha \leq 4$ for open, outdoor environments and in the range $4 \leq \alpha \leq 7$ for constricted, indoor environments.

The PER of the physical layer communication protocol can be analyzed with respect to the signal-to-interference-and-noise ratio (SINR) s given by

$$s = \frac{P_R}{I + N} \tag{6.2}$$

where I is the interference power and N is the noise power at the receiver. In the absence of jamming, we assume the interference power I is zero and refer to the corresponding quantity P_R/N as the signal-to-noise ratio (SNR). We note that this assumption is equivalent to incorporating ambient interference into the quantity N. Under this model, we assume that each transmitter in the network adjusts its transmit power P_T as a function of the fixed distance d to a sufficient level to maintain an SNR of γ at the receiver. The SINR at the receiver can thus be expressed as

$$s = \frac{\gamma}{1 + I/N}. \tag{6.3}$$

The probability of packet error (PER) can be computed as a function $q(s)$ of the SINR s. We note that the exact form of the function $q(s)$ depends on the modulation and coding schemes. Examples of PER function $q(s)$ based on Gaussian noise and interference include

$$q(s) = \beta e^{-\xi s} \tag{6.4}$$

and

$$q(s) = \beta \operatorname{erfc}\left(\sqrt{\xi s}\right), \tag{6.5}$$

where $\operatorname{erfc}(\cdot)$ is the complementary error function for the Gaussian distribution [75], ξ is a constant depending on the modulation and coding schemes, and β is a constant maximum PER. As a particular example, the PER for packets of length L bits using uncoded BPSK or QPSK modulation under a Gaussian noise model has the form

$$q(s) = 1 - \left(1 - b \operatorname{erfc}\left(\sqrt{\xi s}\right)\right)^L, \tag{6.6}$$

which is well approximated by the PER function in (6.5) with $\beta = bL$ for reasonable values of parameters b and L.

Since the constant ξ in (6.4) and (6.5) is a scalar coefficient of s, it can be chosen to fit a reference PER. For example, if interference power $I = mP_R$ is sufficient to cause a PER of p, then using (6.4), we have

$$p = \beta e^{-\xi s} = \beta e^{-\frac{\xi\gamma}{1+I/N}} = \beta e^{-\frac{\xi\gamma}{1+m\gamma}}, \tag{6.7}$$

and the constant ξ can be parameterized by m and p as

$$\xi = \frac{1+m\gamma}{\gamma} \ln\left(\frac{\beta}{p}\right). \tag{6.8}$$

6.3 Adversary Model

Let $\mathcal{J}$ denote a set of jamming nodes which make up the adversarial wireless network. We assume that each jammer $j \in \mathcal{J}$ is able to sense transmissions and infer network flow topology and rates within a particular sensing region around the jammer's location. In addition, we assume that each jammer $j \in \mathcal{J}$ may exchange information with a subset $\mathcal{J}_j \subseteq \mathcal{J}$ of neighboring jammers without interfering with the network of nodes $\mathcal{N}$. This can be achieved by selecting channels orthogonal to those used by the communicating network.

We suppose that each jammer j is constrained by an energy budget E_j, which denotes a finite supply of energy allotted for a particular time interval of attack, since a jamming network unconstrained in terms of energy would be able to brute force jam all nearby flows without the need for any network information. With a constrained energy budget,

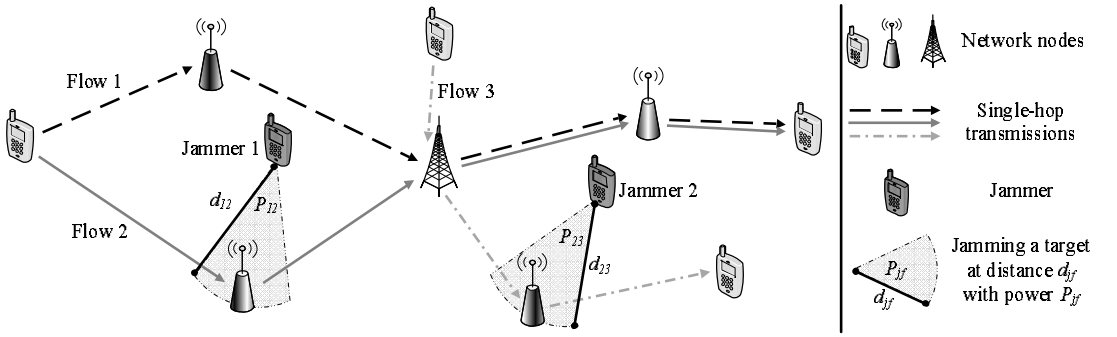


Figure 6.1: An example network and jammer topology is illustrated with three network flows and two jammers. Sample jamming options are indicated by the corresponding minimum distance d_{jf} and power P_{jf}.

on the other hand, the jamming adversary will seek an optimal allocation of resources with respect to the jammers' energy budgets and their affect on the underlying communicating network topology, by incorporating cross-layer information and distributing the jamming workload. We thus define the *jammer-to-flow assignment* variable $x_{jf} \in [0, 1]$ as the fraction of packets in flow f which jammer j will attempt to jam. Since we assumed that the underlying link layer does not admit collisions, we further assume that it is not possible to simultaneously jam multiple packets from non-interfering links with a single narrow-band jamming transmission. Once the assignment variables x_{jf} have been determined, we assume the jammers can coordinate their jamming transmissions such that neighboring jammers do not simultaneously attempt to jam the same packet in a common flow.

We further suppose that each jammer j can select the jamming transmission power P_{jf} used to jam each packet in flow f, which is a primary parameter in determining the energy exhausted by j. The choice of transmission power P_{jf} determines the PER $q(s)$ (e.g. (6.4) or (6.5)) through the received interference power I, which is related according to (6.1). Since I is inversely proportional to the distance from the jammer to the receiver, we assume that the each jammer chooses to jam a multi-hop flow at the receiver closest to itself, thus maximizing the effect of jamming. This minimum distance is denoted by d_{jf}. Using (6.1), the interference power I can thus be expressed as $\rho P_{jf} d_{jf}^{-\alpha}$. Combined with (6.3), this

expression yields the SINR s as a function of P_{jf} and d_{jf}, given by

$$s = \frac{\gamma}{1 + \rho P_{jf} d_{jf}^{-\alpha}/N}. \tag{6.9}$$

The PER $q(s)$ can now be expressed by the function $q(d_{jf}, P_{jf})$ via (6.9), assuming the jammer can estimate the constants ρ, α, and N. For instance, the PER model in (6.4) with $\beta = 1$ is given by

$$q(d_{jf}, P_{jf}) = e^{-\frac{\xi\gamma}{1+\rho P_{jf} d_{jf}^{-\alpha}/N}}. \tag{6.10}$$

This function of d_{jf} and P_{jf} captures the intuitive behavior that the probability of jamming success (PER) increases as a function of transmission power P_{jf} and decreases as a function of distance d_{jf}. Furthermore, given a fixed distance d_{jf}, the function $q(d_{jf}, P_{jf})$ behaves according to a sigmoid, or "s-shaped", function [54], which is useful for optimization techniques.

Given that jammer j is jamming with power P_{jf}, the average energy per packet which is exhausted is equal to a constant cost c times the jamming power P_{jf}, where c may depend on parameters such as modulation and coding schemes, the duty cycle of the jammer, and the packet length. The energy expended on a particular flow is then this quantity times the flow rate r_f and the flow assignment x_{jf}.

For a given set of deployed jammers with energy budgets E_j and set of network flows with rates r_f, the flow-jamming attack is uniquely specified by the jammer-to-flow assignment variables x_{jf} and the jamming transmission powers P_{jf}. Hence, the resource allocation problem of interest is for jammers to collaboratively and optimally determine the assignment variables x_{jf} and transmission powers P_{jf}. An example of a network and jammer topology to illustrate the model is given in Figure 6.1.

The ability for the jammers to collaboratively optimize depends on their ability to exchange locally sensed information about the network flows as well as their own resource constraints. In the case that the set of neighboring jammers $\mathcal{J}_j$ exchanging information is equal to the entire set $\mathcal{J}$, the optimization problem is essentially a centralized optimal resource allocation problem [56]. Such centralized solutions serve as a performance baseline for comparison and are addressed in Section 6.5 for various evaluation metrics introduced in

Section 6.4.2. Noting that extensive jammer communication overhead is counter-productive in practice, we develop a distributed algorithm for flow-jamming in Section 6.6.

6.4 Attack Metrics and Constraints

In order to define the feasible set of allocations for the jamming network, we present a set of constraints which must be satisfied by the flow-jamming attack with respect to the jammer resources and traffic flows. Additionally, we define a set of metrics regarding the effects of the attack in terms of network throughput and jammer resource expenditure, which serve as objective functions for evaluation and optimization of feasible attacks.

We let $\boldsymbol{x}$ and $\boldsymbol{P}$ respectively denote the vectors of jammer-to-flow assignment variables x_{jf} and jamming transmission powers P_{jf}. When convenient, we refer to the sub-vectors of variables $\boldsymbol{x}$ and $\boldsymbol{P}$ corresponding to a single flow f as $\boldsymbol{x}_f$ and $\boldsymbol{P}_f$ and to a single jammer j as $\boldsymbol{x}_j$ and $\boldsymbol{P}_j$. We let $\mathbf{1}$ denote a vector of ones such that $\mathbf{1}^T\boldsymbol{x}$ is equivalent to the ℓ_1 vector sum norm [39] for non-negative $\boldsymbol{x}$.

6.4.1 Attack Constraints

We formulate the base constraints on flow-jamming attacks with respect to the jammer-to-flow assignment vector $\boldsymbol{x}$ and power vector $\boldsymbol{P}$. The first pair of constraints follow the definitions of x_{jf} and P_{jf} and restrict the variables to their corresponding domains as

$$0 \leq x_{jf} \leq 1, \tag{6.11}$$

$$P_{\min} \leq P_{jf} \leq P_{\max} \tag{6.12}$$

for all $j \in \mathcal{J}$ and $f \in \mathcal{F}$, where $P_{\min}$ and $P_{\max}$ are respectively the minimum and maximum jamming transmission powers.

For a given jammer-to-flow assignment vector $\boldsymbol{x}$ and power vector $\boldsymbol{P}$, it is necessary that the resource of each jammer $j \in \mathcal{J}$ does not exceed their energy budget E_j. Since the jammers may be heterogeneous in terms of their energy budgets, we normalize the resource expenditure to the fraction of their available energy that is exhausted. The resource

expenditure is then given by

$$\lambda_j(\boldsymbol{x}_j, \boldsymbol{P}_j) = \frac{c}{E_j} \sum_{f \in \mathcal{F}} P_{jf} r_f x_{jf}. \tag{6.13}$$

The *supply constraint* yielded by these fractional resource expenditures is given by

$$0 \leq \lambda_j(\boldsymbol{x}_j, \boldsymbol{P}_j) \leq 1 \tag{6.14}$$

for all $j \in \mathcal{J}$.

The jamming allocation must additionally satisfy a *flow constraint*, as jammers cannot jam more flow than exists in the network. Since jamming success is probabilistic, we formulate the constraint in the average case by interpreting $q(d_{jf}, P_{jf})$ as the average fraction of jamming attempts which are successful. Since the flow available to downstream jammers is thus dependent on upstream and adjacent jammers, we define the *order* $\pi_f(j)$ of each jammer j along the flow f such that $\pi_f(j_i) < \pi_f(j_k)$ implies that jammer j_i will jam f upstream of (at a node closer to the source than) jammer j_k. Similarly, $\pi_f(j_i) = \pi_f(j_k)$ implies that jammers j_i and j_k will jam the same node in f. We let $\pi_f^{-1}(m)$ denote the set of jammers j with $\pi_f(j) = m$. The desired flow constraint is thus given by

$$\sum_{j \in \pi_f^{-1}(m)} x_{jf} + \sum_{j \in \bigcup_{i<m} \pi_f^{-1}(i)} q(d_{jf}, P_{jf}) x_{jf} \leq 1 \tag{6.15}$$

for all $f \in \mathcal{F}$ and for each order m. We note that if all jamming success probabilities $q(d_{jf}, P_{jf})$ are forced to 1, then the set of flow constraints given by (6.15) for a particular flow f and for all m reduces to the single linear constraint $\mathbf{1}^T \boldsymbol{x}_f \leq 1$. Solving for $\boldsymbol{x}$ in this special case is equivalent to finding the optimal partition of the jammer-to-flow assignment.

Finally, we allow for an optional constraint based on the jammer's desire to jam a certain portion of the network traffic. This *demand constraint* on each flow f imposes a lower bound z_f on the expected fraction of throughput reduction as

$$\sum_{j \in \mathcal{J}} q(d_{jf}, P_{jf}) x_{jf} \geq z_f \tag{6.16}$$

for all $f \in \mathcal{F}$. While the previous constraints always allow a feasible solution, adding the demand constraint further restricts the domain, possibly resulting in an infeasible set of

constraints. We again note that a special case arises when the probabilities $q(d_{jf}, P_{jf})$ are forced to 1. The jammer then has the ability to impose a lower bound of $z_f = 1$ for all flows f, which due to (6.15) is satisfied only when all flows are completely jammed.

6.4.2 Attack Evaluation Metrics

With constraints defining the feasible set, it is natural to define metrics that analyze the effectiveness of a given solution with respect to jammer resource expenditure and network throughput, thus providing objective functions for optimization. If the jamming network is locally optimizing using a distributed protocol, it may result in a solution that violates the overall flow or demand constraints, resulting in undefined metrics. We consider this problem of over-allocation alongside our distributed algorithm in Section 6.6.

To begin, we evaluate the extent to which the jammers in $\mathcal{J}$ are able to reduce network throughput by defining the *jamming impact* $I(\boldsymbol{x}, \boldsymbol{P})$, the average fractional throughput reduction, as

$$I(\boldsymbol{x}, \boldsymbol{P}) = \frac{1}{|\mathcal{F}|} \sum_{\substack{j \in \mathcal{J} \\ f \in \mathcal{F}}} q(d_{jf}, P_{jf}) x_{jf}. \tag{6.17}$$

The average resource expenditure for the set $\mathcal{J}$ of jammers is defined as the fraction $\lambda(\boldsymbol{x}, \boldsymbol{P})$ given by

$$\lambda(\boldsymbol{x}, \boldsymbol{P}) = \frac{1}{|\mathcal{J}|} \sum_{j \in \mathcal{J}} \lambda_j(\boldsymbol{x}_j, \boldsymbol{P}_j). \tag{6.18}$$

By combining (6.17) and (6.18), we can evaluate the overall jamming impact per unit of resource expenditure. Thus, we define the *jamming gain* $G(\boldsymbol{x}, \boldsymbol{P})$ equal to the ratio of jamming impact to resource expenditure as

$$G(\boldsymbol{x}, \boldsymbol{P}) = \frac{I(\boldsymbol{x}, \boldsymbol{P})}{\lambda(\boldsymbol{x}, \boldsymbol{P})}. \tag{6.19}$$

The jamming gain serves as a basis for optimizing the resource efficiency of the attack, instead of optimizing for resource expenditure or impact alone.

The above metrics reflect the average behavior over the set $\mathcal{J}$ of jammers. However, in order to maximize the lifetime of the jamming network [17], we are interested in measuring the ability to fairly distribute the resource expenditure among the jammers. We thus

define the *jamming resource variation* $V(\boldsymbol{x}, \boldsymbol{P})$ equal to the relative difference between the maximum and minimum resource expenditure as

$$V(\boldsymbol{x}, \boldsymbol{P}) = 1 - \frac{\min_j \lambda_j(\boldsymbol{x}_j, \boldsymbol{P}_j)}{\max_j \lambda_j(\boldsymbol{x}_j, \boldsymbol{P}_j)}, \tag{6.20}$$

falling in the range $[0, 1]$. Large variation indicates that some jammers are fully exhausting their energy budgets while others have unallocated energy resources due to network geometry. Small variation implies a balance of relative energy expenditure and allows for prolonged minimum jammer lifetime and flow-jamming attack duration.

As an alternative to imposing the demand constraint (6.16) for all $f \in \mathcal{F}$, we can incorporate a penalty function $\Phi(\boldsymbol{x}, \boldsymbol{P})$ into the metrics which measures the degree of violation of the demand constraint. This also ensures the existence of a feasible solution. The penalty function is thus defined as

$$\Phi(\boldsymbol{x}, \boldsymbol{P}) = \sum_{f \in \mathcal{F}} \left(z_f - \sum_{j \in \mathcal{J}} q(d_{jf}, P_{jf}) x_{jf} \right)^+, \tag{6.21}$$

where $(y)^+ = \max(0, y)$. The demand constraint (6.16) can be written equivalently in terms of the penalty function as $\Phi(\boldsymbol{x}, \boldsymbol{P}) = 0$.

6.5 Flow-Jamming Attack Formulations

In order to quantify the effect of worst-case flow-jamming attacks on network traffic, we formulate these attacks as constrained optimization problems subject to the constraints given in Section 6.4.1, using the evaluation metrics presented in Section 6.4.2 as objective functions. The general formulation yields a non-convex optimization problem, which demonstrates the upper bound on the effectiveness of the attack. We then demonstrate a two-step convex optimization relaxation, which yields a real-time approximation of the desired solution by first solving for transmission powers $\boldsymbol{P}$ and then optimizing the assignment $\boldsymbol{x}$ with respect to the chosen $\boldsymbol{P}$. We discuss the trade-offs between optimality of solutions and associated computational overhead in Section 6.7.

6.5.1 Optimal Flow-Jamming Attacks

Suppose the adversary is interested in maximizing an objective function $g(\boldsymbol{x}, \boldsymbol{P})$ that measures the performance of the jamming attack (e.g. gain, negative resource expenditure, etc.) and minimizing the penalty function $\Phi(\boldsymbol{x}, \boldsymbol{P})$. In order to determine the optimal solution $(\boldsymbol{x}^*, \boldsymbol{P}^*)$ subject to the constraints outlined in Section 6.4.1, the adversary must solve the constrained optimization problem

$$\boxed{\begin{aligned}(\boldsymbol{x}^*, \boldsymbol{P}^*) = \arg\max_{\boldsymbol{x}, \boldsymbol{P}} \; & g(\boldsymbol{x}, \boldsymbol{P}) - \Delta\Phi(\boldsymbol{x}, \boldsymbol{P}) \\ \text{s.t. } & (6.11), (6.12), (6.14), \text{ and } (6.15)\end{aligned}} \tag{6.22}$$

where $\Delta \geq 0$ is a weight on the penalty $\Phi(\boldsymbol{x}, \boldsymbol{P})$ for violating the demand constraint (6.16). The objective and penalty functions are optimized simultaneously, with Δ determining the relative importance of the two functions. A large Δ value would seek primarily to satisfy the demand constraint, using $g(\boldsymbol{x}, \boldsymbol{P})$ as a secondary criterion, with the opposite true for small Δ values. We present the following objective functions $g(\boldsymbol{x}, \boldsymbol{P})$ of interest using the metrics defined in Section 6.4.2.

Case 1 - Maximum Impact Attack: To maximize the overall impact without a demand penalty, the adversary can set $g(\boldsymbol{x}, \boldsymbol{P}) = I(\boldsymbol{x}, \boldsymbol{P})$ with $\Delta = 0$. Alternatively, to impose a demand penalty, the adversary can set $\Delta > 0$ and $z_f > 0$ for at least one flow f, noting that the penalty is assessed individually for each flow f, while the overall impact is averaged over all $f \in \mathcal{F}$.

Case 2 - Minimum Resource Attack: To minimize the resource expenditure $\lambda(\boldsymbol{x}, \boldsymbol{P})$ required to meet the demand constraint (6.16), the adversary can set $g(\boldsymbol{x}, \boldsymbol{P}) = -\lambda(\boldsymbol{x}, \boldsymbol{P})$ and set $z_f > 0$ for at least one flow f to prevent the trivial solution $\boldsymbol{x} = 0$. For sufficiently large Δ, the optimization of this problem will focus on meeting the demand constraint with the minimum resource expenditure.

Case 3 - Maximum Gain Attack: The goals of maximizing impact and minimizing resource expenditure can be combined in maximizing the jamming gain $g(\boldsymbol{x}, \boldsymbol{P}) = G(\boldsymbol{x}, \boldsymbol{P})$. Since maximizing the gain does not necessarily lead to a high jamming impact $I(\boldsymbol{x}, \boldsymbol{P})$, demand penalties with $\Delta > 0$ and $z_f > 0$ for at least one flow f can be imposed.

Case 4 - Minimum Variation Attack: To balance the resource expenditure over the set of jammers $\mathcal{J}$, the adversary can set $g(\boldsymbol{x}, \boldsymbol{P}) = V(\boldsymbol{x}, \boldsymbol{P})$ and set $z_f > 0$ for at least one flow f to prevent the trivial solution $\boldsymbol{x} = 0$. The optimal solution will attempt to meet the demand constraint with the optimal balance of resources over the jammers.

6.5.2 Computational Considerations and Attack Approximations

By inspection of the constraints and objective functions, we see that the flow constraint, demand constraint, and evaluation metrics are dependent on $q(d_{jf}, P_{jf})$, which is sigmoidal in P_{jf}. Hence, the general optimization formulation in (6.22) is a non-convex optimization problem [11], and the determination of the optimal solution $(\boldsymbol{x}^*, \boldsymbol{P}^*)$ must thus rely on heuristic methods. The implication is that only local optimums are guaranteed, and the computation time for finding these points may prohibit real-time jamming attacks.

Given the complications of solving non-convex optimization problems, we present an alternative convex optimization formulation. The goal of our approach is to decompose the joint optimization of variables $\boldsymbol{x}$ and $\boldsymbol{P}$ into two independent optimization problems. Fixing $\boldsymbol{P}$ yields linearity in the constraints (6.14), (6.15), and (6.16), and convexity in the objective and penalty functions (6.17), (6.18), and (6.21), with respect to the optimization variable $\boldsymbol{x}$. Hence, the two-stage decomposition is given by

$$\boxed{\begin{aligned} P_{jf}^* &= \arg\max_{P_{jf}} h(P_{jf}) \\ &\quad \text{s.t. } (6.12) \\ \\ \boldsymbol{x}^* &= \arg\max_{\boldsymbol{x}} g(\boldsymbol{x}, \boldsymbol{P}^*) - \Delta\Phi(\boldsymbol{x}, \boldsymbol{P}^*) \\ &\quad \text{s.t. } (6.11), (6.14), \text{ and } (6.15), \end{aligned}} \tag{6.23}$$

which can be efficiently solved when the objective function $h(P_{jf})$ is convex in P_{jf} and $g(\boldsymbol{x}, \boldsymbol{P}^*)$ is convex in $\boldsymbol{x}$. For $\boldsymbol{x}$, the metric of jamming gain $G(\boldsymbol{x}, \boldsymbol{P})$ is given by ratio of two linear functions in $\boldsymbol{x}$, and can thus be formulated through linear-fractional transformation as a convex optimization problem [11]. The metric of resource variation in (6.20) is generally non-convex, and motivates the need for a similar convex objective.

Attacks aiming to minimize the resource variation $V(\boldsymbol{x}, \boldsymbol{P})$ can be approximated by an alternative formulation which simultaneously minimizes the maximum resource expenditure and maximizes the minimum resource expenditure for each jammer j, effectively tightening the upper and lower bounds on each $\lambda_j(\boldsymbol{x}, \boldsymbol{P})$. Hence, instead of minimizing the variation $V(\boldsymbol{x}, \boldsymbol{P})$, we can instead introduce a variable upper bound λ_U and lower bound λ_L and minimize the difference $\lambda_U - \lambda_L$ with the modified supply constraint

$$\lambda_L \leq \lambda_j(\boldsymbol{x}_j, \boldsymbol{P}_j) \leq \lambda_U, \tag{6.24}$$

for all $j \in \mathcal{J}$, with the bounding constraints $0 \leq \lambda_L \leq 1$ and $0 \leq \lambda_U \leq 1$.

Given that the attack formulation in (6.23) for the given value of $\boldsymbol{P}^*$ is a convex optimization problem, the remaining piece is to define the objective function $h(P_{jf})$ in order to aptly fix the value $\boldsymbol{P}^*$. In what follows, we discuss two heuristic methods that can be used to define this objective function independently of the optimization variable $\boldsymbol{x}$.

Decomposition for High-Gain Attack

The first approach seeks to find $\boldsymbol{P}$ that will maximize the gain of the flow-jamming attack independently of $\boldsymbol{x}$. For a single jammer j and flow f, the contribution I_{jf} toward the impact $I(\boldsymbol{x}, \boldsymbol{P})$ of the flow-jamming attack can be expressed as

$$I_{jf} = x_{jf} q(d_{jf}, P_{jf}), \tag{6.25}$$

where d_{jf} is constant. Likewise, the fraction λ_{jf} of resources used by jammer j to jam packets in flow f can be written as

$$\lambda_{jf} = c P_{jf} r_f x_{jf} / E_j. \tag{6.26}$$

Thus, the individual gain of the single jammer-to-flow assignment can be measured by the ratio I_{jf}/λ_{jf}, which does not depend on x_{jf}. We thus define the objective function $h(P_{jf})$ as

$$h(P_{jf}) = \frac{E_j q(d_{jf}, P_{jf})}{c r_f P_{jf}}. \tag{6.27}$$

The maximum value of the objective function $h(P_{jf})$ for each (j, f) pair can thus be individually chosen independent of the assignment variable x_{jf}. Due to the sigmoidal shape of

the PER function $q(d_{jf}, P_{jf})$, the optimal power P^*_{jf} can be determined analytically [65]. Using these values for $\boldsymbol{P}^*$ yields the optimal gain for a single jammer and flow, though it is not necessarily optimal when multiple jammers and flows are considered. It does, however, allow the assignment variables x_{jf} to be efficiently optimized using (6.23).

Decomposition for High-Impact Deterministic Attack

A second approach is to fix $\boldsymbol{P}$ to maximize the impact of the attack for each node-flow pair. If $P_{\max}$ is sufficiently large, the PER $q(d_{jf}, P_{jf})$ can be made close enough to 1 that the margin $1-q(d_{jf}, P_{jf})$ is negligible. In this case, the choice of objective function $h(P_{jf}) = P_{jf}$ yields the optimal value $P^*_{jf} = P_{\max}$, allowing for a deterministic flow-jamming attack in which all probabilities $q(d_{jf}, P_{jf})$ are assumed to be 1. In this case, as mentioned in Section 6.4.1, the set of non-convex flow constraints (6.15) for flow f and all orders m reduces to the single linear constraint $\mathbf{1}^T \boldsymbol{x}_f \leq 1$. Additionally, this deterministic attack allows the adversary to achieve $z_f = 1$ in the demand constraint (6.16) given sufficient resources. This greedy demand constraint simplifies the penalty function, as the the argument y to the penalty function $(y)^+$ is always non-negative, implying that $(z)^+ = z$, which results in a linear program. Furthermore, a sufficiently high Δ forces the adversary to maximize the jamming impact for all jammers, yielding a corresponding heuristic for high-impact attacks.

6.6 Distributed Flow-Jamming Attacks

As the size of the jamming network increases, the communication and computational overhead of calculating and coordinating the centralized attack algorithm becomes prohibitively expensive. Therefore, in this section we present a distributed flow-jamming attack in which each jammer j computes the jamming transmission power P_{jf} and assignment variable x_{jf} using only local information. Since the optimality of the attack is dependent on the amount of information available, which in turn is dependent on the size chosen for the local region, there is an inherent trade-off between communication overhead and attack performance. However, increasing the required amount of communication may reduce the responsiveness of the jamming attack. In order to demonstrate the feasibility of this attack for resource-constrained jammer networks, we present the case where each jammer j computes only

the local variables $\boldsymbol{x}_j$ and $\boldsymbol{P}_j$ using messages received from jammers in the neighborhood $\mathcal{J}_j = \bigcup_f \pi_f^{-1}(\pi_f(j))$ of jammers targeting the same nodes as j.

Prior to the jamming attack commencing, each jammer senses its local region, determining the initial traffic flow rates r_f as well as the jamming costs of nodes incident to those flows of interest $\mathcal{F}_j \subseteq \mathcal{F}$, and exchanges this information with its neighborhood. After the attack has begun, a down-stream jammer j may sense a packet rate at the nearest node in flow f which is less than the initial packet rate r_f. We thus assume that each jammer j occasionally senses this *residual packet rate*, and we let $r_{jf}^{(t)}$ denote the sampled value of this rate at time t. We assume that two jammers j_1 and j_2 with $\pi_f(j_1) = \pi_f(j_2)$ will sense the same rates $r_{j_1 f}^{(t)} = r_{j_2 f}^{(t)}$ at each time t. We additionally introduce the projected flow rate $\hat{r}_{jf}$ which acts as an estimate of the value of the residual flow $r_{jf}^{(t)}$ after jamming, which is determined from past allocations. Note that initially, $r_f = r_{jf}^{(0)} = \hat{r}_{jf}$.

In this distributed attack formulation, each jammer j individually solves a constrained optimization problem involving only their local view. Since this step involves no coordination with neighboring nodes, this may result in jamming allocation conflicts, where neighboring jammers assignments $\boldsymbol{x}$ sum to more than unity, resulting in sub-optimal performance. To compensate for this, each jammer j exchanges $\boldsymbol{P}_j$ and $\boldsymbol{x}_j$ with its neighbors, and an additional optimization problem is solved for each set of jamming allocation conflicts.

In order to formulate the constrained optimization problem, it is first necessary to reformulate the objective function $g(\boldsymbol{x}, \boldsymbol{P})$, the penalty function $\Phi(\boldsymbol{x}, \boldsymbol{P})$, and the constraints in the scope of a single jammer. Since resource variation is not well-defined in a single jammer context, we focus only on the metrics of jamming impact and jamming gain. The objective function $g(\boldsymbol{x}, \boldsymbol{P})$ and penalty function $\Phi(\boldsymbol{x}, \boldsymbol{P})$ described in Section 6.5.1 can be similarly re-defined as $g_j(\boldsymbol{x}, \boldsymbol{P})$ and $\Phi_j(\boldsymbol{x}, \boldsymbol{P})$ for each jammer j with respect to the local flows $\mathcal{F}_j$ and itself. Since only the jammer j computes only its own allocation, the allocation constraint (6.11) and flow constraint (6.15) can be combined by incorporating the projected flow rate $\hat{r}_{jf}$ for each flow f as

$$0 \leq x_{jf} \leq \frac{\hat{r}_{jf}}{r_f}. \tag{6.28}$$

The supply constraint (6.14) remains unchanged. Use of the simplified formulation yields

the (non-convex) optimization problem

$$\boxed{\begin{aligned}(\boldsymbol{x}_j^*, \boldsymbol{P}_j^*) &= \arg\max_{\boldsymbol{x}_j, \boldsymbol{P}_j} g_j(\boldsymbol{x}_j, \boldsymbol{P}_j) - \Delta\Phi_j(\boldsymbol{x}_j, \boldsymbol{P}_j) \\ &\text{s.t. (6.12), (6.14), and (6.28).}\end{aligned}} \tag{6.29}$$

We again consider the alternative convex formulation in Section 6.5.2 allowing each jammer j to solve the two-stage decomposition

$$\boxed{\begin{aligned}P_{jf}^* &= \arg\max_{P_{jf}} h(P_{jf}) \\ &\text{s.t. (6.12)} \\ \\ \boldsymbol{x}_j^* &= \arg\max_{\boldsymbol{x}_j} g_j(\boldsymbol{x}_j, \boldsymbol{P}_j^*) - \Delta\Phi_j(\boldsymbol{x}_j, \boldsymbol{P}_j^*) \\ &\text{s.t. (6.14) and (6.28).}\end{aligned}} \tag{6.30}$$

After the optimization has been solved for all jammers in $\mathcal{J}$, each jammer j shares its allocations with its neighbors $\mathcal{J}_j$. For each flow $f \in \mathcal{F}_j$, if the subset $\pi_f^{-1}(\pi_f(j)) \subseteq \mathcal{J}_j$ of jammers violate the local flow constraints

$$\sum_{k \in \pi_f^{-1}(\pi_f(j))} x_{kf} \leq \frac{r_{jf}^{(t)}}{r_f} \tag{6.31}$$

at the corresponding target node, the conflict is resolved as follows. For each $k \in \pi_f^{-1}(\pi_f(j))$, let $e_{kf} = cP_{kf}^* r_{kf}^{(t)} x_{kf}^*$ denote the energy allocated to flow f. Each conflicting jammer k then simultaneously uses the exchanged information to solve the sub-problem

$$\boxed{\begin{aligned}(\boldsymbol{x}_f^*, \boldsymbol{P}_f^*) &= \arg\max_{\boldsymbol{x}_f, \boldsymbol{P}_f} \sum_{k \in \pi_f^{-1}(\pi_f(j))} q(d_{kf}, P_{kf}) x_{kf} \\ &\text{s.t. (6.12), (6.31)} \\ &cP_{kf} r_{kf}^{(t)} x_{kf} \leq e_{kf} \text{ for each } k \in \pi_f^{-1}(\pi_f(j)) \\ &0 \leq x_f\end{aligned}} \tag{6.32}$$

after which jamming commences. At the next update time t after δ seconds, jammers sense the current residual flow rate $r_{jf}^{(t)}$ and use the previously sensed rate $r_{jf}^{(t-\delta)}$ to compute the

Distributed Algorithm for Efficient Flow-Jamming

1. At time $t = 0$, initialize and exchange the distances d_{jf} with jammers in $\mathcal{J}_j$. Set the projected flow rates to $\hat{r}_{jf} = r_{jf}^{(0)} = r_f$.
2. Exchange the projected flow rates $\hat{r}_{jf}$ with neighbors and solve the optimization problem in (6.29) or (6.30) using the local information.
3. Exchange the resulting values $\boldsymbol{x}_j^*$ and $\boldsymbol{P}_j^*$ with jammers in $\mathcal{J}_j$.
4. Resolve any allocation conflict by solving (6.32).
5. Jam using the computed parameters $\boldsymbol{x}_j^*$ and $\boldsymbol{P}_j^*$.
6. At the next update time t, sense the new residual flow rate $r_{jf}^{(t)}$ and compute the new projected flow rate using (6.33), then return to step 2.

Figure 6.2: A distributed algorithm for efficient, cooperative jamming of network traffic flows is presented.

projected flow for the next set of optimizations as

$$\hat{r}_{jf} = \begin{cases} r_{jf}^{(t)} + r_{jf}^{(t-\delta)} \displaystyle\sum_{k \in \mathcal{J}_j \setminus \{j\}} x_{kf} & \text{if } r_{jf}^{(t)} \leq r_{jf}^{(t-\delta)} \\ r_{jf}^{(t)} \left(1 - \displaystyle\sum_{k \in \mathcal{J}_j \setminus \{j\}} x_{kf} \right) & \text{else.} \end{cases} \tag{6.33}$$

Thus, each jammer j performs the distributed flow-jamming attack via the iterative algorithm given in Figure 6.2.

We note that certain special cases of the objective functions $h(P_{jf})$ and $g_j(\boldsymbol{x}, \boldsymbol{P})$ yielding linear programming or convex optimization problems may allow for Lagrangian dual decomposition methods to be used [55], though we do not address these special cases in this work. A comparison of the proposed distributed algorithm to the centralized formulation is given in the next section.

6.7 Performance Evaluation

In this section, we evaluate the performance of the flow-jamming attacks using the metrics presented in Section 6.4.2, the centralized optimization problem in Section 6.5, and the

distributed optimization formulation in Section 6.6. We first describe the simulation setup and then illustrate and compare the performance of the centralized and distributed flow-jamming attacks using various objective functions.

6.7.1 Simulation Setup

We use the following setup to obtain our simulation results. A network comprising the set $\mathcal{N}$ of nodes is randomly deployed over a given area, and a link is formed between any pair of nodes within a fixed communication range. Each network flow $f \in \mathcal{F}$ is formed between a randomly selected source s and destination d using a randomized geometric routing algorithm that chooses the next hop toward d from the set of neighbors that are closer to d in terms of either distance or hop count. As discussed in Section 6.2.3, we assume that each transmitting node sets its transmission power to maintain an SNR of γ at the receiving node.

A network comprising the set $\mathcal{J}$ of jammers is deployed over the same area as the nodes in $\mathcal{N}$, and the neighboring subsets $\mathcal{J}_j$ and $\mathcal{F}_j$ for each jammer j are determined by fixed communication and sensing ranges. The path-loss constant ρ and exponent α were set using measurements made in an open environment. We further assume that the interference model is given by the PER function in (6.10), with the constant ξ chosen using the reference parameters m and p in (6.8). Table 6.2 summarizes the default parameter values used in our simulation study, noting that specified parameters are varied in certain cases.

6.7.2 Comparison of Optimization Formulations

We first evaluate the performance of the centralized attack formulation in (6.22) using the objective functions in Cases 1-4 in Section 6.5.1. We illustrate the results of a single network and jammer deployment with each of the four attacks performed using the same data set. Figure 6.3 illustrates the four values for each metric $I(\boldsymbol{x}, \boldsymbol{P})$, $\lambda(\boldsymbol{x}, \boldsymbol{P})$, $G(\boldsymbol{x}, \boldsymbol{P})$ and $V(\boldsymbol{x}, \boldsymbol{P})$, with the results of each metric normalized with respect to the largest result for ease of comparison.

As seen in Figure 6.3, each attack optimizes the corresponding metric in trade for weaker

Table 6.2: We provide a summary of the parameters used to simulate cross-layer jamming attacks.

Parameter	Value
Network area	$100\ m \times 100\ m$
Number of nodes	$\lvert\mathcal{N}\rvert = 200$
Network radio range	$20\ m$
Number of network flows	$\lvert\mathcal{F}\rvert = 20$
Flow rates	$r_f = 1500\ pkts/sec$
Signal-to-noise ratio	$\gamma = 5$
Path-loss constant	$\rho = 2.5 \times 10^{-4}$
Path-loss exponent	$\alpha = 2.7$
Receiver noise	$N = 1\ nW$
PER parameters	$m = 10$, $p = 0.9$
Number of jammers	$\lvert\mathcal{J}\rvert = 10$
Jammer radio and sensing ranges	$50\ m$
Jammer energy supply	$E_j = 10\ mJ$
Jammer cost coefficient	$c = 10^{-6}$
Minimum jamming power	$P_{\min} = 0\ mW$
Maximum jamming power	$P_{\max} = 500\ mW$
Jamming demand	$z_f = 1/2$
Penalty coefficient	$\Delta = 10^4$

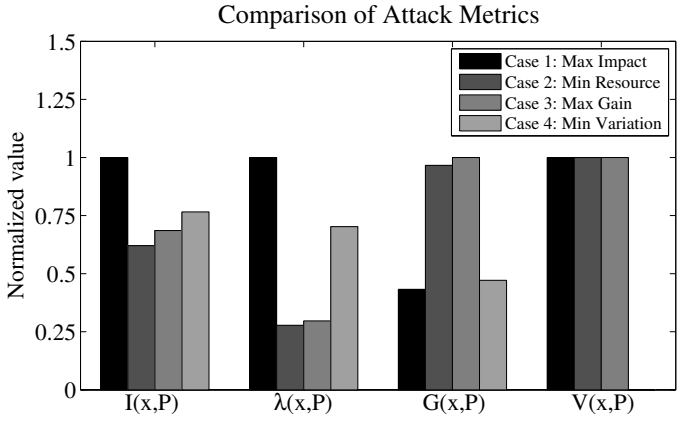


Figure 6.3: The centralized flow-jamming attacks in Cases 1-4 presented in Section 6.5.1 are simulated. The metrics of jamming impact $I(\boldsymbol{x}, \boldsymbol{P})$, resource expenditure $\lambda(\boldsymbol{x}, \boldsymbol{P})$, gain $G(\boldsymbol{x}, \boldsymbol{P})$, and resource variation $V(\boldsymbol{x}, \boldsymbol{P})$ are illustrated for each attack. The value of each metric is normalized by the group maximum.

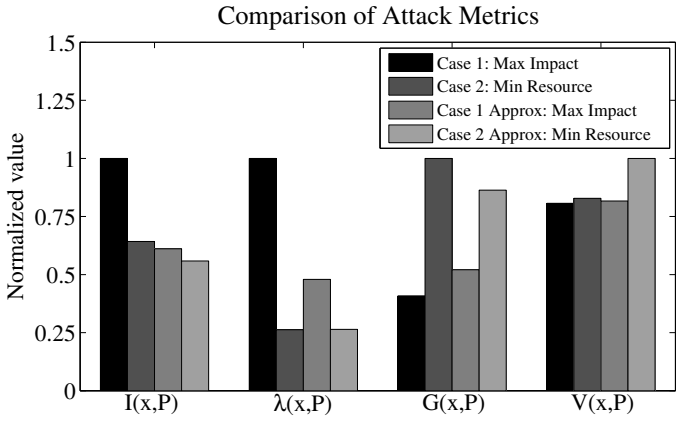


Figure 6.4: The centralized flow-jamming attacks in Cases 1-2 presented in Section 6.5.1 are simulated using both the non-convex formulations and the convex approximations, requiring respective computational run-time of 3178 seconds, 2784 seconds, 0.7 seconds, and 0.4 seconds. The metrics of jamming impact $I(\boldsymbol{x}, \boldsymbol{P})$, resource expenditure $\lambda(\boldsymbol{x}, \boldsymbol{P})$, gain $G(\boldsymbol{x}, \boldsymbol{P})$, and resource variation $V(\boldsymbol{x}, \boldsymbol{P})$ are illustrated for each attack. The value of each metric is normalized by the group maximum.

performance in terms of other evaluation metric. The maximum impact attack in Case 1 yields the highest impact $I(\boldsymbol{x}, \boldsymbol{P})$ but also has the highest resource expenditure $\lambda(\boldsymbol{x}, \boldsymbol{P})$. The minimum resource attack in Case 2 yields the lowest resource expenditure $\lambda(\boldsymbol{x}, \boldsymbol{P})$ but also has the lowest impact $I(\boldsymbol{x}, \boldsymbol{P})$. The maximum gain attack in Case 3 balances the trade-off between impact and resource expenditure, yielding an increase in both metrics over the minimum resource attack in Case 2. We note that all of Cases 1-3 achieve their maximum values by allowing a subset of the jammers to do a majority of the work, leading to a high resource variation $V(\boldsymbol{x}, \boldsymbol{P})$. The minimum variation attack in Case 4 balances this workload and in a further trade-off between impact and resource expenditure, reducing the jamming gain but increasing the impact.

We next compare the non-convex formulation with the performance of the centralized convex attack formulation in (6.23) using the two-stage decomposition with the objective functions in Cases 1-2. Figure 6.4 illustrates the normalized results of these cases. As can be seen, the attacks using convex formulations yield reasonable approximations of the

objective attained by the centralized solution. We note that the computational run-times of 3178 seconds for Case 1 and 2784 seconds for Case 2 are several orders of magnitude greater than the run-times of 0.7 seconds and 0.4 seconds required for the convex approximations. The simulated results and corresponding run-times for the non-convex formulations were obtained using the `glcSolve` solver in the TOMLAB Optimization Environment [1], and the results and run-times for the convex were obtained using the `CVX` package for solving convex optimization problems [34]. These computation times demonstrate that real-time jamming attacks using the non-convex attack formulations are likely impractical. We thus focus our attention on the use of convex approximations for the remainder of this simulation study.

6.7.3 Effect of Parameter Variation

In order to quantify the effect of parameter variation on flow-jamming attack performance, we next simulate attacks using the convex approximation and distributed formulations over a range of parameters. We illustrate the effect of varying the number of jammers, the total jamming energy, the number of traffic flows, and the path-loss exponent α on the resulting jamming impact $I(\boldsymbol{x}, \boldsymbol{P})$.

To evaluate the effect of the size of the adversarial network on jamming efficiency, we vary the number of jammers $|\mathcal{J}|$ while maintaining a fixed total jamming energy $\sum_{j\in\mathcal{J}} E_j$. Figure 6.5 illustrates the jamming impact $I(\boldsymbol{x}, \boldsymbol{P})$ as a function of the number of jammers for attacks using the convex approximations of Cases 1-4 and the distributed algorithm in Section 6.6. As seen in Figure 6.5, dispersing the jamming energy over a larger adversarial network increases the jamming impact via close jammer proximity.

We demonstrate the effect of the jammers' resource constraints on jamming impact by varying the total jamming energy $\sum_{j\in\mathcal{J}} E_j$ for a fixed number of jammers. Figure 6.6 illustrates the jamming impact $I(\boldsymbol{x}, \boldsymbol{P})$ as a function of the total jamming energy for attacks using the convex approximations of Cases 1-4 and the distributed algorithm in Section 6.6. As seen in Figure 6.6, the resulting jamming impact increases with the total available energy, with diminishing returns as the energy increases.

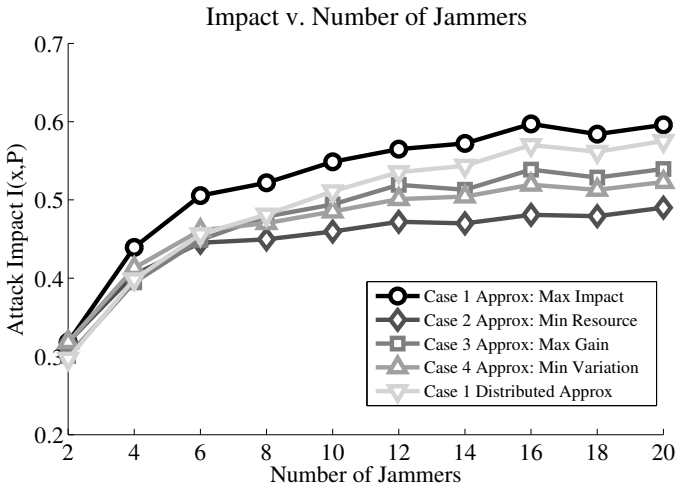


Figure 6.5: The jamming impact $I(\boldsymbol{x}, \boldsymbol{P})$ resulting from each of the centralized convex and distributed flow-jamming attacks is simulated for various numbers of jammers $|\mathcal{J}|$, keeping the total jamming energy $\sum_{j\in\mathcal{J}} E_j$ constant.

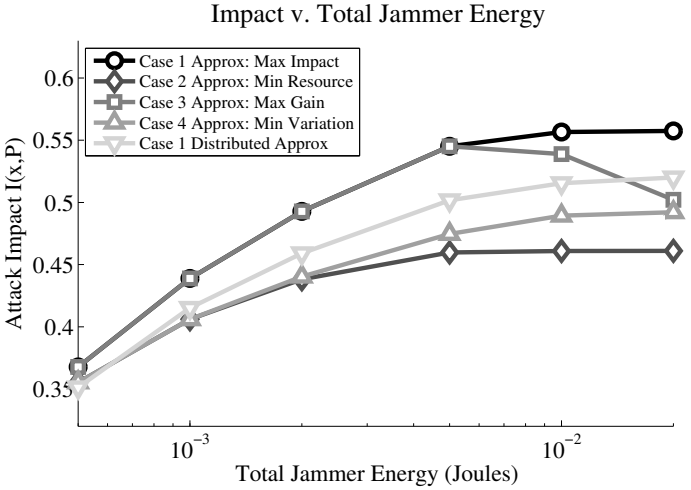


Figure 6.6: The jamming impact $I(\boldsymbol{x}, \boldsymbol{P})$ resulting from each of the centralized convex and distributed flow-jamming attacks is simulated for various values of the total jamming energy $\sum_{j\in\mathcal{J}} E_j$, keeping all other network and jamming parameters constant.

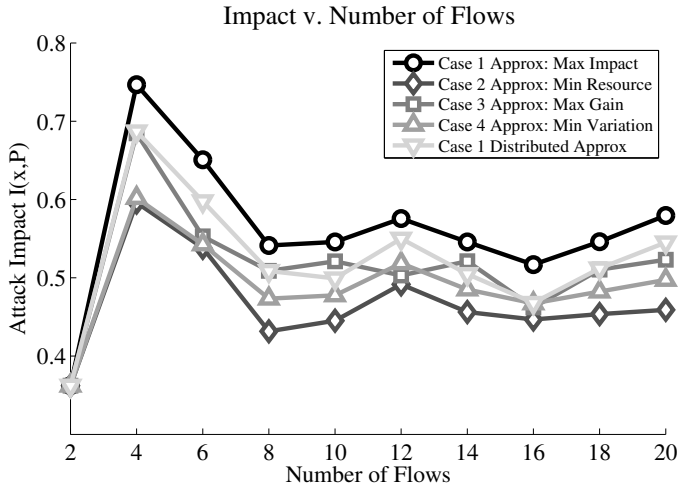


Figure 6.7: The jamming impact $I(\boldsymbol{x}, \boldsymbol{P})$ resulting from each of the centralized convex and distributed flow-jamming attacks is simulated for various numbers of network traffic flows $|\mathcal{F}|$, keeping the total flow rate $\sum_{f \in \mathcal{F}} r_f$ constant.

To show the effect of network flow topology, we evaluate the effect of varying the number of network traffic flows $|\mathcal{F}|$, keeping the total traffic rate fixed. Figure 6.7 illustrates the jamming impact $I(\boldsymbol{x}, \boldsymbol{P})$ as a function of the number of traffic flows for attacks using the convex approximations of Cases 1-4 and the distributed algorithm in Section 6.6. As seen in Figure 6.7, the jamming impact initially increases with the number of flows due to the decreasing minimum distances between jammers and flows. However, as the number of flows increases beyond a threshold, the jamming impact decreases due to the traffic being spread evenly among areas not covered by the adversarial network.

To show the effect of the physical medium, we evaluate the effect of varying the path-loss parameter α. Figure 6.8 illustrates the jamming impact $I(\boldsymbol{x}, \boldsymbol{P})$ as a function of the path-loss exponent for attacks using the convex approximations of Cases 1-4 and the distributed algorithm in Section 6.6. As seen in Figure 6.8, the jamming impact decreases with increasing path-loss exponent due to the increase in transmission power required to maintain the interference power level. In this case, it is also necessary for the network nodes to increase their transmission power to maintain connectivity. We additionally note that the performance of the distributed algorithm converges to that of the centralized convex

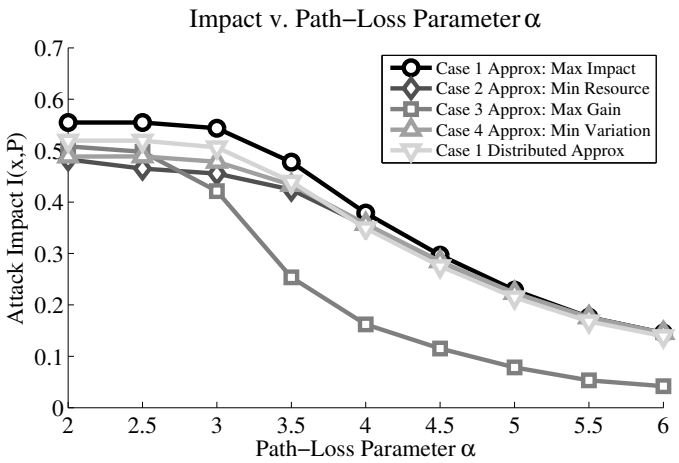


Figure 6.8: The jamming impact $I(\boldsymbol{x}, \boldsymbol{P})$ resulting from each of the centralized convex and distributed flow-jamming attacks is simulated for various values of the path-loss exponent α, keeping all other network and jamming parameters constant.

approximation as the path-loss parameter α increases, demonstrating the highly localized effect of jamming in the lossy environment.

6.8 Summary of Contributions

In this chapter, we showed that cross-layer information and optimization techniques allow a resource-constrained adversary to intelligently allocate jamming resources over an adversarial network. We introduced the class of *flow-jamming attacks* which seek to efficiently reduce network throughput, demonstrating the potential effects of jamming by even the most resource-starved adversarial networks. We presented a set of metrics to quantify the effect of flow-jamming on network flows and jammer resource expenditure and developed a constrained, non-convex optimization problem to model these attacks. Furthermore, in order to demonstrate the feasibility of these attacks in real-time, we proposed convex relaxations and a distributed version of jamming attacks in this framework. We demonstrated the potential impact of cross-layer flow-jamming attacks through a simulation study and showed that the convex formulations closely approximate the non-convex optimization problems. Future work can leverage the understanding of jamming expounded by this work to

design network protocols that are robust to cross-layer jamming attacks.

Chapter 7

CONTRIBUTIONS AND FUTURE WORK

7.1 Contributions in this Dissertation

The realization of robust and secure ad-hoc networking depends on the ability to understand attacks and their impact on network security and performance. The constraints faced in the ad-hoc environment, such as the lack of pre-existing infrastructure, limited resource availability, and unattended operation in a hostile environment, introduce a variety of vulnerabilities and create a variety of challenges to secure networking. In this dissertation, we have investigated the following problems of modeling attacks on network protocols and performance and designing robust networking protocols.

In Chapter 2, we addressed the problem of establishing secure communication links in ad-hoc and sensor networks through the random assignment of symmetric cryptographic keys prior to network deployment. We showed that the protocol designer can probabilistically control the number of nodes sharing each key, allowing for explicit bounds to be placed on the worst-case network connectivity and resilience to node capture attack. We proposed a random key assignment model based on the design and realization of a probability distribution on the number of nodes sharing each key, and we presented a sampling framework for key assignment algorithms that can approximately realize any designed key assignment distribution. We derived a model for probabilistic k-connectivity of randomly deployed networks restricted by radio range and the existence of shared keys. We demonstrated the design of new key predistribution schemes using the proposed model while paying particular attention to the worst-case probability of sharing keys and resilience to node capture. In addition, we presented an approach using the key assignment model to analyze the effect of network extension via addition of nodes to the network. The resulting framework allows for design of key assignment distributions that can realize the average-case security requirements of a given application and improve the worst-case requirements as a design

parameter.

In Chapter 3, we addressed the problem of modeling node capture attacks in order to understand their potential impacts on network security and the problem of developing new vulnerability metrics to jointly analyze key assignment and routing protocols. Instead of focusing on random node capture attacks as in previous works, we showed that an adversary can obtain a significant amount of information by observing or participating in network operations and eavesdropping on message exchanges. We proposed a class of route vulnerability metrics (RVMs) to evaluate node capture attack impact with respect to the joint analysis of routing and link security. We showed that an adversary can similarly use RVMs to formulate an optimal node capture attack as a non-linear integer programming minimization problem and can use the GNAVE algorithm using a greedy heuristic to approximate the NP-hard problem. We demonstrated probabilistic methods that allow for estimation of the RVM value when privacy-preserving protocols prevent an adversary from determining certain quantities. We simulated a variety of different RVMs and compared the performance using the different attack strategies under a variety of routing protocol classes.

In Chapter 4, we investigated the possibility for an adversary to mount an efficient jamming attack on control channels using synchronization information obtained by node capture. This efficient control channel jamming attack effectively bypasses the effect of spread spectrum communication by allowing the jamming adversary to synchronize with the sending and receiving nodes and focus the jamming energy in the precise location of the signal energy, whether variable in the time, frequency, or signal/code domains. In order to mitigate such control channel jamming attacks, we mapped the problem of robust control channel access to the problem of secure key establishment as in Chapter 2. Based on the mapping, we proposed a framework for control channel access based on the redundancy of random key assignment. We proposed and evaluated metrics for resilience to control channel jamming and delay experienced by users in the presence of jamming and demonstrated that the use of random key assignment leads to graceful degradation in resilience and delay as the number of compromised users increases. We presented the GUIDE algorithms for probabilistic identification of compromised users in the system by detecting which control channels are jammed and correlating the information to the set of keys held by each user. We

presented simulation results to demonstrate the trade-offs in the key assignment parameters and the parameters of the identification process using the GUIDE algorithm.

In Chapter 5, we addressed the problem of designing routing protocols that can dynamically adjust the allocation of traffic over multiple paths in order to compensate for the effects of an on-going jamming attack. We propose the use of a source routing protocol that uses statistics relayed from intermediate network nodes along the routing paths in order to adjust the traffic allocation to optimize the throughput delivered to the destination node. We proposed the use of periodic sampling of the loss rate due to jamming and moving average techniques to maintain an estimate of the probability of success as well as the uncertainty in the estimate for each link. We showed that each source node can use the link success estimates to formulate the traffic allocation problem over multiple routing paths as a lossy network flow optimization problem. We presented a mapping of the traffic allocation problem to the problem of financial portfolio selection and made use of this mapping to select an objective function for the optimization formulation. We showed that this centralized optimization problem can be decomposed into a distributed algorithm based on network utility maximization (NUM). We presented simulation results to demonstrate the impact of dynamic and mobile jamming on network throughput and to demonstrate the effectiveness of our traffic allocation algorithm.

In Chapter 6, we investigated the problem of quantifying the impact of cross-layer jamming attacks by multiple coordinated jammers on the achievable network throughput. We formulated jamming attacks as constrained optimization problems that jointly optimize the assignment of jamming workload over the adversarial network and the jamming transmission power levels. We proposed a set of metrics to quantify the impact of jamming on network traffic flows and jammer resource expenditure and used these metrics as the objective functions in the non-convex optimization formulation. To demonstrate the feasibility of these jamming attacks in real-time, we presented convex relaxations and a distributed version of jamming attacks in the optimization framework. We demonstrated the impact of flow-based jamming attacks via simulation and showed that the convex relaxations and distributed attack algorithm closely approximate the non-convex optimization problems.

7.2 Future Research Directions

The work presented in this dissertation falls into a broad class of problems of understanding the impact of attacks and errors on network performance and designing networking protocols that are robust to attack and error. As network applications and design problems continue to evolve, the research areas of adversary modeling and attack mitigation must also evolve. As the space of attacks on network security and protocol performance and the capabilities of an adversary can never be fully understood, the problems of understanding the impact of attacks and designing robust networking protocols are ongoing research areas. In addition to future work on general ad-hoc and sensor networking protocols, the following problems in this space are of interest in the future.

7.2.1 Game-Theoretic Modeling of Adaptive Attack and Defense

Throughout this dissertation, we approached the problems of attack modeling and defense from a one-sided perspective, either taking the view of the network responding to a given attack or of the adversary performing the attack for a given network state. Since the network and the adversary are both operating in the same time and space, it is more likely that they will be operating interactively and adapting to the actions of each other. Hence, it is of interest in the future to model these interactions and the resulting adaptation of each party. Future work will thus approach this modeling problem through the use of game theory, treating the network and the adversary as the two players in a game and allowing each player to make decisions based on particular objectives and the expected decisions of the opponent.

7.2.2 Cyber-Physical System Security

Cyber-physical systems (CPS) arise from the synthesis of computing, networking, and control systems and are already uewd for a wide variety of applications. Logically, a CPS can be envisioned as a control system application running on top of the network protocol stack. However, the reliability of the control application depends inherently on the performance of the underlying network protocols. For example, malicious attacks on the information trans-

mitted through the network, including jamming and packet modification or fabrication, can lead to undesirable input and actuation in the target control system. Hence, The design of CPSs that are robust to errors and attacks on the network information or performance is a problem that belongs to the category of cross-layer design, in that the control application and the networking protocols must be jointly considered for robustness. Future work in the area of CPS security thus involves the cross-layer analysis of control systems and the underlying network protocols in order to understand the potential impact of attacks on network information and protocol performance on the behavior of the control process.

7.2.3 Cognitive Radio Network Security

In a cognitive radio system, network devices are given the ability to be opportunistic in terms of spectum usage, in that any licensed spectrum that is not being used by licensed users can be detected and used by unlicensed cognitive radio users. Allowing for this cognitive ability introduces a number of vulnerabilities to malicious and selfish cognitive radio users in the network, as opportunism can quickly turn to malice or greed. One issue in cognitive radio networks to date is the common assumption of the existence of an underlying system of control channels in order for cognitive radio users to communicate and collaborate toward fair allocation of the available spectral resources. Future investigation of the system described in Chapter 4 may be applicable in the future to this problem of cognitive control channels, though the underlying system model is undoubtedly more complex. The issue of fairness versus selfish behavior among cognitive radio users is another problem of interest in the future. A possible approach to manage fair user interactions is to model the allocation problem as a multi-player game in which players are rewarded with spectral resources but penalized for unfair use of the available spectrum. A third vulnerability in cognitive radio networks is due to the possibility for an unlicensed cognitive radio user to masquerade as a licensed user and convince other cognitive users to stop using the desired resources entirely. This access control problem is complicated due to the signal-based detection mechanisms currently proposed as the basis for cognitive users to detect licensed users' spectral use.

BIBLIOGRAPHY

[1] The TOMLAB optimization environment. `http://tomopt.com/tomlab/`.

[2] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.

[3] I. F. Akyildiz, X. Wang, and W. Wang. Wireless mesh networks: A survey. *Computer Networks*, 47(4):445–487, March 2005.

[4] R. Anderson. *Security Engineering: A Guide to Building Dependable Distributed Systems*. John Wiley & Sons, Inc., 2001.

[5] G. Asada, M. Dong, T. Lin, F. Newberg, G. Pottie, H. Marcy, and W. Kaiser. Wireless integrated network sensors: Low-power systems on a chip. In *Proc. 24th IEEE European Solid-State Circuits Conference (ESSCIRC'98)*, pages 9–12, The Hague, The Netherlands, September 1998.

[6] J. Bellardo and S. Savage. 802.11 denial-of-service attacks: Real vulnerabilities and practical solutions. In *Proc. USENIX Security Symposium*, pages 15–28, Washington, DC, August 2003.

[7] D. Bertsekas and R. Gallager. *Data Networks*. Prentice Hall, 2nd edition, 1992.

[8] C. Bettstetter. On the minimum node degree and connectivity of a wireless multihop network. In *Proc. 3rd ACM International Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc'02)*, pages 80–91, Lausanne, Switzerland, June 2002.

[9] R. Blom. An optimal class of symmetric key generation systems. In *Advances in Cryptology: Proc. EUROCRYPT'84, LNCS 209*, pages 335–338, Paris, France, April 1984.

[10] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung. Perfectly-secure key distribution for dynamic conferences. In *Advances in Cryptology: Proc. CRYPTO'92, LNCS 740*, pages 471–486, Santa Barbara, CA, USA, August 1993.

[11] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge, 2004.

[12] N. Cai and R. W. Yeung. Secure network coding. In *Proc. 2002 IEEE International Symposium on Information Theory (ISIT'02)*, page 323, Lausanne, Switzerland, June/July 2002.

[13] S. A. Çamtepe and B. Yener. Combinatorial design of key distribution mechanisms for distributed sensor networks. In *Proc. 9th European Symposium on Research Computer Security (ESORICS'04), LNCS 3193*, pages 293–308, Sophia Antipolis, France, August 2004.

[14] A. Chan, X. Liu, G. Noubir, and B. Thapa. Control channel jamming: Resilience and identification of traitors. In *Proc. IEEE International Symposium on Information Theory (ISIT'07)*, Nice, France, June 2007.

[15] H. Chan and A. Perrig. PIKE: Peer intermediaries for key establishment in sensor networks. In *Proc. 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'05)*, pages 524–535, Miami, FL, USA, March 2005.

[16] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. In *Proc. 2003 IEEE Symposium on Security and Privacy*, pages 197–213, Oakland, CA, USA, May 2003.

[17] J.-H. Chang and L. Tassiulas. Energy conserving routing in wireless ad-hoc networks. In *Proc. 19th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'00)*, pages 22–31, Tel Aviv, Israel, March 2000.

[18] V. Chvatel. Greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4(3):233–235, August 1979.

[19] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, McGraw-Hill, 2000.

[20] N. Cressie. *Statistics for Spatial Data*. John Wiley & Sons, Inc., 1993.

[21] G. Danezis and R. Clayton. Introducing traffic analysis. In A. Acquisti, S. Gritzalis, C. Lambrinoudakis, and S. di Vimercati, editors, *Digital Privacy: Theory, Technologies, and Practices*. Auerbach, November 2007. Available `http://homes.esat.kuleuven.be/~gdanezis/`.

[22] H. A. David and H. N. Nagaraja. *Order Statistics*. John Wiley & Sons, Inc., New Jersey, third edition, 2003.

[23] R. Diestel. *Graph Theory*. Springer, 3 edition, 2005.

[24] G. Dobson. Worst-case analysis of greedy heuristics for integer programming with nonnegative data. *Mathematics of Operations Research*, 7(4):515–531, November 1982.

[25] W. Du, J. Deng, Y. S. Han, and P. K. Varshney. A pairwise key pre-distribution scheme for wireless sensor networks. In *Proc. 10th ACM Conference on Computer and Communications Security (CCS'03)*, pages 42–51, Washington, DC, USA, October 2003.

[26] W. Du, J. Deng, Y. S. Han, P. K. Varshney, J. Katz, and A. Khalili. A pairwise key predistribution scheme for wireless sensor networks. *ACM Trans. Information and System Security*, 8(2):228–258, May 2005.

[27] W. Du, R. Wang, and P. Ning. An efficient scheme for authenticating public keys in sensor networks. In *Proc. 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'05)*, pages 58–67, Urbana-Champaign, Illinois, USA, May 2005.

[28] P. Erdös, P. Frankl, and Z. Füredi. Families of finite sets in which no set is covered by the union of r others. *Israel Journal of Mathematics*, 51(1-2), 1985.

[29] L. Eschenauer and V. D. Gligor. A key-management scheme for distributed sensor networks. In *Proc. 9th ACM Conference on Computer and Communications Security (CCS'02)*, pages 41–47, Washington, DC, USA, November 2002.

[30] M. Evans, N. Hastings, and B. Peacock. *Statistical Distributions*. John Wiley & Sons, Inc., New York, 3 edition, 2000.

[31] K. Fazel and S. Kaiser. *Multi-Carrier and Spread Spectrum Systems*. Wiley, 2003.

[32] W. Feller. *An Introduction to Probability Theory and Its Applications*, volume 1. John Wiley & Sons, Inc., 1957.

[33] G. Gaubatz, J. P. Kaps, E. Ozturk, and B. Sunar. State of the art in ultra-low power public key cryptography for wireless sensor networks. In *Proc. 3rd IEEE Conference on Pervasive Computing and Communications (PerCom'05)*, pages 146–150, Kauai, HI, USA, March 2005.

[34] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming (web page and software), February 2009. `http://stanford.edu/~boyd/cvx`.

[35] V. Gupta, M. Millard, S. Fung, Y. Zhu, N. Gura, H. Eberle, and S. C. Shantz. Sizzle: A standards-based end-to-end security architecture for the embedded internet. In *Proc. 3rd IEEE Conference on Pervasive Computing and Communications (PerCom'05)*, pages 247–256, Kauai, HI, USA, March 2005.

[36] N. Gura, A. Patel, A. Wander, H. Eberle, and S. C. Shantz. Comparing elliptic curve cryptography and RSA on 8-bit CPUs. In *Proc. 6th Workshop on Cryptographic Hardware and Embedded Systems (CHES'04)*, pages 119–132, Cambridge, MA, USA, August 2004.

[37] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. In *Proc. International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-IX)*, pages 93–104, Cambridge, MA, USA, November 2000.

[38] T. Ho, R. Koetter, M. Medard, D. R. Karger, and M. Effros. The benefits of coding over routing in a randomized setting. In *Proc. IEEE International Symposium on Information Theory (ISIT'03)*, page 441, Yokohama, Japan, June/July 2003.

[39] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, Cambridge, 1985.

[40] D. Huang, M. Mehta, D. Medhi, and L. Harn. Location-aware key management scheme for wireless sensor networks. In *Proc. 2nd ACM Workshop on Security of Ad-Hoc and Sensor Networks (SASN'04)*, pages 29–42, Washington, DC, USA, October 2004.

[41] K. Jain. Security based on network topology against the wiretapping attack. *IEEE Wireless Communication*, 11(1):68–71, February 2004.

[42] Ian R. James. Products of independent beta variables with applications to connor and mosimann's generalized dirichlet distribution. *Journal of the American Statistical Association*, 67(340):910–912, December 1972.

[43] B. R. Johnson. An elementary proof of inclusion-exclusion formulas. *The American Mathematical Monthly*, 87(9):750–751, November 1980.

[44] D. B. Johnson, D. A. Maltz, and J. Broch. *DSR: The Dynamic Source Routing Protocol for Multihop Wireless Ad Hoc Networks*, chapter 5, pages 139–172. Addison-Wesley, 2001.

[45] J. Lee and D. R. Stinson. Deterministic key predistribution schemes for distributed sensor networks. In *Selected Areas in Cryptography (SAC2004), LNCS 3357*, pages 294–307, Waterloo, Canada, August 2004.

[46] T. Leighton and S. Micali. Secret-key agreement without public-key cryptography. In *Advances in Cryptology: Proc. CRYPTO'93, LNCS 773*, pages 456–479, Santa Barbara, CA, USA, August 1993.

[47] R. Leung, J. Liu, E. Poon, A.-L. C. Chan, and B. Li. MP-DSR: A QoS-aware multi-path dynamic source routing protocol for wireless ad-hoc networks. In *Proc. 26th Annual IEEE Conference on Local Computer Networks (LCN'01)*, pages 132–141, Tampa, FL, USA, November 2001.

[48] G. Lin and G. Noubir. On link layer denial of service in data wireless LANs. *Wireless Communications and Mobile Computing*, 5(3):273–284, May 2005.

[49] D. Liu and P. Ning. Establishing pairwise keys in distributed sensor networks. In *Proc. 10th ACM Conference on Computer and Communications Security (CCS'03)*, pages 52–61, Washington, DC, USA, October 2003.

[50] D. Liu, P. Ning, and R. Li. Establishing pairwise keys in distributed sensor networks. *ACM Trans. Information and System Security*, 8(1):41–77, February 2005.

[51] D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, Cambridge, 2003.

[52] H. Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–92, March 1952.

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

[54] F. Meshkati, H. V. Poor, S. C. Schwartz, and N. B. Mandayam. An energy-efficient approach to power control and receiver design in wireless data networks. *IEEE Transactions on Communications*, 53(11):1885–1894, November 2005.

[55] D. P. Palomar and M. Chiang. A tutorial on decomposition methods for network utility maximization. *IEEE Journal on Selected Areas in Communications*, 24(8):1439–1451, August 2006.

[56] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Dover, 1998.

[57] B. Parno, A. Perrig, and V. Gligor. Distributed detection of node replication attacks in sensor networks. In *Proc. 2005 IEEE Symposium on Security and Privacy*, pages 49–63, Oakland, CA, USA, May 2005.

[58] V. Paxson and M. Allman. Computing TCP's retransmission timer. RFC 2988, November 2000. `http://www.ietf.org/rfc/rfc2988.txt`.

[59] M. D. Penrose. On k-connectivity for a geometric random graph. *Wiley Random Structures and Algorithms*, 15(2):145–164, 1999.

[60] A. Perrig, J. Stankovic, and D. Wagner. Security in wireless sensor networks. *Communications of the ACM*, 47(6):53–57, June 2004.

[61] R. A. Poisel. *Modern Communication Jamming Principles and Techniques*. Artech House, 2004.

[62] M. Ramkumar and N. Memon. An efficient random key pre-distribution scheme. In *Proc. IEEE Conference on Global Communications (GLOBECOM'04)*, pages 2218–2223, Dallas, TX, USA, November/December 2004.

[63] T. S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall, 2 edition, 2001.

[64] S. W. Roberts. Control chart tests based on geometric moving averages. *Technometrics*, 42(1):97–101, February 2000.

[65] V. Rodriguez. Maximizing the ratio of a generalized sigmoid to its argument. Technical Report WICAT Tech. Rep. 02-010, Polytechnic University, May 2002.

[66] E. M. Royer and C. E. Perkins. Ad hoc on-demand distance vector routing. In *Proc. 2nd IEEE Workshop on mobile Computing Systems and Applications (WMCSA'99)*, pages 90–100, New Orleans, LA, USA, February 1999.

[67] J. Schiller. *Mobile Communications*. Addison-Wesley, 2000.

[68] C. Schurgers and M. B. Srivastava. Energy efficient routing in wireless sensor networks. In *Proc. Military Communications Conference (MILCOM'01)*, pages 357–361, Washington, DC, USA, October 2001.

[69] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, November 1979.

[70] W. F. Sharpe. *Investors and Markets: Portfolio Choices, Asset Prices, and Investment Advice*. Princeton University Press, 2007.

[71] E. M. Sozer, M. Stojanovic, and J. G. Proakis. Underwater acoustic networks. *IEEE Journal of Oceanic Engineering*, 25(1):72–83, January 2000.

[72] G. L. Stüber. *Principles of Mobile Communications*. Kluwer, 2 edition, 2001.

[73] P. Tague, M. Li, and R. Poovendran. Probabilistic mitigation of control channel jamming via random key distribution. In *Proc. 18th Annual IEEE International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC'07)*, Athens, Greece, September 2007.

[74] D. J. Thuente and M. Acharya. Intelligent jamming in wireless networks with applications to 802.11b and other networks. In *Proc. 25th IEEE Communications Society Military Communications Conference (MILCOM'06)*, pages 1–7, Washington, DC, October 2006.

[75] D. J. Torrieri. *Principles of Secure Communication Systems*. Artech House, Boston, 2nd edition, 1992.

[76] M. Čagalj, S. Čapkun, and J.-P. Hubaux. Wormhole-based antijamming techniques in sensor networks. *IEEE Transactions on Mobile Computing*, 6(1):100–114, January 2007.

[77] A. D. Wood and J. A. Stankovic. Denial of service in sensor networks. *IEEE Computer*, 35(10):54–62, October 2002.

[78] A. D. Wyner. The wire-tap channel. *Bell System Technical Journal*, 54(8):1355–1387, October 1975.

[79] W. Xu, K. Ma, W. Trappe, and Y. Zhang. Jamming sensor networks: Attack and defense strategies. *IEEE Network*, 20(3):41–47, May/June 2006.

[80] W. Xu, W. Trappe, and Y. Zhang. Channel surfing: Defending wireless sensor networks from interference. In *Proc. 6th International Conference on Information Processing in Sensor Networks (IPSN'07)*, pages 499–508, Cambridge, MA, USA, April 2007.

[81] Y. Yu, R. Govindan, and D. Estrin. Geographical and energy aware routing: A recursive data dissemination protocol for wireless sensor networks. Technical Report UCLA/CSD-TR-01-0023, University of California, Los Angeles, Computer Science Department, May 2001.

LIST OF PUBLICATIONS

Journal Publications

1. Patrick Tague and Radha Poovendran, Modeling Adaptive Node Capture Attacks in Multi-hop Wireless Networks, *Ad Hoc Networks*, vol. 5, no. 6, pp. 801-814, August 2007.

2. Patrick Tague and Radha Poovendran, A Canonical Seed Assignment Model for Key Predistribution in Wireless Sensor Networks, *ACM Transactions on Sensor Networks*, vol. 3, no. 4, October 2007.

3. Patrick Tague, David Slater, Jason Rogers, and Radha Poovendran, Evaluating the Vulnerability of Network Traffic Using Joint Security and Routing Analysis, *IEEE Transactions on Dependable and Secure Computing*, vol. 6, no. 2, April-June 2009.

4. David Slater, Radha Poovendran, Patrick Tague, and Brian J. Matt, Tradeoffs Between Jamming Resilience and Communication Efficiency in Key Establishment, *ACM Mobile Computing and Communication Review*, April 2009.

5. Patrick Tague, Mingyan Li, and Radha Poovendran, Mitigation of Control Channel Jamming under Node Capture Attacks, *IEEE Transactions on Mobile Computing*, available online January 2009.

Journal Submissions Under Review

1. Patrick Tague, Sidharth Nabar, James A. Ritcey, and Radha Poovendran, Jamming-Aware Traffic Allocation for Multiple-Path Routing Using Portfolio Selection, submitted to *IEEE/ACM Transactions on Networking*, February 2009.

2. Patrick Tague, David Slater, Guevara Noubir, and Radha Poovendran, Quantifying the Impact of Efficient Cross-Layer Jamming Attacks via Network Traffic Flows, submitted to *IEEE/ACM Transactions on Networking*, April 2009.

Conference Publications

1. Patrick Tague, Jooyoung Lee, and Radha Poovendran, A Set-Covering Approach for Modeling Attacks on Key Predistribution in Wireless Sensor Networks, in *3rd International Conference on Intelligent Sensing and Information Processing* (ICISIP'05-Bangalore), December 2005.

2. Patrick Tague and Radha Poovendran, A General Probabilistic Model for Improving Key Assignment in Wireless Networks, in *4th International Symposium on Modeling and Optimization in Mobile, Ad-hoc, and Wireless Networks* (WiOpt'06), April 2006.

3. Patrick Tague, Mingyan Li, and Radha Poovendran, Probabilistic Mitigation of Control Channel Jamming via Random Key Distribution, in *18th IEEE International Symposium on Personal, Indoor, and Mobile Radio Communication* (PIMRC'07), September 2007, **Best Student Paper**.

4. Patrick Tague, David Slater, Guevara Noubir, and Radha Poovendran, Linear Programming Models for Jamming Attacks on Network Traffic Flows, in *6th Intl. Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks* (WiOpt'08), April 2008.

5. Patrick Tague, David Slater, Jason Rogers, and Radha Poovendran, Vulnerability of Network Traffic under Node Capture Attacks using Circuit Theoretic Analysis, in *27th IEEE Conference on Computer Communications* (INFOCOM'08), April 2008.

6. Patrick Tague, Sidharth Nabar, Jim Ritcey, David Slater, and Radha Poovendran, Throughput Optimization for Multipath Unicast Routing Under Probabilistic Jamming, in *19th IEEE International Symposium on Personal, Indoor, and Mobile Radio Communication* (PIMRC'08), September 2008.

7. Patrick Tague and Radha Poovendran, Modeling Node Capture Attacks in Wireless Sensor Networks, in *46th Annual Allerton Conference on Communication, Control, and Computing*, September 2008.

8. David Slater, Patrick Tague, Radha Poovendran, and Brian J. Matt, A Coding-Theoretic Approach for Efficient Message Verification over Insecure Channels, in *2nd ACM Conference on Wireless Network Security* (WiSec'09), March 2009.

9. David Slater, Patrick Tague, Mingyan Li, and Radha Poovendran, A Game-Theoretic Framework for Jamming Attacks and Mitigation in Commercial Aircraft Wireless Networks, in *AIAA Infotech@Aerospace Conference*, April 2009.

VITA

Patrick Tague received his B.S. degrees in Mathematics and Computer Engineering from the University of Minnesota in 2003. He joined the Network Security Lab at the University of Washington in 2003 and received his M.S. degree in Electrical Engineering in 2007. His research interests focus on modeling and evaluating vulnerabilities in wireless ad-hoc and sensor networks and designing robust wireless networking protocols. He was awarded a DoD/NSA Information Assurance Scholarship in 2005, the Spirit of Community Award in 2006 in recognition of service to the Electrical Engineering Department at the University of Washington, the Best Student Paper Award at the IEEE PIMRC Symposium in Athens, Greece, in 2007, and the Yang Research Award in 2009 in recognition of outstanding graduate research in the Electrical Engineering Department at the University of Washington.