On Counter-Adversarial Resilience in Permeable Networked Systems

by

Hans Walter Behrens

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved May 2023 by the
Graduate Supervisory Committee:

K. Selçuk Candan, Co-Chair
Gail-Joon Ahn, Co-Chair
Dragan Boscovic
Adam Doupé
Stephanie Forrest
Boris Gelfand

ARIZONA STATE UNIVERSITY

August 2023

ABSTRACT

In recent years, a flood of devices has permeated our personal and professional lives, with increasingly interconnected networks playing an ever-growing role in day-to-day activities. As these systems expand in both importance and complexity, their value to attackers and their attack surface simultaneously increase. In this dissertation, I argue that traditional defensive approaches fail to acknowledge this changing landscape. Instead, by focusing on the twin concepts of permeable networks and counter-adversarial behavior, defenders will be able to achieve better outcomes. The dependencies, interactions, and relationships among the growing corpus of connected devices create complex systems that are difficult for both users and operators to reason about. Consequently, despite heightened awareness of security risks, the rate and scale of data breaches continue to accelerate. This underlying complexity renders networks permeable to attackers. In parallel, interest in low- and no-trust distributed computing, such as federated learning, ad hoc networking, and blockchain systems, has been on the rise. In these contexts, users and devices interact cooperatively toward common goals while contending with potentially adversarial participants. Inherently permeable, these cooperative systems benefit from adopting counter-adversarial behaviors. By discarding the traditional goal of strict prevention, defenders can shift their focus to counter-adversarial behaviors instead. These behaviors aim to limit attackers' choices, exhaust their resources, or reduce their effectiveness, collectively disincentivizing attacks. This dissertation leverages these dual concepts to explore counter-adversarial behaviors in the context of permeable networked systems. It further describes methods such as information partitioning, ensemble fusion, and resilience analysis that enable novel counter-adversarial strategies. In doing so, this dissertation provides solutions to the challenges faced by increasingly prevalent permeable systems. Collectively, these form a foundation for the development of more resilient communication architectures.

DEDICATION

To my brilliant wife, whose unwavering support has been the foundation for everything. Thank you for lending me the strength I needed.

To my cherished children, whose unconditional love comforted me during the toughest times. Your laughter and smiles are a reminder of the inspiration behind this journey.

This dissertation serves as a commemoration of the love and encouragement that sustained me, and a tribute in appreciation for those who are my world.

ACKNOWLEDGMENTS

This journey has been filled and defined by people, and they deserve my gratitude.

To my advisor and co-chair Dr. Candan, for his consummate professionalism, incandescent intelligence, and unwavering support throughout this effort. He went to bat for me with the department before we had even met, suffered many long nights sifting through my verbose writing, and always matched or exceeded the effort I invested in a project.

To Dr. Ahn, my committee co-chair, for agreeing to this experiment of having a student under CASCADE instead of a single lab, and for giving me the freedom and time to explore different areas until I found one that fit.

To Dr. Boscovic and the Blockchain Research Lab, for my academic introduction to the field of blockchain. In particular, special thanks to Francis for expanding my boundaries by pushing me in new directions, and to Nakul, for his patience as we hashed out the details of erasure coding.

To Adam, Alice, Arvind, Carlos, Debbie, Erik, Faezeh, Fish, Haehyun, Jackie, JJ, Mike, Tiffany, Yan, and the rest of the SEFCOM crew: my sincere appreciation for never making me feel for a moment that I didn't 100% belong.

To Stephanie and the rest of the Biodesign folks, especially Joe, Kirtus, Pemma, and Tony, for exposing me to many fascinating avenues of computer science off my beaten track. Special thanks to Steph for treating me as a peer before I felt like one, and for never holding back advice when I needed to hear it.

To Boris, for giving me the flexibility to explore my own ideas when they were plentiful, feeding me new ones when they weren't, for acting as a constant sounding board whenever needed, and for asking such good questions. And to the rest of the folks at LANL and the NWCAL, for trusting that Boris knew what he was doing.

To Dr. Sapino, for being a warm and welcoming presence throughout my time at

ASU, always appearing with a flurry of delicious treats, kind words, and practical wisdom for the confused grad students.

To the original Emitlab crew of PhDs: Jung, Mao-Lin, Parth, Shengyu, Sicong, Silvestro, Xinsheng, Xilun, and Yash, who made me feel welcome and guided me through my reintroduction to academia, and to Tasneema for picking up the mantle of "lab admin" without complaint.

To the DataStorm team, especially those who dove deep into the system, Ashish, Fateh, Magesh, and Shubhodeep. We couldn't have built what we did without your willingness to wrangle with legacy code written by long-graduated students.

To the Topl research team, for reminding me that academic progress can be driven by individual curiosity and not grant money alone. And to the rest of the crew, for the moral support as I navigated my PhD journey.

To Mao-Lin, my PhD sibling, for acting as the pragmatist and level-headed voice of reason. For always being willing to pull your weight (and more) when the work piled up, and for putting things in perspective when it became overwhelming.

To Yash, for the friendship, the knowledge, and the honesty. For not laughing when I asked to teach me about L-2 distance, for explaining concepts more effectively than many of my professors, for acting as a pressure relief valve when things were tough, and for the reality checks by the City Hall fountain when necessary.

To Mehrdad, for being my first friend at ASU. From our first Diamondbacks game, to feature embeddings of the local bar scene, to the challenges with your uncle and with your visa, to your wedding! We went through the extrema of PhD life together, and it would have been infinitely harder without you. Your determination and resilience (and advice over drinks) revealed the path forward when I couldn't always see it.

And to the many people whose names I was forced to omit by the Grad College's 2-page limit, your influence has been no less valued – thank you!

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

Historically, the network security domain has emphasized the importance of strong perimeter defenses [97], with less attention paid to considerations of adversarial behavior within that perimeter. However, recent and growing evidence suggests that the permeability of such perimeters is greater than expected [61]. Similarly, networking protocols emphasize efficiency and robustness to random failure, with adversarial resilience seldom considered (and often ignored). The rise of insider threats and advanced persistent threats (APTs) has further underscores the need to reevaluate conventional security practices.

As networks increase in size and complexity, and as the individual devices involved become more capable and computationally sophisticated, ensuring a sterile internal network poses enormous challenges. With many points of failure, often-centralized security policies, and a drastic increase in internal threats such as phishing, these designs can fall victim to overly-optimistic assumptions. Additionally, the proliferation of IoT devices with varying levels of security implementation and oversight has further complicated the process of maintaining a secure network [69].

Recent developments in artificial intelligence further democratize access to adversarial skills and toolsets [26]. When using large language models (LLMs), users need to merely describe the direction or method of attack to receive reasonably effective resources even if they have little practical expertise themselves. This easy access to advanced attack methodologies has put additional pressure on defenders to stay ahead

of the curve and find innovative solutions to counteract such threats.

Fundamentally, the increasing success and availability of offensive behavior speaks to a fundamental dichotomy: offense is easier than defense [50]. A defender must foresee and protect against any conceivable attack, while an attacker need only find a single gap in their opponent's metaphorical armor to succeed. As this imbalance continues to play out and be reinforced in practice, consensus has begun to shift on how best to approach the task of defense at a fundamental level.

Drawing a metaphor to nature, environments with pervasive threats tend to produce a more nuanced approach, such as those seen in immunological response [60]. A coarse-grained external defense such as the skin filters many threats, followed up by layers of additional physical defenses such as hair, mucus, and so on. In expectation of attackers breaching these, a broad spectrum of both systemic and targeted internal mitigations and countermeasures may be deployed as appropriate. This heterogeneity complicates management, but improves security [150].

Recent technical approaches are beginning to appreciate this perspective. For example, we see a change in focus to adopt a zero-trust or perimeter-free approach, in which each individual device is responsible for maintaining its own security rather than making assumptions [125], e.g., about the correctness of incoming messages. Under this paradigm, the collective behavior of individual nodes produces emergent security characteristics, at the cost of increased responsibility at the node level. This shift in mindset helps to address the increasingly complex and interconnected nature of modern networks.

Some network paradigms, such as ad hoc and device-to-device communication, may be inherently incompatible with strong-perimeter approaches since devices may

frequently leave and join the network at will. As mobile, low power, and pervasive devices see growing application to real-world problems, the importance of addressing context-relevant security concerns only increases. This has led to the development of new security models that prioritize adaptability and resilience in the face of changing circumstances, including the presence of active attackers within the network [32].

Despite the rapid rise of pervasive networked devices, practical real-world adoption of ad hoc communication protocols remains slow. One possible reason for this reluctance arises from the security concerns of such approaches: every intermediate device might tamper with, improperly discard, or attempt to read messages that traverse it. Ensuring the integrity and confidentiality of communications in such dynamic environments has thus become a critical area of research.

Hierarchical, distributed, decentralized, and multi-layered network security approaches have grown in relevance to meet the challenges of this shifting landscape [128]. Whether by design, in the case of zero-trust and ad hoc systems, or by accident, in the case of vulnerabilities or internal threats, most practical networks are *permeable* – that is, there exists a point of entry for the adversary to access the network. This concept of permeability acts as a foundational assumption for all the work discussed in this dissertation.

Thus, the improvement of communication security, especially counter-adversarial resilience, remains a critical building block of the global shift to increasingly connected networked systems.

1.1 Counter-Adversarial Resilience

In this era of rapidly evolving technology, the challenge of maintaining secure and resilient network systems is greater than ever before. The proliferation of connected

devices and the increasing complexity of networks necessitate a fundamental shift in our approach to network security. Traditional approaches have proven to be increasingly ineffective in the face of more sophisticated and novel threats.

Choosing the term "counter-adversarial" carefully, we may differentiate from purely preventative behaviors. Instead, we aim to reduce the adversary's power, exhaust their resources, and limit their behavior, but do not try to prevent their attack due to the permeability assumption. This is not to devalue the perimeter-focused aspect of defensive practice, but to highlight the differences brought about by permeability.

As we will explore in this work, novel approaches such as semantic and spatial partitioning, distributed fusion, and automated pattern analysis offer promising avenues for enhancing counter-adversarial resilience in permeable networks. The implementation of these strategies varies depending on the specific application and threat model, but their integration into a holistic security posture will be crucial to the ongoing and future reliability of networks systems.

In this dissertation, we explore this landscape in detail and present several novel strategies that contribute toward this ambitious goal. Through rigorous analysis and experimentation, I demonstrate the effectiveness of these approaches in enhancing network security and resilience.

1.2 Data Separation

As networked devices proliferate, it becomes increasingly important to ensure that they communicate on topics related to their function. By enforcing this segmentation (or partitioning) of concerns, adversaries cannot leverage the exploitation of a device in one scope to interfere with messages in another scope. This focused communication

limits the potential impact of compromised devices and mitigates the risk of information leakage across different functional domains.

Often, this segmentation is accomplished through the application of encryption, in either symmetric or asymmetric modes [106]. By encapsulating messages in a mathematically-impervious shell, the adversary cannot inject, read, or modify messages beyond the scope of the devices it has compromised. Encryption techniques have evolved to provide a higher level of security, including the use of perfect forward secrecy, which ensures that the compromise of a single encryption key does not put all past and future encrypted data at risk.

However, encryption carries other drawbacks that may make it undesirable. Symmetric encryption is efficient, but relies on pre-shared keys that are vulnerable to capture. That is, when a device is compromised, the communication channels for all pre-shared keys that it knows are likewise compromised. Asymmetric encryption solves that problem, but doesn't scale well – managing which key goes to which node across thousands or millions of devices is a challenging problem, and carries substantial added computational costs related to exponentiation. New cryptographic schemes such as post-quantum cryptography, that are being developed to ensure the long-term security of encrypted communications, do not address these fundamental challenges.

Instead, by leveraging a relatively-underused cryptographic primitive, the homomorphic hash, devices can share only the minimum amount of data needed using an idea similar to that used by spies: clandestine cell theory [43]. If each individual node knows only a little information and needs to put it together with data from its fellow nodes to accomplish its task, then each individual node's capture becomes much less damaging. This approach helps to decentralize trust and minimize the impact of

compromised nodes on the overall security of the system.

This approach can be used to constrain adversarial behavior by e.g. limiting transmissions to only topically-relevant queries [15]. Protecting communication channels by their topic, or *semantic partitioning*, improves energy efficiency by reducing transmission costs for non-adversarial nodes under attack. At the same time, the adversary would not be able to make use of the network by sniffing network traffic due to the one-way nature of the hash functions.

1.3 Spatial Partitioning

Semantic partitioning allows for protection of data streams, but the sacrifice of message privacy might be problematic depending on the application. Consider, as an example, a messaging program used to coordinate democratic protests against an authoritarian regime. Existing cellular infrastructure might be disabled or monitored by government agents, but messages passed from phone to phone would still arrive.

However, if an intermediate phone were captured and those messages were intercepted, the recipients' safety could be in jeopardy. Instead, what if the identity of the recipients were *hidden*? Rather than addressing messages to specific recipients, messages could instead be sent to pre-defined locations. Even then, such information might reveal the identities of the recipients or endanger their safety by revealing their location [17]. However, by hiding the target location except to those within the targeted region, participants could then send messages while preserving recipient privacy (and thus, safety).

By leveraging the location information embedded in each message, delivery efficiency may be improved at the same time. Although routing cannot directly target

the hidden location, incremental progress towards the intended region can drastically reduce the inefficiencies inherent to e.g. flooding approaches. Advanced routing algorithms and techniques such as geocasting can be employed to optimize message delivery while minimizing overhead and preserving privacy.

1.4 Distributed Fusion

While data compartmentalization can conceal and protect information, it can also be used to reveal patterns of behavior as well. By collecting data and looking for missing links, it becomes practical to detect and mitigate malicious behavior even when the adversary has a nearly-equal footing with defenders [16].

For example, the previous approach is useful for disseminating information, issuing queries, or sending messages when the recipient location should not be known by the adversary. However, it doesn't help with replies – this requires the introduction of a new concept. If each node in the network is created with a seeded pseudorandom number generator (PRNG) and then given the same seed, every node will produce the same value. Leveraging this property allows the introduction of a new concept called a Trémaux forest.

This structure allows nodes to pool messages efficiently, without ever knowing the entire network's structure or using the same routing tree twice. Consequently, adversaries cannot compromise specific nodes in the hope of partitioning the network. Additionally, if the attacker chooses to drop messages intercepted by adversarial nodes, the tree-like properties allow defenders to identify which nodes precisely did the pruning. Since only an adversary would engage in this pruning behavior, they may then flag the misbehaving nodes and ignore any readings they provide.

In combination, these techniques allow for the collection of replies from a spatially-distributed network of ad hoc nodes in an adversarially-resilient way. This could be useful, for example, in collating information from a deployment of sensors that are monitoring air quality for pollution. Any industrial site that disables or tampers with reporting would be detected immediately, ensuring compliance. Here, the non-compliant site could be considered an adversarial node that *started* within the deployment, the very paragon of permeability.

Moreover, distributed fusion also provides a robust framework for integrating data from disparate sources and maintaining system functionality in the presence of adversarial behavior. This approach can be applied to various other scenarios, such as disaster response or military operations, where the ability to share information and coordinate efforts is crucial for success.

1.5 Patterns of Resilience

The approaches described thus far each address one aspect of adversarially-resilient networked systems. Based on the specific application, security, and resilience criteria, the user must choose a strategy or protocol appropriate to their needs. However in each case, making that choice requires some experience and understanding of the tradeoffs involved. Additionally, the user must understand (or predict) the expected threat model. Judgment failures along any of these axes may potentially reveal weaknesses in the resulting architecture.

To prevent these failures, or when the network designer does not know these criteria, shifting to automated reasoning methods mitigates risks arising from human factors [134]. The lack of training data poses a serious challenge to the development

of such approaches, however. Since red team data and other recorded attacks are network-specific and difficult or expensive to acquire, their availability is limited, complicating automated approaches.

By instead relying on large volumes of synthetic attack data generated by stochastic agent-based simulations, patterns begin to emerge. By quantitatively evaluating the success of each randomly-generated attack, this approach can identify, rank, and report structural or architectural flaws in a network's design to the user, enabling iterative improvement [18]. Machine learning techniques can then be employed to identify patterns and trends in attack data, facilitating the development of more effective defensive strategies.

Users may also compare similar designs to assess the effect of proposed changes on a network's security. In real-world systems, where architectural adjustments must be overlaid onto existing legacy infrastructure, reasoning about the impact of such changes can be extremely complex. By summarizing the pre- and post-change topologies quantitatively, these kinds of iterative deployments can be secured much more economically than conducting cybersecurity assessments such as red-team penetration tests after every change.

In conjunction with the novel defensive protocols and behaviors described previously, this automated analysis method completes the missing piece of the puzzle in building a toolkit for improving the adversarial resilience of networked systems.

Chapter 2

SEMANTIC PARTITIONING

Wireless sensor networks and other power-efficient devices fill increasingly important roles in modern society. At the same time, they also face increasing internal and external threats, such as node capture or protocol disruption by adversarial agents. Providing reliable and secure service in the face of these challenges remains an ongoing problem, and one that is only exacerbated by the computational and power constraints imposed on these devices. In this chapter, I first introduce the concept of on-demand topic channels in the context of ephemeral wireless sensor networks. Then, building on this concept, I introduce three novel messaging protocols to provide secure, authenticated communication between a sensor network and an authorized user while also providing resilience from accidental or adversarial disruption. These protocols leverage homomorphic hashing in innovative ways to trade secrecy against network and computational costs in on-demand topic channel authentication. Finally, I compare and contrast the costs of these protocols, and show that hash-based protocols provide significant implementation-independent improvements to network resilience. This chapter captures results presented in [15].

2.1 Overview

The Internet of Things (IoT) refers to the networks of low-cost, low-power devices used to sense and manipulate the physical world. As energy efficiency increases, and cost decreases, the proliferation and applicability of these devices continues to increase.

2.1.1 Ad-Hoc On-Demand Sensors

Although types of such devices are wide-ranging, I will focus my attention on one particular sub-type, wireless sensor networks (WSNs) supporting *on-demand* queries. These are spatially-distributed deployments of low-powered devices [147], which collectively sense their environment and record or report that information to a user, system, or database over a wireless medium upon request. In contrast to many IoT applications, these do not generally manipulate their surroundings directly. Since these devices' primary responsibility lies in the collection and delivery of this data, and not in transformation or analysis, the computational and power requirements of these devices are much lower than traditional computers. Although the network topology of such sensors may vary, *ad hoc* or *mesh-based* deployments are quite common, as these require neither precision placement of sensors nor powerful coordinator nodes to support evolving network topologies [131].

Due to these devices' relatively low costs, they have broad appeal in solving a wide variety of problems. Industrial process monitoring, for example, can be simplified with robust sensors placed in areas where human monitoring would be difficult or dangerous. Simple logistical trackers permit long supply chains with very little loss, at a scale impossible for manual methods. Low-cost sensors permit users to track real-time changes in their environments in ways not previously possible.

2.1.2 Case Study: Disaster Response

The application of IoT systems for disaster response provides a good case study: the use of such systems has been extensively studied in the literature [95, 120, 123], but significant questions remain unanswered.

Opportunities

The deployment of ad hoc devices in response to natural disasters provides emergency personnel with a much better situational understanding in areas where underlying infrastructure may be damaged or nonexistent. WSN deployments in this context are intrinsically ephemeral, requiring deployment in challenging environments while also providing reliable service for the duration of the response. In many cases, disasters cannot be predicted early enough to allow proactive deployment of sensor networks; instead, emergency responders requiring WSN coverage must rely on reactive deployment. Since existing power and communication infrastructures may be damaged, destroyed, or nonexistent, any sensor network should be entirely self-contained to maximize deployment flexibility. Once deployed, these finite power resources must be managed efficiently to ensure that services can endure throughout the critical response period. Currently, disaster response relies heavily on sensing satellites to fill this requirement [35], but high costs, delayed deployment, and coarse sensing resolution suggest an alternative approach could offer substantial benefits. Localized deployments of comparatively high-resolution WSNs mitigate satellites' associated disadvantages, though they are also more susceptible to power and communication disruptions on the ground than space-based platforms. WSN deployments may be achieved directly by emergency personnel, over a large area via aerial dispersion, or by drone deliveries.

Challenges

Damaged infrastructure may be effectively bypassed through the use of low-cost, battery-powered nodes communicating over an ad hoc wireless network, but the advantages offered by self-contained WSNs come with additional costs. Constraints

on size, complexity, computational resources, and battery life must now be considered. As physical size increases, deployment logistics become more complicated, potentially restricting available delivery methods. As computation and wireless communication needs increase, power drain on the devices' batteries also grows; once a device has been depleted, it may be logistically impossible to recharge or refurbish that unit. Additionally, sensors may intermittently or permanently fail for a number of reasons. Beyond battery depletion, nodes may experience unplanned hardware failure, or an interruption in communications with neighboring nodes. Criminals seeking to exploit a disaster may serve in an adversarial role, attempting to subvert or compromise sensor coverage. The deployed network should therefore be resilient, and remain available for readings despite such failures or attacks. To mitigate the risk of failures, many deployment strategies call for the inclusion of redundant nodes [82], so that some proportion of the participating nodes can fail before network degradation begins.

Furthermore, disaster response activities may necessitate layered, heterogeneous deployments of different sensors. Search and rescue operations, infrastructure monitoring, damage assessments, or warning systems may each require different sensor types, spatiotemporal data resolutions, or security requirements. Despite this wide range of needs, logistical costs associated with delivery may constrain responders to very few, or even one single deployment to cover all these demands.

Emergency responders may need to query different aspects of this heterogeneous deployment relevant to their team's goals, without impacting the activities of other teams by unnecessarily depleting node resources, or sifting through non-relevant data in time-sensitive situations. Due to the diverse nature of these IoT deployments, various streams of information may be available to responders in a disaster scenario. However,

not all data will be relevant to all responders; by compartmentalizing information and releasing only to users who are both relevant and authorized, operational security and network efficiency can both be improved. I use the term *topic channels* to refer to these semantically-grouped heterogeneous sets of sensors. Although some preliminary research in this area has been conducted [130], topic channels have thus far been paired with publish-subscribe frameworks, in an event-driven approach that does not take changing user demand for information into account. By adapting such topic channels to an on-demand paradigm, the number of sensor readings and transmissions by the network may be drastically reduced, security vulnerabilities prevented, and total network service life extended.

2.1.3 Contributions

Given this context, I aim to answer the following questions: (1) Can I provide a protocol supporting on-demand querying of topic channels within a heterogeneous, ad hoc sensor deployment? (2) Can the security of the protocol be guaranteed against a wide variety of attacks and adversarial scenarios, while also maintaining network availability and robustness throughout? (3) Can the protocol support post-deployment modifications to topic channel assignment, without compromising the security of the network? (4) Can the per-node efficiency of the protocol be increased through the use of less computationally-intensive approaches? (5) Can I minimize inter-node message transmission to reduce the network's total energy usage? To address these challenges, I propose and compare three novel network messaging protocols that support adversarially-resistant on-demand topic channels:

- The first protocol is a novel hash-based protocol, leveraging multicast communi-

cation to simplify delivery. This approach combines the security and efficiency benefits of standard encryption protocols, but sacrifices the secrecy of the messages. However, limitations on response authentication in this protocol may leave the network open to exhaustion attacks during the response phase.

- The second protocol introduces the concept of channel validation, which leverages on-node computations by channel members to mitigate the impact of falsified responses sent by adversarial nodes.

- The third protocol builds upon the previous ideas by introducing chain validation, which uses structural properties of the network to constrain message propagation, further reducing power consumption during the response phase.

2.2 Related Work

2.2.1 Topic Channels

The publish-subscribe model [67] describes a system in which streams of information are produced by participating data sources. Data consumers can choose to consume these streams of information based on the topics they are interested in; therefore, these streams are sometimes referred to as *topic channels*. A key point of these systems is that data is generally produced independently of the number of subscribers; that is, it is fundamentally event-driven, or push-based. In applications where real-time perception is desired, this is an ideal model. For example, [130] explores the applications of this technique in low-powered deployments within the disaster response context.

Conversely, in some applications, data is not collected or generated except when it is requested, or pulled, by the consumers. To differentiate this inverted approach,

which could also be termed “subscribe-publish", from more commonly-understood meanings of publish-subscribe and prevent confusion, I refer to this as “on-demand" or “query-response".

This approach is especially beneficial when the number of requests is greatly outnumbered by changes in the sensed data, as it dramatically reduces the amount of communications over the network. I leverage this second model to increase deployment longevity by reducing message transmissions, and to strengthen security by preventing compromised nodes from communicating across the network except in response to valid queries, thereby preventing certain types of DoS attacks.

2.2.2 Hashing Efficiency

Hashing is computationally more efficient on low-powered devices than either message authentication (MAC) or encryption. Although special instructions and optimizations exist to make encryption more efficient on certain chips, these are rarely present on very low-powered devices, such as the sensor nodes used in WSN deployments. These conclusions are supported by the findings of Pereira *et al.*, in their comprehensive benchmarking of WSN-suitable chips for these three classes of functions [117]. Note that the authors did not explicitly evaluate homomorphic hash functions, whose performance characteristics may differ from more traditional collision-resistant one-way functions.

Given the relative efficiency of hashing in the context of low-powered devices, it is not surprising that the use of hashing to increase performance and longevity of ad hoc networks has previously been explored. [33] makes use of variable rounds of hashing to establish secure session keys for encrypted communication between neighboring

nodes; this contrasts with the approach proposed in this chapter, which makes use of deterministic hashing rounds, without encryption. Additionally, once a node in their model has been compromised, it gains the ability to compromise message integrity or exhaust the available resources of the network as a whole, since there is no centralized source of authority.

[92] examined the applications and efficiency advantages of homomorphic hashing for IoT networks in the context of message aggregation. In this application, messages are aggregated within the network itself, using homomorphic cryptographic primitives to secure this process and reducing the number of messages which must propagate back to the base station. Their approach provides several benefits over previous methods, but does not support on-demand topic channels.

2.2.3 Adversarially-Resilient Messaging

Other examinations of adversarially-resistant message delivery and authentication have also appeared in the literature. [91] approaches the problem using cryptographic primitives with a strong adversary, but does not address the problem of node capture attacks.

Message routing itself provides an attack surface for adversaries to exploit. [66] proposed Ariadne, a distributed protocol for securing message routing for ad hoc networks. However, their approach is restricted to the routing messages themselves, which are used to 'learn' the topology of the network. Additionally, it requires either time synchronization, which is rarely possible in low-power deployments, or shared secret keys, which are vulnerable to node capture attacks.

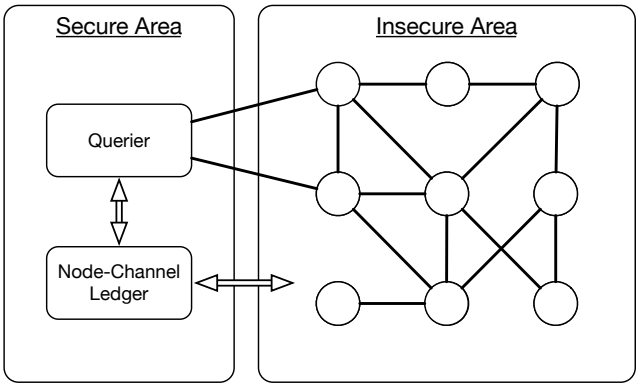


Figure 2.1: Example network model

2.3 Problem Formulation

Before describing the protocols in detail, I first establish the constraints and assumptions that the protocols will obey.

2.3.1 Network Model

As seen in Figure 2.1, an agent in the network can take one of three possible roles: a *node-channel ledger*, a *querier*, or a *sensor node*. I assume that the node-channel ledger and the querier are not resource constrained, as might be represented by a server or local workstation. In contrast, a sensor node has significant computational and power constraints.

Topology

I further assume an implementation-agnostic, graph-like network model, in which each agent may communicate directly with its neighbors, or indirectly with one or more other nodes through multicasting. However, the protocols do not specify the underlying communication or hardware requirements; I assume the network supports

flood-based stateless multicasting, in which the overall topology is not learned by the constituent nodes. It is reasonable to consider this case as the worst-case scenario for multicast implementations; more-efficient approaches [44, 129] may further increase the efficiency of the multicast-based protocols by reducing redundant communication incurred by message broadcasts.

Communication Medium

Although the device network could be either wired or wireless, the former is very rarely seen in ad hoc sensor deployments, and I therefore assume agents communicate wirelessly. A common characteristic of wireless communication is that transmitting messages is significantly more expensive than receiving messages. I therefore primarily attempt to minimize the number of message transmissions by non-compromised sensor nodes.

Ephemerality

All sensor nodes in the network are considered *ephemeral* and *fragile*. Ephemerality describes nodes which may temporarily drop out of the communication network, and reappear later at their previous location; for example, a temporary disruption of service corresponding to network interference, or a power-saving sleep cycle. Fragility is used to describe the fact that nodes may drop out of service unexpectedly and permanently; this could correspond with a depleted battery or an unrecoverable hardware error in the device. Although these may lead to network partitioning, the proposed protocols can efficiently support partitioned networks with very little efficiency loss.

Secure vs. Insecure Areas

I specify two different regions of the network, the *secure area* and the *insecure area*. The secure area corresponds to a well-protected location within the deployment area, such as a field office; the insecure area corresponds to the field deployment itself and represents the area that cannot be reasonably protected against intrusion by an adversary. All sensor nodes deployed in the network are assumed to be located in the insecure area, while both the querier and the ledger are assumed to be in the secure area. I further assume that all communication between participants, regardless of location in a secure or insecure area, is susceptible to an adversarial threat model. However, participants that are located within a secure area cannot be directly compromised by the adversary (either physically or remotely). Conversely, the adversary is able to compromise existing nodes anywhere within the insecure area. Additional capabilities are detailed in Section 2.3.3.

Entrance Nodes

I also specify the concept of *entrance* or bridge nodes. These represent sensor nodes (usually at the periphery of the deployment) that serve as gateways for messages transitioning between the secure and insecure areas. Larger number of entrances reduces the likelihood that the network will be partitioned such that one of the subgraphs is unreachable to either the ledger or the querier. Furthermore, more entrances improves "wear leveling" on these nodes, as messages being transmitted back to the querier may transition through any of the querier's entrance points, rather than forming a transmission bottleneck on only a few nodes. Entrance redundancy also ensures that if the adversary compromises one of these nodes, it is less likely to be

able to sever communication with the network.

2.3.2 On-Demand Sensor Networks

Topic Channels

Given a set of heterogeneous sensor nodes $\mathcal{N}$, with each member N_i containing a set of distinct sensors $\mathcal{S}_i$, I define the concept of a *on-demand topic channel* (ODTC) T as a subset of the set of possible sensor types $\mathcal{S}$, where $\mathcal{S} = \bigcup_{i=1}^{|\mathcal{N}|} \mathcal{S}_i$. I also introduce the concept of a *response rate* to a given ODTC T. This describes the percentage of total nodes in $\mathcal{N}$ which have sensors that are members of T. Note that this explicitly does not correspond to the number of nodes which *actually* respond to a query against a given ODTC, for example due to the proposed concepts of ephemerality and fragility, but to the nodes which *should* respond if all such nodes were functioning.

Node Deployment

Each node is assigned a unique identifier, such as a UUID [83], prior to being deployed into the field. This simplifies the job of the ledger in tracking which secure communication channel is associated with each node, and permits the ledger to address communications directly to specific nodes. Each item in a node's sensor suite is assigned a simple numeric ID, for later use when assigning topic channels. The capabilities of a node, in terms of sensor count and capabilities, must be known to the ledger in order for sensor-topic mappings to be created. Alternatively, if assignments by overall node class are preferred, then distinct node hardware configurations can be considered as single sensor types, and assigned to various topic channels as appropriate.

New sensor nodes may be added into the deployment at any time. To support this,

there must be an existing encrypted channel to communicate securely and secretly with the deployed node, for preliminary handshaking. Generally, this is accomplished by either loading a key during manufacturing, installing a key via physical access prior to deployment, or by using a session key-based communication protocol. Existing approaches may be adopted to establish the secure communication channels in order to satisfy this prerequisite [118].

Since use of an encrypted link may be more expensive to use relative to the proposed hash-based approaches, I aim to minimize messages along this channel, primarily reserving this method for channel assignment or other sensitive administrative tasks.

Querier Authentication

When a querier wants to conduct a query against a ODTC, they must first contact the ledger to request permission to query that channel. This exchange should be conducted securely, as the data exchanged may contain sensitive information which would compromise the network if leaked; for simplicity, I assume an existing secure two-party communication protocol is used. A successfully-authorized querier provides to the ledger a set of one or more topic channels to query, as well as its own credentials (if any) for authentication. Upon receipt and validation of a query request, the ledger returns to the querier the necessary information to issue one query; this process is described in further detail in Section 14.

Messages & Requests

To facilitate message delivery, I will consider two possible network knowledge models. The first assumes that the network topology and changes to it are known in real

time to the querier and ledger. I refer to this method as the "optimistic" or "known" model. In the second case, I assume a true ad hoc deployment, where the network topology (except for an agent's immediate neighbors) is unknown to the participants, nor can it be efficiently learned by the nodes. I refer to this as the "pessimistic" or "ad hoc" model. I also describe four potential message delivery methods, used in different phases of the proposed protocols.

One Message Direct Routing (OD) This approach describes the process of a single message traversing the network between two agents, where the message recipient is predefined. In the case where the network topology is known, and the transmitter is not computationally bound, I also allow the shortest path through the network to be computed. Otherwise, when the topology is not known, direct routing is only possible under paradigms in which the network topology and routing rules are learned; I do not assume such an approach.

N Message Direct Routing (ND) This approach describes the process of delivering distinct messages from one source to multiple recipients. In a general case, this approach can be thought of as an extension to OD, iteratively repeated proportionate to the number of unique messages.

One Message Multicast Routing (OM) This approach describes a single message being multicast to all possible recipients. Each node which receives this message will store a copy locally for potential on-node processing, and will then forward the received message to all neighbors except the neighbor from which it received that message. Other more efficient ad hoc multicast implementations, such

as those described by [70], may also be used.

N Message Multicast Routing (NM) This approach describes the process of multiple, distinct messages, each being delivered to multiple recipients. In a general case, this approach can be thought of as an extension to OM, iteratively repeated proportionate to the total number of messages to be delivered.

2.3.3 Threat Model

I use the NIST 800-154 threat model [113] to describe network security, with the following potential threats:

- *Eavesdropping*: The adversary may listen to and record any message sent on the network, i.e. through the deployment of a very sensitive omnidirectional antenna.
- *Message injection*: The adversary may send messages from a compromised node at any time, to any valid recipient. The number of transmitted messages is constrained by the sending node's power budget. This serves as the primary available interaction to disrupt the network.
- *Black hole attacks*: The adversary may block (i.e. fail to forward) messages traversing routes which pass through an adversarially-controlled node.
- *Node capture attacks*: The adversary may gain control over a given node, either through physical means such as flash dumping, or remotely via exploiting a software vulnerability. In either case, the adversary gains all knowledge known by the compromised node.

- *Replay attacks*: The adversary may re-transmit one or more previously-recorded messages between parties on the network, including valid requests from a querier or responses from sensor nodes.

- *Denial of service attacks*: The adversary may attempt to disable the functionality of the network through the broadcast of malicious messages, either to flood the network with irrelevant information, or to overwhelm the power budgets of the participating sensor nodes.

- *Masquerade attacks*: The adversary may impersonate one or more sensor nodes in the deployment, returning misleading or unintelligible results to the querier.

All data, including both protocol and sensed information, is stored on-device in either RAM or disk storage and is therefore susceptible to adversarial capture on devices located in the insecure area. This data moves through the system wirelessly, using protocols as described in Section 2.4. Securing the wireless transmission itself, or device security to reduce risk of compromise, could serve to augment the security of the proposed protocols.

I assume the adversary may transmit a similar message at the same time as a valid one, but cannot remove or otherwise modify the valid message. The impact of wireless jamming on the network environment is primarily hardware-oriented, and addressed in the literature through specialized anti-jamming work. The objective of the adversary is to disrupt the functionality of the deployment, for example by spoofing node responses to return false information to the querier, or to prevent the querier from collecting relevant results from the network. The defender aims to maintain sensing function despite intentional attack or unintentional degradation.

Finally, I consider the querier as a 'trusted-but-curious' participant. They will obey the protocol and follow all constraints, but may attempt to access information outside of their authorized scope if able to do so. This could include, for example, a curious low-level employee with restricted access to only a few sensors deciding to instead query all topic channels in the network.

2.4 Authentication

In this section, I present three on-demand topic channel authentication protocols for ad hoc IoT networks using several conventions; here, S_i denotes the i^{th} item of a set $\mathcal{S}$, $N.U$ denotes a characteristic U of an object N, $A|B$ represents an operation on A and B which is closed under homomorphism, and `function`() denotes a subroutine.

2.4.1 Baseline: Encryption-Based Protocols

Many secure communication protocols, including those for ad hoc systems, make use of encryption to provide message authentication and secrecy. In general, these approaches may take one of two possible approaches: (A) a single encryption key is shared between all participating nodes, and is used to encrypt communication between all recipients, or (B) unique encryption keys are established for each communicating pair, and the appropriate key-pair is selected based on the sender and recipient of a message. These baseline protocols are undesirable for two reasons. First, the former approach is completely subverted through the node capture of a single node; once the shared key is compromised, the entire network can be disrupted. Second, the latter approach limits the opportunities for intermediate nodes to conduct adversarial mitigation, ultimately undermining network longevity. The proposed protocols aim to

address both of these drawbacks.

2.4.2 Protocol #1: Query Validation

The first proposed protocol sacrifices message secrecy to increase on-node processing efficiency. I do this through the use of *query validation*, a method of certifying the authenticity of a query as it arrives at a node. This protocol uses the concept of request nonces, which are uniquely generated values associated with distinct transactions, and the concept of authentication strings, which are secret values associated with individual topic channels. Together, in conjunction with homomorphic hashing primitives, I am able to provide significant efficiency gains while only relaxing message secrecy relative to encryption-based baseline protocols. The hash function, `hash`(), must (a) be collision-resistant; (b) provide results of a length sufficient to prevent an exhaustive evaluation; and (c) provide the homomorphic property. I do not explicitly require that `hash`() supports keyed hashing; in cases where private information is needed to prevent attacks, I can include such information in the input itself.

Topic Channel Establishment

Topic channel establishment takes place on the node-channel ledger as described in Algorithm 1.

Topic Channel Assignment

To prevent impersonation attacks on the network, the authentication string for each node must be communicated secretly; if not, any node which intercepts the message will be able to masquerade as that node. This necessitates the use of encryption to

Algorithm 1: Ledger creates new topic channel

Description: A new channel T is generated by a local user on the node-channel ledger, with an associated label L, identifier U, and authenticator A, and stored locally in the ledger's set of channels $\mathcal{T}$.
Participants: A node-channel Ledger C.
Input: A local user interacts with the Ledger.
Result: A new topic channel T is created.

1 Instantiate new channel T
2 $T.L \leftarrow$ channel label
3 $T.U \leftarrow$ unique numeric ID
4 $T.A \leftarrow$ randomly-generated string
5 $C.\mathcal{T} \leftarrow C.\mathcal{T} \cup T$

preserve secrecy. Topic channel assignment is described in Algorithm 2. Additionally, pre-assignment of nodes to channels prior to deployment is also possible, through pre-loading of authentication hashes.

Algorithm 3 describes a node's assignment processing. Note that the ability to decrypt the smaller validation key is checked prior to the more-expensive decryption of the full payload. When a node N receives an assignment containing a mapping to a channel T, I say that N is *associated* with T. Prior to forwarding an assignment message, each node that belongs to at least one channel validates the assignment message using the signature of that channel, to ensure assignment messages are not replayed or falsified.

Topic Channel Updates

Since topic channel assignments are overwritten by each request, no explicit deletion or update requests are necessary. Changes for a given node's topic channel assignments can be included in a single request containing the appropriate new authorizations.

Algorithm 2: Ledger sends a new channel assignment

Description: A local user instructs the node-channel Ledger to assign a specific node to one (or more) topic channels. One constructed, these assignments are broadcast into the network addressed to the targeted node.

Participants: A node-channel Ledger C, and a sensor node N.

Input: A local user interacts with the Ledger.

Output: A new topic channel assignment message M.

1 Instantiate new assignment request R
2 R[validation] $\leftarrow (N.U, E)$, where E is a monotonically increasing non-negative integer, maintained by the ledger, and incremented whenever any request is sent
3 R[assignmentNonce] $\leftarrow W$, a randomly-generated unique nonce
4 R[assignmentSequence] $\leftarrow E$
5 R[hashes] $\leftarrow \{T_i.L : \texttt{hash}(E|W|T_i.A)\}\ \forall i \in \mathcal{T}$
6 R[mappings] $\leftarrow \mathcal{C} = \{C_1, C_2, ..., C_k\}$ where C_i is a 2-tuple of $(T_i.U, N_i.S_j)$ representing a channel ID and sensor type that should be associated
7 R[authorizations] $\leftarrow \mathcal{G} = \{(\texttt{hash}(N.U|T_i.A), \texttt{hash}(T_i.A|N.U))\}$, where each two-tuple in $\mathcal{G}$ contains an entry for each *distinct* channel T_i in $\mathcal{C}$)
// M_d (below) only available under known network topologies.
8 R[metadata] $\leftarrow (M_d, M_t)$ where M_d is an upper distance bound between any node and a bridge node, and M_t is the expected maximum number of topic channels
9 Instantiate a new message M
10 M[type] $\leftarrow$ "assignment"
11 M[addressee] $\leftarrow N.U$
12 M[validation] $\leftarrow \texttt{encrypt}(N.U|E)$
13 M[payload] $\leftarrow \texttt{encrypt}(R)$
14 **if** *the topology is known* **then**
15 $\quad \texttt{OD}(N, \texttt{serialize}(M))$
16 **else**
17 $\quad \texttt{OM}(\texttt{serialize}(M))$

Query Processing

The first part of the construction occurs when the query request is made. Additional processing takes place on the ledger, which provides stronger control over the behavior of the querier, who must now contact the ledger before each query of the network. This process is described in Algorithm 4.

Algorithm 3: Sensor receives a topic channel assignment

Description: A sensor node receives an "assignment" message M. If addressed to N, the channel assignments are validated using encryption, and stored on-node; otherwise, the message is checked to ensure that it is not out of sequence and that it originated from the node-channel ledger before being forwarding to other nodes.

Participants: A sensor node N.

Input: An "assignment" message M.

Result: The receiving node stores channel authentication information.

```
1  if M[addressee] == N.U then
2      M.V ← decrypt(M[validation])
3      if M.V[0] == N.U and M.V[1] ≥ N.E then
4          P ← decrypt(M[payload])
5          if P[validation][0] == N.U and P[validation][1] ≥ N.E then
6              N.E ← P[validation][1] N.𝒯 ← P[mappings]
7              N.𝒢 ← P[authorizations]
8  else
9      if M[assignmentSequence] ≤ N.E then
10         drop(M)
11     foreach G_i ∈ N.𝒢 do
12         if hash(M[hashes][T_i.L]|N.U) ≠
             hash(M[assignmentSequence]|M[assignmentNonce]|G_i[1]) then
13             drop(M)
14     retransmit(M)
```

Response Processing

Once a query has been broadcast into the network by the querier, it is processed by a receiving node N as described in Algorithm 5.[1] When a node B receives a response, it processes the message according to Algorithm 6.

[1] I assume that either each deployed node will know the network graph but be computationally-bound and therefore unable to compute the shortest path for direct messaging, or unable to discern the network graph at all. In either case, multicast responses (OM) are required. Since every node must return its own results to the querier, there are always multiple distinct messages to deliver; viewed in aggregate, this means that all protocols use NM to reply.

Algorithm 4: Querier constructs a new request

Description: A Querier and a Ledger (indented) conduct a two-way interaction over a secure, authenticated channel to provide the Querier with information necessary to produce a single, valid query, allowing for mediation of access to channels.

Participants: A Querier Q and a node-channel Ledger C.

Input: A local user inputs a query on the Querier system.

Output: A new, valid query R.

```
1  secure-OD(Ledger, ∅)
2      ∅ received                                          // Ledger C
3      secure-OD(Querier, 𝒯))
4  𝒯 received                                              // Querier
5  𝒯′ ← {T_i}, a subset of 𝒯.
6  W ← a randomly-generated unique nonce
7  secure-OD(Ledger, (𝒯′, W))
8      𝒯′, W received                                      // Ledger C
9      ℋ ← ∅
10     ℋ′ ← ∅
11     W′ ← a new, randomly-generated unique nonce
12     foreach T_i ∈ 𝒯′ do
13         H_i ← hash(A_i|W|C.E)
14         H′_i ← hash(A_i|W′)
15         ℋ ← ℋ ∪ H_i
16         ℋ′ ← ℋ′ ∪ H′_i
17     secure-OD(Querier, (C.E, W′, ℋ, ℋ′))
18 C.E, W′, ℋ, ℋ′ received                                 // Querier
19 Instantiate a new request R R[type] ← "query"
20 R[sequence] ← C.E
21 R[queryNonce] ← W
22 R[querySignature] ← ℋ
23 R[queryDetails] ← 𝒯′
24 OM(serialize(R))
```

Since replies may arrive more than once due the multicast routing, the querier may wish to cache results according to sequence number and node ID in order to de-duplicate results.

Discussion

Through the use of homomorphic hashing, this protocol produces compound signatures, validating that the messages received by the participants did come from the claimed nodes, and that their content was not modified by the adversary. To accomplish this, several new variables are introduced, including nonce values for various phases of the protocol, as well as secret authentication strings known only to the ledger. By restricting the knowledge of these strings to the ledger, I constrain the querier – it is restricted to accessing only those topic channels for which it is authorized, and it may not query topic channels more than once without requesting new permission from the ledger. By using these hashing primitives to replace comparatively-expensive encryption functions, power consumption of the network can be reduced, increasing deployment longevity.

2.4.3 Protocol #2: Query + Channel Validation

Previously, queries were validated as they arrived, and responses were validated as they were delivered to the querier. However, this minimal level of verification leaves the network relatively unprotected against fabricated or falsified responses injected by adversarial nodes. Instead, these improper messages will be received and rebroadcast by the network, magnifying their power-consuming effects. To address this issue, I will now introduce an additional layer of validation, which I call *channel validation*.

This approach permits constituent nodes to verify messages as they traverse the network, and to immediately drop those which can be recognized as false. Although additional computation is required to conduct such validation, substantial reductions in adversarial message transmission are possible, which may lead to an increase in

overall network resilience.

Response Processing

Algorithm 7 includes channel-specific information in the query response, to permit nodes to determine if that reply is valid for that channel. This permits each node to serve as a gatekeeper for its own topic channels[2].

Algorithm 8 describes new validation processes introduced during the re-transmission phase. This prevents falsified messages from improperly responding to a query – if a datum's signature does not pass validation, then the reply is dropped.

Discussion

By further leveraging homomorphic hashing to validate messages as they propagate over the network, using an opportunistic approach, I am able to filter messages generated by adversarial nodes without requiring expensive setup. Instead, each node can make a local decision about the validity of the message as it arrives, and short-circuit the re-transmission if a problem is detected. However, valid messages are still sent to each node in the network, resulting in extraneous transmissions that unnecessarily stress the network's power capacity; this problem is addressed in the following protocol. Note that messages may still be correctly signed by the adversary using a captured node authorized for that channel. In this scenario, the sequence number could be leveraged to prevent such attacks from flooding the network.

[2]Any message must belong to a valid channel (i.e. T_i where $1 \leq i \leq M_t$); as channel size distribution is unknown by nodes, the adversary must therefore choose a valid channel to spoof at random.

2.4.4 Protocol #3: Query + Channel + Chain Validation

I extend the previous concept further, to leverage information about the network topology (if known) to help constrain and shape query response propagation. As discussed in footnote 1, multicasting may result in replies being routed more than needed. Although each node cannot track its own shortest route, the ledger can upper-bound a channel's response propagation through the application of *chain validation*.

Response Processing

Reply creation (Algorithm 9) is modified with the added inclusion of a new key. The first member of this newly-created 2-tuple serves as the current height, while the latter serves as the certification of that height.

Additional validation is added to response processing (Algorithm 10) to validate propagation height. Note that for chain validation, although M_d could be incremented by the adversary to truncate propagation, failing to forward the message at all would be more a more effective attack. Decrementation is disallowed due to the one-way nature of hashing.

Discussion

By leveraging information about network topology, and multicast-based reply propagation, nodes are able to determine a reply's degree of propagation. This is used to prune it from the network if it another identical copy would have already arrived at a bridge node through another route, without knowing that route's details.

I make use of the described hash-based construction to prevent adversaries from resetting the Height to a shorter distance, improperly extending message propagation.

This ensures that although adversaries may prematurely remove messages from the network, they cannot boost the network's power consumption by promoting over-propagation of valid replies. A simple numeric counter, for example, would not provide such a guarantee.

One additional consideration is topological changes triggered by node captures or failures; such modifications may affect the longest-path bounds of a given node. This can be mitigated by increasing the M_d variable to include some flexibility for such changes; larger increases provide more resilience, at the cost of slightly higher transmission redundancy.

2.5 Protocol Assessment

2.5.1 Configuration & Setup

Topologies

Various deployment topologies were examined. Star describes a central node surrounded by neighbors. Tree describes hierarchical, balanced trees with branching factor 4. Line describes a linear sequence of connected nodes. Smallworld networks [89] are generated using the NetworkX v2.1 library [58] with $k = 4, p = 0.2$, where k is the number of linked nearest neighbors, and p is the likelihood of mutation. These parameters produce connected, ring-like networks with "shortcut" chordal edges. Powerlaw (or scale-free) networks [10] are similarly generated, with $m = 1, p = 0.5$ where m represents edges added incident to the existing graph, and p the likelihood of adding a triad. These values balance tightly-clustered graphs against excessive connectivity. 4-way Grids represent a lattice structure of nodes connected to their

vertically- and horizontally-adjacent neighbors. 8-way grids are similar, with the addition of diagonally-adjacent edges.

Network sizes and dimensions were chosen to reflect both likely real-world scenarios as well as topologically-extreme possibilities, highlighting the different behaviors of the protocols under various conditions. Smallworld and powerlaw network generation parameters were chosen to best emulate ad hoc deployments, tuned to equalize node deployment density over the monitored area.

Topic Channel Membership

For topic channel evaluation, 100 channels were created to reflect complex rule-based access control policies, with varying counts of member nodes drawn randomly from the network, and permitting multi-channel memberships. Replies were evaluated with channels of 1%, 10%, and 40% node membership, chosen to reflect high, moderate, and low heterogeneity networks.

Attack Model

Adversarial evaluations assumed a 1% compromise rate of deployed nodes, and a 100% attack rate; that is, all compromised nodes attack simultaneously. This compromise rate balances adversarial resources against the risk of partition, while this attack rate reflects a worst-case scenario for the protocols that most-effectively highlights their strengths and weaknesses.

Environment

Evaluations were conducted using a network simulator implemented in Python3, running on an Intel i5-2500S@2.7GHz with 12GB RAM. Node deployments, channel assignments, and topological details were generated randomly, with results averaged over 25 iterations. Network size was set to $n = 1600$; similar patterns of results appeared for other sizes, but are omitted for brevity.

2.5.2 Characteristics

Message Secrecy

Message Secrecy refers to the ability of nodes in the network, including adversarial nodes, to observe the contents of messages as they pass over the network. Since all three proposed protocols use hashing, they must also include their data payloads in the clear, as hash validation is strictly one-way. As such, their contents cannot be secret from an eavesdropper, and this approach is unsuitable for applications where the data should remain secret.

Capture Resilience

Capture Resilience describes the situation when a node in the network may become compromised by an attacker. Since the proposed protocols store only node-specific information on each device, the adversary may only sign messages using the local credentials of that captured node. Thus, if an adversary has control over 10% of the total nodes, they may only control at most 10% of a query's results. The baseline encryption approaches vary, with the shared-key approach failing totally after node capture, and the unique-key model proving equally secure to the proposed hash-based

protocols.

2.5.3 Evaluation

Exhaustion Resilience

Exhaustion Resilience refers to the ability of a protocol to prevent exhaustion attacks after a node capture; this would include actions like broadcasting duplicate or false messages designed to deplete power resources. In protocol #1, no additional steps are taken to prevent exhaustion attacks, but reduced hashing also results in lower computational costs. In protocol #2, propagation of falsified messages is curtailed, reducing adversarial impact and improving network durability. Finally, in protocol #3, these approaches are combined with chain validation to provide the most exhaustion resilience by upper-bounding the impact any given message, good or bad, can have on the network. The baseline encryption protocols provide an all-or-nothing solution, with results either fully-validatable (shared key) or not validatable at all (unique key). As seen in Table 2.1, for networks with higher average betweenness centrality such as Lines, where the number of routes a message may take between nodes is fewer, the channel validation procedures described in protocol #2 provide reductions in adversarial propagation relative to protocol #1. As nodes associated with the queried channels will drop falsified messages, routes which include such nodes are pruned from the possible routing space. Using chain validation, Protocol #3 narrows these route choices further, increasing resilience by forcing falsified messages to use fewer, more-efficient routes.

Computational Efficiency

Computational Efficiency refers to the fact that all three protocols take advantage of the reduced computational complexity of homomorphic hashing primitives relative to more-expensive encryption-based techniques. As seen in Table 2.2, protocol #1 uses the fewest hash operations, while protocol #2 increases the rate proportional to the topic channel size – fewer, larger channels require more hashing. Protocol #3 varies based on the topology of the network, with increases during validation balancing against reductions from route pruning.

Network Efficiency

Network Efficiency refers to the ability to more effectively utilize network resources to accomplish query submission and response phases, as measured by the number of message transmissions. Query delivery is very efficient in all protocols. During the query response phase, added efficiency is offered by protocol #2 when adversaries are present, and by protocol #3 under certain deployment topologies. Topic channels provide substantial power savings by reducing the number of nodes which must respond to a single query – as seen in Table 2.3, messaging complexity is a function of topic channel membership size. In cases where node heterogeneity is high and channel size is low, such savings can greatly extend the effective lifespan of the deployment without negatively affecting functional performance. As the maximal height of the chain validation is defined by the longest path to a bridge node from any other node, propagation sequences can be more strongly limited when these distances are shorter. Especially for topologies with small diameters, such as Star, protocol #3 provides the largest reductions in transmission costs due to more effective route pruning. Since

protocol #3 also includes the behavior of previous validations, such reductions come in addition to those offered by channel validation.

2.6 Summary

In this chapter, I introduced on-demand topic channels in ad hoc ephemeral sensor deployments, and presented three alternative authentication protocols to provide secure, reliable, authenticated communication while providing resilience from adversarial or accidental disruption. Potential extensions include expanding on protocol behavior under partitioning, characterizing adversarial ability to partition under various topologies, and extending this chapter's entrance node paradigm to include the concept of wear leveling to ensure that nodes within this set are not prematurely exhausted due to higher-than-usual communication loads. Additionally, intelligent message aggregation and forwarding provides several promising alternatives for further reducing network load.

Algorithm 5: Node replies to query, protocol #1

Description: A node receives a query request, and first determines if it should be forwarded. Any message associated to that node must pass query validation before forwarding. If N is in a queried channel, it then constructs a response, including a validation hash of the returned data.

Participants: A sensor node N.

Input: A "query" message M.

Output: A valid response R.

```
1  R ← ∅
2  T ← M[queryDetails]
3  H ← M[querySignature]
4  retransmit ← ∅
5  foreach T_i ∈ T do
6      if T_i is associated with N then
7          retransmit ← retransmit ∨ False
8          if hash(N.G_{i,1}|M[queryNonce]|M[sequence]) == hash(N.U|H_i) then
9              R ← R ∪ T_i
10             retransmit ← True
11 if retransmit ≠ False then
12     retransmit(M)
13 if R ≠ ∅ and N.E < M.E then
14     N.E ← M.E D ← ∅
15     V ← ∅
16     foreach R_i ∈ R do
17         V_i ← hash(M[queryNonce]|D_i|N.G_{i,1})
18         D_i ← recorded datum
19         V ← V ∪ V_i
20         D ← D ∪ D_i
21     Instantiate new reply R
22     R[type] ← "response"
23     R[sequence] ← N.E
24     R[queryNonce] ← M[queryNonce]
25     R[node] ← N.U
26     R[data] ← D
27     R[dataSignature] ← V
28     OM(serialize(R))
```

Algorithm 6: Node receives reply, protocol #1

Description: A node receives a response to a query. If it is the Querier, it confirms the validity of the message. The Ledger drops all incoming messages, while intermediate sensor nodes forward such messages without additional validation.

Participants: One of: a Querier, a Ledger, or a sensor node.

Input: A "response" message M.

Result: The Querier validates and records the response, the Ledger drops the response, or the node forwards the response.

```
1  D ← M[data]
2  V ← M[dataSignature]
3  switch node type do
4      case "Querier" do
5          foreach D_i ∈ D do
6              if hash(M[queryNonce]|D_i|M[node]|H') == hash(V_i|W') then
7                  include D_i as a valid datum in the aggregation
8      case "Ledger" do
9          drop(M)
10     else
11         retransmit(M)
```

Algorithm 7: Node replies to query, protocol #2

Description: As Algorithm 5, except channel-related information from the query is also included in the response for later validation.

```
1  R ← ∅
   [...]                                          // as in Algorithm 5
28     R[queryDetails] ← T                            // Channel info
29     R[querySignature] ← H                      // Channel signature
30     OM(serialize(R))
```

Algorithm 8: Node receives reply, protocol #2

Description: As Algorithm 6, with the addition of channel validation logic at the sensor node level. Responses which fail the channel validation are dropped from the network.

```
1  D ← M[data]
   [...]                                                  // as in Algorithm 6
11     drop ← False                                       // channel validation
12     foreach T_i ∈ T ∧ i ≤ M_t do
13         if ¬drop ∧ T_i is associated with B then
14             if hash(G_{i,1}|W|E) == hash(B.U|H_i) then
15                 drop ← drop ∨ (hash(W|D_i|N.U|G_{i,2}) == hash(V_i|B.U)
16             else
17                 drop ← True
18     if ¬drop then
19         retransmit(M)
```

Algorithm 9: Node replies to query, protocol #3

Description: As Algorithm 7, except chain validation information is also generated and included in the reply payload.

```
1  R ← ∅
   [...]                                                  // as in Algorithm 7
30    R[height] ← (0, hash(0|W|E|G_{i,1}))                // chain info
31    OM(serialize(R))
```

Algorithm 10: Node receives reply, protocol #3

Description: As Algorithm 8, except chain validation is performed on responses prior to forwarding. Responses which fail this additional validation are dropped from the network.

```
1  D ← M[data]
   [...]                                                  // as in Algorithm 8
19        K = M[height]                                   // chain validation
20        S = K_1|K_1 − 1|...|1|0                         // ordered seq.
21        if K_1 ≤ M_d ∧ hash(K_2|B.U) == hash(S|W|E|N.U|G_{i,2}) then
22            M[height] ← (K_1 + 1, hash(K_1 + 1|K_2))
23            retransmit(M)
```

	Protocol 1	Protocol 2	Protocol 3
Star	2.558E+04	2.098E+04	1.600E+01
Tree		3.743E+03	8.651E+02
Line		1.536E+02	1.536E+02
Smallworld		2.007E+04	1.700E+04
Powerlaw		8.528E+03	8.286E+03
4-way Grid		1.902E+04	1.896E+04
8-way Grid		2.066E+04	2.057E+04

Table 2.1: Transmissions Per Attack

	Protocol 1	Protocol 2	Protocol 3
Star	9.557E+02	4.148E+05	2.652E+03
Tree	9.593E+02	4.084E+05	2.871E+05
Line	9.506E+02	3.985E+05	2.813E+08
Smallworld	9.631E+02	4.214E+05	4.496E+06
Powerlaw	9.623E+02	4.125E+05	4.044E+06
4-way Grid	9.680E+02	4.256E+05	1.485E+07
8-way Grid	9.544E+02	4.144E+05	1.041E+07

Table 2.2: Hash Operation Count per Query

	Protocol 1	Protocol 2	Protocol 3	Protocol 1	Protocol 2	Protocol 3	Protocol 1	Protocol 2	Protocol 3
	1% Channel Membership Rate			10% Channel Membership Rate			40% Channel Membership Rate		
Star	2.597E+04		1.624E+01	2.523E+05		3.496E+02	1.025E+06		1.344E+03
Tree	2.504E+04		2.088E+03	2.570E+05		2.283E+04	1.018E+06		8.953E+04
Line	2.523E+04			2.564E+05			1.026E+06		
Smallworld	2.504E+04		2.501E+04	2.605E+05		2.601E+05	1.023E+06		1.022E+06
Powerlaw	2.638E+04		2.337E+04	2.578E+05		2.281E+05	1.026E+06		9.111E+05
4-way Grid	2.373E+04			2.523E+05			1.017E+06		
8-way Grid	2.699E+04			2.584E+05			1.028E+06		

Table 2.3: Transmissions Per Query

Chapter 3

SPATIAL PARTITIONING

Wireless sensor networks and other widely-deployed low-power systems provide users with monitoring capabilities which were previously difficult to achieve. As the use of such systems increases, their security takes on increasing importance. Improvements in this area should not compromise existing requirements for efficacy and efficiency, however. One common approach found in the literature is the use of location to guide message delivery, commonly called geographic routing. Existing approaches, while efficient and reliable, require networks to reveal the target location explicitly, which may not be desirable for systems operating under adversarial threat. I address this limitation by introducing WindRose, a novel message authentication and routing protocol designed for low-power mesh networks. Users may address messages to nodes within geographic regions, using message metadata to guide efficient routing without revealing the specific location of interest to any participant outside that area. This enhances network usability, privacy, and security without compromising efficiency. To support this functionality, I also introduce the homomorphic Merkle tree, an innovative data structure for asymmetric data validation. I evaluate the proposed protocol both qualitatively and quantitatively, discuss the strengths and weaknesses of its security, and provide experimental results to support my contributions. This chapter captures results presented in [17].

3.1 Overview

Wireless sensor networks and other widely-deployed low-power systems provide users with monitoring capabilities which were previously difficult to achieve. As the use of such systems increases, their security takes on increasing importance. Oblivious routing protocols [8] use strictly local information to inform which neighboring node should receive a message. By avoiding reliance on shared information or global state, they offer strong fault tolerance and resilience to adversarial behavior. However, they sacrifice efficiency, with heuristics that most closely resemble greedy behavior [71].

Note that such protocols are especially well-aligned to use with low-powered and ad hoc networks. The power constraints prevent high-level or long-range coordination, while the ad hoc nature of the communication provides rich opportunities for adversarial interference. These twin constraints have previously been explored in the literature, but primarily individually (see [7] for power constraints in ad hoc networks and [66] for adversarial interference). When considered in conjunction, however, further challenges appear.

3.1.1 Challenge: Location Privacy-Preserving Queries/Updates in Ad-hoc Networks

Consider a company that wishes to dispatch software updates to a deployment of low-powered IoT devices. An adversary, unaware of any specific security vulnerabilities within the network, adopts an alternative approach. Instead of attacking the devices traditionally, they will wait for an update to be released by the vendor and dispatched to the devices. When this occurs, they then disrupt the message delivery and/or front-run the message and exploit the vulnerability before it can be patched. However, by masking the location of the nodes to be patched until after the patch arrives, the

adversary is unable to effect their plan and the devices remain secure.

In another example, consider a network equipped with surveillance devices. The agency operating this network has been informed through a separate channel about a potential adverse event by a malicious third-party, and wants to reconfigure the relevant portion of the network to be sensitive and alert for this event. However, the agency also does not want to send the relevant commands in the open to avoid providing advance warning to the malicious agents operating in the surveilled portion of the network.

3.1.2 Contributions: WindRose

In this chapter, I propose WindRose, an adversarially-resistant oblivious routing protocol that utilizes masked geographic information. Specifically, WindRose makes the following contributions:

- geographic routing with masked targets,
- spoof- and network-attack resistance for queries, and
- efficient message delivery.

To achieve these goals, I can sacrifice the secrecy offered by encryption-based cryptographic primitives for the computational efficiency of hashing, and gain several benefits. The protocol relies on an innovative homomorphic Merkle tree (HMT) data structure which is used to efficiently compute location membership and message validity, without requiring vulnerable shared symmetric keys, or difficult-to-scale public key encryption infrastructure.

3.2 Related Work

3.3 Homomorphic Hashing

Several aspects of this work leverage the class of collision-resistant homomorphic hashing functions for information-masked authentication. Specifically, for a function $H(x)$, it is homomorphic if and only if $H(a \odot b) = H(a) \odot H(b)$, for some operator $\odot$. This property permits "incremental" hashing, enabling hash composition without requiring access to the unobfuscated inputs. Such functions were originally described in the literature by [19], with later work based on the discrete log problem [78] and elliptic curves [96] proposed as alternative constructions.

The use of homomorphic hashing to secure ad hoc routing protocols has been previously explored [15], but does not consider the relative efficiency of the delivery mechanisms, relying primarily on flood-based approaches. Other applications [76] focus on their usage for authentication, rather than their obscurative properties. Recent work [88] has examined their use to improve resilience against adversarial behavior, but does not discuss potential applications for routing.

Although hashing in general is more computationally efficient than encryption or message signing [117], homomorphic hashing can introduce additional challenges. For example, discrete log approaches do not compress well, leading to much larger hash signatures than competing approaches. Conversely, elliptic curve methods provide good length compression, but can be more computationally costly.

3.3.1 Geographic Routing

Using geographic information for oblivious routing has been extensively discussed in the literature. [84] proposed a protocol which introduces the concept of "progress" towards the target, while geometrically-guided approaches were summarized by [79]. Other approaches have focused on maximizing power efficiency [132, 142, 144, 145] by leveraging geographic information.

Note that most geographically-informed routing protocols rely on the target location being explicitly stated to facilitate route discovery. In contrast, WindRose assumes that only the original querier will know the precise region targeted, and that participant nodes will have only information on the heading on which the target region lies. WindRose also assumes that nodes will be able to detect their own location at least once, but this is not always possible for ad hoc networks. [122] proposes an alternative solution which does not explicitly require nodes to be able to collect location information. Instead, it is inferred from sentinel nodes around the perimeter of the network deployment.

Many proposed solutions also assume that participant nodes are honest and will follow a specified protocol. Previous works [66, 72] have examined the impact of adversarial nodes on routing in ad hoc networks, but the presence of such nodes complicates the task of limiting information leakage and delivering messages reliably, especially for geographically-guided routing protocols.

3.3.2 Range Query Privacy & Leakage

Obscuring targeted regions from adversarial detection has been explored in the literature from other perspectives than routing. In effect, geographically-targeted

routing in wireless sensor networks can be interpreted as a type of multidimensional range query search, in which the spatial coordinates of the nodes serve as a bounding region of interest, and their returned sensor values are observed features in a higher-dimensional space. In this way, existing approaches can be adapted to apply to this context [65].

Often, the problem is framed as a user performing obfuscated queries against a target region, with an adversary attempting to identify the area of interest. Information that is exposed in this way is commonly termed as "leakage". [81] showed that even in cases where the source and the target are both fully encrypted, leakage of both the region of interest and the information contained therein can potentially be exposed. I do not claim to eliminate leakage, but to reduce it to acceptable levels for sufficiently few queries.

3.4 Problem Formulation

Before delving into the details of the system, I first describe the high-level behaviors enabled by this approach.

3.4.1 Query Model

Area of Interest

Users interact with a distributed, ad hoc network of devices which they have previously deployed, but whose topology and distribution is potentially unknown. The user identifies a geographic region, the "area of interest", that they wish to query. They do not need to know if there are any nodes in that region, or if there are, what their precise locations may be.

This area of interest can be considered a polygon, centered at a point such that the angular bounds define the minimum width of the polygon. Compatible polygons are describable using (1) this centroid, (2) a single distance measure, and (3) the angular bounds. As in Fig. 3.1, this could include a circle defined by its diameter, a square defined by its edge, or an isosceles trapezoid defined by its length. Note that these regions have different areas and positions in the target arc, both defined by the centroid's distance from transmission source.

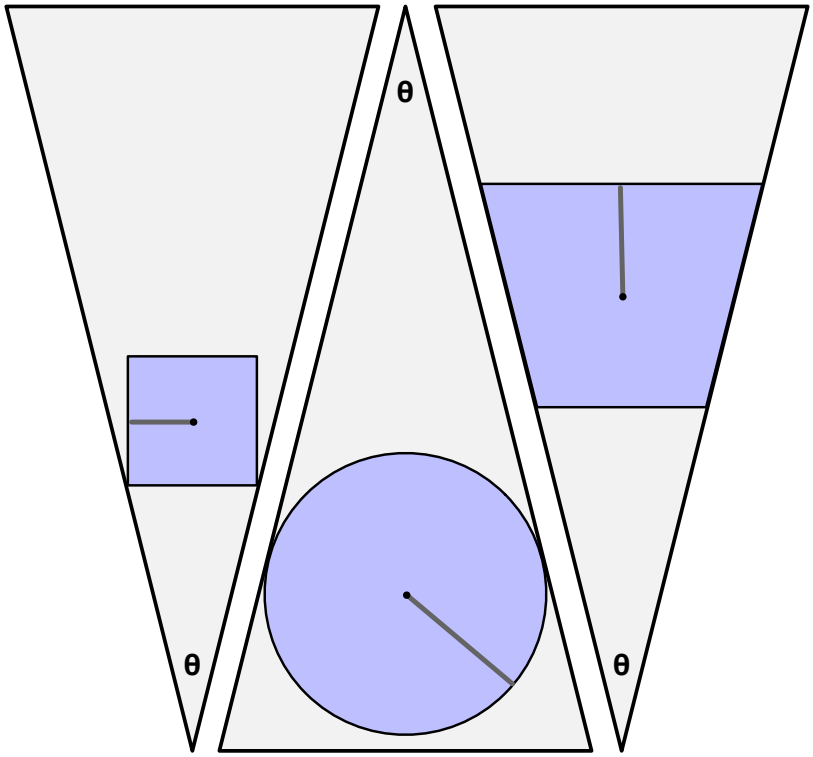


Figure 3.1: Three types of region polygons, defined by a centroid and a distance.

Queries

Queries carry a payload, which carries some semantic meaning to the recipient. Some payloads may instruct receiving nodes to engage an actuator or turn on a recording device, while others might request a reconfiguration or provide a software update, for example.

Query Modalities

The query message is constructed and transmitted into the network through an endpoint local to the user, which is termed the *source node*. Queries can be sent in two modes, *best-effort-any* and *best-effort-all*. The first mode will not take any special actions (beyond those defined by the protocol) in order to reach the area of interest. The second mode, in contrast, will attempt to determine when it has reached the area of interest and aim to maximize the number of nodes in the area of interest which receive the query.

3.4.2 Adversarial Model

Ad hoc systems are often deployed over wide areas. As such, it can be difficult or impossible to monitor, control, or secure deployments against adversarial access. Adversaries can eavesdrop and collect information on all transmissions which traverse their nodes. To represent these capabilities, I model adversaries' network access as a modified Dolev-Yao model, with a locality restriction. That is, an adversary cannot suppress or modify transmissions which do not involve an adversarial node.

Node Capture Attacks

A key capability of an adversary against ad hoc systems is the ability to compromise nodes, termed *node capture attacks*. Given physical access to a device and effectively unlimited time, it is reasonable to assume the adversary would be able to compromise the device and dump its memory through a variety of attacks, potentially even if the device had a trusted memory module. However, the adversary is limited in their ability to do so by the time investment necessary, and can therefore only capture a variable fraction of all nodes in the network.

Sybil Attacks

Additionally, adversaries could add listening stations or Sybil nodes within the deployment area. Adversaries can also inject messages originating at compromised or Sybil nodes, and may or may not obey protocol rules, at the adversary's discretion.

Black Hole and Wormhole Attacks

Adversaries may execute *black hole attacks* by dropping transmissions which use an adversarially-controlled route. They can also execute *wormhole attacks*, where messages overheard at one point can be transmitted directly to a different adversarial node at another location in the network.

Privacy Leakage

The adversary can make inferences about queries, especially the area of interest, and other network behaviors, which I term *privacy leakage*.

I assume that all compromised nodes are controlled by a single adversary, who has

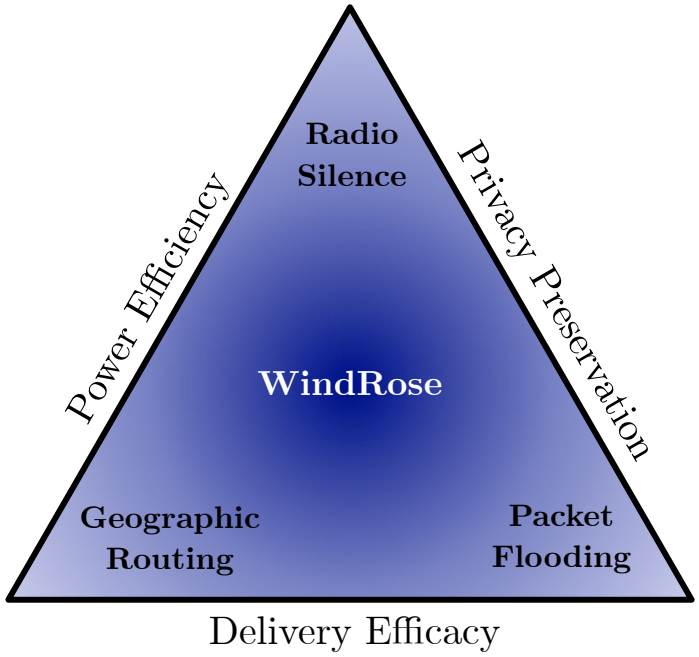


Figure 3.2: WindRose seeks an equilibrium between competing criteria.

the ability to pool information collected from the network instantly, subject to the previously-described capabilities. On the other hand, I also polynomially-bound the adversary's computational capabilities. That is, the attacker cannot find two distinct input values which cause the collision-resistant homomorphic hash functions to hash to the same value. Note that this does not provide perfect forward secrecy, only a realistic bound for their computational power.

3.4.3 Design Criteria for WindRose

To achieve the stated goals, I present WindRose, a novel message authentication and routing protocol designed for low-power mesh networks. Users may address messages to nodes within geographic regions, using message metadata to guide efficient routing without revealing the specific location of interest to any participant outside that

area. This enhances network usability, privacy, and security without compromising efficiency or functionality. As described above, the deployment environment imposes multiple, conflicting constraints on system functionality and the proposed protocol aims to satisfy the following criteria:

- maximize the number of nodes in the target region which receive a given query, and
- prevent the adversary from determining the area of interest.

Moreover, achieving these goals may require costly trade-offs. For example, using a single route between a source and a destination would be the most efficient option, but if an adversary lies on that route, then the message will never be delivered. Flooding packets into the network would guarantee delivery, but require many transmissions, prematurely depleting network power resources. Consequently, the protocol also aims to address a third criterion:

- minimize the total number of transmissions used for a given query.

WindRose aims to provide an equilibrium point between these three criteria (Figure 3.2), according to the functional requirements of the network, and to enable users to tune the performance of the system parametrically to best suit their needs.

3.5 Homomorphic Merkle Tree

For several algorithms underlying WindRose, I compare hash signatures for equality, to validate that information from two sources matches. Merkle trees are commonly used for this purpose [98, 107], but have several requirements: Merkle trees are binary,

balanced, and bushy. These are easily satisfied when participants are working from the same base data, but are a liability when inputs are unknown, or when all leaves must be frequently re-hashed.

To mitigate this drawback, I propose a Homomorphic Merkle Tree structure (HMT), which similarly uses a tree-like structure to encode hierarchical relationships between recursive *homomorphic* hashes of its neighbors. Like Merkle trees, when comparing hash equality, only the root hashes must match to ensure the data was constructed from the same leaf information. However, in contrast to Merkle Trees, HMTs relax the binary, balanced, and bushy requirements. The homomorphic nature of the underlying hash function allows the root node's value to be conceptualized as an abstract representation of the in-order traversal of the tree (given an appropriate homomorphic combination operator). However, it also sacrifices the $O(log(n))$ complexity characteristic offered by Merkle trees.

Note that collision-resistant functions closed under homomorphism are rarely also closed under transitivity; it is therefore occasionally necessary to store two branch hashes, corresponding to the two symmetric orderings of the leaves. Expressed mathematically, $h_1 = H(A \odot B), h_2 = H(B \odot A)), h_1 \neq h_2$, and in the absence of either A or B, both h_1 and h_2 must be stored if needed. In such a scenario, trees containing A or B can still be reconstructed, but the original values are *hidden*. This offers the ability to "freeze" trees by placing hidden values in the first and last leaf positions, constraining reconstruction verifiability to participants who know this hidden data (or their parent branches).

HMTs are used for validation in several stages of the protocol, such as message integrity, and regional membership, as described below.

3.6 WindRose Protocol

In this section, I provide an overview and details of the WindRose protocol, including message integrity verification, geographically-masked routing heuristic, and evaluation of hidden region membership.

3.6.1 Parameters and Notation

As described in the previous section, satisfying the key protocol criteria necessitates certain trade-offs, subject to parametric tuning to suit deployment requirements. Here, I summarize the key parameters that WindRose uses to allow adjustment of the algorithm to best fit the deployment requirements:

- P_{key}: A pre-shared security variable, known temporarily by the nodes and permanently by the sources
- $P_{query\text{-}strategy}$: The propagation strategy followed during query delivery, from: {*best-effort-any*, *best-effort-all*}
- $P_{query\text{-}budget}$: The maximum number of neighbors to which a message can be transmitted
- $P_{query\text{-}margin}$: The size of the routing margin
- $P_{query\text{-}deviation}$: The size of the maximum permissible deviation from the candidate heading
- $P_{max\text{-}size}$: The maximum queryable region size.
- $P_{slice\text{-}resolution}$: A security parameter for membership validation

In the rest of this section, I use these parameters to describe deployment, and query creation and routing processes.

3.6.2 Nonces and Caching

Cycle elimination and nonces. Cycle elimination plays a critical role in any oblivious routing protocol. Unique node identifiers, a common approach, can pose scalability challenges for networks with very many nodes. WindRose instead makes use of unique message identifiers in the form of randomized nonces. In addition to complicating precomputation attacks by the adversary, these nonces permit each node to maintain a small, local cache for cycle elimination, with a simple least-recently-used eviction strategy. Since each node is a low-powered device, these caches should be as small as practical for the query load of the network.

Caching. I assume that the service requirements or characteristic load of the network indicate that few queries will be issued simultaneously. This minimizes cache size, which is a useful side effect for low-powered devices that may not have much onboard memory to spare. Cycles may be further reduced due to angular heuristics discouraging backtracking, but are still possible in some configurations. Cache sizes should be tuned based on query requirements and network density, corresponding to the largest expected simple cycle.

3.6.3 WindRose Network Deployment Process

Network key. When initially creating a deployment, the operator generates the network's P_{key}, and pre-loads this value on the nodes prior to installation in the monitored area. As this value could be extracted from a captured node, it is used for

node initialization, as described below, and then destroyed immediately following the initial deployment.

Node Initialization. Upon initial startup, a node N localizes itself through a convenient mechanism, and stores this location locally as N_{loc}. It also computes and stores $N_{sig} = H(N_{loc} \odot P_{key})$ and $N_{sig'} = H(P_{key} \odot N_{loc})$.

Using its location, the node creates a square $P_{max\text{-}size} \times P_{max\text{-}size}$ region centered on itself, and partitions it into a two-dimensional matrix with $\lceil \frac{P_{max\text{-}size}}{P_{slice\text{-}resolution}} \rceil^2$ cells. For each column $j \in X, N_{x\text{-}slice_j} = H(N_{loc} \odot j \odot P_{key})$, and likewise for each row $i \in Y, N_{y\text{-}slice_i} = H(P_{key} \odot i \odot N_{loc})$. Thus, finer slice resolution require more initial hashes, but provide stronger protection against brute force attacks.

Heading and relative bearing computation. In addition to self-localization, each node must also be able to compute or derive absolute heading. This may be accomplished in a variety of ways [48], but onboard localization hardware makes this process both rapid and efficient. Additionally, each node must be able to determine the relative bearing between itself and a reachable neighbor. The exact process by which this is accomplished can vary according to hardware capabilities, but in general has been shown to be accurately computable and resilient to spoofing [52].

Deployment risks. The usage of pre-shared information leaves the network vulnerable to supply chain attacks. Node capture prior to initialization could also lead to network compromise through cold boot attacks. Collectively, these two attack vectors represent the most severe threat to the security of WindRose. Consequently, when the node localization process is over, the node must immediately destroy the local copy of P_{key}, overwriting it if necessary to prevent later reconstruction. Nodes need not include a trusted execution module or other protected memory mechanism,

Algorithm 11: Generation of a new query by a source node N.

Output: A message M ready for propagation.

1 $M_{nonce} \leftarrow$ `generate-nonce`
2 $M_{seq} \leftarrow$ `get-next-seq`()
3 $M_{src} \leftarrow N_{loc}$
4 $M_{left} \leftarrow$ `bearing`(N, left-most perspectivity)
5 $M_{right} \leftarrow$ `bearing`(N, right-most perspectivity)
6 $R_{centroid} \leftarrow$ `trunc`(centroid of region, $P_{slice\text{-}resolution}$)
7 $M_{payload} \leftarrow$ A message payload.
8 $M_{seal} \leftarrow H(P_{key} \odot R_{centroid} \odot P_{key})$
9 $M_{s\text{-}sig} \leftarrow H(R_{centroid} \odot P_{key})$
10 $M_{m\text{-}sig} \leftarrow H(P_{key} \odot M_{seq} \odot M_{payload} \odot M_{nonce} \odot M_{src} \odot M_{left} \odot M_{right} \odot M_{s\text{-}sig})$

as the sensitive information is only needed during setup and discarded thereafter. On the other hand, a finer-resolution $P_{slice\text{-}resolution}$ requires more hashes, and will take more time to compute, increasing the attack window during deployment. Since node capture prior to initial startup presents a security risk, the setup process should be initiated reliably prior to deployment. This could be triggered manually, with a time-delay mechanism, or through an onboard accelerometer that detects aerial dispersion, for example.

Finally, nodes may be pre-programmed with a set of potential source locations.New sources can be added or updated later through followup transmissions, but inclusion prior to deployment reduces network traffic and does not compromise security, since source nodes' locations are not private.

3.6.4 Query Creation and Routing in WindRose

The query creation algorithm is detailed in Algorithm 11, while query routing is described in Algorithm 12. In this section, I will discuss the role played by each of the

phases of these algorithms[1].

Query creation. When a query message is created, a nonce and sequence number are used to limit precomputation and replay attacks, respectively. The source node is specified, and serves as the anchor point for the message. From this anchor, the left and right bounds are derived by projecting the delivery intent into the deployment space and calculating the perspectivity. The remaining components of the message are used to ensure messages and target regions can be validated as having come from a source node, and that any modifications to the message will cause it to fail integrity checks.

Query routing. Ad hoc deployments, especially wireless sensor networks, often consist of power-constrained devices which must subsist on a finite power budget allocated during deployment, such as an internal battery. It is therefore crucial that delivery of messages use as few resources as practical while still maintaining operational characteristics.

When transmitting the message into the network, the source node may have several potential candidates to choose from as the first node(s) to select – this ranking process is, in fact, identical for all downstream nodes transmitting the message. Intuitively, forwarding every message to every node (flooding) will maximize delivery, but also quickly exhaust network power resources. Conversely, forwarding a message along the shortest path between two nodes will minimize power usage, but such routes can be difficult or impossible to precisely discover, while also being susceptible to adversarial attack e.g. through black hole attacks, as well as being brittle to network changes,

[1]Note that, to simplify the presentation, the algorithmic descriptions have been simplified to elide details such as over- and under-flows in angular operations, octant-relative conditionals, and other implementation minutiae.

such as those caused by failing nodes.

WindRose attempts to balance these competing requirements by providing tuning mechanisms to control the resilience-efficiency trade-off. Specifically, each node is provided with a *transmit budget*, which represents the maximum number of neighbors to which a node can transmit a single message. A budget of one is efficient but susceptible to failure, while a budget of three or more is likely to reach the target at the expense of extraneous transmissions. Additionally, the WindRose protocol specifies a *maximum deviation constraint*, which restricts to what degree a message may deviate from the optimal direction of transmission. This can, for example, prevent messages from propagating backwards, reducing both transmission cost and delivery likelihood. Finally, the protocol also specifies an *angular margin*, which represents the degree to which a message can deviate outside of the target angular region before deciding it is too far off-track and rerouting back towards the center. Larger values allow for better circumnavigation of network holes, but also can potentially lead to messages overshooting and missing the target, or taking overly circuitous routes.

Heading selection: When a message is first injected into the network, it can compute the midpoint between the outer bounding rays to determine an intended heading. The concept of a *desired heading* is important in WindRose, differentiating between a current heading (ϕ_{srel}), the heading which represents progress towards the target region (ϕ_{orig}), and the heading which the message will actually take (ϕ_{poss}) are distinct concepts. The heading which the message intends to take, as computed by the forwarding node, is the desired heading (ϕ_{des}). Angular relationships between the current node and neighboring nodes are computed, and nodes are ranked in descending order of their deviation from the desired angle. The relationships between these angles,

and the methods by which they calculated, are detailed in Algorithm 12. Then, the top-most candidate nodes are chosen as the next recipients.

Note that the above ranking approach produces deterministic routing behavior, which may not be desirable due to network load balancing, for example. Alternatively, a linear or non-linear candidate rank permutation algorithm may be used, such as RankEdge [28]. This permits a finer level of control over the exploration / exploitation trade-off than would otherwise be possible. In this context, resilience to network failure or adversarial disruption is mitigated by exploration, while increasing routing efficiency is analogous to exploitation. Alternatively, a stochastic approach could be used, which would decrease efficiency while increasing route privacy.

Off-track message routing: When a message arrives at an intermediate node, the source-relative angle is computed. If the source-relative angle falls outside of the left-right bounding rays of the message, plus a parametric margin of error, then the message is considered to be *off track*. Such messages are given priority routing orthogonal to the target angle, back the bounded region. This is taken by computing the interior angles of the right triangle formed between the source node, the current node, and a virtual Euclidean point forming the right angle between the current node and the central target ray. This mechanism is related to the virtual Euclidean Steiner tree creation used in [143], and prevents the formation of *reachability shadows*, areas of the network which are unlikely to be reached by any messages by virtue of falling on the far side of a network hole relative to the original source node.

On-track message routing: Conversely, for messages which are on track, i.e. fall within the bounded region of the message, the protocol instead forms an isosceles triangle from the source to the current node at the apex of the triangle. The com-

plementary transmission angle then becomes half of the central angle of this triangle. Conceptually, this means that near to the source, messages are more likely to oscillate around the central target ray, while asymptotically approaching the central ray as distance from the source increases.

When a message is received by a node within the region, and the node's membership within the region has been confirmed (see Sec. 3.6.5), one of several possible alternatives take place. If propagation uses a best-effort-any approach, propagation continues following normal rules, with no change in behavior. This makes detection of the target region essentially impossible , since no behavioral change takes place, and therefore no density gradient can be computed because none exists (see Sec. 3.8.2). Alternatively, the message may choose a best-effort-all approach. In this case, a node that falls within the target region and which is within the *core* of the area - that is, more than one transmission radius from the boundary - changes its propagation behavior. Instead, it forwards the message to all nodes within transmission range (broadcasting), excluding the node from which it received this message.

Note that, regardless of the behavior a node takes when forwarding a message, it also caches the query nonce (see Section 3.6.2). Messages whose nonces hit the local cache are dropped and do not forward further, to short-circuit routing cycles and prevent unneeded transmissions.

3.6.5 Target Region Detection in WindRose

When evaluating whether a message has arrived in a target region, each node compares the proof embedded in the message, which I call the *geographic seal*, with the potential regions in which the node may be a member. These are generated on the

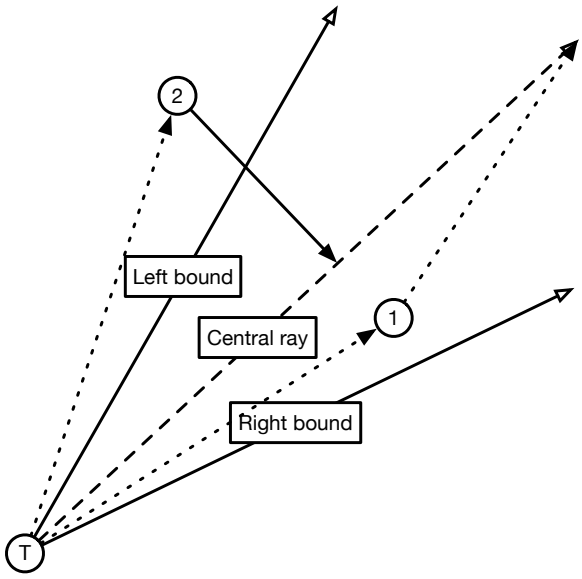


Figure 3.3: A node T, with a transmission budget of 2, forwards to its chosen candidate nodes '1' and '2'. '1' re-transmits along its reciprocal angle, while '2' is off-track, and attempts to return to the bounded region.

fly by quantizing the $\mathbb{R}$-valued search space into pre-defined buckets corresponding with the slice size: intuitively, the central ray can be interpreted as intersecting a node's local slice grid. This means that queries against larger regions will produce greater numbers of candidate cells, and correspondingly require more hashes to validate membership.

Since each node must only evaluate regions in which it may be a member, the number of potential hash evaluations it must complete are strictly bounded to only those regions. To prevent such bounding from making exhaustive precomputation possible for non-compromised nodes, the seal incorporates both a randomly-generated nonce unique to that query, as well as the pre-shared signing string which was previously hashed and discarded by the participant node. Slice information is not

explicitly encoded, since node locations are not known by the source. This information asymmetry prevents computation of valid candidate regions at arbitrary locations, restricting it to the location of a given node at the time of its deployment. However, this also forbids post-deployment node relocation, since the slice grid must be re-computed; this is an expected trade-off to improve network security.

Note that, as I have introduced in Section 3.4, the area of interest can be considered a polygon, centered at a point such that the angular bounds define the minimum width of the polygon describable using (1) this centroid, (2) a single distance measure, or (3) the angular bounds (Fig. 3.1). To evaluate their membership within a given polygon, an individual node must know the shape type. This can be pre-defined at a protocol level, or included in the message and signature.

3.7 Security Discussion

In this section, I discuss the security considerations of WindRose, including its resilience to adversarial interference and its ability to preserve the privacy of a query region.

3.7.1 Protection against Query Spoofing

As seen in Algorithm 11, the M_{sig} hash is computed using the P_{key} value. As the adversary does not have access to this value, as described in Sec. 3.6.3, this value cannot be falsified. Every node validates this signature prior to forwarding the message and a falsified query will be dropped from the network once it arrives at an uncompromised node.

3.7.2 Protection against Network Attacks

In addition to protocol-specific spoofing, WindRose also provides resilience to several other more general attacks through existing mitigation measures:

Replay attacks. The M_{seq} counter prevents message replay attacks, as out-of-order messages are discarded.

Rainbow table attacks: The inclusion of a randomized nonce prevents precomputation (also known as rainbow table) attacks: each hash must be re-computed for each new nonce.

Black hole attacks: The transmission budget allows users to tune a network's resilience to black hole attacks. Higher budgets produce more transmissions, but also decrease the likelihood that a route between source and area of interest will be severed by a small number of well-placed adversaries. In cases where the adversary carefully places Sybil nodes to create voids in the network, increasing the maximum deviation will improve the ability to route around these obstacles.

Brute force attacks. Brute force attacks are not generally effective against CRHFs, but can play a role in certain phases of WindRose. Specifically, the HMT evaluation on line 21 of Algorithm 13 is susceptible to brute force evaluation in which the adversary evaluates every possible centroid against the seal. However, potential candidate centroids can be only be validated by nodes when they are within $P_{max-size}$ distance, and then must be evaluated potentially many times based on the $P_{slice-resolution}$. This protects against a single node compromise weakening overall network security against a computationally-powerful adversary. Rather, as $P_{max-size}$ and $P_{slice-resolution}$ decrease, the required number of compromised nodes and the quantity of hashes evaluated respectively increase, quadratically.

Denial of service (DoS) attacks. DoS attacks against ad hoc networks take two primary forms, transmission jamming and premature exhaustion. Jamming primarily takes place at the hardware layer, rather than the protocol layer, and is addressed through different mechanisms. Additionally, jamming in a large-scale deployment requires prohibitive power consumption due to the inverse-square law. Localization spoofing (e.g. via GPS) is possible, but only effective if it takes place during the initial deployment setup phase (see Section 3.6.3).

Premature exhaustion relies on message spoofing to achieve transmission amplification, where falsified messages improperly propagate through the network through correctly-behaving nodes, consuming more resources than initially spent by the adversary. Since queries cannot be spoofed, such attacks are not possible in WindRose.

3.7.3 Protection against the Leakage of the Area of Interest

Protecting the privacy of a region of interest represents a core goal of the WindRose protocol. The primary vector by which this information can be inferred by an attacker is the observed variations in message density of queries.

Message density variations are highly dependent on network topology and adversarial distribution, and are therefore difficult to reason about qualitatively. I evaluate privacy leakage quantitatively in Sec. 3.8.4. As the experiments show, in query routing, variations in message density produced by best-effort-any or best-effort-all are difficult to differentiate from background noise. Adversaries within the area of interest can identify the region with perfect accuracy, which is expected behavior. However, since this will not occur until after the message has arrived, it mitigates the damage that foreknowledge of the target region might cause. Teleportation attacks could increase

the impact of such events, but would require extensive communication infrastructure and a relatively large quantity of compromised nodes both at the periphery and within the deployed volume.

3.8 Experimental Results

3.8.1 Experimental Setup

Experiments were conducted using a network routing simulator implemented in Python3, built on the NetworkX, Numpy, and Matplotlib libraries. Each parametric configuration was executed on a stochastically-generated topology with uniform density; reported results reflect an average over 25 runs. Experiments used 1000 nodes in a 100×100m space, with transmission range of 7m. These parameters produce connected networks with high likelihood, and with low betweenness centrality ($\propto 0.0085$). Adversaries compromise nodes randomly, with a likelihood of 0%, 10%, or 35%.

3.8.2 Evaluation Criteria

Messaging Efficacy

A critical metric by which any geographically-targeted message delivery protocol must be measured is how effectively messages are delivered to that region. If the shortest, most efficient routes are chosen, but these routes bypass the area of interest or give up prior to arrival, then the protocol has failed in its primary purpose.

I evaluate this in the context of the user's strategy choice, as well. Specifically, for best-effort-any delivery, if any node in the target region is reached, then the

routing is considered as having succeeded. I represent this with a success rate of 100%; correspondingly, failure to reach any nodes is 0%.

For best-effort-all evaluation, I use a different metric. Here, I report the *query delivery rate* of a given query as a ratio, $Q_{\%} = \frac{N_r}{N_t}$; the number of nodes which received a query, N_r, over the total number of nodes in the target region N_t.

I evaluate these metrics for both query strategies, best-effort-any and best-effort-all, and report mean and standard deviation. When testing efficacy, the adversary black-holes every message a compromised node receives, in order to maximally degrade network performance.

Messaging Efficiency

To quantify the power efficiency of the protocol, I measure the total number of transmissions for a given message. This allows for an approximation of energy cost for a transmission, irrespective of the actual hardware technology selected for the deployment.

Target Area Leakage

Quantifying the target area leakage raises some difficulties: this leakage is best described by the message density within the network, but it also necessitates modeling the capabilities of the attacker. Conceptually, an adversary which controls several nodes over the deployed area can collect data about the observed density and leverage this information to differentiate between a hypothetical *baseline* or *"null case"* where no region is targeted, and the alternative, where a specific region has been targeted. Figure 3.4 illustrates how changes in message densities (represented by node size) can

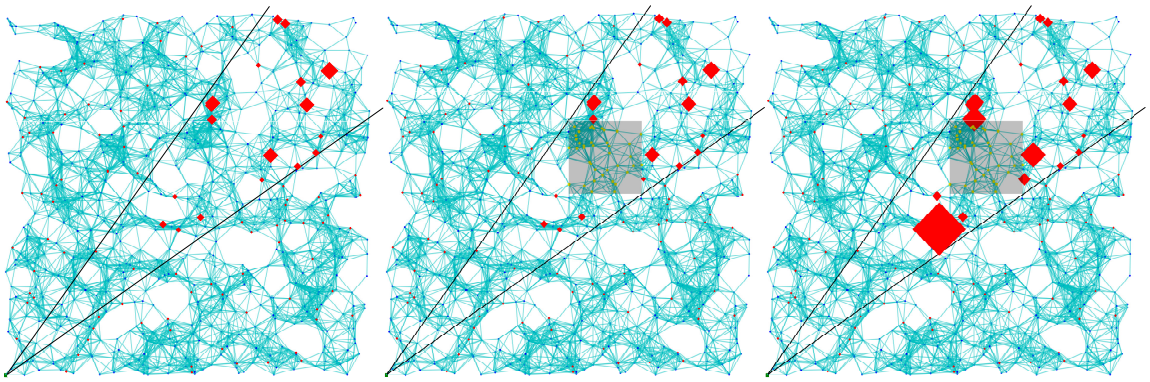


Figure 3.4: Visualizations of message density for three cases: [A] null, [B] private, and [C] leaking. Distributions between A and B are identical, successfully masking the region, while (artificially injected) leakage in C allows the adversary to infer its location.

potentially leak the region of interest.

Consequently, to assess the relative effectiveness by which queries are hidden, I consider the specific network topology, create a null case distribution for this topology, and evaluate the relative message density. To maximize the adversary's discriminative power, I assume that compromised nodes will follow delivery protocols in order to maximize the amount of data collected. Additionally, adversarial nodes disregard cache hits under this paradigm in order to maximize network traffic. Using this paradigm, I generate a series of (topology, source, target) 3-tuples that define a transmission environment. For each configuration to evaluate, I execute WindRose against that list of environments using the specified configuration. Then, I compare differences in behavior independent of environmental perturbation. I also generate a baseline behavior in which no region is specified.

To compare the densities, each adversarial node is placed into an absolute ordering

based on their distance from the source, with their values equivalent to the total number of messages of any kind observed by that node. The observation vector is then normalized. I call this a *message density distribution*. I compare the normalized forms of these distributions between different configurations using Kullbach–Liebler divergence, or *relative entropy* [80].

3.8.3 Discussion of the Results

Query Efficacy

In measuring the relative delivery efficacy of queries (Tables 3.2 and 3.3), I have measured both best-effort-any and best-effort-all strategies, for five budgets and three adversarial rates.

First, I observe that best-effort-any substantially outperforms best-effort-all. This is expected, as it only requires a single target node be reached in order to count as a success. It also highlights the fact that increasing budgets may provide incremental improvements, but these are only beneficial if more nodes must be reached. If one recipient node is sufficient, lower budgets may be a better choice.

I also observe that for best-effort-all, the redundancy offered by larger budgets can offer substantial improvements in delivery rate. However, as budgets increase, these gains level off. Areas of the network with fewer possible routes restrict the path diversity offered by larger budgets, setting an upper bound on their impact.

Finally, note that the relatively high range observed can be attributed to a small number of topologies in which the adversary controlled one or more nodes very close to the source node. Under these conditions, it was possible for the adversary to partition the network between the source and the target region, before path diversity from

larger budgets was able to route around the blockage.

Query Efficiency

Table 3.4 shows the results for query deliveries with different strategies and budgets. I do not measure efficiency differences between adversarial rates, as I assume that adversarial nodes will obey the protocol and transmit normally when attempting to maximize energy usage.

Although message redundancy increases the number of transmissions significantly, especially going from a budget of one to two, it does not continue to scale exponentially. Instead, the cost levels off, as the number of distinct, reasonable routes a message can take is fixed. This suggests that messages which propagate through these bottlenecks will be dropped due to cache hits, preventing over-propagation.

Further increases in transmission budget follow a near-linear increase proportional to the number of nodes in the network. Thus, in cases where delivery efficacy and privacy preservation (discussed below) are desirable, and energy constraints permit, increasing query budgets can be a very effective strategy.

Additionally, transmission costs between best-effort-any and best-effort-all strategies do not show large differences. This is likely due to the fact that messages "overshoot" the target region to avoid leakage; the costs of this continued propagation dominate the overall costs of delivery. User choices between strategies should therefore be based on efficacy and privacy considerations, and not delivery efficiency.

3.8.4 Query Privacy

I have evaluated leakage by the best-effort-all strategy for budgets of different sizes, and for different adversarial rates; these results are presented in Table 3.5. Note that for the best-effort-any strategy, no leakage is possible due to identical routing behavior relative to the null case.

First, I observe that even when the adversary has perfect knowledge of the null case, differentiation between the distributions can be quite difficult, sometimes on the order of a single message at one observing node. Any noise or variability in the network behavior, or any mistakes or gaps in the adversary's null case model, would make this task impossible. Thus, in cases where reaching more nodes would be favorable, best-effort-all does not pose a large privacy risk to the user.

Second, I observe that when more adversarial nodes are present, inference becomes easier. Intuitively, this makes sense, since they have more observations from which to extrapolate. However, the increases are lower than might be expected. For small budgets, the relative increase is low, although the higher base discrimination produces a higher absolute increase. For larger budgets, the relative increase is higher, but since the starting values are already low, discrimination may still be quite difficult.

Finally, note that as the transmission budgets increase, it becomes increasingly difficult for the adversary to find variations in the density distribution. Initially, it might seem that greater messages in the network would *increase* discrimination, due to more potential observations in the distribution. However, results indicate the opposite - discriminability actually decreases.

Since messages can arrive at target nodes from more routes, the broadcasting behavior of best-effort-all is less likely to be useful in reaching target nodes (which

would have already been reached via other routes). Thus, the behavior more closely resembles the null case distribution, making differentiation more difficult.

3.9 Summary

In this chapter, I introduced WindRose, a novel protocol for adversarially-resistant geometric routing which masks location information from participating nodes. Using relatively weak assumptions and strong adversarial capabilities, I have shown that users may have efficacy, efficiency, and privacy preservation simultaneously, without requiring specialized node features such as trusted computing modules. Leveraging innovative Homomorphic Merkle trees (HMTs), WindRose is able to provide message delivery and region validation without revealing sensitive information to participating nodes. At-rest data is only a risk to the node on which it is stored, and thus if that node is captured, no broader impact can be achieved by the adversary.

Table 3.1: Notation of the WindRose Protocol

Notation	**Meaning**
$A \odot B$	An operation closed under homomorphism
ϕ_{label}	A heading or navigational bearing
θ_{label}	An angle between two rays
δ_{label}	A distance in Euclidean space
N_d	A node N, and the data d it stores
M_d	A message M and its encapsulated data d
P_d	A globally shared configuration parameter P with a setting d
R_d	A geographic polygon R with associated regional metadata d
$H()$	A homomorphic hash function closed under $\odot$
$\texttt{in}(A, B, C)$	True $\iff$ A is between B and C
$\texttt{order}(\mathcal{N}, mode)$	A preferentially-weighted ordering of a set $\mathcal{N}$
$\texttt{loc-check}(M)$	Determine membership in a target region (Algorithm 13)

Algorithm 12: Query forwarding heuristic.

Input: A message M (in query format) is received by a node N.
Result: M is dropped or forwarded according to WindRose heuristics, with minor updates to local node state.

if $H(N_{loc} \odot M_{seal} \odot N_{loc}) \neq H(N_{sig} \odot M_{s\text{-}sig} \odot N_{loc})$ **then**
  Invalid seal signature, **drop**
if $H(N_{loc} \odot M_{m\text{-}sig} \neq H(N_{sig} \odot M_{seq} \odot M_{nonce} \odot M_{src} \odot M_{payload} \odot M_{left} \odot M_{right} \odot M_{s\text{-}sig})$ **then**
  Invalid message signature, **drop**
if M_{nonce} *in* $N_{cache} \vee M_{seq} <= N_{seq}$ **then**
  Repeated sequence or cache hit, **drop**
N_{cache}.`insert`(M_{nonce})
$N_{seq} \leftarrow M_{seq}$
$\phi_{srel} \leftarrow$ `bearing`(M_{src}, N)
$\phi_{orig} \leftarrow (M_{left} + M_{right}) * 0.5$
$\theta_{dev} \leftarrow \phi_{orig} - \phi_{srel}$
if $in(\phi_{srel}, M_{left}, M_{right})$ **then**
  $member \leftarrow$ `loc-check`(M)
  if $member \neq False$ **then**
    `process`($M_{payload}$)
  $\phi_{des} \leftarrow \phi_{orig} - \theta_{dev}$
else if $in(\phi_{srel}, M_{left} - P_{margin}, M_{right} + P_{margin})$ **then**
  $\phi_{des} \leftarrow \phi_{orig} - \theta_{dev}$
else
  $\theta_{int} \leftarrow 90 - \theta_{dev}$
  $\phi_{des} \leftarrow \phi_{orig}^{-1} - \theta_{int}$
$candidates \leftarrow \{\}$
$local\text{-}devation \leftarrow P_{query\text{-}deviation}$
$local\text{-}budget \leftarrow P_{query\text{-}budget}$
if $member = Core$ **then**
  $local\text{-}deviation \leftarrow 360$
  $local\text{-}budget \leftarrow |N_{neighbors}|$
foreach nb in $N_{neighbors}$ **do**
  $\phi_{poss} \leftarrow$ `bearing`(N, nb)
  $\theta_{valid} \leftarrow \phi_{des} - \phi_{poss}$
  if $\theta_{valid} > local\text{-}deviation$ **then**
    continue
  $candidates \cup \phi_{poss}$
$candidate\text{-}list \leftarrow$ `order`($candidates$)
for $i \leftarrow 1$ to $local\text{-}budget$ **do**
  if $i < candidate\text{-}list_{length}$ **then**
    `transmit`($M, candidate\text{-}list[i]$)

Algorithm 13: Validate the region defined by message M.

Input: A message M encoding a query region.
Output: A membership label from: {*False*, *True*, *Core*}

$\phi_{srel} \leftarrow$ bearing from M_{src} to N
$\phi_{orig} \leftarrow (M_{left} + M_{right}) * 0.5$
$\theta_{dev} \leftarrow \phi_{orig} - \phi_{srel}$
$\theta_{range} \leftarrow \phi_{orig} - \phi_{left}$
$\delta_{srel} \leftarrow L^2$ distance from M_{src} to N
$\delta_{progress} \leftarrow cos(\theta_{dev}) * \delta_{srel}$
$\delta_{size} \leftarrow tan(\theta_{range}) * \delta_{progress}$
$\delta_{dist} \leftarrow tan(\theta_{dev}) * \delta_{progress}$
$\delta_{max} \leftarrow \delta_{progress} + \delta_{size}$
$\delta_{min} \leftarrow \delta_{progress} - \delta_{dist}$
$\delta_{xmit} \leftarrow$ transmission range of N
while $\delta_{min} \leq \delta_{max}$ **do**
    $\delta_{radius} \leftarrow tan(\theta_{range}) * \delta_{min}$
    $R_{centroid} \leftarrow M_{src} + \delta_{min}$
    $R_{region} \leftarrow \texttt{polygon}(R_{centroid}, \delta_{radius})$
    if $N_{loc} \notin R_{region}$ **then**
        δ_{min} += *granularity*
        continue
    else
        $j, i \leftarrow$ the column-major slice coordinates of $R_{centroid}$
        if $N_{x\text{-}slice_j} \odot R_{centroid} \odot N_{y\text{-}slice_i} \neq N_{loc} \odot j \odot M_{seal} \odot i \odot N_{loc}$ **then**
            continue
        if $P_{query\text{-}strategy}$ = *"best-effort-any"* $\vee \delta_{radius} <= \delta_{xmit}$ **then**
            return *True*
        $R_{core} \leftarrow \texttt{polygon}(R_{centroid}, \delta_{radius} - \delta_{xmit})$
        if $N_{loc} \in R_{core}$ **then**
            return *Core*
        else
            return *True*
return *False*

Table 3.2: Query Efficacy by Budget and Adversary Rate (B-E-Any)

		$P_{query\text{-}budget}$				
A%		**1**	**2**	**3**	**4**	**5**
0	Mean	40%	92%	100%	100%	100%
	StDev	50%	27.69%	0%	0%	0%
10	Mean	12%	76%	92%	96%	96%
	StDev	33.17%	43.59%	27.69%	20%	20%
35	Mean	0%	16%	48%	72%	76%
	StDev	0%	37.42%	50.99%	45.83%	43.59%

Table 3.3: Query Efficacy by Budget and Adversary Rate (B-E-All)

		$P_{query\text{-}budget}$				
A%		**1**	**2**	**3**	**4**	**5**
0	Mean	10.62%	71.74%	93.09%	95.22%	97.42%
	StDev	21.64%	30.57%	11.99%	11.19%	8.01%
10	Mean	0.93%	43.32%	70.40%	80.63%	83.37%
	StDev	2.71%	35%	29.59%	21.60%	20.63%
35	Mean	0%	5.75%	21.42%	38.45%	43.81%
	StDev	0%	15.26%	25.79%	27.55%	27.24%

Table 3.4: Query Cost by Strategy and Budget

	$P_{query\text{-}budget}$				
	1	**2**	**3**	**4**	**5**
B-E-Any (Mean)	18.52	306.60	684.24	1016.64	1295.12
(StDev)	13.69	157.71	258.56	382.67	490.66
B-E-All (Mean)	28.88	342.24	728.80	1064.20	1341.36
(StDev)	36.89	180.23	277.44	400.59	507.04

Table 3.5: Privacy Leakage by Adversary Rate (B-E-All)

		$P_{query\text{-}budget}$				
A%		**1**	**2**	**3**	**4**	**5**
10	Mean	6.02E-02	5.11E-03	5.07E-04	1.70E-05	0.00E+00
	StDev	2.31E-01	1.18E-02	1.41E-03	8.50E-05	0.00E+00
35	Mean	6.39E-02	4.15E-03	2.99E-04	5.95E-05	3.35E-05
	StDev	1.90E-01	7.90E-03	6.01E-04	1.98E-04	6.75E-05

Chapter 4

DISTRIBUTED FUSION

Ad hoc communication networks provide a robust and low-power method for sensors to return their observations to an authority. However, ensuring that the returned values accurately represent the environment being sensed poses challenges. In particular, malicious sensors controlled by an adversary may collude to return systematically misleading or falsified results. Previous literature examines this challenge for weak adversaries and under strong assumptions, limiting practical applicability. In this chapter, I propose Pando, a novel approach for mitigating the Byzantine distributed sensor fusion problem. I introduce an approach for the decentralized creation of a forest ensemble, made up of overlapping, hierarchical message passing routes that provide robust and efficient delivery while also mitigating adversarial interference. I then leverage a modified, homomorphic Merkle tree structure to create a novel Byzantine-tolerant message passing protocol. Finally, I propose a classification-informed data fusion algorithm based on these methods. I evaluate the correctness and efficiency of the approach under several attack models, and discuss functionality improvements over the state of the art. This chapter captures results presented in [16].

4.1 Overview

Many distributed sensing applications require sensor deployments in remote, dangerous, or inhospitable environments. Due to the likelihood of loss, damage, or destruction in these scenarios, such sensors commonly use inexpensive, low-power components and self-contained power sources, and rely on ad hoc communication

methods to compensate for absent or inaccessible network infrastructure [4]. Ad hoc approaches also allow for dynamic re-routing as nodes become inaccessible, evolving the network topology [27].

Despite these advantages, ad hoc networks suffer from the disadvantage that, if any nodes intentionally diverge from the established protocol in order to maliciously undermine communication effectiveness, the message will not arrive.

As an example, consider a smart city in which sensors distributed throughout the water supply network monitor municipal water quality. If an adversary wishes to illegally dump industrial runoff, it can mislead these sensors by injecting plausible baseline measurements, allowing the environmental harm from its pollutants to go undetected.

Similarly, a governmental organization surveilling a remote area to reduce the flow of smuggling deploys a number of sensors to notify personnel of unauthorized activity in the area. In response, criminals may compromise a subset of these sensors, creating a blind spot in the network. As a result, movement of illicit goods may proceed unimpeded, without alerting officials to the problem.

In both cases, malicious interference during sensing allows the provision of false data, resulting in worse outcomes than if data were simply absent. Consequently, mitigating adversarial interference forms a critical foundation for sensing, improving security for a broad variety of potential applications. Towards this goal, I make the following contributions:

- I propose a method for distributed creation of an ensemble of spatially-distributed overlay networks using strictly node-local information.

- I propose a novel messaging format utilizing homomorphic Merkle trees to enable

self-authenticating payloads.

- Using these structures, I describe a protocol for Byzantine-tolerant distributed sensor fusion.

By detecting message tampering, unexpected absence, and illicit modification, the proposed protocol offers a foundation for distributed sensing platforms to provide a more robust level of service in scenarios with adversarial interference.

4.2 Related Work

4.2.1 Sensor Fusion

Many modern devices and systems contain multiple sensors, either co-located on a single device to collect heterogeneous data, or dispersed across many devices to sense a wider area. Often, due to their low power, low cost, or harsh operating conditions, these sensors return unreliable measurements.

The process of "sensor fusion" refers to combining this ensemble of data into a single, reliable representation of the environment, and serves a crucial role in many domains, such as self-driving cars, UAVs, and other applications with autonomous agents interacting with the environment [41, 74].

To address the most common challenges arising from corrupted or noisy sensors, statistical methods can smooth out natural variability in data [57], but when dealing with distributed systems, these approaches become difficult to apply due to the distribution of data across potentially many devices.

4.2.2 Distributed Sensing

Xiao *et al.* [146] extends statistical smoothing to distributed sensor systems, relying on the concept of message passing to share a weighted average of the observations so far. This approach proves robust against unreliable links and measurements, but does not address malicious interference.

A similar scheme proposed by X. Zhang *et al.* [151] adopts a Bayesian sensor fusion approach using graph information to inform consensus. This distributes responsibility for fusion across the network, but does not address the presence of adversarially-controlled nodes in the topology.

K. Zhang *et al.* [149] examine efficiency improvements offered by a mobile data collection agent, reducing both the total messages exchanged and the sensor density required to achieve effective coverage. However, they again do not consider Byzantine activities.

4.3 Resilience Toward Adversaries

Previous literature has explored networks' sensitivity to adversarial interference, in particular from the perspective of routing. Awerbuch *et al.* [6] proposes using statistical inference of link failures to identify malicious nodes in an ad hoc context, but does not address modifications to message payloads directly. Hu *et al.* [66] describes a method which protects payloads, but requires expensive per-node key management scaling linearly with network size.

Other work explores the concept of node capture resilience [32], establishing communication channels within the local neighborhood of a node that are robust to post-deployment compromise. However, messages which directly traverse a compro-

mised node may have their payloads changed, an undesirable consequence in most applications.

Liu *et al.* [93] propose the use of a trust model for detecting and excluding adversarial nodes. However, the trust models considered assume a uniform distribution of information importance, with protection amortized across all messages, rather than protecting individual rounds of sensing. Further, trust-based models can be susceptible to betrayal low-frequency, high-reward attack scenarios.

In contrast, the approach proposed by Abrardo *et al.* [2] provides strong guarantees about message fusion in the presence of adversaries, including a theoretically optimal bound. However, this approach is computationally expensive; a followup work by the same authors [3] addresses this challenge through the use of Monte Carlo Markov Chains (MCMC), with both approaches requiring that adversarial nodes do not collude.

4.4 Resilience Toward Adversaries

Previous literature has explored networks' sensitivity to adversarial interference, in particular from the perspective of routing. Awerbuch *et al.* [6] proposes using statistical inference of link failures to identify malicious nodes in an ad hoc context, but does not address modifications to message payloads directly. Hu *et al.* [66] describes a method which protects payloads, but requires expensive per-node key management scaling linearly with network size.

Other work explores the concept of node capture resilience [32], establishing communication channels within the local neighborhood of a node that are robust to post-deployment compromise. However, messages which directly traverse a compromised node may have their payloads changed, an undesirable consequence in most

applications.

Liu *et al.* [93] propose the use of a trust model for detecting and excluding adversarial nodes. However, the trust models considered assume a uniform distribution of information importance, with protection amortized across all messages, rather than protecting individual rounds of sensing. Further, trust-based models can be susceptible to betrayal low-frequency, high-reward attack scenarios.

In contrast, the approach proposed by Abrardo *et al.* [2] provides strong guarantees about message fusion in the presence of adversaries, including a theoretically optimal bound. However, this approach is computationally expensive; a followup work by the same authors [3] addresses this challenge through the use of Monte Carlo Markov Chains (MCMC), with both approaches requiring that adversarial nodes do not collude.

4.4.1 Homomorphic Hash Trees in Sensing

Homomorphic hashing [19, 78, 88, 96] differs from traditional cryptographically secure one-way hash functions by satisfying the following relationship: $H(A) \odot H(B) == H(A \odot B)$ for an operator $\odot$.

In prior work [15, 17], we have seen that I can extend the well-known Merkle tree [102] using homomorphic hashing to allow for imbalanced, mixed-degree trees, which I refer to as *homomorphic Merkle trees* (HMTs). In particular, I proposed these HMTs as a self-authenticating data structure, and applied them in various message authentication contexts [15, 17]. In this chapter, I apply these techniques to the data collection and fusion problem, expanding the previous focus on resilient node addressability.

4.5 Problem Formulation

4.5.1 Setting

I assume that each node distributed in a monitored region contains hardware for sensing and local communication with its nearby neighbors, yet insufficient for long distance communication. Each node can also localize itself, such as through the use of satellite navigation, and localize its neighbors in a spoof-resilient manner using a protocol such as [52]. Localization should occur during initial deployment to mitigate the impact of localization spoofing. All nodes in a deployment contain identical capabilities.

A fusion coordinator (FC) is tasked with collating sensor information and making a final determination on a fused value, and should be able to collect information from arbitrary nodes within the deployment. Recovery logistics could include physical mobility [149] or long-distance wireless communication [46]. Sensing occurs in synchronous rounds as a response to a query from the FC, with each round assigned a monotonic nonce generated randomly by the FC. Authenticated query delivery may use any appropriate method [15, 17].

For simplicity, I assume that sensor observations are drawn from a finite categorical dictionary and, without loss of generality, that the ad hoc network lies in a two dimensional space.

4.5.2 Adversarial Model

A percentage of deployed nodes may be compromised by the adversary, who gains access to all local information stored on the captured node at the time of compromise.

This is a reasonable assumption, as given physical access to sensors, extracting the data stored on these devices poses a surmountable problem to a motivated adversary. Furthermore, this information may be shared among the captured nodes to coordinate deception strategies and malicious behavior.

The adversary aims to modify the network behavior such that the system comes to consensus on a value of the adversary's choice (the "incorrect" value), replacing the value which would have been produced in the absence of malicious behavior (the "correct" value) using three possible attacks: Adversarially-controlled sensor nodes

- report incorrect observations (*value injection*);
- generate synthetic observations corresponding to nodes which do not exist (*node injection*); or
- drop incoming observation traffic to give malicious observations more weight (*traffic dropping*).

I consider five deception strategies, where an attacker

- does not drop any observations to better blend in and hide its adversarial nature (*passive*);
- drops all observations except its own, to maximally disrupt sensing functionality (*destructive*);
- drops all observations including its own, to disrupt sensing and hide its presence (*silent*);
- drops observations with a preponderance of correct values, and keeps those with incorrect values to negatively impact the fused result (*selective*); or

Table 4.1: Symbols and Notation

Operators	**Meaning**
$A \odot B$	An operation closed under homomorphism
$H()$	A traditional collision-resistant hash function (CRHF)
$\mathcal{H}()$	A CRHF homomorphic under $\odot$
$\emptyset, [], \{\}$	Set, array, or key-value map respectively
$\|A\|$	The cardinality of A
A^+	The absolute value of A

- drops observations with a preponderance of incorrect values to mislead Pando's defenses (*sneaky*).

I discuss the impacts of each of these deception strategies in more detail in Section 4.7.

4.6 Pando Protocol

Pando [1] tackles attacks through the decentralized creation of an ensemble of Trémaux forests. Even though these forests are constructed using only local node information, they serve as secure hierarchical routes for diverse, efficient message passing, mitigating adversarially-inserted network partitions. In this section, I detail this 3-phase process, involving: (Phase 1) ensemble creation; (Phase 2) data collection; (Phase 3) ensemble-based fusion determination.

[1] Pando, meaning "I spread out" in Latin, is the name of a colony of aspen trees determined to be a single living organism, with a shared root system.

Prior to deployment, each node ν receives a unique serial (ν_{id}), derives a node signing key ($\nu_{sig} = \mathcal{H}(\nu_{id} \odot \mathcal{S})$) using a shared secret $\mathcal{S}$, and stores ν_{id} and ν_{sig} locally for later use during HMT creation. The FC must also securely store $\mathcal{S}$ for later use validating node authenticity. Although leaking $\mathcal{S}$ would invalidate protocol security, deployed nodes only know a hash of $\mathcal{S}$, and thus post-deployment node capture does not undermine overall network security. During deployment, nodes also determine and store their position as ν_{pos}.

4.6.1 Phase 1, Creation of an Ensemble of Forests

A Trémaux tree describes a hierarchical spanning tree rooted at a particular vertex, such that all pairs of adjacent nodes have a absolute ordering with respect to the root vertex [40]. While ideally one spanning tree would be created, to reduce network costs and other overhead, Pando may instead create a forest of Trémaux trees that collectively span the network. By leveraging their hierarchical nature, these trees allow Pando to route messages efficiently to each root.

To initiate a query, the FC randomly generates a nonce η, and transmits it into the deployment as described in Section 4.5.1. Upon receiving the nonce, each node follows a series of steps to generate a globally-shared ensemble of hierarchical forests. No node knows the entire network topology, and instead produces oblivious routing decisions conducted using local information only. Collectively, this results in useful emergent routing behavior that is both efficient and robust.

Trémaux Forest Ensembles

Each node initializes a local, cryptographically secure pseudorandom number generator P with the received nonce, then generates an ensemble coordinate vector, $\vec{e}$, of length m, the pre-determined ensemble size. These coordinates (referred to as "virtual roots") are drawn uniformly at random from the deployment area using a two-dimensional uniform distribution[2]. Note that, as all nodes use the same η, they also produce an identical vector $\vec{e}$.

Subsequently, for each virtual root[3] $\varphi = \vec{e}[i]$ $(1 \leq i \leq m)$, a node computes its routing behavior for the forest rooted at φ. It first computes an absolute ordering of its neighbors (and itself) using their respective Euclidean (L^2) distance from φ. Using this information, it then generates routing rules for a given (η, φ) pair and stores them locally.

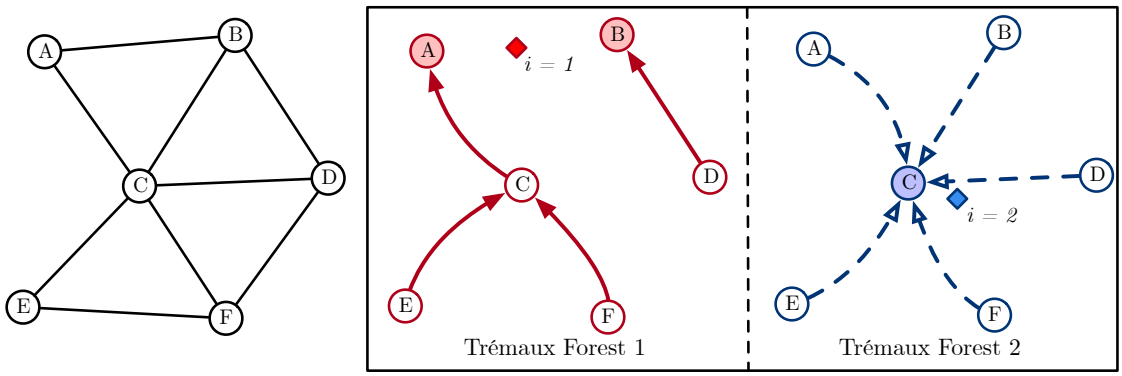
I refer to the emergent routing behavior generated by these collective rules as a *Trémaux forest* ($\mathcal{F}_\varphi$) for a specific virtual root $\varphi \in \vec{e}$, and to the set of all forests enabled by the vector $\vec{e}$ as a *forest ensemble* ($\mathcal{E}$) for all $\varphi \in \vec{e}$ (Figure 4.1).

Node-Local Process for Forest Ensemble Creation

Alg. 14 presents the node-local computations producing the forest ensemble. For each $\varphi \in \vec{e}$, a node computes its meta-relations, labeling neighbors with the following terminology:

[2]Unimodal Gaussian sampling centered on the deployment produces shorter, more homogeneous trees, improving transmission efficiency but reducing resilience; bimodal Gaussian sampling biased toward the periphery produces the reverse behavior. In the evaluations, I consider uniform sampling, which provides a balance between these extremes.

[3]For simplicity, in the rest of the chapter, I will use the notation $\varphi \in \vec{e}$.



(a) Connectivity (b) Forest ensemble, $m = 2$

Figure 4.1: An example forest ensemble, with virtual roots.

- *Descendants*: Neighbors further from φ.
- *Ancestors*: Neighbors closer to φ.
- *Parent*: The ancestor closest to φ.

I also use *child* to refer to the reciprocal relationship of *parent* – this relationship cannot be easily determined by a node, but can be discovered by the FC by analyzing routing information embedded in the HMTs. The nodes also classify themselves as

- *Roots*: Nodes with no ancestors,
- *Leaves*: Nodes with no descendants

as appropriate.

As mentioned above, for some topologies, Algorithm 14 may partition an otherwise fully-connected graph, generating a forest, with multiple trees, each partially spanning the graph. In the next phase, the FC may choose to use data from all roots (an *unpruned* strategy), or may collect data from roots within a fixed distance of each

Algorithm 14: Generation of a new forest ensemble.

```
   GENERATEENSEMBLE(m, η):
 1 e⃗ = []
 2 for i = 0; i < m; i ++; do
 3     seed = η
 4     for j = 0; j < i; j ++ do
 5         seed = H(seed)
 6     e⃗[i] = (x, y) ← P_seed ∼ U // Uniform distribution
 7 foreach φ ∈ e⃗ do
 8     ν_dist = L²(ν_pos, φ)
 9     parent, ancestor, descendant = ∅
10     min-dist = ∞
11     foreach neighbor ∈ neighbors do
12         neighbor_dist = L²(neighbor_pos, φ)
13         if neighbor_dist < min-dist then
14             parent = neighbor
15             min-dist = neighbor_dist
16         if neighbor_dist < min-dist then
17             ancestor = ancestor ∪ neighbor
18         else
19             descendant = descendant ∪ neighbor
```

virtual root (a *pruned* strategy). In practice, one root usually dominates each forest, and thus the pruned strategy provides nearly identical results while greatly simplifying data-gathering logistics for the FC.

4.6.2 Phase 2, Data Collection within a Trémaux Forest

Once the rules are generated, data observation and collection may begin. This involves (a) data transmission, in parallel, for each forest within the ensemble, and (b) collection of the data from the forest ensemble by the FC.

Data Transmission within a Trémaux Forest

Each node makes a local observation o, accepts messages from its descendants, and sends aggregated results to its parent. Nodes maintain sensing timeout windows sized inversely proportionate to the node's distance from φ, waiting to determine if its descendants will return any data. Longer windows provide better sensor coverage, at the expense of higher latency, and should be tuned for usage requirements. This sensing timeout prevents deadlocks caused by observation dependency cycles between descendants across $\mathcal{E}$.

Leaves may sign and transmit their observations immediately. In either case, the node also optionally notifies its non-parent ancestors that it has completed its observation to further reduce latency, at the expense of more transmissions.

I refer to each message received from a descendant as a *subtree*, in reference to their structure as a homomorphic Merkle tree. Nodes may choose to perform validation on received subtrees prior to forwarding, incurring an added computational cost bounded by a maximum validation depth parameter, δ. This process mirrors the validation conducted by the FC detailed in Algorithm 16.

Intuitively, since the FC validates the entire HMT in the next phase to prevent node injection, intermediate validation only reduces falsified message propagation, and δ should therefore be tuned to balance computational and transmission costs. This effectively limits adversarial interference to subtree pruning: although it can make at most one false observation per compromised node, it could ignore potentially many uncompromised descendants by dropping their results.

Once a node receives a message from all descendants, or the sensing window closes, it merges the received subtrees with its own local observation, and sends the combined

Algorithm 15: Transmission of observation message.

```
TransmitObservation(η, φ):
subtree = messages received from descendant children
obs = node-local observation
η_F = H(η|φ)
relation-sig = ∅
relation = ∅
foreach child_i ∈ Sort(subtree_keys) do
    relation-sig = relation-sig ⊙ child_i
    relation_i = child_i
relation-sig = H(relation-sig ⊙ ν_sig)
obs-sig = H(η_F ⊙ obs ⊙ ν_sig)
msg = subtree
msg_{ν_id} = (obs, obs-sig, relation, relation-sig)
Transmit msg to parent of ν, if one exists for φ.
```

information to its parent (Algorithm 15). The process repeats until reaching a root, where it awaits further action by the FC.

Data Collection from Roots

Finally, once the results have propagated to the roots of the forests, the FC collects the data. Using the nonce η, the FC generates the ensemble vector $\vec{e}$, using the geographic information to guide data collection; this could include physical mobility [149], long-distance wireless communication [46], secure node-to-node sensor network protocols [112], or other mechanisms.

4.6.3 Phase 3, Ensemble-based Fusion Determination

Prior to interpreting the data, the FC must validate the composition of each collected subtree within the ensemble (via Algorithm 16), discarding results from roots whose signatures do not authenticate. This filters out all node injections from the data, although the remaining data may still contain value injections (see Sec. 4.5.2).

Algorithm 16: Validation of observation message.

```
VALIDATE(message, curr-depth, δ):
if H(η_F ⊙ obs ⊙ obs-id ⊙ ν_sig) ≠ H(obs-sig ⊙ ν_id) then
    return False
if curr-depth < δ then
    if child-id_1 ⊙ ... ⊙ child-id_n ⊙ parent-id ⊙ ν_sig ≠ relation-sig ⊙ ν_id then
        return False
    foreach child ∈ ν_children do
        if ¬VALIDATE(message, curr-depth + 1, δ) then
            return False
return True
```

Normalized Observational Keyscore

After evaluating the forest ensemble, the FC computes a *keyscore*, $\kappa(o)$, for each observation, o, according to the following equation:

$$\kappa(o) = \sum_{\mathcal{F} \in \mathcal{E}} \frac{\sum_{\nu \in \mathcal{F}} obs(\nu, o, \mathcal{F})}{|\mathcal{F}|}. \tag{4.1}$$

Here, $\mathcal{E}$ denotes the ensemble, $\mathcal{F}$ denotes a forest, $|\mathcal{F}|$ denotes the number of nodes in the forest, and $obs(\nu, o, \mathcal{F}) = 1$ if the node ν reported observation o for forest $\mathcal{F}$ (and 0 otherwise). Intuitively, the keyscore quantifies the prevalence of an observation within and across forests in the ensemble.

Deception Classification

Since under certain attack models, the adversary may drop observations, Pando attempts to classify node behavior. Specifically, the FC assesses structural differences between forests in $\mathcal{E}$ to gather information about routing behavior of participating nodes (Alg. 17).

Algorithm 17: Classification-based weight generation.

```
   CLASSIFICATION(E):
 1 was-parent, was-child = {}
 2 N = {}
 3 valid-subtree = ∅
 4 foreach F ∈ E do
 5 |   valid = VALIDATE(F, 0, ∞)
 6 |   if valid then
 7 |   |   valid-subtree = valid-subtree ∪ F
 8 |   |   foreach ν ∈ F do
 9 |   |   |   was-parent[ν] = was-parent[ν] + 1
10 |   |   |   foreach child ∈ children(ν, F) do
11 |   |   |   |   was-child[child] = was-child[child] + 1
12 |   |   |   |   N[child] = N[child] ∪ ν
13 |   |   |   |   N[ν_id] = N[ν] ∪ child
14 ω = {}
15 foreach ν ∈ N do
16 |   ω[ν] = 0.5
17 |   observed = (was-parent[ν] > 0 ∨ was-child[ν] > 0)
18 |   imbalance = abs(was-parent[ν] − was-child[ν])
19 |   if imbalance > |N[ν]| then
20 |   |   ω[ν] = 0
21 |   else if observed then
22 |   |   ω[ν] = 1
23 return valid-subtree, ω
```

To do so, the FC evaluates how many times a node served as a parent, how often it was a child, and counts the number of distinct neighbors with which the node interacted. Using this information, Pando computes a heuristic-based behavioral classification score for each node.

Let $children(\nu, \mathcal{F})$ denote the set of children of the node ν on the forest $\mathcal{F}$, and let $parent(\nu, \mathcal{F})$ denote the set of nodes which claimed ν as a descendant.

Also let $pf(\nu)$ denote the number of times node ν reported having a child:

$$pf(\nu) = \sum_{\mathcal{F} \in \mathcal{E}} [|children(\nu, \mathcal{F})| > 0], \tag{4.2}$$

and let $cf(\nu)$ denote the count of occurrences where ν was recorded as a child of another node in the ensemble:

$$cf(\nu) = \sum_{\mathcal{F} \in \mathcal{E}} \sum_{u \in \mathcal{F}} [\nu \in children(u, \mathcal{F})]. \tag{4.3}$$

Intuitively, as the ensemble size increases, I expect $pf \sim cf$, and a deviation from this could highlight attempted deception. I thus define node trust $\tau(\nu)$ as:

$$\tau(\nu) = \begin{cases} 1, & (cf(\nu) = 0) \text{ or } (\frac{pf(\nu)}{cf(\nu)} > 1 - \sigma) \\ 0, & \text{otherwise} \end{cases}. \tag{4.4}$$

Here, σ represents risk tolerance, with larger values corresponding to a greater willingness to trust nodes with imbalanced roles, and smaller values representing a preference for strict balance. Too high, and adversarial nodes will be improperly trusted; too low, and it may reject non-malicious nodes solely due to stochastic fluctuations in tree formation.

Due to the method used to generate φ (see Sec. 4.6.1), roots are always counted as their role cannot be spoofed. Conversely, by maximizing Eq. 4.2 I prevent leaves' observations from always being ignored. However, nodes toward the center of the deployment (e.g. with higher connectivity) may exhibit more role variability than less well-connected nodes. To account for this, I incorporate the connectivity neighborhood $\mathbb{N}(\nu)$ as a normalizing factor, allowing more active nodes greater leeway before penalizing the role imbalance. Consequently, nodes at the periphery of the deployment

will almost always be classified as malicious, since they only report child values when φ falls nearby.

Classification-Informed Keyscore Boosting

I then leverage $\tau(\nu)$ to perform *classification-informed keyscore boosting* (CIKB) as follows:

$$\kappa_b(o) = \sum_{\mathcal{F} \in \mathcal{E}} \frac{\sum_{\nu \in \mathcal{F}} obs(\nu, o, \mathcal{F}) * \tau(\nu)}{\sum_{\nu \in \mathcal{F}} \tau(\nu)}. \tag{4.5}$$

Note that, since the $\tau(\nu)$ term is considered for both numerator and the denominator of the observation counting in forests, the cumulative sum of keyscores across all entries in the observation set remains equal to 1.0.

Fusion Determination

Once the FC computes $k_b(o)$ for all distinct, valid observations, it makes the final fusion determination by ordering the observations by descending keyscore and selecting the observation with the highest keyscore as the fused observation.

4.6.4 Messaging Costs

Under tree-based routing, each node will send a message only once per ensemble; thus, the number of transmissions during the data gathering phase of Pando is $O(nm)$, where n represents the number of nodes and m represents the size of the ensemble. This aligns with efficiency bounds reported by other message-passing approaches [3], although Pando operates under a stronger "colluding" adversarial model. FC data collection costs are proportional to the number of roots in the forest ensemble, and depend on the collection mechanism.

4.7 Evaluation

4.7.1 Experimental Setup

To evaluate Pando, I performed network simulations to trace message flows, including adversarial behavior, under different scenarios. All reported results represent averages across 100 executions. Network topologies were generated randomly using a uniform distribution, and naturally-partitioned networks were re-generated; although not required for Pando functionality, this simplifies evaluation without reducing generality. I tested against $n = 1000$ nodes with average degree ~ 14, for ensemble sizes of $m = \{1, 2, 5, 10, 20\}$ and $\sigma = 0.25$ (appropriate for uniformly distributed networks). Observation cardinality was two, the binary case.

Compromised nodes were chosen at random from the base topology, with adversarial population sizes of $\{0, 10, 20, 30, 40, 50\}\%$. Since Pando is able to detect and filter out all node injections, in the experiments I consider the *worst case* where the compromised nodes perform value injection but not easily-detected node injection. For the traffic dropping behavior, I considered all five deception strategies reported in Section 4.5.2).

4.7.2 Effectiveness Measure

I report observation correctness of a sensing round as the effectiveness measure. More specifically, since I evaluate the binary case, with true, $\top$, and false, $\bot$, observations from the network, and since confidence-boosted keyscores across all observations sum to 1, a keyscore for the true observation $\kappa_b(\top)$ less than 0.5 indicates fusion error (failure).

Observation Confidence

Correctness requires access to ground truth information, an impractical requirement for real-world deployments. As an alternative, we also measure observation confidence, providing the FC with a metric to measure the level of adversarial behavior in the network. Since the FC roughly knows the number of nodes in a deployment, and the adversary's primary mechanic for disrupting fusion lies in pruning observations, the percentage of observed nodes provides a simple method for estimating pruning activity which we refer to as "confidence". However, as the adversarial population increases, pruning becomes less necessary, as the adversary begins to rely more heavily on correctly-signed (but false) observations. Thus, the confidence value initially decreases, then increases again. Instead, the FC computes the pooled mean and pooled standard deviation of the confidence across the forest ensemble. The U-shaped pattern holds for the pooled mean, but the pooled standard deviation decreases near-monotonically. By evaluating both, the FC can differentiate between the two extremes of the curve, providing a weighted confidence score. Intuitively, this reflects that for a network with few adversaries, the forest ensemble should have many "tall" forests, and a few "short" forests, while for a network with many adversaries, the ensemble will contain mostly forests of the same (shorter) height.

Each forest in the ensemble provides a predominant observation; if every forest in the ensemble reports the same value, either (A) the value is likely correct, or (B) the adversary totally dominates the network. Thus, as the adversarial population (or rate of malicious behavior) increases, the variance across the ensemble will initially increase, and then decrease.

Classification Accuracy

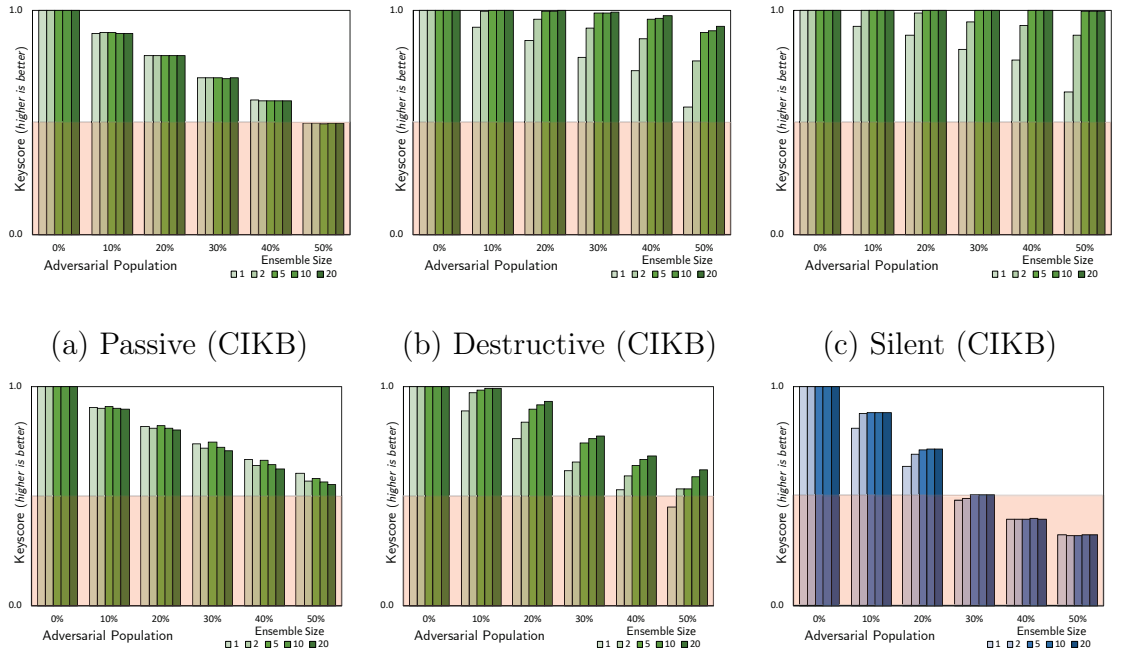
In cases where the FC does detect adversarial interference, it may wish to identify the nodes exhibiting malicious behavior. Therefore, we measure the performance of its classification in Phase 3 using F1 score. Due to the construction of the classification heuristic, we expect that nodes around the periphery (true leaves) are likely to be false positives, but if the FC can identify nodes as persistently malicious (for example, across multiple observation rounds), it can further improve fusion correctness by silently ignoring observations from those nodes.

We evaluate our approach from three perspectives: (1) correctness of observed values, (2) confidence in observed values, and (3) efficiency of data collection. Correctness effectively measures the resilience of Pando to adversarial interference, by comparing the value to which the network fused against the expected value. However, ground truth availability is limited in practice.

Figure 4.2(a) shows that for a passive adversary, which does not perform pruning, but injects false observations, the final keyscore reflects its share of the population. Since the adversary does no pruning under this attack model, and relies entirely on injecting false observations, no mitigation can be effective, since the adversary obeys the protocol.

In contrast, for destructive or silent adversaries who aggressively prune the network to reduce the amount of data reaching the FC, Figures 4.2(b) and 4.2(c) show that these approaches are easily detectable by the classifier, even for small ensemble sizes and when the adversary controls many nodes.

In the sneaky model (Figure 4.2(d)), the attacker attempts to mislead the FC into misclassifying good nodes by actively (self-)pruning malicious nodes. However,



(a) Passive (CIKB) (b) Destructive (CIKB) (c) Silent (CIKB)

(d) Sneaky (CIKB) (e) Selective (CIKB) (f) Selective (without CIKB)

Figure 4.2: Results by deception strategy, adversarial population, and ensemble size; shaded region indicates fusion error

since the heuristic relies on detecting the pruning behavior itself, this actually hurts the adversary more than it helps. In fact, by working against its own interest, it actually ends up doing relatively little pruning because correct values dominate the propagation.

Since increased ensemble sizes primarily aid in detecting pruning behavior, no corresponding increase is observed in this scenario. When considering the most-challenging selective case, as in Figure 4.2(e), the classification heuristic provides a strong countermeasure against the attack model by significantly boosting keyscore – even an ensemble size of two provides protection against an adversary controlling 50%

of the network.

To illustrate the contrast, I also examine the selective model's performance under a scenario where the classification heuristic is artificially disabled (Figure 4.2(f)). As seen here, when CIKB is not used, the adversary would be able to pass the fusion error threshold when controlling only ∼30% of the network, even for larger ensembles; at the 30% threshold at which it was previously effective, even a single forest is sufficient to prevent fusion error. For ensembles of 10 or more, passive actually outperforms selective for high adversarial populations.

Interestingly, I conclude that the adversary's best choice for maximizing fusion error lies in adopting a passive attack model. While this appears counter-intuitive, it aligns well with previous work assessing game theoretic optimality for message-passing fusion protocols [2].

4.8 Summary

In this chapter, I introduced Pando, a method for addressing the Byzantine distributed sensor fusion problem. I proposed a method for the creation of decentralized forest ensembles using minimal node-local storage and shared information. Using these ensembles, I described a message passing and validation framework using homomorphic Merkle trees (HMTs) for efficient and secure communication and data aggregation. Finally, I detailed a classification-informed keyscore metric, which leverages topological information embedded in HMTs for robust and reliable sensor fusion.

Chapter 5

PATTERNS OF RESILIENCE

In recent years, the severity and number of threats to networked devices have increased, highlighting the need for effective network security. However, functional requirements often override security considerations, leading to subtle effects on safety. Additionally, heterogeneous network architectures, advanced persistent threats, and automated attacks further complicate matters.

To address these challenges, I propose DART, a comprehensive tool that quantifies networks' resilience to attacks, identifies and ranks deployment substructures' impact on security, and recommends improvements to network defensive posture. DART leverages a abstract multilayer graph derived from both logical and credential connectivity to establish an attack context. Then, I execute agent-based simulations to model possible attacks and capture novel metrics on the network under threat. I dynamically adjust simulation costs, automatically terminating when appropriate. Finally, I identify and rank causally-linked substructures associated with successful attacks, providing insights to designers about security flaws in both connectivity and credentialing.

I evaluate my approach against graph-theoretic alternatives with several synthetic and real-world topologies, demonstrating that adopting DART's recommendations significantly reduces adversarial success and increases exploitation costs without undue functional disruption. This chapter captures work presented in [18].

5.1 Overview

The increasing ubiquity of networked devices is leading to a corresponding increase in the complexity and heterogeneity of these networks. The adoption and utilization of computational resources across various domains further contributes to the expansion of these networks and their impact. These trends necessitate effective defense against rising threats, such as ransomware, cyberwarfare, and other forms of sophisticated malicious behaviors, which pose significant risks to both users and practitioners. Traditional security approaches that rely on prevention have continually failed to provide adequate protection due to misconfigurations or software vulnerabilities. Detection-based approaches require human operators to screen false positives [63, 133], an expensive and possibly insufficient process for responding to automated threats. Therefore, layered defenses that integrate multiple approaches offer the best chance of success against determined attackers, based on principles and best practices learned through experience.

Even heavily-layered defenses may be inadequate to protect sensitive assets [136]. This is in part because networks must accomplish functional tasks that rarely reflect a static reality. For example, "quick fixes" to address functional problems in deployed, operational networks may undermine the most secure initial design. Unplanned administrative access routes, which can leak credentials to malicious eavesdroppers, may bridge networks in unexpected ways. As deployments grow and evolve, incremental changes can accrue to produce substantial security weaknesses. Addressing this tension between operational, functional realities and ideal security constraints is challenging. In this chapter, I propose a solution to this challenge, DART, which helps identify and highlight structural flaws in a (potentially evolving) network. DART consists of four

phases:

- First, *I leverage abstraction to simplify the problem to a more tractable one.* While many existing approaches expect a highly-detailed view of the network, such as the operating system or installed software on each node, obtaining this information may be difficult or impossible. Additionally, network topologies might contain nodes that subvert standard assumptions, such as electromechanical switches, optical interconnects, or virtualized software agents. Therefore, I exploit the logical connectivity and trust relationships between nodes to build a directed, multilayer graph to encode these abstract relationships. A key motivation for this abstraction lies in limiting onerous data collection or specification, minimizing the gap between design and implementation.

- Second, *I compute a preliminary heuristic evaluation of the relative risk of each node in the compound graph.* In this phase, I let users optionally label nodes on this graph with source and/or goal weights, representing the risk of compromise and the value to an attacker respectively. For example, a public-facing web server might have a higher source weight than a client laptop, which in turn would have a higher weight than an air-gapped mainframe. On the other side, a domain controller might have a higher goal weight to an attacker than a load balancer, while a thin client might have no value to an attacker at all.

- Different network topologies may be vulnerable to different types of attacks. For example, networks with strong but shallow defenses may be susceptible to fast, brute-force attacks based on zero-day exploits, while deeply-layered defenses may be more vulnerable to slow, subtle approaches based on credential reuse.

To account for this, *I next execute an agent-based model to simulate en ensemble of potential attacks against the network.* Since the tactical or strategic decisions of an adversary may be unpredictable or surprising, I avoid assigning definitive characteristics to my attackers – instead, I rely on stochastic sampling from the space of all possible attack behaviors to discover attacks that prove to be particularly effective. DART automatically adapts the number of simulations to the characteristics of the network.

- Finally, to capture nuances such as attacks that wasted attacker resources for minimal gain, *I assess each attack and assign an efficacy score, which are then used to project potential attack features from the observed compromises.* This helps in decoupling the sequencing of an attack's evolution from its efficacy and enables us to look for commonalities among the most-effective attack features, rank these substructures based on their relative contributions, and inform the user about which substructures of their network most undermined its overall security.

To confirm that addressing the vulnerabilities identified by DART decreases adversarial success and increases exploitation cost, in §5.7, I evaluate DART against both synthetic and real-world network topologies encoding several different deployment scenarios. I then generate alternative topologies based on interventions recommended by DART, then re-evaluate the revised networks' resilience to attack. Experiments show the effectiveness of my approach and illustrate its real-world practicality.

5.2 Related Work

5.3 Network Defense

In the literature, past works typically target the prevention or detection of adversarial movement in the network. Traditionally, prevention-based techniques attempt to keep the adversary from gaining a foothold at all [119]. Unfortunately, the variety of vulnerabilities and the widening attack surface of modern devices have shown that prevention alone rarely guarantees security. Indeed, the proliferation of computing power greatly complicates this process [75], making perfect prevention impractical. I refer to this concept as *permeability*, and use this term throughout to refer to networks that exhibit imperfect perimeter security.

Detection-based approaches work with this assumption by taking the presence of attacks as given and aiming instead at detecting them. These approaches face other challenges, such as the well-known "long tail" of network traffic [42] which complicates inference-based approaches for anomaly detection. In [63], analysis of login events detects the presence of adversarial lateral movement with high probability. However, for every real attack detected, defenders must sift through hundreds of false positive alerts, and the approach requires network-specific filters to achieve even this level of efficacy. Additionally, mitigation steps to address a detected attack are often left as an exercise for the reader. Unfortunately in many cases, post hoc mitigation cannot remedy the damage already inflicted, and approaches that work in real time may be preferred.

As exploitation and attacks undergo additional automation [37], it is far from clear that systems that rely on humans in the loop can address these attacks on the same

time scales as automated systems. Though some works make progress toward this goal [20], difficulty in accurately classifying attacks and reticence toward disrupting normal traffic have so far undermined such tools.

5.3.1 Network Simulation

Given that many networks follow a scale-free structure [11], exhaustive evaluation can prove prohibitively expensive. To that end, many graphical approaches have adopted simulation-based techniques to sample from this large space. In particular, modeling techniques that rely on training data [45] produce mixed results dependent on the quality of that data.

One of the most well-known network simulators, NS3 [124] takes a high-fidelity approach to simulating networks. Reaching all the way down to the physical layer, this approach produces a detailed trace of network traffic. This fidelity also proves a drawback for some models, where connections may be abstract, and also introduces significant computational costs.

Other approaches have leveraged network simulation for security purposes, such as [31]. The authors evaluate multiplex network and power graphs to assess the security of smart grid systems. However, the fidelity required to accurately detect specific attacks in this problem domain forestalls more generalized applications of these techniques.

5.3.2 Security Quantification

Past attempts to measure security rely primarily on high-fidelity, probabilistic representations of the networks [47]. Specific software vulnerabilities (and their

severity) guide analysis, which can be difficult to obtain and ignores the rising threat of zero-day exploits. Other approaches [104] rely on causality-preserving inference while still relying on functional and configuration-specific data rather than architectural features; these high-fidelity models may be expensive to build, maintain, and evaluate.

Other approaches attempt to concretize the problem by choosing a specific dimension (e.g. time) and measuring it [87, 101]. These approaches rely on qualitative estimations as input, such as the skill of attackers, which limits an assessment's accuracy to the quality of the assumptions.

An ongoing challenge to security measurement lies in the assessment of impact. Namely, data leakage, degradation of functionality, frequency of compromise, and other measures form plausible but function- and context-dependent valuations. Though some frameworks have been proposed [54, 94], consensus remains elusive.

5.4 Problem Formulation

5.4.1 Network Model

The foundation of DART relies heavily on graphs. I first define a set of vertices $\mathcal{V}$, representing a semantic mapping to system components. These connect in a multilayer graph defined by two sets of edges: a "connectivity" layer $\mathcal{E}_C$ and a "trust" layer $\mathcal{E}_T$.

- **Connectivity edges** reflect bidirectional interactions between nodes, such as logical connectivity or observed network traffic. An adversary may use these edges to propagate their influence within the network, but only with difficulty – for example, capturing a connected device on a local subnet might require the discovery and

Table 5.1: Notation & Conventions

$\mathcal{V}$	The set of nodes
$\mathcal{E}_C$	The set of communication edges
$\mathcal{E}_T$	The set of trusted edges
$\mathcal{V}_s$	The (sub)set of source nodes
$\mathcal{V}_g$	The (sub)set of goal nodes
$\mathcal{V}_c$	The (sub)set of nodes under adversarial control
$\mathcal{V}_t$	The (sub)set of nodes under threat
X_i	The state of a variable X at step i
$(A \to B)$	An edge from A to B
$\langle a, b, c \rangle$	The ordered series of a, b, c
k	Feature length
ℓ	Attack budget

exploitation of a zero-day vulnerability[1]. Since connectivity edges reflect *possible* communications, I treat these edges as undirected. For simplicity, in this chapter I assume (but do not require) a connected graph of communication edges.

- I also define **trust edges**, which reflect special relationships between nodes, such as implicitly trusted links, Kerberos tickets, or SSH keypairs. An adversary who holds a position of trust over another node can exploit the relationship to extend their area of control with relatively low risk. Since trust relationships often require asymmetry, I treat these edges as unidirectional. Note that not all graphs may contain trust edges.

For simplicity, in this chapter I assume traditional computer networks, but note that my approach could adapt to alternative domains, such as social network interactions, company org charts, or other graph-based structures encoding interaction and trust.

5.4.2 Adversarial Model

I further define a set of **source nodes** ($\mathcal{V}_s \in \mathcal{V}, \mathcal{V}_s \neq \emptyset$), locations accessible to the adversary irrespective of edge connectivity. These might represent nodes at risk of compromise via phishing, or publicly-facing web servers, for example. Each source node optionally pairs with a weight to allow for nonuniform risk profiles.

Similarly, I also define a set of **goal nodes** ($\mathcal{V}_g \in \mathcal{V}, \mathcal{V}_g \neq \emptyset$) which correspond to vertices with intrinsic value to the adversary – that is, capturing a goal node represents an added benefit to the adversary (potentially beyond its relative position or connectivity in the graph). Goals might describe a critical piece of equipment,

[1]Here, I treat the risk of traffic bypassing software-defined networking (SDN) rules as orthogonal to my approach, and therefore focus on logical links.

a database with sensitive information, or a domain controller. Optional weightings allow for fine-grained tuning of nodes' perceived sensitivity. The source and goal node subsets need not be distinct.

I assume, prior to the attack, my adversary perceives all nodes in the source subset but no other nodes or edges, and must discover the network (as well as the goal nodes) through exploration: the adversary may spread within the network and extend their influence by **capturing nodes** in the network and using the outgoing edges to learn and **threaten** other nodes in the network. The adversary **knows** the captured subgraph, the set of threatened nodes, and all edges originating from the captured subgraph. I refer to the dynamic subsets of captured and threatened nodes as $\mathcal{V}_c$ and $\mathcal{V}_t$, respectively. In cases where I must refer to a dynamic set at a given sequence step i, I write $\mathcal{V}_{c(i)}$.

An **attack**, a sequence of **action**s by an adversary, on a given network starts at a source node and can proceed through different mechanisms:

- Adversaries may **exploit** threatened nodes, an abstraction representing capturing a node without leveraging a trust relationship. For example, an exploit might describe a spearphishing campaign against an administrator, or utilizing a zero-day vulnerability to gain elevated privileges on a server. To maximize adversarial power, I conservatively assume that such exploits always succeed once launched. To counterbalance this power, I also assume a proportionately higher cost associated with exploits and bound them within an **exploit budget**[2], ℓ.

[2]Such an assumption aligns with practice; for example, the well-known Stuxnet attack used four zero-day software vulnerabilities [108], plus at least one operation to plant USB drives, for an exploit budget $\ell = 5$.

- Adversaries may also **login**; these actions leverage trust relationships to move between systems without requiring an exploit. These reflect the discovery and re-use of credentials, or the use of trusted connections, for example. More generally, these represent mechanisms for lateral movement by the adversary. To benefit from a trust relationship, the adversary must have captured the source node for that trust edge.[3]

I refer to a collective sequence of actions by an adversary as an "attack". An attack continues until either all goals are captured, or no further actions are possible – that is, an exhaustion of the exploit budget combined with no trust edges originating in the capture region. In cases where I must refer to the action taken at step i of an attack A, I write A_i. I label attacks as **successful** or **unsuccessful** based on whether they satisfy success criteria – different success criteria, such as "*all goal nodes are captured*", "*any goal node is captured*", or "*more than X% of cumulative goal weight are captured*", are possible. Without loss of generality, in this document, I use the first of these criteria, but DART implements the other options for users with different risk tolerance profiles. Note that here, I do not consider the duration of the attack when defining success – therefore, the success criteria are agnostic of time scale and can represent both slow, methodical behaviors and fast, automated ones.

Since attacks rely on both a given graph $\mathcal{G}$ and the exploitation budget, $\ll$, of the adversary, I define an **attack context** as $\mathcal{C}(\mathcal{G}, \ell)$. For a given attack context, there typically exists many possible valid attacks – I refer to the space of all possible attacks

[3]Since detecting abnormal behavior by valid credentials remains a challenging problem [63], I do not put a bound or budget on the use of captured credentials. Indeed, recent work has shown that credential reuse dominates zero-day exploits in practice [139] for advanced persistent threat actors. Thus, I expect the role of logins to play a critical role in the overall security of the system.

as the *attack space*. This attack space contains both **effective** and **ineffective** attacks (I address the concept of attack effectiveness in Section 5.5.1).

5.4.3 Budgeted Ensemble-based Assessment of the Network

My goal is to provide network operators with informative and actionable results, especially information about adversarial outcomes, including statistics and bounds, as well as a ranked list of sensitive graph substructures.

In practice, the attack space can be large and sparse (i.e. may contain only a few successful attacks); therefore, the large size of this space prohibits exhaustive enumeration for the discovery of successful attacks. DART therefore leverages simulation ensembles to explore many alternative attacks sampled from the attack space.

I refer to a single execution of an attack simulator as a **simulation** or a **sample**, each of which produces an attack as described previously[4]. A collection of such simulated attacks is referred to as an **ensemble**. Since this ensemble is analyzed for identifying critical features[5] of an attack, the size of a simulation ensemble (the **simulation budget**) can directly influence the quality of its results. Choosing an appropriate budget for a simulation ensemble is challenging, since underestimation may miss critical attacks and overestimation wastes resources. I address this challenge in §5.6.

[4]I intentionally avoid the application of heuristics for the generation of "realistic" attacks – an adversary aware of these heuristics could tailor their behavior to produce attacks that leverage unexpected behavior. Similar to the rejection of security through obscurity described in [56], my approach aims for efficacy even against an adversary aware of my approach.

[5]I define this in Section 5.5.4.

5.5 DART Description

I now describe DART[6]. As I described in the introduction, DART has four phases, the first of which (abstraction), is outlined in the previous section. Here, I discuss the three remaining phases: (a) initial (rough) quantification of a network's defensive characteristics to measure its resilience to attack; (b) simulation ensemble-based feature extraction to identify network's critical substructures; and (c) generation of a feature ranking based on the (statistical) relevance to the attacker.

5.5.1 Measures of Criticality and Resilience

As described earlier, DART leverages simulation ensembles to explore many alternative attacks sampled from the attack space. In this phase, DART leverages simulation ensembles to quantify several (initial and rough) criticality measurements at different resolutions that I leverage in later phases.

Impact of a Node (within a given Attack Sequence)

At its most granular, a network consists of nodes that form a natural starting point for any system relying on comparative assessment. Intuitively, the portions of the network with higher node importance (or impact) scores would reflect regions that the adversary is more able to attack. Note that the importance of a node depends on the nodes it trusts, since if its trusted neighbors become compromised, the node itself becomes vulnerable to login attacks exploiting that relationship. Nodes that trust no other nodes retain the majority of their importance, while nodes that trust many nodes dilute their value by distributing that weight amongst those nodes. Note also

[6]Defensive Assessment and Resilience Tool

that a particular node's impact during an attack depends upon the network state at the step it was captured. Capturing a single goal node may alter the reward landscape substantially, motivating an adversary to seek out additional goals and penalizing those who continue to capture now-unimportant nodes in the neighborhood of a former goal.

Given this, in order to obtain an initial, and rough, assessment of a node's significance during an attack, I leverage the well-known Personalized PageRank (PPR) [114] to integrate several different characteristics such as centrality, betweenness, and depth. Intuitively, the random walk behavior intrinsic to PPR approximates the unpredictable adversarial propagation that I assume: the underlying process models an agent which chooses a node at random from the seed vector and begins a random walk, where, at each step, the walker chooses a new edge to traverse, or teleports to any seed node. The steady-state distribution of the walker's location at the reachable nodes forms the PPR score of the network. Intuitively, these scores capture both the adversarial threat and risk propagating through the network. I compute the PPR scores, with a seed vector populated by nodes useful to the attacker; more specifically, I define **node impact** for a node v at step i as follows:

$$NI(v, i) = \text{PPR}(\mathcal{G}, \mathcal{S})[v], \tag{5.1}$$

where $\mathcal{S} = (\mathcal{V}_s \cup \mathcal{V}_g) \setminus \mathcal{V}_{c(i)}$ denotes the set of seed nodes consisting of the source and destination nodes, minus nodes in a captured state at time i. The inclusion of both source and goal nodes further improves scoring by providing a smooth transition from easily reachable but comparatively low-value nodes to harder to reach, higher-value nodes nearer to the goals: I include source nodes in the seed vector to assign greater node impact to easily-reachable nodes; similarly, by including goal nodes, I increase

the impact of nodes proximate to valuable vertices.

Note that, while, as I later show in the experiments, PPR is not necessarily a good measure of criticality in and of itself, it provides a good starting point for analysis.

Effectiveness of an Attack

As defined above, the concept of the *node impact* ignores the broader context of an attack. At first glance, the accumulation of individual impacts might reflect the collective importance of the sequence of actions and a naive approach to associate value to an attack might sum the impact values of the captured nodes, but doing so subtly alters the reward landscape: attackers maximize impact by capturing more nodes *near* to the goals, rather than targeting goals themselves.

Given a sufficient attack budget (assuming that the network is not partitioned) any attack will eventually succeed but the most impactful attacks are those that do so using the fewest steps possible. Therefore, I define the *degree of effectiveness* for an attack A of length ℓ as follows:

$$AE(A) = \frac{\sum_i^l NI(A_i, i)}{\ell} \tag{5.2}$$

This construction rewards adversaries who explore new areas of the network toward uncaptured goals and penalizes those who remain in known regions, meander aimlessly, or move into irrelevant portions of the network.

Attack Resilience of a Network

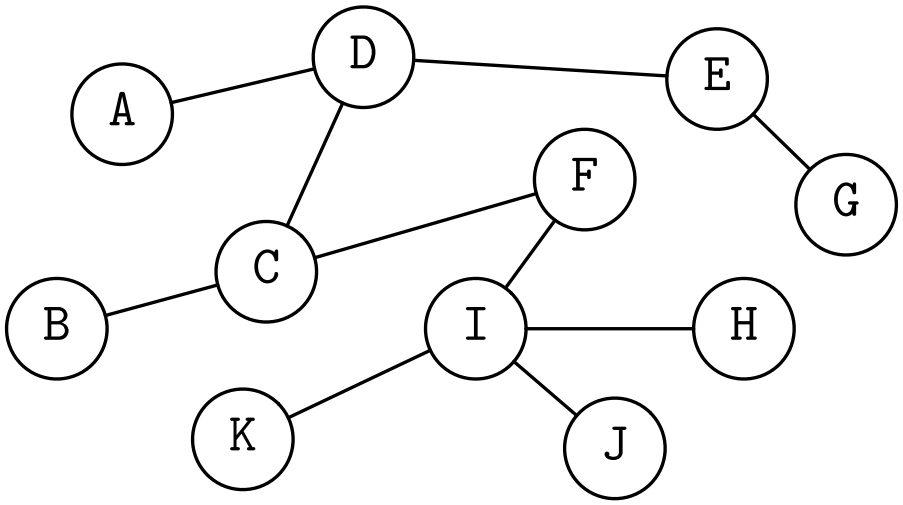
Assuming a sufficiently large simulation budget[7], the attacks in the ensemble can be used to quantify several measures of criticality and resilience:

[7]I address the challenge of picking an appropriate simulation budget in §5.6.

- *Attack success ratio:* What percent of attacks ended in success for the attacker?
- *Exploit bounds:* what were the minimum, mean, and median number of exploit actions used during successful attacks?
- *Login bounds:* what were the minimum, mean, and median number of login actions used during successful attacks?

Answers to these questions can help construct a high-level characterization of the network's security. For example, an attack success rate ≈ 0 suggests that a network's security likely reflects best practices. When attacks do succeed, the bounds on required effort illuminate security posture; for example, a network with low average exploit usage but high logins likely indicates a brittle perimeter, and a highly-interconnected interior with substantial implicit trust.

In theory, users could leverage these comparisons for A-B testing [22] to evaluate whether proposed topological changes or node deployments would produce positive, negative, or neutral effects on network resilience. In practice, however, these high-level findings are informative but not actionable – users may not know which changes would best improve their network. Secure-by-design network topologies typically rely heavily on the experience and expertise of subject matter experts. However, the vast majority of deployment environments do not have the resources to retain an expert, and untrained users likely will not know how network changes will affect security. DART provides further analysis to provide actionable recommendations.



Attack: A,D,C,B,F,E,I,G

$F(k = 2)$: {AD,DC,DE,CB,CF,FI,EG}

Figure 5.1: Features of length (k) 2, identified on a sample attack

5.5.2 Attack Features

Attacks encode sequences of actions – but, the specific ordering of that sequence may or may not matter to the successful evolution of the attack: by definition, an attack against a non-source node must be launched from any previously-captured node, but the path by which the exploit is delivered could arrive along many possible routes. Further, the sequence may reflect an attack with many evolving phases interleaved together – an attacker need not launch their next attack from the most recently captured node.

To determine if the specific sequence of actions does matter, I *project* my attacks into a feature domain where I can address an attack's causal constraints on edge

traversal, in the space of equivalent, valid orderings. More specifically, given an attack context, $\mathcal{C}$, an attack, A, and a feature length, $k < \ell$, I define the set of **attack features** as follows (Fig. 5.1):

$$\mathcal{F}(\mathcal{C}, A, k) = \{\langle p_1, p_2, \ldots, p_k \rangle \mid \forall_{1<i\leq k} \quad (p_i \in A) \ \wedge \\ (p_{i-1} \rightarrow p_i) \in (\mathcal{E}_C \cup \mathcal{E}_T)\}. \tag{5.3}$$

That is, given an attack, I evaluate the (constrained) power set, consisting of all possible (potentially non-continuous) k length subsequences that are compatible with the structure of the underlying graph: for any node in an attack feature, it must either (a) appear as the first node in the attack feature, or (b) there must exist an edge from the preceding node to this one. The edge constraint reveals the route-based nature of attack features by speculating on possible paths by which an adversary captured a given node during the attack. Note that, since attack features are drawn from attacks, and the constituent actions in an attack do not repeat, by definition all attack features are acyclic subpaths.

Note that due to the use of attacks as inputs to the projection process, the distribution of attack features contains a complex observational bias: (a) First, shorter features appear more frequently than larger ones. Since there are fewer causal constraints for shorter features to satisfy, they may be projected more often. (b) Second, features whose constituent nodes appear proximate to source nodes are easier to discover, since entrances are naturally more well-explored than the periphery. Since many sampling and statistical techniques rely on an independence assumption that each feature is drawn randomly from the space of possible features, these nonuniformities complicate further analysis. I address these biases and how to mitigate their influence

in §5.6.

5.5.3 Feature Relevance

Using the measures of node and attack criticality and attack features previously proposed, I now turn my attention to interpreting the importance of attack features themselves.

Each attack in the ensemble is either successful or not. However, focusing solely on the success ignores the efficiency/inefficiency of the attack. I, therefore, introduce the concept of **feature relevance** that helps differentiate among features by taking into account both the frequency with which they appear in the ensemble and with respect to the performance of the attacks in which they occur:

- Attack features that consistently produce high-impact attacks can be reported to the user for mitigation or removal.
- Attack features that correspond to especially low-impact attacks highlight areas of the network that waste adversarial resources, evaluating honeypot or honeynet deployments.
- Relevance rankings form a real-valued absolute ordering over all observed attack features, simplifying interpretation.[8]

Attack Relevance

Let us consider the effectiveness of an ensemble of (simulated) attacks (e.g. Fig. 5.2). The shape of this plot would reflect the latent attack resilience of the underlying

[8]Any recommended edge removals prohibited by functional requirements (e.g. partitioning) can be ignored in favor of subsequent alternatives.

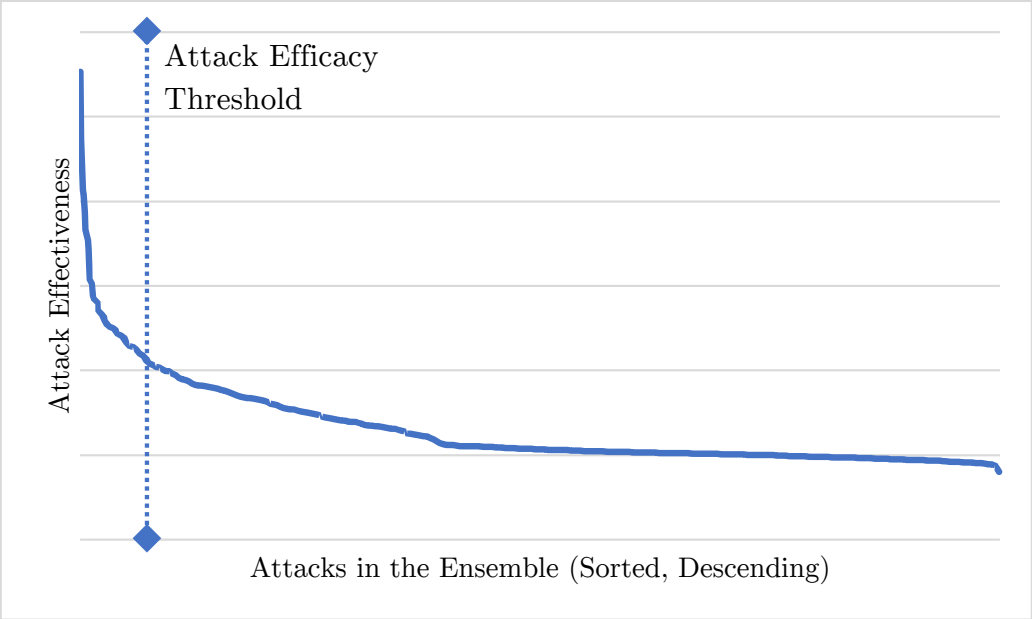


Figure 5.2: Effectiveness scores of attacks in an ensemble. Attacks on the left of the threshold have high normalized effectiveness (they are *relevant*), while those to the right contribute much less (*irrelevant*). We use this to guide feature selection.

network topology. For example, in fully-connected networks, the gradient is likely to be relatively linear, or even flat: the attackers' actions play little role in their success or failure. Conversely, in topologies with more nonuniform connectivity, the importance of the attackers' choices would manifest more directly, with some attacks marked significantly more effective than others: when a relatively few routes exist from source to goal, the attackers who discover these routes will dramatically outperform the average.

This raises the question: can the distribution's structural variability be used to inform my feature relevance computation? In particular, I partition the attacks into

two categories: **relevant** ($\mathcal{R}$) and **irrelevant** ($\mathcal{I}$) attacks[9]. The first set contains attacks whose features most likely indicate success and thus are more likely to be of interest to attackers. The second set consists of attacks that failed or wasted the attacker's exploit resources. In cases containing a uniform or linear distribution, the ensemble can be partitioned into two balanced classes equally around the median. In nonuniform cases, however, such a partitioning may not be appropriate; instead, I identify the elbow of the distribution using the well-known Kneedle approach [127] and use this point for bipartitioning the ensemble of attacks. The resulting uneven partitions however can result in biases in analysis; to tackle this, I adopt the hybrid SMOTE-TL approach for over- and under-sampling as proposed in [13]. Intuitively, oversampling in the minority class using SMOTE boosts recognition of key features, while under-sampling in the majority class using Tomek Links prunes redundant information.

At the conclusion of this step, each attack maps to (1) a set of attack features derived from its underlying action sequence, and (2) a relevance classification label. I next leverage this to compute the relevance of attack features.

Computing Feature Relevance

To compute the relevance of my attack features using the ensemble, I compare statistical properties of attack features in relevant ($\mathcal{R}$) and irrelevant ($\mathcal{I}$) attacks sets. Intuitively, attack features which often appear in relevant simulations and rarely appear in irrelevant ones are more interesting to an attacker. Inspired by the relevance

[9]In other words, in the context of assessing attacks, I use the terms relevance and efficiency interchangeably, with the caveat that all unsuccessful attacks are ineffective

feedback process in the information retrieval literature [126], I compute the **relevance score** of an attack feature, f (observed in the ensemble), as follows:

$$RS(f) = \log \frac{p(f|\mathcal{R})}{1 - p(f|\mathcal{R})} - \log \frac{p(f|\mathcal{I})}{1 - p(f|\mathcal{I})}, \tag{5.4}$$

where the notation $p(f|\mathcal{X})$ describes the percentage of attacks in the set $\mathcal{X}$ which contain the feature f.

Note that, when considering features of different lengths, the biased feature sizes described in §5.5.2 become critical: since shorter features are much more likely to be observed, this relevant frequency can introduce bias relative to the much rarer longer features. To ensure that DART identifies features of any length, I de-bias the ranking by comparing features only against features of the same length. This eliminates problems that would have occurred if comparing attack features of different lengths.

5.5.4 Critical Features

Once the relevance score has been computed for each feature (of a target length, k) observed in the ensemble, I prepare final recommendations for the user, listing the features in ranked order of relevance:

- Features with large positive scores (**critical features**) indicate substructures of the network that are critical for the success and effectiveness of an attack.
- Features with large negative scores indicate substructures of the network that negatively impact an attack by leading the attacker to get lost in the network.
- Features with scores close to 0 have minimal impact on the success and effectiveness of an attack.

In cases where critically-relevant edges align with connectivity edges, the user can restructure the network topology to remove those links through e.g. logical connectivity changes at the SDN layer. Alternatively, the user may wish to break particular trust edges (or chains of trust edges) through e.g. key invalidation. When removing a particular attack feature that may violate functional constraints, the user may consider the following feature in the list.

5.6 Budget Estimation

As described in the previous sections, DART relies on a simulation-based approach to assess the criticality of network components. Naturally, as the number of simulations in the ensemble increases, the predictive accuracy would likewise increase, but with well-known diminishing returns [141]. Thus, choosing an ensemble budget requires balancing three desiderata:

- **Desideratum #1:** Minimize computational and time costs,
- **Desideratum #2:** Identify potentially-rare critical subpaths, and
- **Desideratum #3:** Establish a bound on subpath discovery costs.

In this section, I describe a budget estimation strategy that combines these desiderata.

Coverage Target

Given a network to be evaluated by DART, the user does not know where in the network any particular flaws might be found, or indeed if such structural flaws exist. Intuitively, an exhaustive enumeration and analysis of every possible subpath would include any such vulnerable attack features but would be computationally intractable for a network

of even moderate complexity. Instead, I ask the user to specify a **coverage target** (ϵ), representing the percentage of valid attack features they wish DART to examine. Here, *v*alidity refers not only to the causal consistency of the attack features themselves but also to whether such a feature could occur in a given simulation context. As such, the set of valid attack features represents a relatively small subset in the space of all possible attack features, which itself reflects a small subset of the space of possible subpaths.

Progress Target

Since generation of the ensemble proceeds stochastically, I also require the provision of a **progress target** (θ), reflecting the likelihood that the ensemble will continue making reasonable progress toward the required coverage. While the coverage target dictates how much of the network to explore, the progress target allows the simulation to terminate when coverage may not be (practically) reached.

Estimating Feature Coverage

To compute the coverage provided by an ensemble, I must first compare the features in a simulation against a baseline set of attack features. That is, for each simulation produced by the simulator, I wish to compare what *was* observed ($\mathcal{F}_{obs}$) and what *could* have been observed ($\mathcal{F}_{poss}$). I refer to the ratio between these two sets as the *miss rate*:

$$MR = \frac{|\mathcal{F}_{poss} \setminus \mathcal{F}_{obs}|}{|\mathcal{F}_{poss}|}. \tag{5.5}$$

Unfortunately, this definition relies on the $\mathcal{F}_{poss}$ set which would be intractable to exhaustively generate. Moreover, I cannot rely on features collected during simulation to form my baseline set, because the sampling bias prevents independence assumptions necessary for statistical conclusions. Instead, I propose a hybrid random walk-based feature sampling mechanism[10] to approximate $\mathcal{F}_{poss}$, given an attack budget value, ℓ, and feature length, k:

1. Node (non-source) oriented feature sampling:

   (a) I assign each (non-source) node i a local exploration radius ϱ_i, equal to ℓ minus the minimum hop distance to any source node.

   (b) I remove from the graph any (non-source) node whose $\varrho_i < 0$, as it will be unreachable in practice during simulation.

   (c) For each remaining (non-source) node, I execute m random walks of length $\min(\varrho_i, k)$.

   (d) I remove duplicate features from the feature sets obtained in these walks to produce a set of attack features $\mathbb{F}_{node}$.

2. Source-oriented feature sampling:

   (a) For each source node, I execute m random walks of length $\Lambda >> \ell$.

   (b) I remove duplicate features from the feature sets obtained in these walks to produce $\mathbb{F}_{source}$.

3. I compute the final sample set $\widetilde{\mathcal{F}}_{poss} = \mathbb{F}_{node} \cap \mathbb{F}_{source}$.

[10] By basing the sampling on random walks, rather than frontier expansion as in the agent-based models, I can leverage the broad literature on fast random walks [138] to do so efficiently.

Above, Λ is a large random walk length selected in a way that permits full traversal of the network many times over, ensuring a uniform observation space.

Note that these random walks do not obey the same causal constraints used during simulations: The features in the set $\mathbb{F}_{node}$ do obey the attack length constraint, ℓ, but may violate directionality of the attack. The features in the set $\mathbb{F}_{source}$ obey directionality but ignore ℓ. By intersecting these two sets, I include in $\widetilde{\mathcal{F}}_{poss}$ only those features that satisfy both the directionality and attack length constraints.

Estimating the Size of the Simulation Budget

Using the above, I can now compute an appropriate simulation budget for creating an ensemble. As noted by Glynn *et al.*, systems based on simulation obey an asymptotic efficiency law [53]. To minimize computation costs, I terminate when I reach either of two points:

- when I have exceeded my coverage target, or
- when I have failed to reach my progress target.

When estimating the simulation budget, I rely on evaluating progress in an iterative, stepwise manner. This approach is inspired by two-phase sampling [85], in which a simulation is executed for a fixed (typically small) *initial budget* b_0. Then, the statistical properties of the results obtained in the first phase of execution inform the number of additional simulations required to reach the statistical targets.

To address the first termination case, I maintain a cumulative observation set $\breve{\mathcal{F}}_{obs}$ as the running union of features observed in the simulations executed so far:

$$\breve{\mathcal{F}}_{obs}(i) = \bigcup_{s=1}^{s=i} \mathcal{F}_{obs}(s), \tag{5.6}$$

where s represents a particular simulation instance's position in the ensemble and i indicates the total number of simulations executed thus far. Using this, I can compute the *cumulative miss rate*, $\breve{MR}_i$, at step i as

$$\breve{MR}_i = \frac{|\widetilde{\mathcal{F}}_{poss} \setminus \breve{\mathcal{F}}_{obs}(i)|}{|\widetilde{\mathcal{F}}_{poss}|}. \tag{5.7}$$

The first termination case is satisfied when $1 - \breve{MR}_i \geq \epsilon$, where ϵ is the coverage target. When true, this satisfies the first termination case, otherwise the process continues.

Since each newly-discovered feature may be a critical subpath, for evaluating the second termination condition, I adopt a conservative approach and assess changes in the cumulative observation set to determine *feature observation novelty*, $o(i)$, at step i:

$$o(i) = \begin{cases} 1 & \text{when } i = 1 \text{ or when } \breve{\mathcal{F}}_{obs(i)} \neq \breve{\mathcal{F}}_{obs(i-1)} \\ 0 & \text{otherwise} \end{cases} \tag{5.8}$$

which enables us to compute a *discovery efficiency*, $d(i)$, at step i:

$$de(i) = \frac{\sum_{x=1}^{i-1} o(x)}{i-1}, \tag{5.9}$$

which can also be treated as the likelihood of discovering a new feature at step i based on the number of observations up to that point, assuming a uniform distribution of observations. Unfortunately, however, feature observations are not uniform – I discover many new features in the first few simulations but in later simulations, the likelihood of observing new features drops significantly. In fact, the discovery efficiency $de(i)$ over the ensemble is well-described by a double exponential decay function (Fig. 5.3):

$$\widetilde{de}(i) = ae^{-bi} + ce^{-di}, \tag{5.10}$$

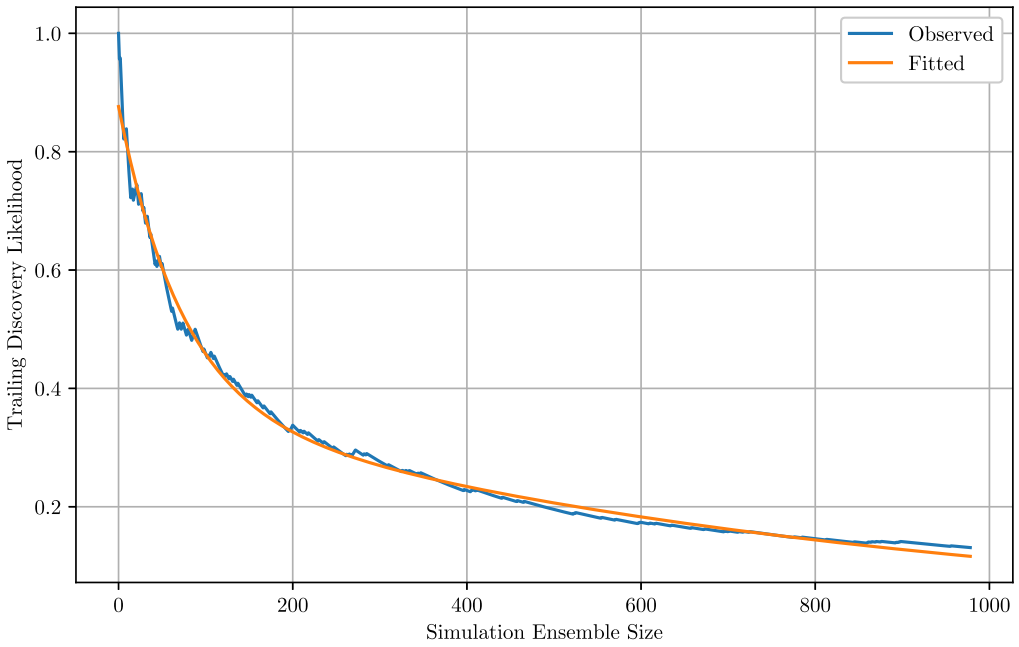


Figure 5.3: Comparing observations against analytic predictions for fitted double-exponential decay.

where a and c are the initial values of the decaying exponentials, and b and d control the decay rates[11]. Thus, instead of relying on the discovery efficiency as defined in Equation 5.9, I rely on a modified definition

$$de^*(i) = \frac{\sum_{x=j}^{i-1} o(x)}{i - j - 1}, \tag{5.11}$$

where j is an elbow position on the curve, identified using the Kneedle approach [127], based on the current ensemble[12]. Since the discovery efficiency flattens beyond the elbow position, this leads to a better estimation of the observation likelihood of new features in the (near) future.

[11]To discover these values, I fit the observation curve using least-squares minimization [111].
[12]As the ensemble grows, the elbow position is regularly updated.

I define a *progress efficiency* in terms of the number of future simulation steps $w \in \mathbb{Z}^+$:

$$pe(w) = 1 - (1 - de^*(i))^w \tag{5.12}$$

where $de^*(i)$ is the current feature discovery efficiency at step i.

From this progress efficiency, I can now compute the smallest stride $w' \in \mathbb{Z}^+$ within which DART should observe at least one novel feature with likelihood greater than the progress target θ:

$$\text{argmin}_{w'} \{pe(w') \,|\, pe(w') \leq \theta\} \tag{5.13}$$

The user can optionally bound the stride size w within a range of integers $w \in \langle \lfloor w \rfloor, \lceil w \rceil \rangle \in \mathbb{Z}^+$. This can be useful from a computational practicality standpoint, such as to avoid overly-long periods between stride re-evaluation, but is not mathematically required. When bounded in this way, the optimization function changes to evaluate only the selected range:

$$\text{argmin}_{w'=\lfloor w \rfloor}^{\lceil w \rceil} \{pe(w') \,|\, pe(w') \leq \theta\} \tag{5.14}$$

When the optimization function cannot be satisfied or is infeasible, then the simulation ensemble will fail to make meaningful progress. In other words, the marginal likelihood of observing a single new feature in the upcoming w' simulations must be below the progress target, and since the ensemble's efficiency in terms of producing new features is now too low, the ensemble terminates.

5.7 Evaluation

In this section, I evaluate the proposed DART approach to assessing the various features of a network for resiliency against attacks. All results were generated using

the DART tool, leveraging Python 3.11 with supporting libraries[13]. Experiments were conducted on a cluster provided by [73] equipped with AMD EPYC 7763 CPUs and 256GB RAM.

5.7.1 Baselines

In this section, I compare DART against alternative graph-theoretical approaches, with *betweenness centrality* and *personalized page rank*-based ranking of the edges. Note that these methods focus on single edges and therefore cannot capture multi-step structures; I argue that this capability remains one of the advantages of DART. When comparing against PPR-A, PPR-D, and EBC approaches, I fix $k = 2$ since these methods do not support longer features.

Edge Betweenness Centrality (EBC)

The first baseline leverages the concept of edge betweenness centrality [25] (EBC) to identify edges critical to data flow. This measure captures the proportion of all-pairs shortest paths that contain a given edge. Edges with high EBC therefore appear more often in point-to-point communications between distant nodes. If effective attacks most often make use of shortest paths, I would expect mitigations suggested by this measure to improve security.

PPR Difference (PPR-D)

DART uses PPR [114] as an intermediate measure when computing node impact, a fundamental building block of my proposed approach. Here, I use the PPR scores

[13][14, 58, 59, 68, 86, 99, 100, 116, 127, 140]

of each node directly as a baseline. In particular, I identify edges that go from low-scoring to high-scoring nodes and rank them by difference. If PPR alone sufficiently captures the flow of information from less-connected (or credential-holding) nodes to more central ones, then mitigations using PPR-D should offer some protection to the network.

PPR Average (PPR-A)

As an alternative PPR-based method, I rank edges by the average PPR scores of the endpoints. Intuitively, this approach identifies edges connecting highly central nodes which may act as critical routing backbones.

Inverse DART (INV)

As a sanity check, I also consider an approach where I select not the highest DART scoring feature, but the *lowest* scoring feature. If DART is indeed effective in differentiating between critical and non-critical network features, relying on the features selected by INV should lead to *worse* performance among all baselines.

5.7.2 Data Sets

In this section, I consider both synthetic and real graphs:

Synthetic Graphs

Since adversarial strategies and their effectiveness may vary by topology, I evaluate against five families of randomly-generated graphs: Barabasi-Albert [11], Erdős-Renyi [38], Newman-Watts-Strogatz [110], power law [64], and scale-free [23]. I further

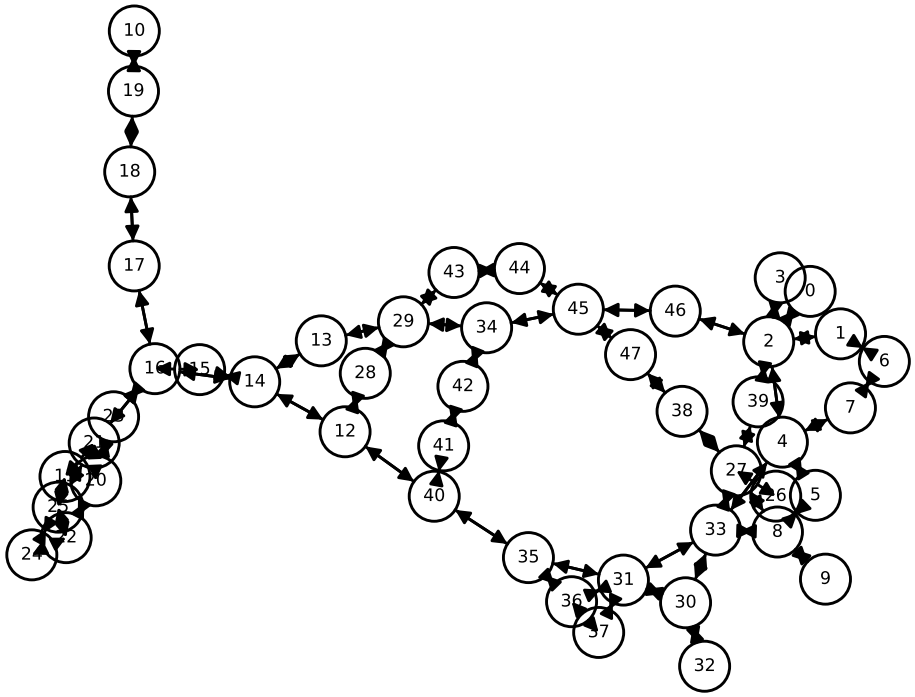


Figure 5.4: Bell Canada, an international telecommunications network.

elaborate on the generation process, similarities and differences between these families in Section 5.7.4. To ensure statistical soundness, I generate 200 topologies for each family. Each contains 10 nodes; the number of edges varies by family between 21 and 65, with a mean of 33. All graphs contain single, disjoint source and goal node pairs and one inserted credential edge between a randomly-selected pair of nodes.

Real-World Graphs

To provide a more realistic assessment of DART's performance, I additionally evaluate against several real network topologies [77]. Similar to the synthetic topologies, and in the absence of DART-specific metadata in the dataset, I generate 50 permutations

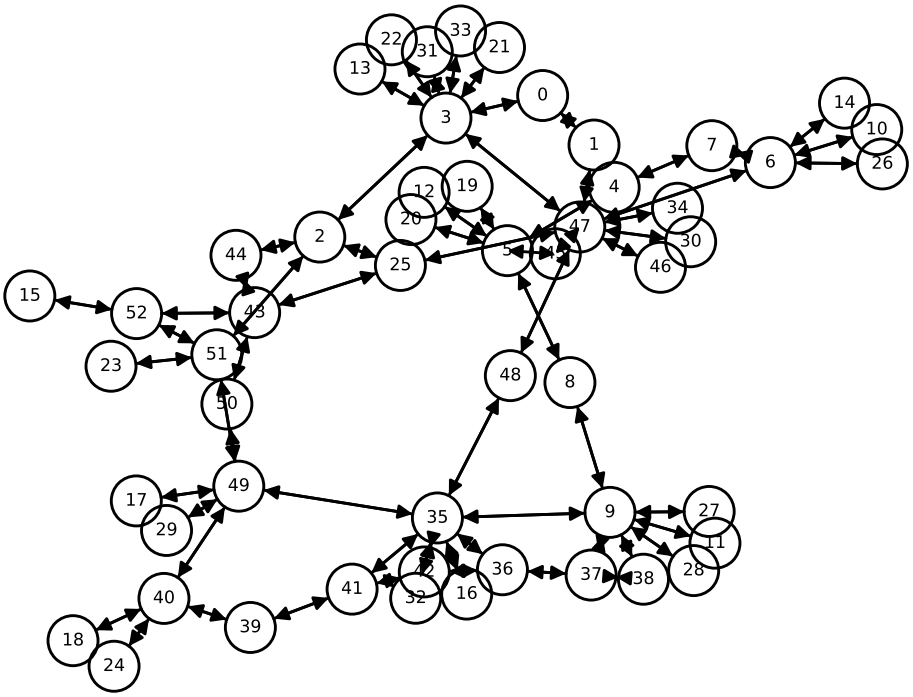


Figure 5.5: BtN, an international telecommunications network.

of each topology by choosing at random a single, disjoint node pair to act as source and goal node. For each permutation, I also insert a two-hop, three-node credential chain into the network at random.

Bell Canada. This topology (48 nodes:128 edges) describes the corporate network for various sites across Canada (Fig. 5.4).

BtN. This topology (53:130) represents telecommunication infrastructure connecting U.S. and global sites (Fig. 5.5).

CESNET. This topology (52:126) encodes the national infrastructure network for Czechia used for education and research (Fig. 5.6).

GARR. This topology (61:150) describes another national research and education

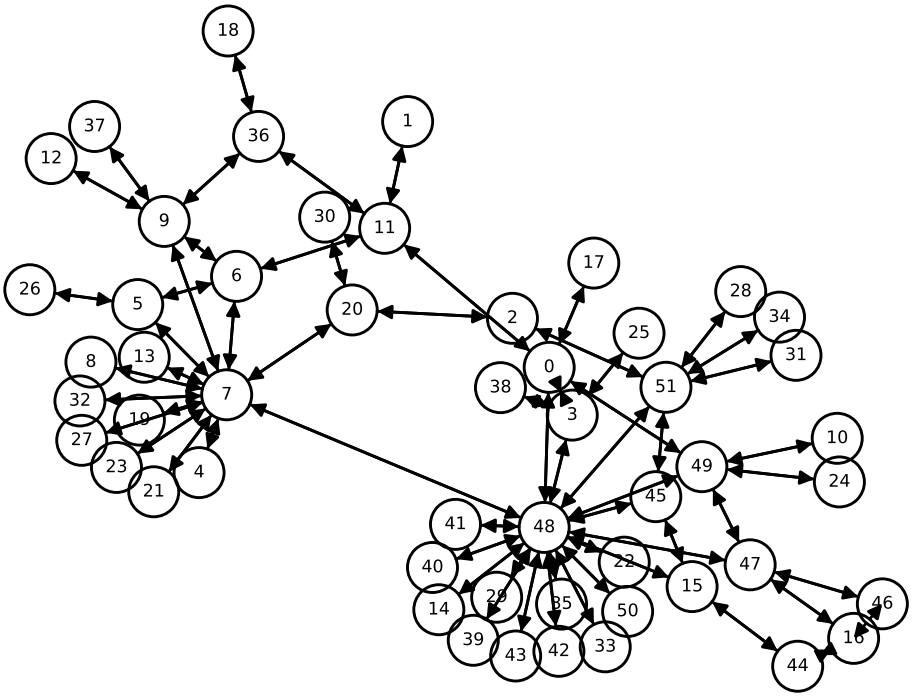


Figure 5.6: CESNET, the national research network of Czechia.

network in Italy (Fig. 5.7).

TINET. This topology (53:178) represents the global IP backhaul network provided by the now-defunct TINET company (Fig. 5.8).

5.7.3 Experiment Parameters

The number of simulations per topology varies according to my budgeting algorithm (§5.6). For the underlying PPR computations, I set $\alpha = 0.85$, a commonly-accepted default [21]. For the synthetic results, I let $\ell = 5$, $\Lambda = |\mathcal{E}_C \cap \mathcal{E}_T|$, $m = 1000000$, $\epsilon = 0.99$, and $\theta = 0.99$; real-world topologies use $\ell = 25$ instead. My sampler has an initial budget $b_1 = 1000$ and a stride of 5000.

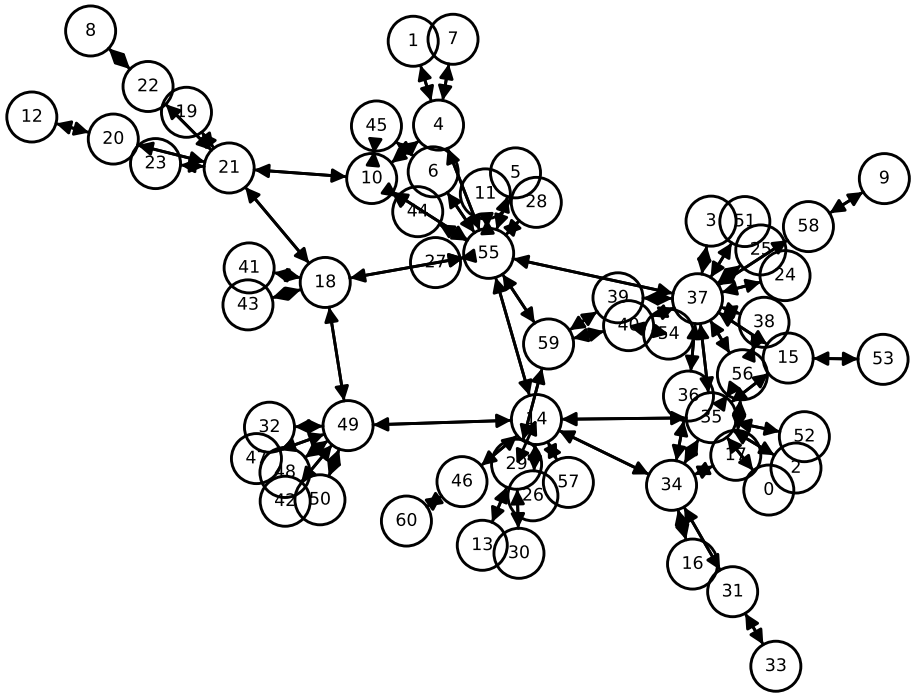


Figure 5.7: GARR, the national research network of Italy.

5.7.4 Random Graph Generation

To evaluate the proposed approach, generating random graphs eliminates the possibility of 'cherry picking' topologies that perform well, and instead evaluating the average performance across topologies. However, for comparisons within the topology population to be meaningful, they should share characteristic traits determined during the generation process. As a consequence, the process of stochastically generating graphs opens up significant discussion surrounding both the specific methods chosen and their resulting outputs [36].

In practice, random graph generation that targets realistic output often produces results whose degree distribution follows a power law – that is, most nodes connect to

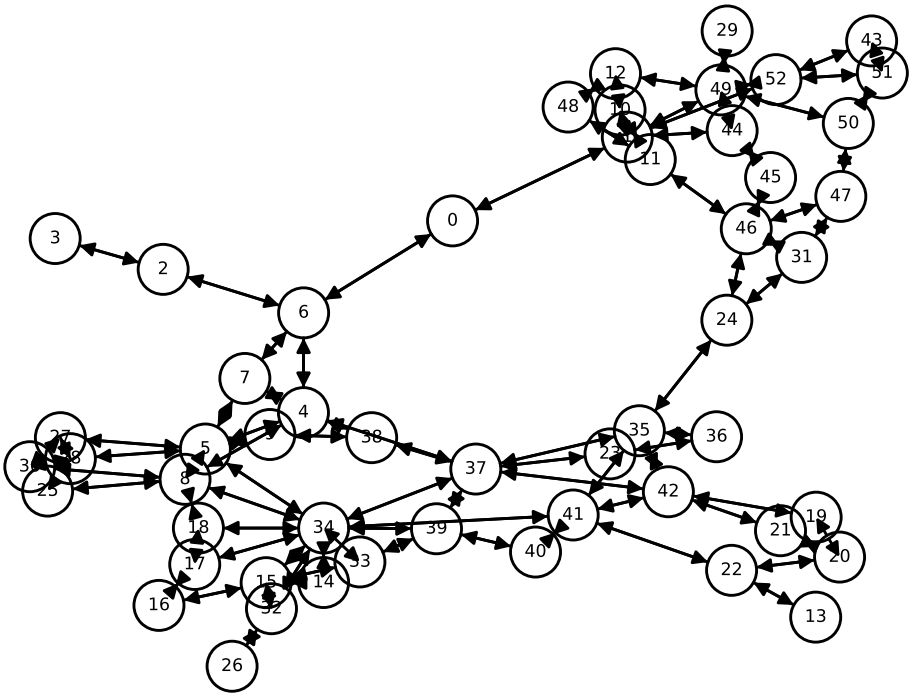


Figure 5.8: TINET, an international telecommunications network.

few others while a minority of central nodes act as key bridges across the network. This pattern appears in areas across science, from human social interactions [105] to natural plant structures [135]. The degree to which such networks are (dis-) similar relies on the mechanism of action and parametric configurations of the generative approach used.

In Section 5.7.2, I introduce several synthetic graph families that exhibit potentially-similar structures, in that a small world graph is a type of power law graph, for example. Thus, to avoid confusion, I will first elaborate on the generation mechanisms that differentiate the described families. In all cases, different properties can be said to act as a *goal* or an *outcome*. In the former case, the characteristic is a design goal

of the generation mechanism and is therefore used to guide the generation process directly. In the latter case, the property may emerge naturally from the method used to generate stochastic connectivity, which considers an alternative mechanism as the driving function.

Note that the key differentiating factor between different families can therefore be described by the respective sets of goals and outcomes. I will now discuss several of these characteristics, and how they relate to the generation process.

Preferential Attachment

The concept of *preferential attachment* refers to the idea that newly-added nodes will prefer to connect with nodes that already have many connections [109]. This approach is used as one of the design goals in Barabasi-Albert, power law, and scale-free graph generation.

Growth

The concept of growth refers to the addition of nodes to the network slowly over time. Conversely, in random graphs with a fixed number of nodes in which only the nodes are added stochastically do not reflect the concept of growth. Barabasi-Albert, power law, and scale-free graphs all model growth, while Erdős-Renyi and Newman-Watts-Strogatz graphs start with a static number of nodes.

Random Edge Addition

When adding an edge between two nodes is independent of the nodes' degree or other edges, then it does not necessarily follow a distribution (e.g. power law). While

these graphs are quite stochastic, they may not be as effective and representing real-world scenarios as accurately. This random approach is used by Erdős-Renyi and Newman-Watts-Strogatz graphs.

Triadic Closure

Triadic closure aims to influence not only the degree distribution of the network, but the clustering coefficient as well. Here, when node A is connected to both node B and node C, there is a correspondingly high likelihood that node B is also connected to C. Only power law graphs (as described here) use this as a generative mechanism.

Rewiring

Similar to triadic closure, rewiring attempts to influence the clustering coefficient of the resulting graph, but does so in a different way. Instead, edges are directly moved or added to change the clustering coefficient of the graph directly; the only generation method that uses this mechanism is the Newman-Watts-Strogatz graph.

Node Fitness

In some cases, it is desirable to enforce heterogeneity in the network structure by directly evaluating the fitness of nodes according to a heuristic value, which is then used to adjust how connections are formed. This approach is used by the scale-free approach, which takes additional pains to enforce heterogeneity on the network to ensure that the resulting graph is truly scale-free.

Table 5.2: Synthetic Generation Details

Family	Approach	NetworkX Function	Parameters
Erdős-Renyi	[12]	`fast_gnp_random_graph`	$n = 10, p = 0.5$
Barabasi-Albert	[11]	`barabasi_albert_graph`	$n = 10, m = 2$
Newman-Watts-Strogatz	[110]	`newman_watts_strogatz_graph`	$n = 10, k = 2, p = 0.5$
Power Law	[64]	`powerlaw_cluster_graph`	$n = 10, m = 2, p = 0.5$
Scale-Free	[23]	`scale_free_graph`	$n = 10, \alpha = 0.41, \beta = 0.54, \gamma = 0.05, \delta_{in} = 0.2, \delta_{out} = 0$

Method Summary

To summarize the differences in these graphs, I focus less on the resulting configurations and more on the mechanisms by which these graphs emerged. For example, power law graphs and Newman-Watts-Strogatz graphs both exhibit high clustering coefficients, but approach that goal in a different way. Similarly, all graphs except Erdős-Renyi exhibit low average path length due to their mechanism of construction. Finally, I note that the Barabasi-Albert, power law, and scale-free graphs all follow a power law distribution in terms of degree, but reach that goal in different ways.

I have detailed the generation details, as well as the parametric configurations, in Table 5.2. When choosing parameters I used default parameters when the method specified them, and attempted to maintain consistent semantics (e.g. similar probabilities and degree constraints) otherwise.

5.7.5 Results on Synthetic Topologies

Attack Success Ratio

Recall that attack success ratio encodes the percent of attackers able to capture all goal nodes. Each strategy evaluates the baseline topology, identifies and knocks out

the edge determined to be the most critical to attack, and then re-evaluates adversarial success in the new environment. I restrict edge knockout to the top-ranked edge which would not partition the network, to avoid disrupting functional requirements. I also include the *inverse* ranking, INV, which chooses the *worst*-ranked edge to knock out. Doing so illustrates that identifying and removing edges that hurt the attacker can provide a detrimental effect, making attacks succeed more often.

In Fig. 5.9, I see the efficacy of different intervention strategies in preventing adversarial success. I observe that in all cases, DART reduces the success ratio below the baseline and does so more effectively than any other method. Interestingly, though PPR acts as the foundational metric for DART, usage by PPR-A and PPR-D meets with limited success, coming close only in the Erdős-Renyi family. This emphasizes the importance of the additional information gained via simulation relative to purely graph-theoretic measures. I also see that, as expected, the INV approach provides a modest but consistent boost for the attacker's success.

Note that in some cases, it may be desirable to maximize defense, even by disconnecting the graph. Due to the nature of its construction, the power of disconnection is most pronounced in the scale-free case, which has an average edge connectivity [39] of only 1.28. Thus, removal of a single edge will likely disconnect the graph. Fig. 5.9(f) shows that, absent the connectedness constraint, much greater improvements are possible. No substantial difference was observed for the other topological families.

Attack Exploitation Cost

These results indicate that DART can effectively reduce the likelihood of success of an attacker. However, not all attacks were created equal – the number of exploits

required to succeed captures an important cost factor to the attacker. I record this as the mean number of exploits used across all simulations for a specific topology, which allows for fractional exploitation costs. Note that the cost cannot fall below two in these scenarios, since the attacker uses one to enter the network and at least one more to capture the goal node.

In Fig. 5.10, I see that the cost associated with each attack increases after DART intervention. In particular, I see that even when an attack cannot be completely prevented, the number of exploits required still grows. Given the development costs [1] and relative risk [9] of deploying exploits, increasing these costs provides a second penalty to the attacker. That is, DART causes an attacker to succeed less often, and to pay a higher price when it does succeed. Note that, similar to the attack success results, ignoring the connectedness constraint (Fig. 5.10(f)) forces even higher costs on the attacker. Additionally, INV aids the attacker by reducing attack costs below the baseline, and in some cases down to the minimum possible.

5.7.6 Results on Real-World Topologies

Interestingly, while the overall pattern of the results remains, the real-world datasets exhibit greater variability between topologies. While graph-specific differences across synthetic networks may be obscured by the population's structural variability, the real datasets only differ in source and goal nodes and therefore show more consistent behavior. This suggests that the underlying structure of a network does indeed impact security and that discovering such structures offers promising defensive avenues.

Attack Success Ratio

Examining the results, I see that in the Bell Canada topology, DART significantly outperforms PPR-A, while in BtN both methods provide a sizeable but similar improvement. In contrast, in the GARR topology, the interventions remain relatively close to the baseline in comparison to the other graphs, suggesting an architecture that is harder to improve. In all cases, DART illustrates both a defensive improvement versus the baseline, as well as the largest improvement among the interventions. In the disconnected case for CESNET (Fig. 5.11(f)), the defensive improvements by DART become even more pronounced.

Attack Exploitation Cost

Similar to the patterns seen in the synthetic results, I see consistently higher costs to the attacker by DART across all topologies. The disconnected CESNET costs in Fig. 5.12(f) show that DART is again most effective when unconstrained. However, I also see that defensive improvements shown by PPR-A sometimes *reduce* attacker costs. This can occur when interventions prevent the adversary from making inefficient choices.

To understand this tradeoff, Fig. 5.13 plots attack success ratio against exploitation cost, showing that DART optimizes both measures more effectively than other interventions. In short, attacks succeed less often, and when they do succeed, they require more exploits from the attacker.

5.7.7 Ablation and Sensitivity Studies

To assess the important and impact of different components of the DART approach, I provide a more detailed breakdown along alternate axes of evaluation.

Estimated $\widetilde{\mathcal{F}}_{poss}$ vs. True $\mathcal{F}_{poss}$

To evaluate the correctness of my sampling-based approach for establishing baselines, I compare against a naive, exhaustive traversal of the graph. For each graph, I compare the ground truth features against the sampled features and report the mean and standard deviation across all 50 permutations for precision, recall, and F-1 score in Table 5.3. These results use $k = 3$ and $\ell = 25$ as a conservative scenario; higher values would penalize the sampler due to the larger sample space, but also pose tractability challenges for the naive approach. First, I observe that for every graph, precision is always exactly 1 – this means that the sampler never observes features that cannot be observed. This is critical, because it would set an artificially-high bar that the simulation ensemble could not possibly reach. I do see some drop in recall, which is expected for a sampling-based approach, but the F-1 scores across all topologies remains consistently high. The lowest mean F-1, for GARR, was still 0.9786; this suggests that my approximation comes extremely close to the true value.

Note that TINET does not appear in these results. This is due to the fact that the topology of TINET cannot be tractably enumerated using the exhaustive method, even for the same k and ℓ values. Indeed, it is not always possible to preemptively predict which topologies will be tractable, due to depending on many possible factors of the simulation context. However, the sampling-based approach does not encounter such challenges and performs consistently well against all tested topologies.

	Bell Canada			BtN			CESNET			GARR		
	Precision	Recall	F-1	Precision	Recall	F-1	Precision	Recall	F-1	Precision	Recall	F-1
μ	1.0	0.9655	0.9823	1.0	0.9689	0.9837	1.0	0.9758	0.9872	1.0	0.9616	0.9786
σ	0.0	0.0232	0.0121	0.0	0.0418	0.0228	0.0	0.0448	0.0237	0.0	0.0794	0.0469
Med.	1.0	0.9662	0.9828	1.0	0.9727	0.9862	1.0	1.0	1.0	1.0	1.0	1.0

Table 5.3: Comparison Between $\widetilde{\mathcal{F}}_{poss}$ vs. True $\mathcal{F}_{poss}$

Varying Coverage

Intuitively, coverage allows users to bound the amount of effort expended for a given topology. Most attack features can be easily discovered, but a long tail of hard-to-discover features will remain. However, choosing a lower ϵ permits users to analyze the "low-hanging fruit" of the more common features at a relatively low cost. As seen in Table 5.4, I see this pattern clearly as total simulation costs rise 54-236% over the lower threshold.

The disparity observed across topologies highlights the concept that the difficulty of feature discovery is nonuniform and dependent on graph details. For example, both CESNET and GARR achieve 90% coverage essentially using only the initial budget, suggesting that many features can be easily reached by adversaries. Bell Canada required the highest budget at $\epsilon = 0.9$, while TINET was the highest at $\epsilon = 0.99$. This suggests that Bell Canada has a low proportion of easily reachable features, and that more of its features are difficult to discover. In comparison, TINET has a higher proportion of accessible features than Bell Canada, but the higher cost at higher ϵ indicates that TINET also contains a few features that are exceeding difficult to discover.

	Bell Canada		BtN		CESNET		GARR		TINET	
	$\epsilon = 0.9$	$\epsilon = 0.99$	$\epsilon = 0.9$	$\epsilon = 0.99$	$\epsilon = 0.9$	$\epsilon = 0.99$	$\epsilon = 0.9$	$\epsilon = 0.99$	$\epsilon = 0.9$	$\epsilon = 0.99$
μ	1099.9	1698.16	315.08	1058.06	200	595.52	202	542.8	722.9	1955.18
σ	805.284	610.331	349.688	561.855	0	444.52	14.142	418.369	863.484	507.589
Med.	1138.5	1646	200	1119	200	496	200	336	300	1910
Max	2865	3130	1964	2333	200	1906	300	1811	3793	3793
Min	200	300	200	200	200	200	200	200	200	300
Total	54995	84908	15754	52903	10000	29776	10100	27140	36145	97759

Table 5.4: Comparison, ϵ Coverage Targets by Ensemble Size; $b_1 = 200, \theta = 0.99$

Varying Threshold

When evaluating threshold, it is important to note that the scaling of the threshold proceeds inversely to most thresholds. That is, higher numbers are *less* restrictive, not more. Put more simply, choosing $\theta = 0.99$ indicates that execution should continue when there is a 99% chance of observing a new feature in the next w executions, while choosing $\theta = 0.99999$ asks for a much higher likelihood of observing a new feature. Though the likelihood of observing a new feature decreases exponentially in w, and the difficulty of each individual observation follows a double-exponential decay (Figure 5.3), the elbow at j tends to stabilize, bounding the difficulty by stabilizing $de^*(i)$. Thus, I expect Equation 5.12 will fall relatively slowly for fixed w.

Looking at the results in Table 5.5, I see that as the threshold falls, the effect on total simulation costs proves dramatic. Again, I consider Bell Canada as an exemplar, where I see the lower threshold takes 351% of the higher. This suggests that Bell Canada has a long tail of features, which aligns with my conclusions about ϵ – the higher proportion of hard-to-discover features leads to a more slowly-falling threshold. Though TINET requires more in absolute numbers, the increase between thresholds is less than half that of Bell Canada.

	Bell Canada		BtN		CESNET		GARR		TINET	
	$\theta = 0.99999$	$\theta = 0.99$	$\theta = 0.99999$	$\theta = 0.99$	$\theta = 0.99999$	$\theta = 0.99$	$\theta = 0.99999$	$\theta = 0.99$	$\theta = 0.99999$	$\theta = 0.99$
μ	483.36	1698.16	356.74	1058.06	227.62	595.52	252.14	542.8	1187.46	1955.18
σ	533.9781	610.3308	321.8889	561.855	118.5112	444.52	105.0487	418.3687	549.0446	507.5888
Med.	200	1646	200	1119	200	496	200	336	1332.5	1910
Max	2304	3130	1583	2333	1013	1906	641	1811	2184	3793
Min	200	300	200	200	200	200	200	200	200	300
Total	24168	84908	17837	52903	11381	29776	12607	27140	59373	97759

Table 5.5: Comparison, θ Coverage Targets by Ensemble Size; $\epsilon = 0.99, b_1 = 200$

Impact of b_1 on Budget Estimation

Before execution DART, users must choose an initial simulation budget b_1 that builds the context to establish termination conditions. In Table 5.6, I see the effects of this choice. In most cases, I see that choosing a lower initial budget typically plays only a marginal role in the overall simulation cost. For example, in the TINET topology, the average number of simulations across all variations differed by less than 2% even when increasing the baseline size. This suggests that the dynamic approximation can effectively find the correct equilibrium point even when the initial budget is quite small.

However, the trend is not universal – I can see that choosing a lower value does provide a modest savings for the Bell Canada case. Looking at the scores, this highlights a pattern shared across all examples of a higher standard deviation resulting from outlier cases in which only the minimum, initial number of cases were required. This can be caused by certain permutations being relatively simplistic. For example, if there are few choices for an adversary between a source and target along a single path, that specific permutation might satisfy its coverage requirement in only the initial budget.

	Bell Canada		BtN		CESNET		GARR		TINET	
	$b_1 = 20$	$b_1 = 200$	$b_1 = 20$	$b_1 = 200$	$b_1 = 20$	$b_1 = 200$	$b_1 = 20$	$b_1 = 200$	$b_1 = 20$	$b_1 = 200$
μ	1526.9	1698.16	997.96	1058.06	604.72	595.52	535.96	542.8	1988.28	1955.18
σ	765.4255	610.3308	616.735	561.855	479.0733	444.52	424.9078	418.3687	540.3945	507.5888
Med.	1578.5	1646	983.5	1119	432.5	496	360.5	336	1944.5	1910
Max	2813	3130	2302	2333	1960	1906	1645	1811	3795	3793
Min	20	300	20	200	20	200	120	200	320	300
Total	76345	84908	49898	52903	30236	29776	26798	27140	99414	97759

Table 5.6: Comparison, b_1 Coverage Targets by Ensemble Size; $\theta = 0.99, \epsilon = 0.99$

Varying Attack Budget (ℓ)

For all the real-world topologies evaluated, I chose $\ell = 25$. To determine whether and how this choice influences the results, I fix the other variables and investigate attack budget. Naturally, as the attack budget rises, the observed exploitation cost rises (Figure 5.15), because one controls the other. However, note that the changes in observed behavior result simply in spreading the distribution across the Y-axis. That is, the patterns persist: TINET consistently has lower success, with GARR showing the reverse. Since DART relies on the relative value of attacks by normalizing across the number of attacks, the choice of ℓ in evaluating exploitation cost therefore makes little difference.

In terms of success ratio, the influence is more obvious. Since a larger budget necessarily implies an attacker that can spread further through the network, it is not surprising that success ratio rises along with ℓ. This is best illustrated in the Bell Canada topology. In the $\ell = 20$ case, the success ratio remains somewhat low before a sharp increase around two-thirds of the way through. When increasin to $\ell = 35$, that increase remains but is increased from the high-70s to the low-90s. However, the preceding variations are affected more significantly, and pass the fiftieth percentile for

more than 90% of the variations.

This suggests that choosing an appropriate ℓ is important. If too high, most attacks will succeed and discrimination will be difficult, and the features identified might be relevant only to a highly-resourced attacker. That is, DART will focus its attention on high-value features with low practical utility, looking for worst-case scenarios. Conversely, choosing a too-low value might never allow the attacker to succeed, again complicating discrimination. This second-order effect of adversarial behavior serves as an interesting starting point for future work in predicting the required budget for a successful attack.

5.8 Summary

In this chapter I present DART, a comprehensive tool that leverages topological information to improve network security. Rather than preventing or detecting attacks, I identify structural weaknesses in the underlying deployment, rank their relative importance to an adversary during an attack, and report that information to operators for remediation. These recommendations can be incorporated at design time, prior to deployment, or assessed post-hoc to enable A-B impact assessments of potential changes. I evaluated my system against synthetic and real-world topologies before and after the recommended interventions, and showed that DART's suggestions consistently reduced attack success and increased adversarial cost relative both to the baseline network as well as other graph-theoretic measures.

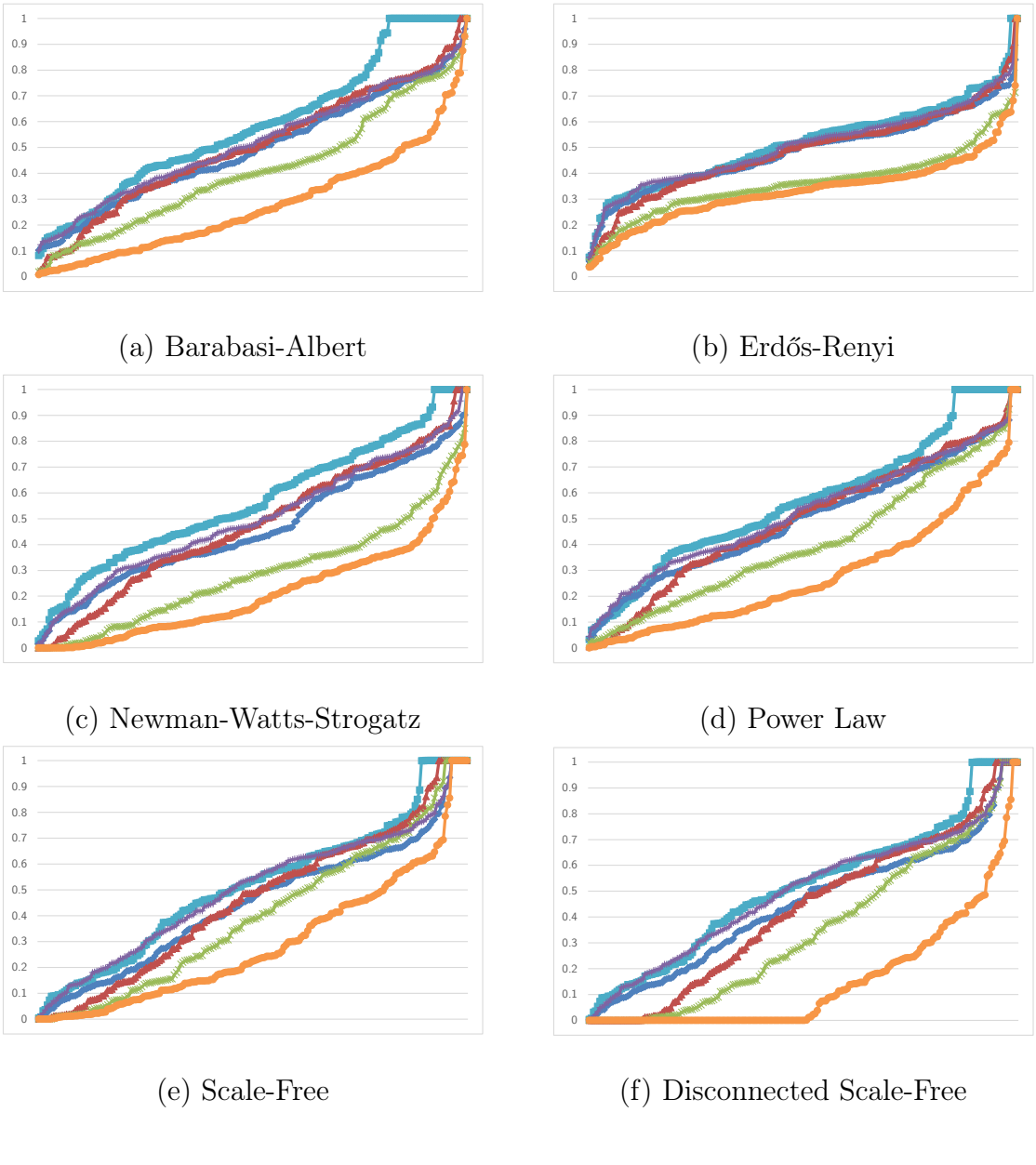
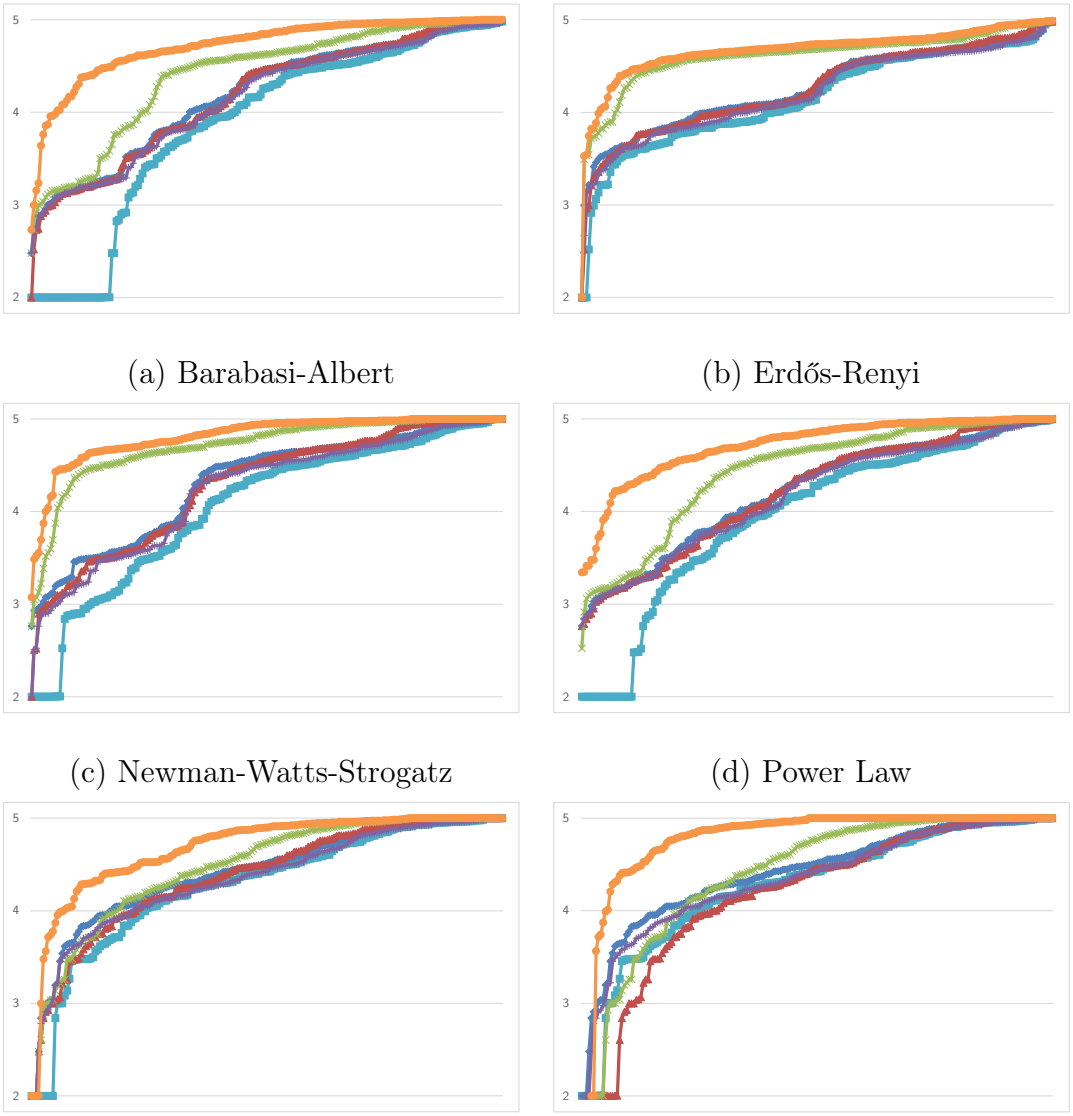


Figure 5.9: Attacker's success ratios for different synthetic topological families (the lower the better)



(a) Barabasi-Albert

(b) Erdős-Renyi

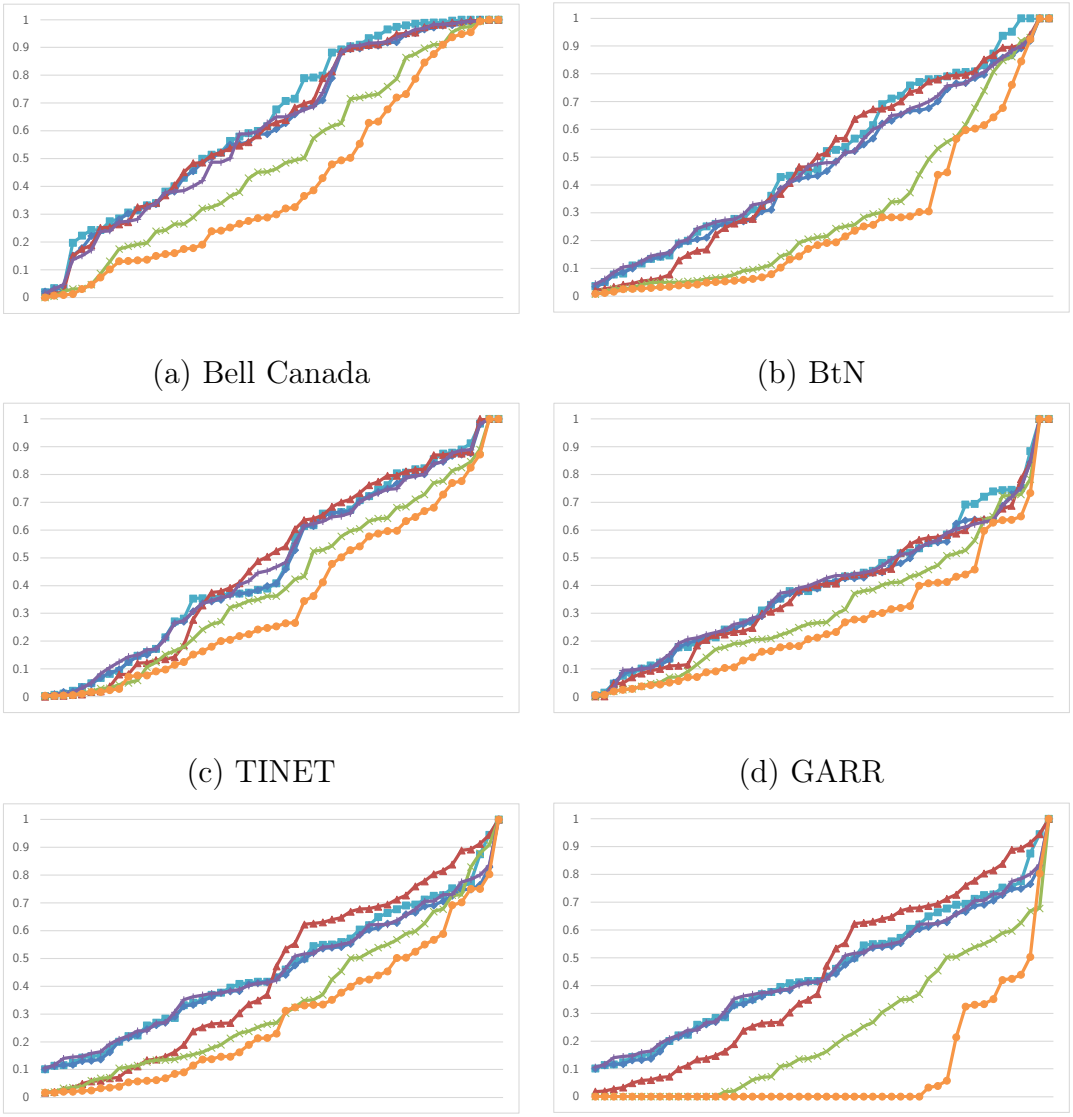
(c) Newman-Watts-Strogatz

(d) Power Law

(e) Scale-Free

(f) Disconnected Scale-Free

Figure 5.10: Attacker's exploitation costs for different synthetic topological families (the higher the better)

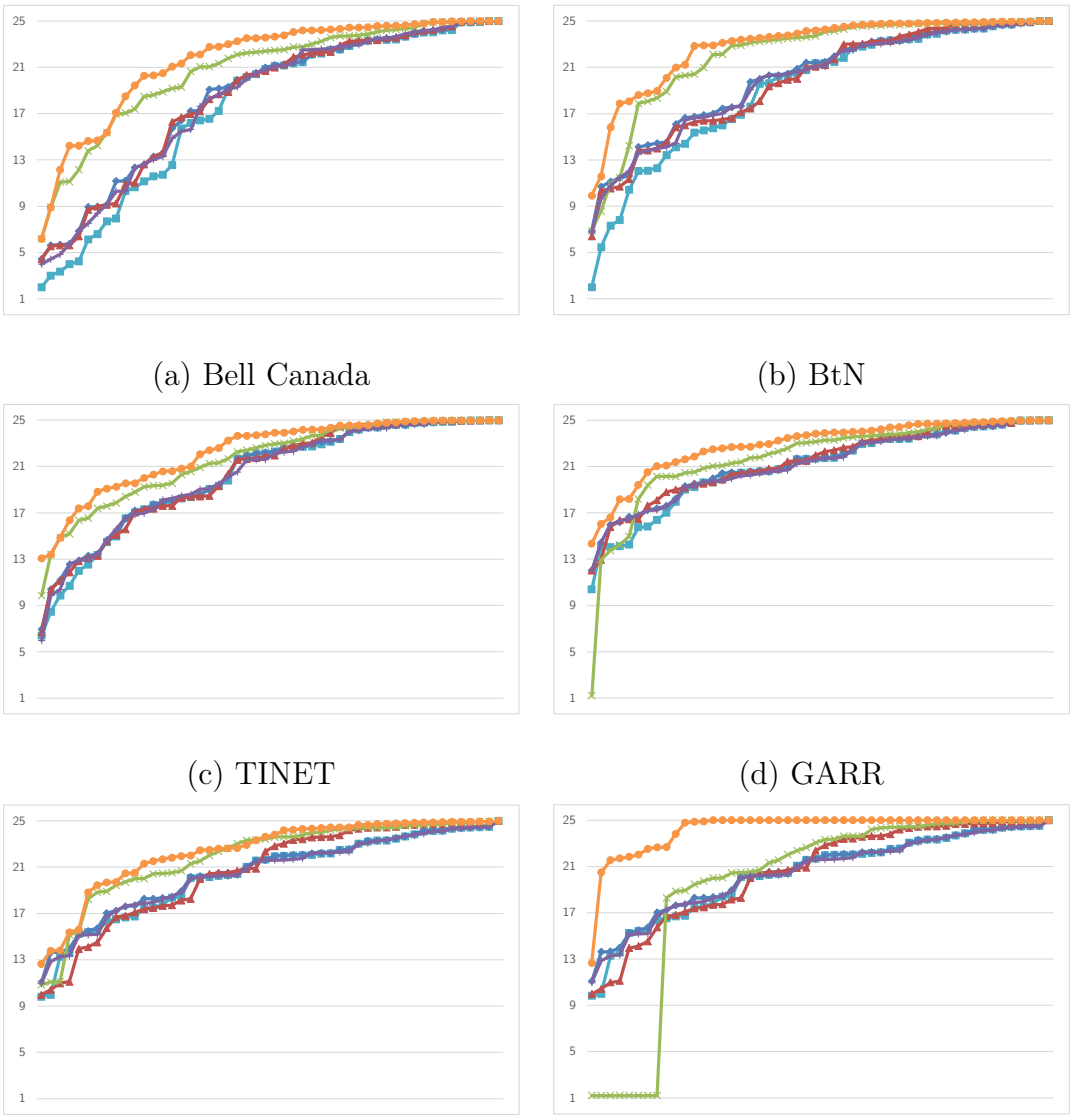


(a) Bell Canada (b) BtN

(c) TINET (d) GARR

(e) CESNET (f) Disconnected CESNET

Figure 5.11: Attacker's success ratios for different real topologies (the lower the better)



(a) Bell Canada (b) BtN

(c) TINET (d) GARR

(e) CESNET (f) Disconnected CESNET

Figure 5.12: Attacker's exploitation costs for different real topologies (the higher the better)

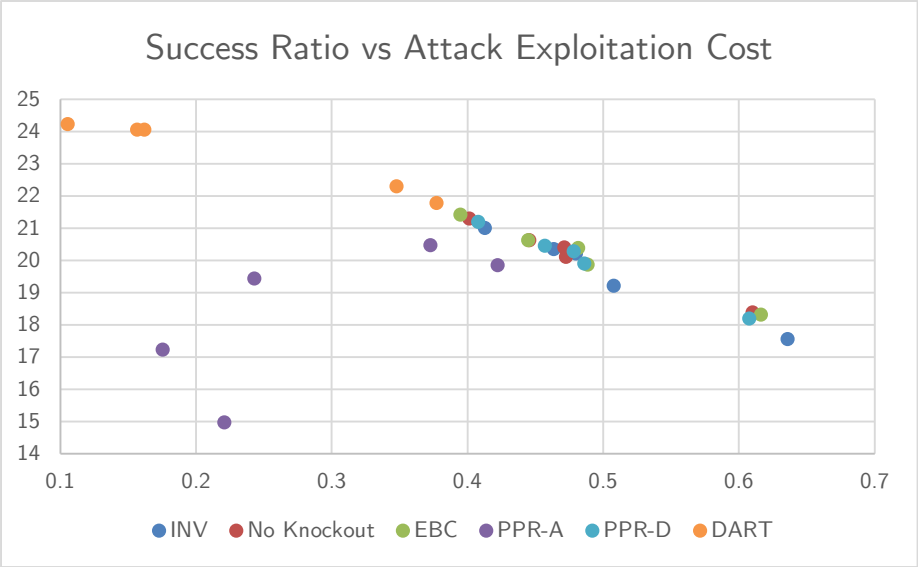


Figure 5.13: Success ratio vs exploitation cost – each dot corresponds to the performance of a given approach for a different real topology. (Top-left indicates attacker's highest cost and lowest success)

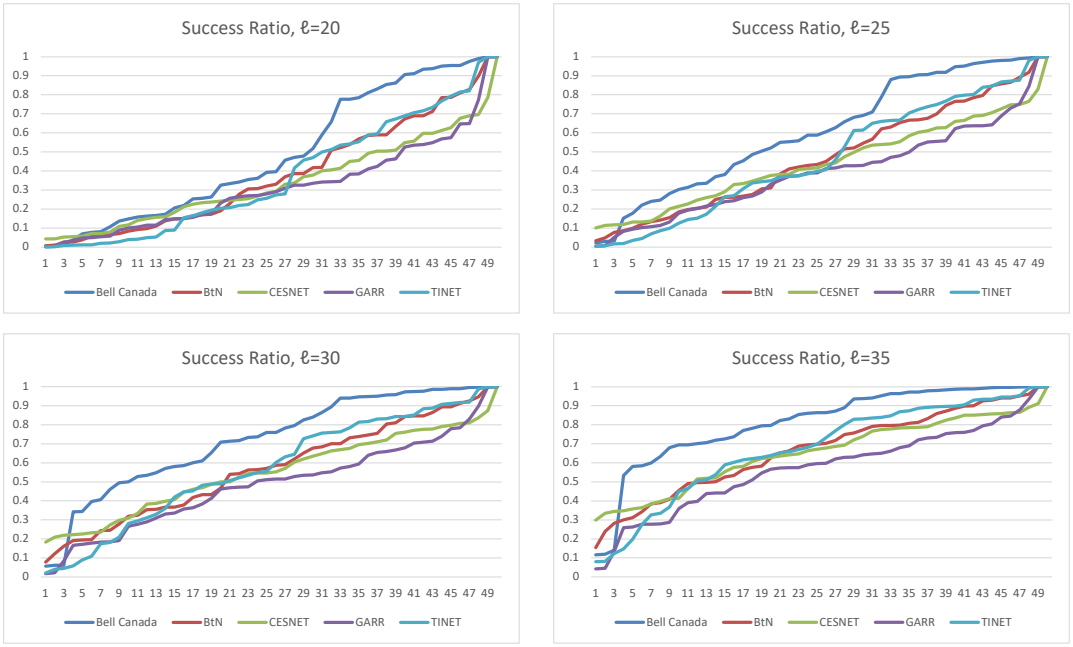


Figure 5.14: Attacker's success ratio for alternate values of ℓ

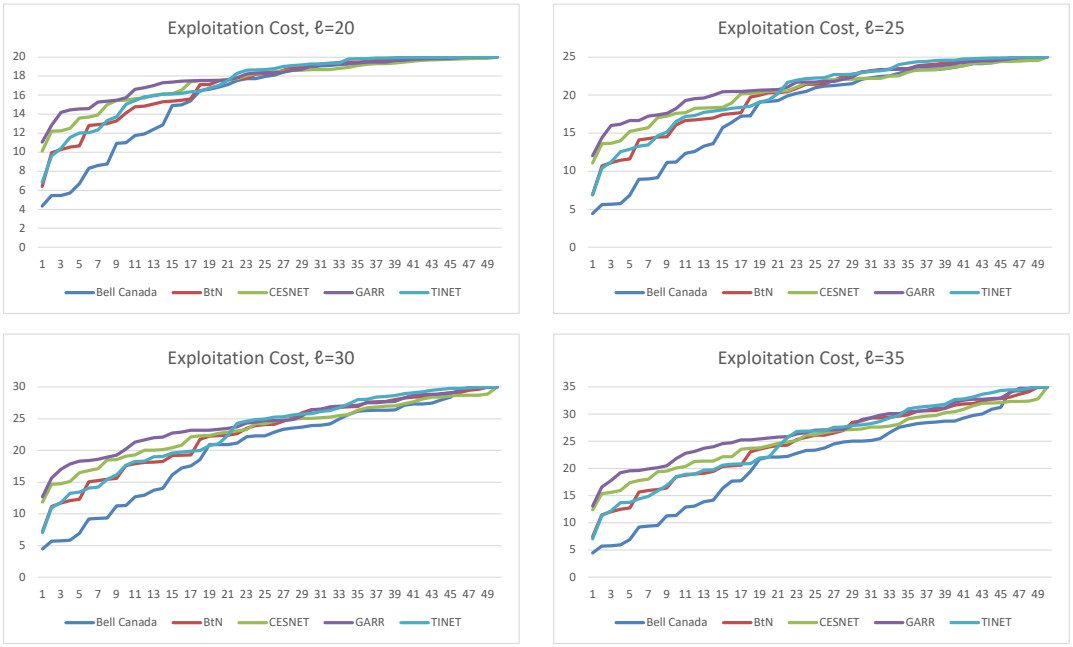


Figure 5.15: Attacker's exploit cost for alternate values of ℓ

Chapter 6

CONCLUSION

In this dissertation, we have explored a range of concepts and strategies for enhancing counter-adversarial resilience in permeable networks. It is crucial to recapitulate the core ideas covered throughout this work to emphasize their significance and the potential implications they have on network security.

The underlying assumption across all the discussed work is the concept of permeability. That is, I assume that an attacker will be able to capture at least some participants of the network under evaluation, whether they are low-powered sensor nodes deployed in remote areas or even an abstract concept of a node in the automated pattern recognition domain. This idea of permeability acts as a foundational thread, weaving these concepts together.

Moreover, all the cases presented aim to restrict the adversary's power, flexibility, or choices, which I refer to as "counter-adversarial" choices. I make this distinction because other work may focus on identifying, stopping, or removing attackers. In contrast, my work concentrates on methods that make the attacker's task more challenging, limit their successes, or exhaust their resources while still maintaining other network desiderata.

A key method employed in all the on-line approaches discussed is the homomorphic Merkle tree (HMT). This data structure leverages the homomorphic hash cryptographic primitive to allow for imbalanced subtrees within the overall Merkelized structure. The HMT enables different participants the flexibility to exchange information in a

semi-aggregated way.

In the following sections, I will summarize some of my findings, discuss the impact of these results, identify limitations and challenges common to these approaches, and conclude by outlining promising avenues for future work. Finally, I will reflect on the challenges of improving counter-adversarial resilience in the permeable network context.

6.1 Progress in Partitioning

Ad hoc communication mechanisms have thus far not gained critical momentum outside specific industrial and governmental applications. However, these trends seem to be changing. The rapid adoption of IoT devices has led to a variety of incompatible ad hoc protocols, resulting in ecosystem fragmentation. Industry groups have released standards, such as Matter [29], which present a unified ad hoc communication mechanism. However, this standard focuses primarily on interoperability and convenience, overlooking the danger of permeable networks.

By adopting semantic and spatial partitioning schemes, users could regain much more fine-grained control over their devices' connections, data sharing, and communication partners. Furthermore, the proposed resilience mechanisms could be interoperable with a common communication protocol, so their adoption would not necessarily require a forced choice between interoperability and security.

While IoT devices usually appear in relatively constrained areas and thus do not see as much benefit from spatial partitioning, mobile devices remain a crucial segment. Recent technologies like 5G contain the foundations for device-to-device (D2D) communication [137], but these techniques are still rare in practice. Ostensibly due to high power costs, lack of buy-in from entrenched network provides also acts as

a cooling effect on innovation in this direction.

Other approaches, such as the Helium Network [49], attempt to leverage blockchain technologies to secure and incentivize participation in the underlying ad hoc protocol. However, inefficiencies and costs introduced by the incentivization layer make this system prohibitively expensive at present.

6.2 Advances in Distributed Fusion

Although distributed sensor fusion has remained a relatively niche application, this arguably stems from a lack of imagination more than a lack of hardware. Indeed, centralized approaches that collect information from wide swathes of devices such as phones, cars, laptops, or even public areas are only increasing in prevalence. So why have distributed approaches remained elusive?

Mobile devices act as a rich source of sensor data for much of the world. However, privacy concerns around the collection and use of this data continue to grow [24]. Companies, meanwhile, face significant fines for misusing sensitive user data [62]. This has encouraged the rise of new directions of research, such as federated learning, which allow for analysis and processing of this data without directly accessing it.

Unlike federated learning approaches, which require a great deal of effort to preserve user privacy from the central data processor [103], HMT-based approaches such as Pando allow for the partitioning of data in such as a way that individual observations could be aggregated and anonymized at the edge, addressing these concerns without involving a centralized party until the last step of the process.

These techniques may become more important as cars and other vehicles are equipped with more comprehensive sensor suites. Self-driving cars and other "smart"

vehicles would benefit from sharing and comparing sensor data to improve predictive accuracy and safety for passengers. However, the costs of malicious behavior are correspondingly high – potentially life-and-death. Mitigating these risks may become a key role of distributed fusion approaches.

6.3 Insights in Pattern Analysis

A crucial challenge in offline network security analysis is the scarcity of realistic attack data for learning purposes. Conducting simulations of realistic attacks on networks is often impractical due to the high costs and replication difficulties associated with human-driven attacks, such as red team exercises. Indeed, several current approaches use relatively more-available legal information [51] rather than try to collect such data.

While recent trends in machine learning have focused on supervised and semi-supervised learning, the utilization of automated pattern analysis highlights the potential of unsupervised approaches and data science concepts. This relatively unexplored territory offers a wealth of opportunities for enhancing network security research and development. In particular, when human attackers know learning approaches will be used, their behavior changes to undermine its efficacy [5]. We must find ways to sidestep this challenge.

For example, the structural aspects of a network offer a concrete source of data irrespective of malicious or benevolent behavior. Since this information is relatively static due to the operational constraints of the network, it offers a contrast to learning-based approaches that rely on dynamically changing situations, which can be susceptible to adversarial deception if they have knowledge of the underlying network. Leveraging

the relative independence of network architectures suggests that further analysis could offer a rich avenue of discovery for permeable network research.

Insights garnered from automated pattern analysis suggest that further exploration of unsupervised learning methods and data science techniques can significantly contribute to network security advancements. By considering additional factors, such as the impact of time on attacks and defenses, researchers could gain a more comprehensive understanding of adversarial tactics and devise more effective countermeasures.

6.4 Impact of Proposed Strategies

I have introduced several concepts that could be adapted to enable or improve counter-adversarial resilience in permeable networks in real time. In addition, the automated pattern analysis techniques proposed could serve to democratize access to network security. While existing systems require domain specialization and expertise on both the attackers' and defenders' sides, running DART against a network topology requires very little training. This could allow small and medium-sized businesses, local governments, and other lower-resource organizations to find and secure weaknesses in their networks without requiring substantial additional investment.

Growing interest in federated systems [121], such as ActivityPub, suggests that there is a real and growing demand for decentralized systems in practice. By providing additional tools to create secure and resilient networked systems, I hope to remove some of the remaining barriers that block further adoption of decentralized approaches. In federated systems, which typically integrate with existing network infrastructure, decentralization occurs at the software level. However, I also hope to spur adoption of ad hoc communication.

There remains a core barrier to the adoption of decentralized and ad hoc systems that cannot be solved technically: incentive alignment. All networks must be built and deployed before being used, and that process necessarily consumes time and resources. Since those resources are typically provided by a single entity, such as a corporation or government, there is little incentive for cooperation and interoperability between networks versus limiting participants to a single, proprietary approach. Even real-world ad hoc deployments of opportunistic backhaul, such as Amazon's Sidewalk or Apple's Find My networks [34], restrict users' access to the network and instead only enable proprietary behaviors.

6.5 Mobility and Dynamics

As self-driving vehicles and other intrinsically-mobile connected platforms proliferate, it naturally raises questions about how mobility and network dynamics influence the concepts discussed thus far. The twin ideas of permeable networks and counter-adversarial behavior do not address the orthogonal challenges of changing or undiscovered topologies, for example.

In the approaches described so far, the underlying structure of the network is treated as as a static and known (or discoverable) baseline. This enables location information to be hashed and stored in an obscured format to maintain privacy, such as when masking delivery location. Alternatively, the structures identified during pattern analysis may no longer exist in the dynamic setting.

Static topology assumptions fail when looking at spatially-dynamic networks such as mobile ad hoc or vehicular ad hoc networks (MANETs and VANETs, respectively [115]). Similarly, in topologies with significant node churn – that is, where nodes

enter or exit the network stochastically – the topology formed at any given moment is dependent on which nodes are active at that time.

While fully mitigating these limitations is beyond the scope of this document, we can explore potential approaches that adapt the proposed methods to function more effectively. First, consider the dynamic topology case. Spatial partitioning will unfortunately not apply to these scenarios, but semantic partitioning will continue to function as normal since the underlying topology is orthogonal to its goals.

Pattern analysis may become more difficult, but indexing web pages (the stated motivation for the PageRank algorithm) necessarily also must compensate for dynamism in page content – substantial effort has been paid to updating and maintaining these structures in the dynamic case [55, 148]. One possible approach could rely on caches of recently-evicted and recently-discovered structures to assess the impact network changes have had, generating secondary heuristic estimations on the importance of seen but not analyzed sub-paths. Other approaches might rely on a continuous refinement strategy, where agents move through the dynamic network and continually emit outputs while stale observations are pruned over time. Probabilistic edge weights also allow for the representation of uncertain existence; the proposed model could rely on these weights to describe dynamic networks instead.

These approaches could work well when the topology is mutating relatively slowly, but would likely fail in e.g. the VANET case, where the entire topology is drastically changing over a matter of seconds. Instead, novel approaches such as those based on graph neural networks offer promising alternatives to handle these scenarios [30].

In cases undergoing topological churn, ad hoc methods adapt well because they align naturally with this assumption. Since ad hoc devices typically use low-power

commodity hardware, each node may degrade or fail and therefore solutions must adapt to this reality. Partitioning-based approaches and distributed fusion, which target this use case, will function in the high-churn case without additional adjustment.

6.6 Limitations and Challenges

While the security and reliability of networked systems serve as critical dimensions for assessing their efficacy, these are not the only metrics, and in practice, they may not even be the primary consideration. For example, when evaluating a network, an operator might also care about its overall data throughput or the average latency of messages that traverse it.

In all the real-time defensive mechanisms described thus far, and more generally in ad hoc and decentralized systems overall, both throughput and latency suffer when compared against fully-trusted, centralized systems. These drawbacks have been explored and mitigated for decades in the literature, but nevertheless remain a key barrier to the adoption of decentralized systems.

Another key limitation shared by several of the described approaches stems from the presence of an underlying shared secret during a setup phase. While each approach takes care to minimize the amount of time this information remains on the device by deleting it immediately after deployment, these kinds of approaches remain susceptible to supply chain attacks. If an attacker were able to capture a device prior to deployment, it would fundamentally undermine several of the described approaches.

In terms of offline analysis, such as that conducted by DART, computational scalability remains a challenge. Since the space of possible attacks scales quickly with the complexity and size of the network, conducting a thorough analysis of extremely

dense or very large networks would require further work in tackling this scalability challenge.

The knowledge of source and target nodes further complicates applicability to scenarios where no such information exists. Consider networks where the attacker's goal is to partition the network – while fundamentally a structural attack, current approaches would fail to identify the critical nodes without explicitly calling out the threat.

6.7 Future Research Directions

Each research avenue described in this dissertation unlocks other possibilities for future work, building upon their respective foundations. Semantic partitioning currently relies on a defined and agreed-upon set of topics, but in practice, these channels may evolve and change over time at a potentially-rapid rate. Improving the approach to allow for these changes, such as via the inclusion of methods inspired by dynamic topic modeling, would broaden the applicability of these concepts.

Structural attacks such as partitioning remain a challenging problem to detect and mitigate under the permeability assumption. Here, weaknesses in connectivity require a broader perspective on network topologies, which may be impossible to achieve with local knowledge. Methods combining ensemble fusion techniques with periodic heartbeat or local sensing behaviors could be a promising direction to mitigate such attacks.

Additional work to improve data compartmentalization approaches' resilience to structural attacks remains. For example, existing methods such as [15] assume the presence of an entry node to manage a portion of the authentication information.

While this may be reasonable in practice, it does constrain applications to specific contexts. Weakening or removing this assumption would allow for more widespread use of these techniques.

Similarly, spatial partitioning relies on a set of assumptions that could be expanded to apply to more scenarios. For example, the current implementation assumes nodes within a network are stationary [17], but real-world scenarios may involve mobile nodes. Future work could explore how spatial partitioning can be adapted for mobile networks, or for networks in which the location of nodes can change over time. This would require developing new techniques for tracking and dynamically updating the network as nodes move.

In the context of real-time defenses, additional work could be done to improve the overall efficiency and effectiveness of the proposed strategies. Investigating more efficient algorithms for detecting and responding to adversarial activity would be beneficial, particularly in large-scale, high-traffic networks where computational resources may be constrained. Furthermore, exploring ways to detect and counter future adversarial techniques, such as those that rely on AI-based attacking agents, would help improve the security of decentralized networks even further.

Future research could focus on optimizing the computational process for large-scale networks used by [18], as well as improving its ability to identify and respond to dynamic threats. For instance, incorporating adaptive or dynamically-updating approaches to automatically adapt the analysis to changing network topologies or emerging threat patterns could prove valuable. Additionally, extending the scope of DART to include more nuanced types of connections represented as a hypergraph could increase its overall applicability and usefulness.

The current automated pattern analysis landscape does not thoroughly examine the influence of time on attacks and defenses. In real-world scenarios, Advanced Persistent Threats (APTs) often exercise extreme patience, taking considerable time to infiltrate and spread slowly throughout a network. However, this extended operational duration also increases the window for potential detection. Incorporating a more nuanced approach that accounts for the attacker's speed could facilitate comparisons between low-and-slow and smash-and-grab strategies, ultimately informing defensive architectural design.

As discussed in Section 6.5, dynamic topologies and especially those with rapidly-changing connectivity will pose substantial challenges in future work. As autonomous vehicles become more prevalent, dynamic solutions will only continue to increase in importance. Future explorations in this area could extend the concepts of semantic partitioning introduced here, using the separation of responsibility to enable more robust and secure structures despite constantly-changing location data. For example, a three-party model consisting of automobile manufacturers, governmental licensing agencies, and vehicle owners would allow for practical and Sybil-resistant authentication while still dispersing responsibility among multiple parties to satisfy permeability.

To address the problem of imperfect information, such as networks where not all nodes or connections are known, future efforts may expand probabilistic approaches. For example, assigning a connection's existence likelihood as an edge weight would allow the application of existing probabilistic methods. Similar methods have been used in other partially-observable systems such as the border gateway protocol [90], and could be adapted to this context as well. Alternatively, iterative model refinement based on streaming network observations also offers promising avenues of exploration.

As in any dynamic model, care must be taken with respect to adversarial input shaping. However, such approaches would preserve the best aspects of globally-planned methods, such as efficient route planning.

Finally, one further research direction to explore is the development of incentive mechanisms for promoting interoperability and collaboration in decentralized and ad hoc networks. Finding ways to encourage entities to share resources, participate in network upkeep, and contribute to the overall security of the network could be crucial in overcoming some of the existing barriers to widespread adoption of such systems. This may involve investigating the application of game-theoretic approaches, reputation systems, or even blockchain-based solutions to rebuild trust and foster cooperation in decentralized network environments.

The work presented in this dissertation can serve as a foundation for further advances in the field of counter-adversarial resilience in decentralized and ad hoc networks. This future research, in turn, has the potential to create more secure, reliable, and resilient networked systems that can better withstand adversarial attacks and serve the ever-evolving needs of our increasingly connected world.

6.8 Final Reflections

In today's rapidly evolving digital landscape, the security and resilience of our networks are of paramount importance. The research presented in this dissertation seeks to address these challenges by investigating counter-adversarial techniques and strategies, aiming to create a more secure foundation for decentralized and ad hoc networks, or any network adopting the permeability assumption. Throughout the course of this work, a range of novel approaches have been explored, evaluated, and

ultimately refined, contributing to the broader understanding of network security and resilience.

The introduction of DART represents a significant step forward in the development of counter-adversarial strategies, offering a promising framework for detecting and mitigating threats in arbitrary networks. While there is still room for improvement and further development, the results obtained from the evaluation of DART demonstrate its potential for enhancing the security and resilience of such networks in the face of adversarial attacks.

Moreover, the investigation of partitioning as a technique for improving network resilience has shed light on the potential benefits and limitations of this approach. This work not only contributed to our understanding of data partitioning but also opened up new avenues of research for improving the efficiency and applicability of these techniques in more complex, dynamic environments.

As the world becomes increasingly interconnected and reliant on digital systems, the need for secure and resilient networks will continue to grow. The findings and contributions presented in this dissertation represent important advancements in the field of network security, providing valuable insights and a strong foundation for future work.

Ultimately, the pursuit of counter-adversarial resilience in permeable networks is a complex and multifaceted challenge. By building upon the research presented in this dissertation, I hope that future efforts will continue to push the boundaries of our understanding and capabilities, driving the development of increasingly secure, resilient, and adaptable networked systems that can withstand the diverse and evolving threats that our interconnected world faces.

REFERENCES

[1] Ablon, L. and A. Bogart, *Zero days, thousands of nights: The life and times of zero-day vulnerabilities and their exploits* (Rand Corporation, 2017).

[2] Abrardo, A., M. Barni, K. Kallas and B. Tondi, "A Game-Theoretic Framework for Optimum Decision Fusion in the Presence of Byzantines", IEEE TIFS (2016).

[3] Abrardo, A., M. Barni, K. Kallas and B. Tondi, "A message passing approach for decision fusion in adversarial multi-sensor networks", Information Fusion (2018).

[4] Akyildiz, I. F., W. Su, Y. Sankarasubramaniam and E. Cayirci, "Wireless sensor networks: A survey", Computer Networks **38**, 4, 393–422 (2002).

[5] Anderson, H. S., A. Kharkar, B. Filar and P. Roth, "Evading machine learning malware detection", black Hat **2017** (2017).

[6] Awerbuch, B., D. Holmer, C. Nita-Rotaru and H. Rubens, "An On-demand Secure Routing Protocol Resilient to Byzantine Failures", in "ACM Workshop on Wireless Security", (2002).

[7] Azar, Y., E. Cohen, A. Fiat, H. Kaplan and H. Räcke, "Optimal oblivious routing in polynomial time", Journal of Comp. and Systems Sci. (2004).

[8] Bansal, N., A. Blum, S. Chawla and A. Meyerson, "Online Oblivious Routing", in "Symposium on Parallel Algorithms and Arch. (SPAA)", (2003).

[9] Bao, T., Y. Shoshitaishvili, R. Wang, C. Kruegel, G. Vigna and D. Brumley, "How shall we play a game?: A game-theoretical model for cyber-warfare games", in "2017 IEEE 30th Computer Security Foundations Symposium (CSF)", pp. 7–21 (IEEE, 2017).

[10] Barabási, A.-L., "Scale-free networks: a decade and beyond", science **325**, 5939, 412–413 (2009).

[11] Barabási, A.-L. and R. Albert, "Emergence of scaling in random networks", Science (New York, N.Y.) **286**, 5439, 509–512 (1999).

[12] Batagelj, V. and U. Brandes, "Efficient generation of large random networks", Physical Review E **71**, 3, 036113 (2005).

[13] Batista, G. E. A. P. A., R. C. Prati and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data", ACM SIGKDD Explorations Newsletter **6**, 1, 20–29, URL `https://doi.org/10.1145/1007730.1007735` (2004).

[14] Behnel, S., R. Bradshaw, C. Citro, L. Dalcin, D. S. Seljebotn and K. Smith, "Cython: The Best of Both Worlds", Computing in Science and Engineering **13**, 2, 31–39, URL https://doi.org/10.1109/MCSE.2010.118 (2011).

[15] Behrens, H. W. and K. S. Candan, "Adversarially-Resistant On-Demand topic channels for wireless sensor networks", in "2018 IEEE 37th Symposium on Reliable Distributed Systems (SRDS)", pp. 83–92 (2018).

[16] Behrens, H. W. and K. S. Candan, "Pando: Efficient Byzantine-Tolerant Distributed Sensor Fusion using Forest Ensembles", in "ICC 2020 - 2020 IEEE International Conference on Communications (ICC)", pp. 1–6 (2020).

[17] Behrens, H. W. and K. S. Candan, "WindRose: Private Spatial Targeting in Adversarially-Resilient Geographic Routing", (2020).

[18] Behrens, H. W., B. Gelfand and K. S. Candan, "Defensible Network Topologies viaUnsupervised Attack Simulation", in "The 18th International Conference on Availability, Reliability and Security", (2023).

[19] Bellare, M., O. Goldreich and S. Goldwasser, "Incremental cryptography: The case of hashing and signing", in "CRYPTO", (1994).

[20] Benyo, B., P. Pal, R. Schantz, A. Paulos, D. J. Musliner, T. Marble, J. M. Rye, M. W. Boldt and S. Friedman, "Automated Self-Adaptation for Cyber-Defense – Pushing Adaptive Perimeter Protection Inward", in "2013 IEEE 7th International Conference on Self-Adaptation and Self-Organizing Systems Workshops", pp. 47–52 (2013).

[21] Berkhin, P., "A Survey on PageRank Computing", Internet Mathematics **2**, 1, 73–120, URL https://doi.org/10.1080/15427951.2005.10129098 (2005).

[22] Bhat, N., V. F. Farias, C. C. Moallemi and D. Sinha, "Near-Optimal A-B Testing", Management Science **66**, 10, 4477–4495, URL https://pubsonline.informs.org/doi/abs/10.1287/mnsc.2019.3424 (2020).

[23] Bollobás, B., C. Borgs, J. Chayes and O. Riordan, "Directed scale-free graphs", in "Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms", SODA '03, pp. 132–139 (Society for Industrial and Applied Mathematics, USA, 2003).

[24] Boyles, J. L., A. Smith and M. Madden, "Privacy and data management on mobile devices", Pew Internet & American Life Project **4**, 1–19 (2012).

[25] Brandes, U., "A faster algorithm for betweenness centrality", The Journal of Mathematical Sociology **25**, 2, 163–177 (2001).

[26] Brewster, T., "GPT-4 Can't Stop Helping Hackers Make Cybercriminal Tools — forbes.com", https://www.forbes.com/sites/thomasbrewster/2023/03/16/gpt-4-could-help-stupid-hackers-become-good-cybercriminals/, [Accessed 25-Apr-2023] (2023).

[27] Bruno, R., M. Conti and E. Gregori, "Mesh networks: Commodity multihop ad hoc networks", IEEE Communications Magazine (2005).

[28] Chandrasekhar, A., D. M. Gordon and S. Navlakha, "A distributed algorithm to maintain and repair the trail networks of arboreal ants", Scientific Reports (Nature) (2018).

[29] Connectivity Standards Alliance, "Release V1.0.0 Release · project-chip/connectedhomeip", https://github.com/project-chip/connectedhomeip/releases/tag/v1.0.0 (2022).

[30] da Silva, E. S., H. Pedrini and A. Santos, "Applying graph neural networks to support decision making on collective intelligent transportation systems", IEEE Transactions on Network and Service Management (2023).

[31] de Souza, E., O. Ardakanian and I. Nikolaidis, "A Co-simulation Platform for Evaluating Cyber Security and Control Applications in the Smart Grid", in "ICC 2020 - 2020 IEEE International Conference on Communications (ICC)", pp. 1–7 (2020).

[32] Delgado-Mohatar, O., A. Fúster-Sabater and J. M. Sierra, "A light-weight authentication scheme for wireless sensor networks", Ad Hoc Networks (2011).

[33] Delgado-Mohatar, O., A. Fúster-Sabater and J. M. Sierra, "A light-weight authentication scheme for wireless sensor networks", Ad Hoc Networks (2011).

[34] Despres, T., S. Patil, A. Tan, J.-L. Watson and P. Dutta, "Where the sidewalk ends: Privacy of opportunistic backhaul", in "Proceedings of the 15th European Workshop on Systems Security", EuroSec '22, pp. 1–7 (Association for Computing Machinery, New York, NY, USA, 2022).

[35] Douglas, J., T. Usländer, G. Schimak, J. F. Esteban and R. Denzer, "An open distributed architecture for sensor networks for risk management", Sensors **8**, 3, 1755–1773 (2008).

[36] Drobyshevskiy, M. and D. Turdakov, "Random graph modeling: A survey of the concepts", ACM computing surveys (CSUR) **52**, 6, 1–36 (2019).

[37] Enoch, S. Y., Z. Huang, C. Y. Moon, D. Lee, M. K. Ahn and D. S. Kim, "HARMer: Cyber-Attacks Automation and Evaluation", IEEE Access **8**, 129397–129414 (2020).

[38] Erdos, P., A. Rényi *et al.*, "On the evolution of random graphs", **5**, 1, 17–60 (1960).

[39] Esfahanian, A.-H., "Connectivity algorithms", Topics in structural graph theory pp. 268–281 (2013).

[40] Even, S., *Graph Algorithms* (Cambridge University Press, 2011).

[41] Fang, W., L. Zheng, H. Deng and H. Zhang, "Real-time motion tracking for mobile augmented/virtual reality using adaptive visual-inertial fusion", Sensors (2017).

[42] Feldmann, A. and W. Whitt, "Fitting mixtures of exponentials to long-tail distributions to analyze network performance models", Performance evaluation **31**, 3-4, 245–279 (1998).

[43] Fellman, P. V. and R. Wright, "Modeling terrorist networks, complex systems at the mid-range", arXiv preprint arXiv:1405.6989 (2014).

[44] Feng, C.-H., Y. Zhang, I. Demirkol and W. B. Heinzelman, "Stateless multicast protocol for ad hoc networks", Trans. on Mobile Computing (2012).

[45] Feng, H. and Y. Shu, "Study on network traffic prediction techniques", in "Proceedings. 2005 International Conference on Wireless Communications, Networking and Mobile Computing, 2005.", vol. 2, pp. 1041–1044 (2005).

[46] Flickenger, R., S. Okay, E. Pietrosemoli, M. Zennaro and C. Fonda, "Very Long Distance Wi-fi Networks", in "ACM SIGCOMM Networked Systems for Developing Regions (NSDR)", (2008).

[47] Frigault, M., L. Wang, A. Singhal and S. Jajodia, "Measuring network security using dynamic bayesian network", in "Proceedings of the 4th ACM Workshop on Quality of Protection", QoP '08, pp. 23–30 (Association for Computing Machinery, New York, NY, USA, 2008).

[48] Gade, K., "The seven ways to find heading", J. of Navigation **69**, 5, 955–970 (2016).

[49] Garewal, K. S. and K. S. Garewal, "The helium network", Practical Blockchains and Cryptocurrencies: Speed Up Your Application Development Process and Develop Distributed Applications with Confidence pp. 279–312 (2020).

[50] Gartzke, E. and J. R. Lindsay, "Weaving tangled webs: offense, defense, and deception in cyberspace", Security studies **24**, 2, 316–348 (2015).

[51] Gerstenfeld, J., "Understanding the connection between hackers andtheir hacks: Analyzing usdoj reports for hackerprofiles", International Journal of Cybersecurity Intelligence & Cybercrime **6**, 1, 59–76 (2023).

[52] Gil, S., S. Kumar, M. Mazumder, D. Katabi and D. Rus, "Guaranteeing spoof-resilient multi-robot networks", Autonomous Robots **41**, 6, 1383–1400 (2017).

[53] Glynn, P. W. and W. Whitt, "The Asymptotic Efficiency of Simulation Estimators", Operations Research **40**, 3, 505–520 (1992).

[54] Grobauer, B., T. Walloschek and E. Stocker, "Understanding Cloud Computing Vulnerabilities", IEEE Security Privacy **9**, 2, 50–57 (2011).

[55] Guo, W., Y. Li, M. Sha and K.-L. Tan, "Parallel personalized pagerank on dynamic graphs", Proceedings of the VLDB Endowment **11**, 1, 93–106 (2017).

[56] Guo, W., Q. Wang, K. Zhang, A. G. Ororbia, S. Huang, X. Liu, C. L. Giles, L. Lin and X. Xing, "Defending Against Adversarial Samples Without Security through Obscurity", in "2018 IEEE International Conference on Data Mining (ICDM)", pp. 137–146 (2018).

[57] Gustafsson, F., *Statistical Sensor Fusion* (Studentlitteratur AB, 2010).

[58] Hagberg, A., P. Swart and D. Chult, "Exploring network structure, dynamics, and function using networkx", Los Alamos Nat. Lab (2008).

[59] Harris, C. R., K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke and T. E. Oliphant, "Array programming with NumPy", Nature **585**, 7825, 357–362, URL `https://www.nature.com/articles/s41586-020-2649-2` (2020).

[60] Hassan, M. A., S. Ali, M. Imad and S. Bibi, "New advancements in cybersecurity: A comprehensive survey", Big Data Analytics and Computational Intelligence for Cybersecurity pp. 3–17 (2022).

[61] Hassan, W. H. *et al.*, "Current research on internet of things (iot) security: A survey", Computer networks **148**, 283–294 (2019).

[62] Hemphill, T. A. and P. Longstreet, "Financial data breaches in the us retail economy: Restoring confidence in information technology security standards", Technology in Society **44**, 30–38 (2016).

[63] Ho, G., M. Dhiman, D. Akhawe, V. Paxson, S. Savage, G. M. Voelker and D. Wagner, "Hopper: Modeling and detecting lateral movement", in "USENIX Security 2021", pp. 3093–3110 (USENIX Association, 2021).

[64] Holme, P. and B. J. Kim, "Growing scale-free networks with tunable clustering", Physical Review E **65**, 2, 026107, URL `https://link.aps.org/doi/10.1103/PhysRevE.65.026107` (2002).

[65] Hore, B., S. Mehrotra, M. Canim and M. Kantarcioglu, "Secure multidimensional range queries over outsourced data", VLDB (2012).

[66] Hu, Y.-C., A. Perrig and D. B. Johnson, "Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks", Wireless Networks; New York **11**, 1-2, 21–38 (2005).

[67] Hunkeler, U., H. L. Truong and A. Stanford-Clark, "Mqtt-s - a publish/subscribe protocol for wireless sensor networks", in "COMSWARE", (2008).

[68] Hunter, J. D., "Matplotlib: A 2D Graphics Environment", Computing in Science Engineering **9**, 3, 90–95 (2007).

[69] Hussain, F., R. Hussain, S. A. Hassan and E. Hossain, "Machine learning in iot security: Current solutions and future challenges", IEEE Communications Surveys & Tutorials **22**, 3, 1686–1721 (2020).

[70] Junhai, L. and et.al., "A survey of multicast routing protocols for mobile ad-hoc networks", Comm. Surveys & Tutorials (2009).

[71] Kaklamanis, C., D. Krizanc and T. Tsantilas, "Tight bounds for oblivious routing in the hypercube", Mathematical systems theory (1991).

[72] Karlof, C. and D. Wagner, "Secure routing in wireless sensor networks: Attacks and countermeasures", in "IEEE Workshop on Sensor Network Protocols and Applications", (2003).

[73] Keahey, K., P. Riteau, D. Stanzione, T. Cockerill, J. Mambretti, P. Rad and P. Ruth, "Chameleon: A Scalable Production Testbed for Computer Science Research", in "Contemporary High Performance Computing: From Petascale toward Exascale", edited by J. Vetter, vol. 3 of *Chapman & Hall/CRC Computational Science* (CRC Press, Boca Raton, FL, 2018), first edn.

[74] Kelly, J. and G. S. Sukhatme, "Visual-Inertial Sensor Fusion: Localization, Mapping and Sensor-to-Sensor Self-calibration", IJRR (2011).

[75] Khan, M. A. and K. Salah, "IoT security: Review, blockchain solutions, and open challenges", Future Generation Computer Systems **82**, 395–411, URL `https://www.sciencedirect.com/science/article/pii/S0167739X17315765` (2018).

[76] Kim, Y.-S. and J. Heo, "Device authentication protocol for smart grid systems using homomorphic hash", J. of Comm. and Networks (2012).

[77] Knight, S., H. Nguyen, N. Falkner, R. Bowden and M. Roughan, "The internet topology zoo", Selected Areas in Communications, IEEE Journal on **29**, 9, 1765–1775 (2011).

[78] Krohn, M. N., M. J. Freedman and D. Mazieres, "On-the-fly verification of rateless erasure codes for efficient content distribution", in "IEEE Symposium on Security and Privacy, 2004. Proceedings. 2004", pp. 226–240 (2004).

[79] Kuhn, F., R. Wattenhofer, Y. Zhang and A. Zollinger, "Geometric ad-hoc routing: Of theory and practice", in "ACM PODC", (2003).

[80] Kullback, S. and R. A. Leibler, "On information and sufficiency", The annals of mathematical statistics (1951).

[81] Lacharité, M.-S., B. Minaud and K. G. Paterson, "Improved reconstruction attacks on encrypted data using range query leakage", in "S&P", (2018).

[82] Lazarescu, M. T., "Design of a wsn platform for long-term environmental monitoring for iot applications", Journal on Emerging and Selected Topics in Circuits and Systems (2013).

[83] Leach, P. J., M. Mealling and R. Salz, "A universally unique identifier (uuid) urn namespace", (2005).

[84] Lee, S., B. Bhattacharjee and S. Banerjee, "Efficient geographic routing in multihop wireless networks", in "MobiHoc", (2005).

[85] Legg, J. C. and W. A. Fuller, "Chapter 3 - Two-Phase Sampling", in "Handbook of Statistics", edited by C. R. Rao, vol. 29 of *Handbook of Statistics*, pp. 55–70 (Elsevier, 2009), URL `https://www.sciencedirect.com/science/article/pii/S0169716108000035`.

[86] Lemaître, G., F. Nogueira and C. K. Aridas, "Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning", Journal of Machine Learning Research **18**, 17, 1–5, URL `http://jmlr.org/papers/v18/16-365.html` (2017).

[87] Leversage, D. J. and E. J. Byres, "Estimating a System's Mean Time-to-Compromise", IEEE Security Privacy **6**, 1, 52–60 (2008).

[88] Lewi, K., W. Kim, I. Maykov and S. Weis, "Securing Update Propagation with Homomorphic Hashing", Tech. Rep. 227, Facebook, Inc. (2019).

[89] Lewis, T. G., "Small-world networks", Network Science: Theory and Practice pp. 131–175 (2009).

[90] Leyba, K. G., J. J. Daymude, J.-G. Young, M. Newman, J. Rexford and S. Forrest, "Cutting through the noise to infer autonomous system topology", in "IEEE INFOCOM 2022-IEEE Conference on Computer Communications", pp. 1609–1618 (IEEE, 2022).

[91] Li, Q. and G. Cao, "Multicast authentication in the smart grid with one-time signature", IEEE Transactions on Smart Grid (2011).

[92] Li, Z. and G. Gong, "Data aggregation integrity based on homomorphic primitives in sensor networks.", in "ADHOC-NOW", (Springer, 2010).

[93] Liu, X. and J. S. Baras, "Using trust in distributed consensus with adversaries in sensor and other networks", in "FUSION", (2014).

[94] Luna, J., H. Ghani, D. Germanus and N. Suri, "A security metrics framework for the Cloud", in "Proceedings of the International Conference on Security and Cryptography", pp. 245–250 (2011).

[95] Maalel, N., E. Natalizio, A. Bouabdallah, P. Roux and M. Kellil, "Reliability for emergency applications in internet of things", in "Distributed Computing in Sensor Systems (DCOSS)", (2013).

[96] Maitin-Shepard, J., M. Tibouchi and D. F. Aranha, "Elliptic Curve Multiset Hash", The Computer Journal **60**, 4, 476–490 (2017).

[97] Marin, G. A., "Network security basics", IEEE security & privacy **3**, 6, 68–72 (2005).

[98] Maymounkov, P. and D. Mazieres, "Kademlia: A peer-to-peer information system based on the xor metric", in "IPTPS", (2002).

[99] McKerns, M. M., L. Strand, T. Sullivan, A. Fang and M. A. G. Aivazis, "Building a Framework for Predictive Science", arXiv:1202.1056 [cs] URL `http://arxiv.org/abs/1202.1056` (2012).

[100] McKinney, W., "Data Structures for Statistical Computing in Python", in "Python in Science Conference", pp. 56–61 (Austin, Texas, 2010), URL `https://conference.scipy.org/proceedings/scipy2010/mckinney.html`.

[101] McQueen, M. A., W. F. Boyer, M. A. Flynn and G. A. Beitel, "Time-to-Compromise Model for Cyber Risk Reduction Estimation", in "Quality of Protection", edited by D. Gollmann, F. Massacci and A. Yautsiukhin, vol. 23, pp. 49–64 (Springer US, Boston, MA, 2006).

[102] Merkle, R. C., "A Certified Digital Signature", in "Advances in Cryptology (CRYPTO) Proceedings", (1990).

[103] Mothukuri, V., R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha and G. Srivastava, "A survey on security and privacy of federated learning", Future Generation Computer Systems **115**, 619–640 (2021).

[104] Muñoz-González, L., D. Sgandurra, A. Paudice and E. C. Lupu, "Efficient Attack Graph Analysis through Approximate Inference", ACM Transactions on Privacy and Security **20**, 3, 10:1–10:30 (2017).

[105] Muchnik, L., S. Pei, L. C. Parra, S. D. Reis, J. S. Andrade Jr, S. Havlin and H. A. Makse, "Origins of power-law degree distribution in the heterogeneity of human activity in social networks", Scientific reports **3**, 1, 1783 (2013).

[106] Mushtaq, M. F., S. Jamel, A. H. Disina, Z. A. Pindar, N. S. A. Shakir and M. M. Deris, "A survey on the cryptographic encryption algorithms", International Journal of Advanced Computer Science and Applications **8**, 11 (2017).

[107] Nakamoto, S., "Bitcoin: A peer-to-peer electronic cash system", (2008).

[108] Naraine, R., "Stuxnet attackers used 4 Windows zero-day exploits", https://www.zdnet.com/article/stuxnet-attackers-used-4-windows-zero-day-exploits/ (2010).

[109] Newman, M. E., "Clustering and preferential attachment in growing networks", Physical review E **64**, 2, 025102 (2001).

[110] Newman, M. E. J. and D. J. Watts, "Renormalization group analysis of the small-world network model", Physics Letters A **263**, 4, 341–346, URL `https://www.sciencedirect.com/science/article/pii/S0375960199007574` (1999).

[111] Newville, M., T. Stensitzki, D. B. Allen, M. Rawlik, A. Ingargiola and A. Nelson, "Lmfit: Non-linear least-square minimization and curve-fitting for python", Astrophysics Source Code Library pp. ascl–1606 (2016).

[112] Nguyen, K. T., M. Laurent and N. Oualha, "Survey on secure communication protocols for the Internet of Things", Ad Hoc Networks **32**, 17–31 (2015).

[113] NIST, "Sp 800-154: Guide to data-centric system threat modeling", (2016).

[114] Page, L., S. Brin, R. Motwani and T. Winograd, "The PageRank citation ranking: Bringing order to the web.", Technical Report 1999-66, Stanford InfoLab / Stanford InfoLab (1999).

[115] Pathan, A.-S. K., *Security of self-organizing networks: MANET, WSN, WMN, VANET* (CRC press, 2016).

[116] Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and É. Duchesnay, "Scikit-learn: Machine Learning in Python", Journal of Machine Learning Research **12**, 85, 2825–2830, URL `http://jmlr.org/papers/v12/pedregosa11a.html` (2011).

[117] Pereira, G. C. C. F., R. C. A. Alves, F. L. da Silva, R. M. Azevedo, B. C. Albertini and C. B. Margi, "Performance Evaluation of Cryptographic Algorithms over IoT Platforms and Operating Systems", Security and Communication Networks **2017**, 2046735–2046735 (2017).

[118] Perrig, A., J. Stankovic and D. Wagner, "Security in wireless sensor networks", Communications of the ACM **47**, 6, 53–57 (2004).

[119] Provos, N., M. Friedl and P. Honeyman, "Preventing Privilege Escalation", USENIX Security p. 12 (2003).

[120] Qiu, T., Y. Lv, F. Xia, N. Chen, J. Wan and A. Tolba, "Ergid: An efficient routing protocol for emergency response internet of things", J. of Network and Computer App. (2016).

[121] Raman, A., S. Joglekar, E. D. Cristofaro, N. Sastry and G. Tyson, "Challenges in the Decentralised Web: The Mastodon Case", in "Proceedings of the Internet Measurement Conference", IMC '19, pp. 217–229 (Association for Computing Machinery, New York, NY, USA, 2019).

[122] Rao, A., S. Ratnasamy, C. Papadimitriou, S. Shenker and I. Stoica, "Geographic routing without location information", in "MobiHoc", (2003).

[123] Reina, D., M. Askalani, S. Toral, F. Barrero, E. Asimakopoulou and N. Bessis, "A survey on multihop ad hoc networks for disaster response scenarios", Intl. Journal of Distributed Sensor Networks (2015).

[124] Riley, G. F. and T. R. Henderson, "The ns-3 Network Simulator", in "Modeling and Tools for Network Simulation", edited by K. Wehrle, M. Güneş and J. Gross, pp. 15–34 (Springer, Berlin, Heidelberg, 2010).

[125] Rose, S., O. Borchert, S. Mitchell and S. Connelly, "Zero trust architecture", Tech. rep., National Institute of Standards and Technology (2020).

[126] Salton, G. and C. Buckley, "Improving retrieval performance by relevance feedback", Journal of the American Society for Information Science **41**, 4, 288–297 (1990).

[127] Satopaa, V., J. Albrecht, D. Irwin and B. Raghavan, "Finding a "Kneedle" in a Haystack: Detecting Knee Points in System Behavior", in "2011 31st International Conference on Distributed Computing Systems Workshops", pp. 166–171 (2011).

[128] Sengupta, S., A. Chowdhary, A. Sabur, A. Alshamrani, D. Huang and S. Kambhampati, "A survey of moving target defenses for network security", IEEE Communications Surveys & Tutorials **22**, 3, 1909–1941 (2020).

[129] Shaikh, A., S. Faizullah, D. Vasan and Z. Ahmed, "Slim-simple lightweight and intuitive multicast protocol for manets", International Journal of Computer Applications (2014).

[130] Shih, C.-S., P.-C. Hsiu, Y.-H. Chang and T.-W. Kuo, "Framework designs to enhance reliable and timely services of disaster management systems", in "Computer-Aided Design (ICCAD)", (2016).

[131] Shirkande, S. D. and R. A. Vatti, "Aco-based routing algorithms for ad-hoc network (wsn, manets): A survey", in "Communication Systems and Network Technologies (CSNT)", (2013).

[132] Shu, L., Y. Zhang, L. T. Yang, Y. Wang, M. Hauswirth and N. Xiong, "Tpgf: geographic routing in wireless multimedia sensor networks", Telecommunication Systems (2010).

[133] Siadati, H. and N. Memon, "Detecting Structurally Anomalous Logins Within Enterprise Networks", in "Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security", CCS '17, pp. 1273–1284 (Association for Computing Machinery, New York, NY, USA, 2017).

[134] Speicher, P., M. Steinmetz, J. Hoffmann, M. Backes and R. Künnemann, "Towards automated network mitigation analysis", in "Proceedings of the 34th ACM/SIGAPP symposium on applied computing", pp. 1971–1978 (2019).

[135] Strogatz, S. H., "Romanesque networks", Nature **433**, 7024, 365–366 (2005).

[136] Talbot, E. B., D. Frincke and M. Bishop, "Demythifying Cybersecurity", IEEE Security Privacy **8**, 3, 56–59 (2010).

[137] Tehrani, M. N., M. Uysal and H. Yanikomeroglu, "Device-to-device communication in 5g cellular networks: challenges, solutions, and future directions", IEEE Communications Magazine **52**, 5, 86–92 (2014).

[138] Tong, H., C. Faloutsos and J.-y. Pan, "Fast Random Walk with Restart and Its Applications", in "Sixth International Conference on Data Mining (ICDM'06)", pp. 613–622 (2006).

[139] Ussath, M., D. Jaeger, F. Cheng and C. Meinel, "Advanced persistent threats: Behind the scenes", in "2016 Annual Conference on Information Science and Systems (CISS)", pp. 181–186 (2016).

[140] Virtanen, P., R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris,

A. M. Archibald, A. H. Ribeiro, F. Pedregosa and P. van Mulbregt, "SciPy 1.0: Fundamental algorithms for scientific computing in Python", Nature Methods **17**, 3, 261–272, URL `https://www.nature.com/articles/s41592-019-0686-2` (2020).

[141] Watts, J., "Scale Dependency in Agent-Based Modeling: How Many Time Steps? How Many Simulations? How Many Agents?", in "Uncertainty and Sensitivity Analysis in Archaeological Computational Modeling", edited by M. Brouwer Burg, H. Peeters and W. A. Lovis, Interdisciplinary Contributions to Archaeology, pp. 91–111 (Springer International Publishing, Cham, 2016).

[142] Wu, S. and K. S. Candan, "GPER: Geographic power efficient routing in sensor networks", in "ICNP", (2004).

[143] Wu, S. and K. S. Candan, "GMP: Distributed Geographic Multicast Routing in Wireless Sensor Networks", in "ICDCS", (2006).

[144] Wu, S. and K. S. Candan, "Power-aware single-and multipath geographic routing in sensor networks", Ad Hoc Networks (2007).

[145] Wu, S. and K. S. Candan, "GPEB: Power-efficient geographic broadcasting in sensor networks", in "MobiHoc", (2008).

[146] Xiao, L., S. Boyd and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus", in "Info. Proc. in Sensor Networks (IPSN)", (2005).

[147] Yang, S.-H., "Wireless sensor networks: Principles, design and applications", Springer (2014).

[148] Zhang, H., P. Lofgren and A. Goel, "Approximate personalized pagerank on dynamic graphs", in "Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining", pp. 1315–1324 (2016).

[149] Zhang, K., J. Du, J. Wang, C. Jiang, Y. Ren and A. Benslimane, "Distributed Hierarchical Information Acquisition Systems Based on AUV Enabled Sensor Networks", in "ICC", (2019).

[150] Zhang, M., L. Wang, S. Jajodia, A. Singhal and M. Albanese, "Network Diversity: A Security Metric for Evaluating the Resilience of Networks Against Zero-Day Attacks", IEEE Transactions on Information Forensics and Security **11**, 5, 1071–1086 (2016).

[151] Zhang, X., K. Hengster-Movrić, M. Šebek, W. Desmet and C. Faria, "Consensus-based distributed sensor fusion over a network", in "Conf. on Control Tech. and Applications (CCTA)", (2017).