Secure Network Coding for Wireless Mesh Networks: Threats, Challenges, and Directions

Jing Dong[*,a], Reza Curtmola[b], Cristina Nita-Rotaru[a]

[a]*Department of Computer Science, Purdue University*
305 N. University St., West Lafayette, IN 47907
[b]*Department of Computer Science, New Jersey Institute of Technology*
218 Central Ave, Newark, NJ 07102

Abstract

In recent years, network coding has emerged as a new communication paradigm that can significantly improve the efficiency of network protocols by requiring intermediate nodes to mix packets before forwarding them. Recently, several real-world systems have been proposed to leverage network coding in wireless networks. Although the theoretical foundations of network coding are well understood, a real-world system needs to solve a plethora of practical aspects before network coding can meet its promised potential. These practical design choices expose network coding systems to a wide range of attacks.

We identify two general frameworks (inter-flow and intra-flow) that encompass several network coding-based systems proposed in wireless networks. Our systematic analysis of the components of these frameworks reveals vulnerabilities to a wide range of attacks, which may severely degrade system performance. Then, we identify security goals and design challenges in achieving security for network coding systems. Adequate understanding of both the threats and challenges is essential to effectively design secure practical network coding systems. Our paper should be viewed as a cautionary note pointing out the frailty of current network coding-based wireless systems and a general guideline in the effort of achieving security for network coding systems.

Key words: Wireless network coding, network coding attacks, network coding security

1. Introduction

Network coding is a promising paradigm that has been shown to improve throughput and provide elegant solutions to problems that were traditionally considered difficult, such as congestion control and reliability. The core principle of network coding is that intermediate nodes actively mix (or *code*) input packets and forward the resulting coded packets. Several practical systems have been proposed to bridge theory with practice in the context of wireless mesh networks [8, 50, 49, 31, 37, 12].

Although the theoretical foundations of network coding are well understood, real-world systems need to solve a plethora of practical aspects before network coding meets its promised potential. The wireless communication medium has inherent particularities, such as high error rates and unpredictable signal strength, creating a complex, unpredictable and challenging environment. As a result, network coding systems need to make numerous practical design choices and optimizations that are essential to leverage network coding and achieve good performance. Unfortunately, in the quest for performance, security aspects are disregarded: Many of these design choices result in protocols that have numerous security vulnerabilities.

For example, the very nature of packet mixing makes network coding systems vulnerable to a severe security threat known as *pollution attacks*, in which attackers inject corrupted packets into the network. Although packet injection is not a new attack, its impact on network coding is devastating. This is because as long as there is one corrupted

[*] Corresponding author. Phone: 1-765-496-9398 Fax: 1-765-494-0739
Email addresses: dongj@cs.purdue.edu (Jing Dong), crix@njit.edu (Reza Curtmola), crisn@cs.purdue.edu (Cristina Nita-Rotaru)

packet that an intermediate node uses during the coding process, then all the coded packets forwarded by the node will be corrupted. The result is an epidemic propagation of corrupted packets, as further nodes code and forward more corrupted packets.

Previous work relevant to the security of network coding has focused exclusively on the *packet pollution* attack. Several solutions were proposed to combat this attack [9, 52, 47]. However, packet pollution is only one of many potential attacks. The complexity of network coding systems, as well as the inherent vulnerability of multi-hop wireless networks, create numerous opportunities for attacks against network coding systems for wireless networks.

In this paper, following a well-known security principle that states a system is as secure as its weakest link, we focus on the security of a network coding system in its entirety and examine all of its components. We analyze the security of the various components of a network coding system. Most of the practical network coding systems [8, 50, 49, 31, 37, 12] we are aware of have been proposed for wireless mesh networks (WMNs). Thus, the main focus of our analysis is network coding systems designed for WMNs. We describe two general frameworks that encompass several network coding-based systems proposed for wireless networks. Depending on how they leverage the benefits of network coding, we classify these systems into *intra-flow* network coding systems [8, 50, 49], which mix packets within the same individual flows, and *inter-flow* network coding systems [31, 37, 12], which mix packets across multiple different flows.

We systematically analyze the components of these frameworks and identify potential security vulnerabilities that may severely degrade system performance. To the best of our knowledge, this is the first paper to systematically analyze the security of each component in practical network coding-based wireless systems. As a proof of concept, we experimentally demonstrate the severity of attacks in network coding systems. Our experiments show that even a single attacker whose only action is dropping packets can cause over 50% of throughput degradation for over 80% of flows.

We identify the security goals and challenges in addressing the security vulnerabilities and discuss several directions for designing defense mechanisms. The multitude of attack avenues presented by current network coding-based systems leads us to conclude there is a tension between the performance of such systems and their security. Mostly due to protocol complexity, it becomes extremely difficult to secure such systems in their entirety. Our paper should be viewed as a cautionary note pointing out the frailty of current network coding-based wireless systems.

In the rest of the paper, we first review related work in Section 2. We then state the network model and provide a classification of network coding systems for wireless networks in Section 3. We present a systematic analysis of the threats in network coding systems in Section 4. In Section 5, we present security goals and design challenges for achieving secure network coding systems. Finally, we demonstrate the impact of the attacks through simulations in Section 6, and conclude the paper in Section 7.

2. Related Work

Previous work related to the security of network coding focuses exclusively on the packet pollution attack. Our goal in this paper is to look beyond packet pollution and systematically analyze the security of every component in a network coding system.

Pollution attacks. The study of pollution attacks has only focused on intra-flow network coding systems and, to the best of our knowledge, addressing pollution attacks for inter-flow network coding systems is still an open problem. Current solutions to packet pollution attacks in intra-flow coding systems can be categorized into cryptographic approaches, information theoretic approaches, and approaches based on network error correction coding.

Cryptographic approaches rely on augmenting the network coded packets with additional verification information; this allows intermediate nodes to verify the validity of coded packets and filter out polluted packets. Existing schemes use specialized homomorphic hash functions or homomorphic digital signatures. In hash-based schemes [36, 21], the source uses a homomorphic hash function to compute a hash of each native data packet and sends these hashes to intermediate nodes via an authenticated channel. The homomorphic property of the hash function allows nodes to compute the hash of a coded packet out of the hashes of native packets. The scheme proposed in [36] has a high computational overhead, but this limitation is overcome in [21] by using probabilistic batch verification in conjunction with a cooperative detection mechanism. The work in [32] also presents a scheme to overcome the high computation overhead of homomorphic hash schemes by leveraging the null space of coded packets to achieve packet authentication. Most of the schemes based on digital signatures [9, 39, 52, 47] require reliable distribution of a new public key for every new file that is sent and the size of the public key is linear in the file size. This requirement limits their scalability for large-scale content distribution. The only exception is a recent scheme [6] which achieves constant-size public key at the cost of using expensive bilinear maps. The scheme presented in [48] avoids using homomorphic signatures or hashes all together by relying solely on much more efficient symmetric key encryptions, however, the drawback is the significantly larger bandwidth overhead.

Information theoretic approaches do not filter out polluted packets at intermediate nodes; instead, they either encode enough redundant information into packets which allows receivers to detect the presence of polluted packets [24], or use a distributed protocol which allows receivers to tolerate pollution and recover native packets [28]. However, given that polluted packets are not filtered out, the throughput that can be achieved by the protocol is upper-bounded by the information-theoretic optimal rate of $C - z_O$, where C is the network capacity from the source to the receiver and z_O is the network capacity from the adversary to the receiver. Thus, if the attacker has a large bandwidth to the receiver, the useful throughput can rapidly degrade to 0. Wang et al. [45] propose to reduce the capacity of the attacker by only allowing nodes to broadcast at most once in the network. This model requires trusted nodes and differs vastly from practical systems for wireless networks, where each intermediate node in general forwards multiple coded packets.

Finally, there are approaches based on a *network error correcting coding theory* for detecting and correcting corrupted packets in network coding systems [44, 34, 46, 7]. In principle, the network error correction coding theory is parallel to classic coding theory for traditional communication networks, and also exhibits a fundamental trade-off between coding rate (bandwidth overhead of coding) and the error correction ability. Such schemes have limited error correcting ability and are inherently oriented toward network environments where errors occur infrequently. In an adversarial wireless environment, the attackers can easily overwhelm the error correction capability of the scheme by injecting a large number of polluted packets, resulting in incorrect decoding.

Secure routing in multi-hop wireless networks. In addition to combating packet pollution attacks, there has been significant work to achieve secure routing in multi-hop wireless networks. Relevant work includes protocols for secure route establishment and secure packet forwarding. Secure route establishment [27, 25, 22, 43] protects the route selection process from being tampered by attacker nodes. Secure packet forwarding protocols address packet injection and packet dropping attacks [40, 4, 11, 13]. However, they are designed for traditional routing protocols without the use of coding. For example, monitoring-based defenses (e.g., watchdog [40]) have been proposed for addressing packet modification and dropping for traditional routing protocols. With network coding, such schemes are no longer effective, as an upstream node usually cannot decode or verify the correctness of packets coded by a downstream node in a network coding system.

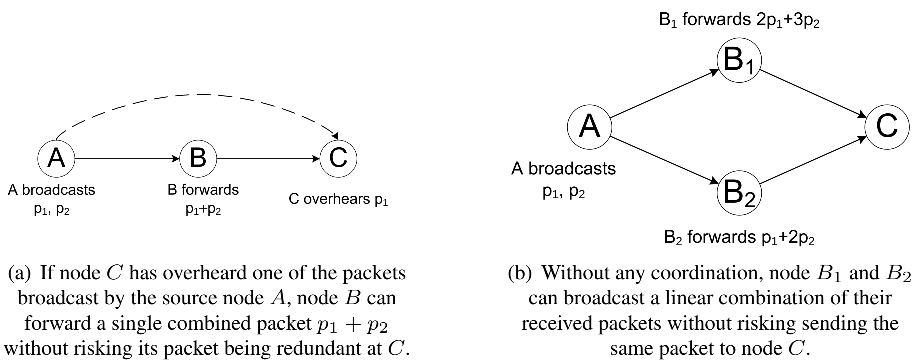
3. Network Coding-based Wireless Systems

In this section, we present the general frameworks for network coding systems for wireless mesh networks. There are two general approaches for applying network coding to wireless mesh networks, intra-flow network coding and inter-flow network coding. Both approaches exploit the *broadcast advantage* and *opportunistic listening* in wireless networks to reduce transmissions and improve performance. However, these benefits are realized differently: Intra-flow network coding systems mix packets within individual flows, while inter-flow network coding systems mix packets across multiple flows.

3.1. Network Model

Network coding has been leveraged as a generic technique in several types of wireless networks, vehicular ad hoc networks (VANETs) [38], sensor networks (WSNs) [23], and Disruption-Tolerant Networks (DTNs) [18, 51]. As our focus is on network coding for WMNs, we point out several specific characteristics of WMNs.

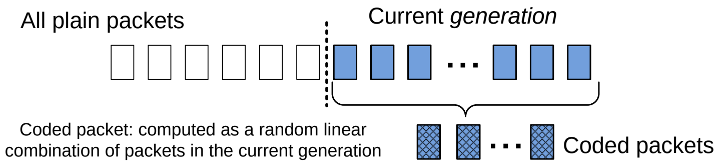
WMNs usually consist of a set of stationary nodes that communicate using the wireless transmission medium, as such they offer several benefits over other wireless architectures. They have better coverage than wireless local area networks (WLANs) because of the multi-hop communication. WMNs also have less restrictive mobility and energy constraints than MANETs or VANETs since routers are fixed and only clients can be mobile and may have energy



(a) If node C has overheard one of the packets broadcast by the source node A, node B can forward a single combined packet $p_1 + p_2$ without risking its packet being redundant at C.

(b) Without any coordination, node B_1 and B_2 can broadcast a linear combination of their received packets without risking sending the same packet to node C.

Figure 1: Illustrative examples for intra-flow network coding.



All plain packets
Current *generation*
Coded packet: computed as a random linear combination of packets in the current generation
Coded packets

Figure 2: An illustration of plain packets, generation, and coded packets in intra-flow network coding systems.

constraints. Nodes can maintain state (by storing it on non-volatile memory) and are able to perform public-key cryptographic operations (such as digital signatures). The typical mesh network configuration contains redundancy, which translates in multiple possible routes between sources and destinations. Finally, WMNs have support for increased bandwidth by using nodes equipped with multiple interfaces. These features create opportunities for increased robustness and scalability, resulting in higher quality of service and information availability. The community-oriented nature of WMNs facilitates group applications, such as webcast, distance learning, online gaming, video conferencing, and multimedia broadcasting.

WMNs follow a multi-hop communication pattern: the source node sends data to the destination nodes through a set of intermediate nodes. The intermediate nodes connect the source and destination nodes either through a single path or through multiple paths. Any of the nodes in the network can be either a source or a destination node. However, in the case of a infrastructure access WMN, the nodes that are connected to the wired infrastructure are likely to be source or destination nodes.

For illustrative purposes, we describe the capabilities of a typical node used in two of the practical network coding systems [8, 31]. A node is a desktop-class device (*i.e.*, a PC), equipped with a 802.11 wireless card (bandwidth up to 54 Mbps) attached to an omni-directional antenna. The wireless card operates in ad hoc mode and transmits at a power level of 15-18 dBm.

3.2. Intra-flow Network Coding

The general idea of intra-flow network coding is to leverage the opportunistic overhearing with packet coding to avoid redundant transmissions of packets in the network, thus improving performance. Fig. 1 shows two illustrative examples of how intra-flow coding benefits. In both Fig. 1(a) and 1(b), node A has two packets p_1 and p_2 to send to node C, and broadcasts these two packets locally. In Fig. 1(a), if node C has overheard one of the packets, node B can broadcast a single coded packet $p_1 + p_2$ to allow node C to recover both packets, without coordinating with C to avoid sending a redundant packet that has already been received by C. In Fig. 1(b), the two intermediate nodes B_1 and B_2 can broadcast a random linear combination of their received packets, avoiding the possibility that they both send the same packet to node C. Below, we first give a description of a general system architecture for intra-flow network coding, then explain in detail the coding and decoding operations.

In intra-flow network coding systems, packets are delivered in batches, referred to as *generations*. Each node forwards *coded packets* for the current generation, which are computed as a random linear combination of the packets in the generation (as illustrated in Fig. 2). To send a generation of packets, the source node continuously broadcasts coded packets for the current generation until an acknowledgment (ACK) is received from the destination. The coded packets are relayed to the destination via a set of intermediate nodes, referred to as *forwarder nodes*. Each forwarder node stores linearly independent coded packets it overhears and forwards new coded packets by combining the coded packets stored in its buffer. When the destination node receives enough linearly independent coded packets, it decodes the packets by solving a system of linear equations and unicasts an ACK packet to the source node, allowing the source to start sending the next generation of packets.

An intra-flow coding system consists of the following components: Forwarding node selection and rate assignment, data packet forwarding, acknowledgment delivery, and the coding/decoding procedure.

Forwarding node selection and rate assignment. This process determines the forwarder node set and the rate at which each forwarder node forwards coded packets. The optimal selection of forwarder nodes and rate assignment takes into consideration several factors, such as the topological distance of each node to the source and destination nodes, the interference among nodes, and the fairness among different flows. The topological position of a node refers to the topological distance of the node to the source and destination, as measured by certain routing metric, such as ETX [10]. The topological position determines a node's ability to contribute to the delivery of data packets for a particular flow. A forwarding node selection and rate assignment algorithm strives to select nodes that have large enough contribution in the forwarder set and assign large forwarding rate to nodes that can make large contributions. In contrast, nodes that make little or no contribution are not included in the forwarder set and thus do not participate in the data delivery. The interference relationship among nodes also affects the node selection and rate assignment process. An overly aggressive rate assignment can result in interference, and reducing the protocol performance; on the other hand, an under-assignment of forwarding rate delays the packet delivery process, also resulting in a reduced protocol performance. Due to the global nature of the input required, existing intra-flow coding protocols, such as MORE [8], DICE [49], and [50], use a centralized approach, where the computation is based on a link state graph maintained at each node as in a link state routing protocol. The source performs the centralized computation of the forwarding node set and rate assignment, and disseminates this information to the other nodes piggybacked on data packets.

Data packet forwarding. The forwarder nodes and the destination node maintain a buffer of linearly independent packets that they have overheard. Each forwarder node generates new coded packets by computing random linear combinations of coded packets stored in its buffer and broadcasts them at a pre-assigned rate as discussed above. When the destination node overhears enough linearly independent coded packets, it decodes the packets by solving a system of linear equations, and initiates the acknowledgment process as described below.

Acknowledgment delivery. The ACK packet is delivered from the destination to the source using the traditional single path routing process via the best quality path. The timely and reliable delivery of ACK is critical to ensure that the source moves to the next batch quickly. Thus, each intermediate node delivers ACK packets with high priority and ensures reliability by mandating an explicit acknowledgment from the next hop.

Packet coding and decoding. A key component of an intra-flow network coding system is the coding and decoding operations performed by nodes, which we present as follows.

Let G denote a generation of n *plain packets* $p_1, p_2, \cdots, p_n$. In intra-flow network coding, each plain packet p_i is viewed as a column vector of m elements in a finite field $\mathbb{F}_q$ of size q, i.e.

$$\vec{p_i} = (p_{i1}, p_{i2}, \cdots, p_{im})^T, p_{ij} \in \mathbb{F}_q.$$

Then, a generation G of n packets can be viewed as a $m \times n$ matrix, i.e.

$$G = [\vec{p_1}, \vec{p_2}, \ldots, \vec{p_n}],$$

with each plain packet being a column in the matrix.

A coded packet consists of two components, $(\vec{c}, \vec{e})$, referred to as *coding vector* and *coded data*, respectively. The coding vector $\vec{c} = (c_1, c_2, \ldots, c_n)$ is a random vector in ${\mathbb{F}_q}^n$. The coded data $\vec{e}$ is computed as a linear combination of packets in G using the components of $\vec{c}$ as the coefficients, i.e.

$$\vec{e} = \sum_{i=1}^{n} c_i \vec{p_i}.$$

In the matrix form, we can write

$$\vec{e} = G\vec{c}.$$

The source node computes coded packets by selecting a random coding vector $\vec{c}$, and then computing its corresponding coded data $\vec{e} = G\vec{c}$ from the plain packets in the generation G. Each forwarder node computes new coded packets by forming random linear combinations of the coded packets it has received. Let $(\vec{c_1}, \vec{e_1}), (\vec{c_2}, \vec{e_2}), \ldots, (\vec{c_k}, \vec{e_k})$ be the set of coded packets a forwarder node has received, to compute a new coded packet $(\vec{c'}, \vec{e'})$, it first selects a random vector $\vec{h} = (h_1, h_2, \ldots, h_k)$, and then computes $\vec{c'} = \sum_{i=1}^{k} h_i \vec{c_i}$ and $\vec{e'} = \sum_{i=1}^{k} h_i \vec{e_i}$. It is easy to verify that $(\vec{c'}, \vec{e'})$ is a valid coded packet with $\vec{e'} = \sum_{i=1}^{n} c'_i \vec{p_i}$.

When the destination node receives n linearly independent coded packets $(\vec{c_1}, \vec{e_1}), (\vec{c_2}, \vec{e_2}), \ldots, (\vec{c_n}, \vec{e_n})$, it can establish a system of linear equations $\{\vec{e_i} = \sum_{i=1}^{n} c_i \vec{p_i}\}$, which consists of n equations and n unknowns ($\vec{p_i}$s are unknowns). By solving this system of linear equations, it can recover the plain packets $\vec{p_1}, \vec{p_2}, \ldots, \vec{p_n}$.

3.3. *Inter-flow Network Coding*

Inter-flow network coding exploits opportunistic listening and wireless broadcast with *opportunistic coding* at intermediate nodes. The key idea is that when a node has a set of packets for different flows to be delivered to different next hop nodes, instead of unicasting each packet individually to its corresponding next hop node, the node combines the packets together and broadcasts the combined packet once for all the next hop nodes. Therefore, inter-flow coding reduces multiple individual unicast transmissions to only one broadcast transmission. Fig. 3(a) shows a simple illustrative example of how inter-flow network coding benefits. In this figure, node A has a packet m_i to send to each of the downstream node R_i. Instead of unicasting each packet individually, node A can take advantage of the packets that have already been overheard by node R_is and broadcast a single coded packet to allow each R_i to recover its desired packet. Below we present an overview of the system architecture for inter-flow network coding systems and the essential system components.

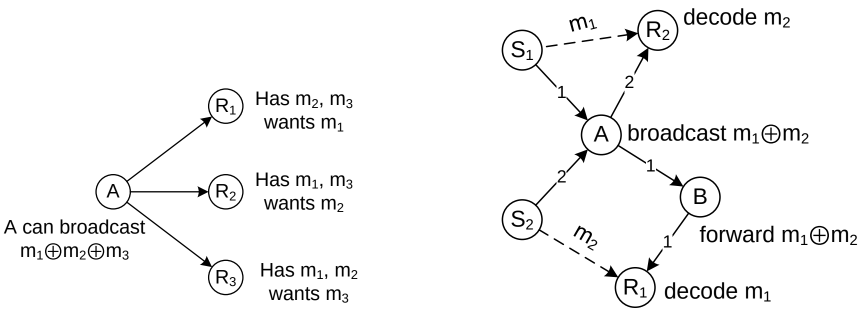
Inter-flow network coding systems are generally designed on top of traditional routing protocols. Given a set of paths for different flows, inter-flow network coding identifies a set of nodes at the intersections of paths to combine unicast transmissions of plain packets for different flows into broadcast transmissions of coded packets. The downstream nodes decode the coded packets using their overheard packets. Thus, unlike intra-flow network coding where all forwarder nodes perform coding and only receiver nodes perform decoding, in inter-flow network coding only nodes at the intersections of flows perform coding operations and any downstream nodes that have overheard necessary packets can perform the decoding operation.

In general, an inter-flow coding system consists of the following four components: Discovery of coding opportunities, packet coding and decoding, packet forwarding, and routing integration for increased coding opportunities.

Discovery of coding opportunities. Coding opportunities at a node consist of the packets that can be coded together for transmission such that the packet is decodable at downstream nodes. Based on the scope considered for coding opportunities, we can classify inter-flow coding protocols into *localized coding protocols* (only one hop neighbors of a node are considered for potential coding opportunities) and *global coding protocols* (all the nodes in the network are considered). In both cases, the discovery of coding opportunities requires a node to collect information about packet reception at other nodes.

In localized coding protocols (*e.g.*, COPE [31] and [15]), each node periodically reports its packet reception to its neighbors via local broadcasts. For example, in Fig. 3(a), each node R_i reports its received packets to node A, allowing node A to broadcast a single coded packet $m_1 \oplus m_2 \oplus m_3$ and be sure that it can be decoded by all the downstream nodes. To deal with loss of reception reports, link qualities are also used to guess whether a neighboring node receives a packet. For example, if a neighbor of a node has very good link quality to the previous hop of a packet, then the node can infer that this neighbor also receives the packet with high confidence.

In global coding protocols (*e.g.*, DCAR [37]), each node keeps track of all other nodes in the network that can overhear a packet by maintaining the neighboring node set of all the nodes on the path of the packet. To achieve this,



(a) Local coding systems

(b) Global coding systems

Figure 3: Illustrative examples for inter-flow network coding. The number on edges represents the flow ID that the edge is on. Dashed lines represent packet overhearing.

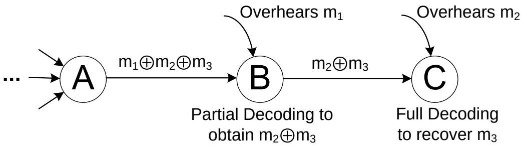


Figure 4: An example scenario illustrating partial and full decoding in an inter-flow network coding system.

the protocol follows a common flood-based on-demand route discovery process (e.g. DSR [29]), except that each node includes in the route request message its neighboring node set in addition to its own identifier. An example of a global coding protocol is shown in Fig. 3(b), where node A knows that nodes R_1 and R_2 are the neighboring of nodes S_2 and S_1, respectively. Thus, it can broadcast a single coded packet $m_1 \oplus m_2$ to allow both receiver nodes R_1 and R_2 to decode and recover their desired packet using their overheard packet.

Packet coding and decoding. Both the coding and decoding operations in inter-flow network coding are bit-wise XOR operations on packets. More specifically, given a set of plain packets $m_1, m_2, \ldots, m_k$ for distinct flows, a coded packet e is computed as $e = m_1 \oplus m_2 \oplus \ldots \oplus m_k$. The decoding operation performed by a node may be either a *full decoding*, which results in a plain packet, or a *partial decoding*, which results in another (more simple) coded packet. Given a coded packet, a node performs a full decoding if it has received all but one of the packets that are XORed to form the coded packet. Otherwise, the node performs a partial decoding to obtain another (more simple) coded packet. For example, given a coded packet $e = m_1 \oplus m_2 \oplus \ldots \oplus m_k$, if a node has overheard packet $m_2, \ldots, m_k$, it can perform a full decoding of e to recover the plain packet m_1 by computing $e \oplus m_2 \oplus \ldots \oplus m_k$. If it has only received packets $m_3, \ldots, m_k$, it performs a partial decoding by computing $e \oplus m_3 \oplus \ldots \oplus m_k$ to obtain a new coded packet $m_1 \oplus m_2$. Fig. 4 gives a simple example demonstrating partial and full decoding in a network coding system. In this figure, as node B has only overheard packet m_1, on receiving a coded packet $m_1 \oplus m_2 \oplus m_3$, it performs a partial decoding and delivers the partially decoded packet $m_2 \oplus m_3$ to node C, which allows node C to perform a full decoding using its overheard packet m_2 to recover its desired packet m_3.

Packet forwarding. Unlike traditional routing protocol where each node only forwards a packet to one next hop node, in inter-flow network coding a node needs to deliver a coded packet to multiple next hop nodes. In order to achieve a reliability guarantee similar to the 802.11 link layer reliability, a node needs to ensure that a coded packet is received by all the intended next hop nodes. However, 802.11 broadcast lacks the reliability of unicast communication. To address this problem, a *pseudo-broadcast* technique ([31, 37]) has been commonly adopted. With pseudo-broadcast, the sender node sends coded packets with 802.11 unicast using one of the intended next hop nodes as the MAC receiver. The packet will be retransmitted multiple times until the designated MAC receiver receives the packet and acknowledges it. The multiple retransmissions also allow the other next hop nodes more opportunities to receive the packet. To guarantee full reliability, the other next hop nodes are also required to acknowledge the packet, which is achieved by piggybacking the ACK on other packets broadcast by the node. If the ACKs of some next hop nodes are not received after a timeout period, the sender node retransmits the packet for such nodes by either sending them individually or coding them with other packets.

Routing integration. An inter-flow coding protocol can be designed independently of routing protocols, where coding opportunities arise from incidental path intersections. A natural extension to further improve the performance is to design coding-aware routing protocols, so that paths are selected to maximize the benefits of coding. Such coding-aware routing protocols are usually realized with new coding-aware metrics, which discount the cost of links that allow coding. The optimal path selection based on such metrics can be performed in either distributed or centralized fashion. For example, in DCAR [37], the metric aggregation and path selection follows the same steps of traditional source routing protocols (e.g. DSR [29]), except that each node also considers coding opportunities when computing path metrics. In contrast, [12] adopts a centralized link state routing-like approach, where each node floods its own coding-aware link metrics and local flow information in the network. The source node then computes the optimal paths based on the complete network information.

4. Threats in Network Coding Systems for Wireless Networks

In this section, we present potential security threats in current network coding systems. We focus solely on attacks on the network coding system that aim to disrupt the data delivery process. Below we first describe the assumptions we make about the attacker nodes and then systematically describe attacks against each component of the two classes of the network coding systems.

4.1. Adversarial Model

In this section, we describe the general adversarial model which applies to all considered network coding systems. This general model is later refined for intra-flow and for inter-flow network systems, in their respective sections.

The network may come under attack from both *outside* and *inside* attackers. Outside attackers are limited to attacking the network without having access to network resources; this includes eavesdropping, injection, modification and replay of packets. Inside attackers, having obtained access to key material, can pose as authorized participants and attack the network from the inside. Inside attackers are either nodes that have been compromised or honest nodes that have turned malicious. Attacker nodes are assumed to have the same capabilities as regular nodes (*i.e.*, desktop-class devices). We do not assume that nodes in the network are tamper-resistant.

The main objective of the attackers is to degrade network performance (*i.e.*, throughput) while minimizing the amount of expended resources. We differentiate between two types of attacks by adversarial nodes that directly manipulate data packets and those that target to improve the attacker's participation in the network. The effectiveness of an attack by a node that manipulates data packets (*e.g.*, drops or modifies) is directly proportional with the amount of packets that pass through the node. Under benign conditions, nodes should receive, process and forward an amount of packets proportional to their position and the quality of the links in the network. We refer to this as the *benign rate*. We call a *luring attack* any attack which results in an increase of a node's rate over its benign rate. Luring attacks can be used as amplifiers for other attacks such as packet dropping or packet pollution.

4.2. Intra-flow Network Coding

We analyze security vulnerabilities in each component of intra-flow network coding systems.

4.2.1. Forwarding node selection and rate assignment

A key input to the forwarding node selection and rate assignment process is the link state graph, which is maintained as in a link state routing protocol as follows. Each node monitors its local link qualities, and periodically floods the information in the entire network. Albeit being simple, there are numerous security vulnerabilities that can result in incorrect link state graphs at nodes, and consequently incorrect selection of forwarding nodes and rate assignment.

- **Link quality falsification or modification.** The attacker node can claim false metrics for its own adjacent links. Such attacks are difficult to prevent, as this information is local to the attacker node. A reactive approach, *e.g.* in [13], that detects the attack and then reactively identifies and isolates the attacker, may be a more viable solution. Alternatively, the attacker node may modify link qualities reported by other nodes as they are flooded in the network. Such attacks can be prevented using message authentication, such as digital signatures.

• **Wormholes.** Wormhole attacks can introduce fictitious links between honest nodes, and distort their perception of network topology. Although existing wormhole solutions, such as packet leashes [26] and TrueLink [17] can be applied, they typically incur substantial overhead, which can potentially nullify the performance gain of network coding.

Designing a secure and efficient link state protocol is a challenging task. To our knowledge, there is no existing solution that ensures correct link state propagation in wireless networks in the presence of colluding attackers.

4.2.2. Data forwarding

The data forwarding process is subject to packet pollution and packet dropping attacks.

• **Packet pollution.** Packet pollution attacks are the most well-known and most studied attacks against network coding systems. In packet pollution attacks, the attacker node injects corrupted packets into the network. Using the notation defined in Section 3.2, a packet $(\vec{c}, \vec{e})$ is *corrupted* if $\vec{e} \neq \sum_{i=1}^{n} c_i \vec{p_i}$. In other words, a packet is corrupted if it is not a valid linear combination of the packets in the generation as claimed by the coding vector $\vec{c}$.

As each forwarder node combines received packets to form new coded packets, pollution attacks can cause an epidemic effect, where the corrupted packets from one affected honest node further affect other honest nodes. As a result, by injecting even a few corrupted packets, the attacker can degrade the performance significantly. Existing defense techniques are generally heavy-weighted (see Section 2), leading to a significant negative impact on the protocol performance. Recently, a lightweight solution based on efficient linear checksums and time asymmetry was proposed [14]. However, designing a lightweight pollution defense for intra-flow coding systems in wireless networks that does not rely on time synchronization is still an open problem.

• **Packet dropping.** Intuitively, network coding systems should be resilient to packet dropping attacks due to the inherent redundancy in multi-path routing and opportunistic listening. However, in current protocols, the selection of forwarding nodes and rate assignment are carefully optimized to reduce interference and the total number of transmissions. As a side effect, this results in fragile systems that are sensitive to node misbehaviors, even as simple as packet dropping. As demonstrated by our experiments (Section 6), even a single packet dropping attacker can result in over 40% of degradation in performance for most flows. Addressing packet dropping attacks in network coding systems is more challenging than in traditional routing protocols, as the number of packets a node transmits and the time of transmissions are contingent on the opportunistic receptions at the node. As a result, traditional approaches, *e.g.*, watchdog [40], no longer apply.

4.2.3. Acknowledgment delivery

The timely and reliable delivery of ACK messages is critical to the performance of the protocol. This process is vulnerable to the following attacks.

• **ACK injection or modification.** The attacker forges a bogus ACK or modifies an ACK packet causing the source to move onto the next batch prematurely. As a result, the destination may receive only partial batches, and consequently is not able to decode any data packets. Such attacks can be prevented with message authentication, such as digital signatures.

• **ACK dropping.** If the attacker node lies on the ACK delivery path, it can drop all the ACK packets. This can prevent the source node from advancing through batches and cause it to keep transmitting one batch of packets forever. An attacker can enhance their chance of being selected on the ACK delivery path by manipulating path metrics or using wormhole attacks.

• **ACK delay.** The attacker node delays the delivery of ACK packets, instead of dropping ACK packets completely. This attack is more stealthy than ACK dropping attacks and can also cause a significant throughput degradation, as it prolongs the time required for sending a batch of packets.

4.3. Inter-flow Network Coding

We next analyze each of the components of inter-flow coding systems for potential security vulnerabilities.

4.3.1. Discovery of coding opportunities

The correct discovery of coding opportunities at a node relies on the correct packet reception information that the node collects from other nodes. The process of collecting the packet reception information of other nodes is subject to various types of attacks for both localized and global coding protocols as follows.

• **Packet reception information mis-reporting.** In localized coding protocols (*e.g.*, COPE [31]), an attacker can impersonate honest nodes and report incorrect packet reception information to their neighbors. Such an attack can cause a node to send coded packets that cannot be decoded by the intended next hop nodes. Since such non-decodable packets cannot be acknowledged, it will cause the sender node to continuously transmit useless packets. This attack can be addressed with message authentication schemes. However, given the high frequency of packet reception reports, the authentication scheme needs to be extremely lightweight.

• **Link state pollution.** Localized coding protocols also rely on the link quality between nodes to infer packet reception status at other nodes. Therefore, attacks on link state routing protocols which cause incorrect link state graph at nodes can also cause a node to infer incorrect packet reception information and consequently send non-decodable packets.

• **Neighbor set pollution.** In global coding systems (*e.g.*, DCAR [37]), a node determines coding opportunities based on the neighboring node set information collected during the route discovery process. An attacker can cause the collection of incorrect neighboring node set information either by direct modifications of route request packets or by using wormholes to introduce fictitious links. The resulting incorrect neighboring node set can cause a node either to miss coding opportunities, or worse yet, to send coded packets that cannot be decoded by downstream nodes, which can potentially degrade the throughput of the targeted flow to zero. Existing approaches for secure source routing protocols, such as Ariadne [27], can be used to authenticate the neighboring node information, and prevent malicious modifications. However, techniques for defending wormhole attacks are also necessary for securing the neighbor set information.

4.3.2. Transmission of coded packets

The packet transmission process in inter-flow coding systems is also subject to various types of attacks as follows.

• **ACK injection or modification.** By injecting bogus ACKs or modifying ACKs, the attacker node can cause premature ending of necessary packet retransmissions in the pseudo-broadcast technique, resulting in the failure of packet reception at next hop nodes. Again, message authentication schemes can be used as a counter-measure; however, due to the high frequency of ACK messages, it is crucial that the scheme selected is efficient in terms of both computation and bandwidth.

• **Packet pollution.** Like for intra-flow coding schemes, in a packet pollution attack against inter-flow coding systems, the attacker node injects corrupted packets into the network. However, the definition of corrupted packets is different for inter-flow coding, given the nature of inter-flow coding systems which perform coding across different flows. A packet e is *corrupted* if it is labeled as coded from packets $m_1, m_2, \ldots, m_k$, but $e \neq m_1 \oplus m_2 \oplus \ldots \oplus m_k$, where each packet m_i belongs to a different flow i. Note that this definition includes the case when a coded packet e is labeled as being coded from a single packet m_i (generated by the source of flow i), but $e \neq m_i$.

By injecting only a few corrupted packets an attacker can cause epidemic corruption of packets. The devastating effect of a pollution attack becomes evident if we consider corrupted packets that reach coding nodes; as these nodes are at the intersection of several flows, the corruption will propagate downstream on all the flows that intersect at the coding nodes. Existing pollution defense schemes proposed for intra-flow coding, such as homomorphic signatures, cannot be applied in this context, as coded packets are formed from packets generated by different sources. Defending against pollution attacks in inter-flow coding systems is still an open problem.

• **Packet over-coding.** Coding nodes are supposed to code packets from different flows as specified by the discovery of coding opportunities phase. This ensures that downstream nodes will be able to perform decoding correctly. However, a malicious coding node can execute an over-coding attack by coding together more packets than it is supposed to. For example, consider a malicious coding node that is at the intersection of three flows and is supposed to code packets from two of the three flows; however, the node codes together packets from all the three flows, causing incorrect decoding at the nodes downstream.

The difficulty of defending against such an attack comes from the fact that the packets forwarded by the attacker do not match the definition of a corrupted packet. Thus, even if a defense scheme was able to detect polluted packets, it would still not be effective against packet over-encoding attacks. An effective defense would have to incorporate information from the discovery of coding opportunities phase and would have to ensure that the established coding opportunities are enforced. Finally, we note that the packet over-coding attack is specific to inter-flow network coding systems.

• **Packet under-decoding.** This attack is similar to a packet over-coding attack, but is performed by a decoding node. The discovery of coding opportunities phase relies on the fact that decoding nodes use a certain number of their overheard packets to decode (or partially decode) the coded packets they receive. However, a malicious decoding node may use less overheard packets for decoding than it is supposed to. As a result, the packets forwarded by the malicious node cannot be decoded by its downstream nodes.

Just like the packet over-coding attack, the packet under-decoding attack is specific to inter-flow network coding systems and will not be detected by schemes that only protect against packet pollution.

• **Packet dropping.** Compared to traditional routing protocols, inter-flow coding systems encourage path sharing in an effort to increase coding opportunities. By exploiting such a tendency in the path selection, an attacker can manage to be selected by many paths, and hence can disrupt the communication of many flows via packet dropping. We differentiate between packet dropping performed by forwarding nodes and coding/decoding nodes. When a forwarding node drops packets, this can be detected with traditional defense techniques such as watchdog [40], because the node is supposed to forward the same packets it has received. However, the watchdog technique no longer applies when a coding/decoding node drops packets. For example, consider a malicious coding node that is at the intersection of three flows is supposed to code packets from the three flows; however, it drops packets from one of the flows and only codes packets from the other two flows. In this example, its upstream nodes cannot use overhearing to verify the correctness of the forwarded coded packets, because they cannot decode these packets.

4.3.3. Routing integration

In order to select an optimal route that considers coding opportunities, a coding-aware routing protocol not only requires the correctness of link and path metrics as in traditional routing protocols, but it also requires the correctness of coding benefits reported by each node. Thus, in addition to manipulating link and path metrics, an attacker node can disrupt the protocol by manipulating coding opportunities. For example, by reporting very high coding opportunities, the attacker can improve its chances to be selected on the path and gain the control over the flow. Since coding opportunities not only depend on the network topology, but also depend on the current flow structure, it is more challenging to ensure the correctness of coding opportunities reported by a node than ensuring the correctness of pure topological metrics (*e.g.*, link or path qualities).

5. Research Challenges and Defense Directions

In this section we discuss possible security goals for network coding systems and the numerous challenges for designing secure network coding for wireless systems.

5.1. Security Goals

The general security objectives of any protocol are data confidentiality, data authentication, data integrity, data freshness, data availability, and graceful performance degradation. The main goal of integrating the network coding paradigm into a wireless networked system is to improve the system's throughput. Thus, from the perspective of network coding, the main security goal is the data availability and graceful performance degradation. In other words, the goal is to be able to maintain the high level of performance provided by network coding systems even in the presence of attackers. The other security goals such as data confidentiality, integrity, freshness and authentication have been extensively studied for many protocols and are more appropriately addressed at the upper layers.

Data availability. Although network coding has been shown to improve the system performance significantly [8, 31, 50, 49], a side-effect of the performance improvement is the more complex system design and numerous new vulnerabilities that may be exploited by attackers. Based on the experience with previous systems, caution should be exercised when switching to an adversarial environment, especially for systems designed to optimize performance. For example, high-throughput routing metrics [10, 5, 3, 33, 16] can significantly improve throughput compared to the traditional hop-count metric. However, in the presence of attackers, a high-throughput metric favors aggressive path selection and acts as an attack amplifier, allowing the attacker to completely control and disrupt the traffic for a large number of flows in the network [13]. For network coding, due to the use of packet coding, a similar effect of large scale disruption is also observed for packet pollution attackers (see Section 6). Thus, an important security goal for network coding systems is to eliminate large scale attacks and maintain the benefit of increased performance even under an adversarial environment.

Graceful degradation. In many unsecure wireless networked systems, when the adversarial presence increases, the performance of the system decreases. For example, if we consider the routing problem, throughput decreases linearly with the number of attackers for simple attacks (*e.g.*, black hole attack with randomly-placed adversaries), and it decreases much more sharply for more sophisticated attacks (*e.g.*, wormhole attacks with strategically-placed adversaries) [4, 11]. A similar effect was observed for pollution attacks in intra-flow network coding systems [14]. However, the impact of the attacks on the routing protocol is drastically reduced when security mechanisms are applied on top of the unsecure system. We expect that provisioning a network coding system with security mechanisms will also significantly reduce the impact of attacks. Indeed, this was shown to be true for pollution attacks in intra-flow network coding systems [14]. Therefore, another important security goal is to ensure at most a *graceful degradation* in performance as the number of attackers increases.

We summarize the security goals that are desirable for a secure wireless network coding system as follows:

1. Under an adversarial environment, a secure network coding system should maintain its benefits of increased performance when compared to the system not using network coding. In other words, the overhead of the security mechanisms and the impact of attacks should not negate the performance gains of network coding.
2. The performance of a secure network coding system should be resilient to an increase in the number of attackers. In other words, the performance of the protocol should remain high in the presence of a few attackers and degrade gradually as the number of attackers increases.

5.2. Challenges

In this section we discuss some of the major challenges involved in creating defense mechanisms for attacks against network coding systems and derive several guiding principles for designing secure network coding systems. We focus only on general design guidelines that are applicable across different network coding systems. Designing concrete security schemes for specific network coding systems is beyond the scope of the paper.

Given the complexity of existing network coding systems and the numerous security vulnerabilities, instead of patching existing systems, a more fundamental approach is to design new network coding protocols with security consideration in the first place. Such protocols may be less optimized in performance compared to existing protocols, however, the security guarantees provided can make them attractive choices for adversarial environments. Based on a well-known principle for designing secure systems which states that security should be considered during the design process and not as an add-on once the system has already been designed, we define the following guiding principle applied to network coding:

Principle 1. *The security of the network coding system should not be considered in isolation of the system itself.*

The trust assumptions between each component of the network coding system should be clearly stated and information about malicious behavior detected in one component should be shared across other components. For example, if a node has been identified as a polluting attacker, then it should not be included in the forwarding node selection phase.

One of the design challenges for a secure coding system is that the different components of the network coding system are very dependent on each other and attacks against different components will have a similar impact on the system, even though the effort of the adversary may vary substantially. Based on the well-known general security principle that states that a system is as secure as its weakest link, we derive the following guiding principle:

Principle 2. *Each one of the components in a network coding system should have a relatively close attack resilience.*

This principle advocates that each component should be equally-well defended to prevent attackers from exploiting the weakest link in the system. For example, an intra-flow coding system may be immune to pollution attacks, but if the ACK delivery component is not secure, then the whole system becomes very vulnerable.

It is a formidable task to design a defense scheme that pro-actively prevents all the aforementioned threats and is sufficiently lightweight such that the performance gain of network coding is preserved. Several proactive solutions proposed for network coding rely on expensive cryptographic mechanisms. While such solutions have the benefit of detecting attackers, they significantly degrade the performance of the system when no attacks are present. This leads us to derive the next guiding principle:

Principle 3. *When employing proactive mechanisms, the performance of the secure system under normal conditions (i.e., no adversaries) should be comparable with the performance of the unsecure system under the same conditions.*

This principle states that the overhead of the protection mechanism should be minimal, so that the performance of the system is not significantly affected when no adversaries are present in the system. Although this principle may seem obvious, it is not followed by many of the existing work (e.g. [9, 39, 52, 47]) that propose proactive cryptographic solution for addressing pollution attacks in intra-flow network coding systems, since they focus primarily on the security of the proposed scheme without regard to their impact on the system performance. A further implication of this principle is that although the security of scheme may be proved analytically, it is generally necessary to evaluate its overhead on the system to ensure acceptable performance impact by the defense scheme itself.

Several other challenges stem from the unique aspects of network coding principle and wireless medium characteristics. A major challenge that secure network coding designers face is due to the fundamental different role that intermediate nodes play. In a typical one-to-one communication, end-to-end security mechanisms can be used to achieve guarantees such as confidentiality and authenticity. This may lead the source and destination nodes to detect the presence of an attack and take appropriate measures. However, relying exclusively on end-to-end attack detection is ineffective for a network coding system because by the time the attack is detected, the performance advantage of network coding is cancelled. This problem is even more exacerbated in a wireless environment where bandwidth is a scarce resource. Thus, a network coding system cannot exclusively rely on end-to-end security mechanisms to achieve the additional goal of increased performance compared a system that does not perform network coding.

Network coding, particularly when combined with other optimization techniques, creates an environment where honest nodes can be participants in an attack, even without their knowledge. The identification and isolation of attackers is exacerbated by the epidemic nature of attacks like pollution, which causes honest nodes affected by the attack to exhibit attacker-like behavior. For example, a node (a receiver in the case of intra-flow network coding or even an intermediate node in the case of inter-flow network coding) can detect after decoding that a packet was corrupted, but it may not be able to immediately identify the attacker as one of the neighbors since the corruption may have been propagated by honest nodes without their knowledge. A promising direction in this case is to design schemes that combine light-weight proactive monitoring with reactive defense schemes, where nodes take corrective actions only when abnormal conditions occur.

The wireless transmission medium has particular features that make it appropriate for network coding. Many of the systems shown to be practical were specifically designed for a wireless setting. However, general defenses for attacks against network coding are not always appropriate in wireless environments. For example, many cryptographic mechanisms proposed for peer-to-peer systems have prohibitive cost in wireless networks, and information theoretic approaches which try to add enough redundancy make assumptions not valid in wireless networks. Due to the constrained bandwidth of the wireless medium, there is a long term benefit in detecting the presence of the attackers and defenses should be customized for such networks appropriately.

Finally, wireless network coding requires a larger extent of cooperation among nodes than traditional routing protocols. Intermediate nodes not only have to cooperate in the more complex path selection and packet forwarding process, but they also need to actively participate in the packet coding and decoding process. In a network environment where participant nodes belong to different parties, nodes may execute the attacks for selfish reasons. For example, nodes may manipulate the link quality to divert traffic away from themselves or drop packets to save energy consumption. In such an environment, incentive and cooperation enforcement mechanisms such as [53, 19] can provide a first line of defense in discouraging such behaviors and encouraging cooperation in the network. However, packet dropping and link quality falsification can also be due to malicious intent and many of the attacks presented in Section 4, such as packet pollution, are inherently malicious. Thus, defense mechanisms for such attacks are still necessary to ensure the correct network functions.

6. Experimental Evaluation

We present simulation results for evaluating the severity of security threats in network coding systems. Although network coding systems are vulnerable to a myriad of threats, we focus on attacks that target the coding, decoding, and the packet forwarding process, as these attacks are more system independent and more specific to network coding systems.

6.1. Methodology.

We conducted our experiments on the Glomosim [1] simulator. Below we describe the detailed experimental setup for both intra-flow and inter-flow network coding systems.

Experiment setup for intra-flow network coding. We use a representative intra-flow coding protocol, MORE [8], for our experiments. We use the setup for MORE as recommended by [8]: $\mathbb{F}_{2^8}$ as the finite field for network coding, the batch size is 32 packets, and the packet size is 1500 bytes. For each experiment, we select a source node and a destination node at random. The average throughput graph is plotted as the average of 20 different random runs. The throughput CDF graph is generated from 100 random pairs of source and destination nodes.

We enhance Glomosim to use a trace-driven physical layer based on traces from the real-world link quality measurements in the MIT Roofnet [2], an experimental 802.11b/g mesh network of 38 nodes widely used in research papers [20, 10]. We use 802.11 MAC with 5.5Mbps raw bandwidth and 250 meter nominal range.

We examine the following three attack scenarios that represent the main threats to the packet forwarding process of intra-flow network coding systems:

- Pollution: attacker nodes inject corrupted coded packets.
- Drop-Data: attacker nodes drop data packets.
- Drop-ACK: attacker nodes drop ACK packets; data packets are delivered normally.

In all three types of attacks, the attacker nodes are selected at random among all forwarding nodes.

Experiment setup for inter-flow network coding. As existing inter-flow network coding systems vary significantly on the process used for the discovery of coding opportunities and the specific coding conditions being used, we implement a general inter-flow coding protocol, referred to as ICODE. Since we focus on the attacks on coding, decoding, and packet forwarding process, ICODE is implemented in an off-line manner using the global knowledge of the network. Specifically, we first select a set of source and destination pairs, and establish a path between each pair using the ETX metric [10] . Then each node on a path is examined for coding opportunities. At runtime, each node forwards or codes packets according to the off-line analysis results. Thus, ICODE utilizes all possible coding opportunities with global knowledge. In this respect, ICODE encompasses all existing protocols, including [30, 31, 37, 12, 15, 42, 41].

As inter-flow network coding requires the participation of multiple flows, we use a larger network of size 100 nodes placed uniformly at random in a 1500 m by 1500 m area. Up to 30 nodes are randomly selected among all the nodes to be attacker nodes. A variable number of flows are established between randomly selected source and destination pairs. The total offered load of all the flows is kept at 1 Mbps, which is sufficient for full utilization of the network bandwidth without causing over-congestion. We experimented with different numbers of flows. However, due to lack of space, we only present results for the case of 25 flows. (The results for the other cases are similar.)

We examine the following attacks that are the main threats to the packet forwarding process of inter-flow network coding systems:

- Pollution: attacker nodes on selected paths inject corrupted (plain or coded) packets to their downstream nodes.
- Over-coding: malicious coding nodes inject over-coded packets to their downstream nodes.
- Under-decoding: malicious decoding nodes refuse to decode packets that they are supposed to decode; other packets are forwarded normally by the node.
- Drop-Data: attacker nodes on selected paths drop data packets.

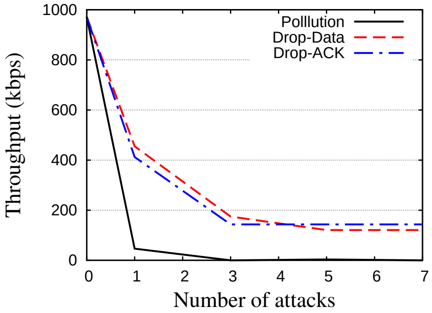


Figure 5: Average throughput under multiple packet dropping attackers.

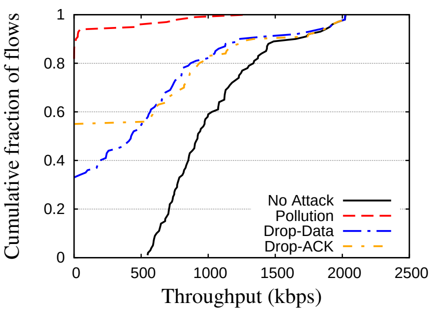


Figure 6: Throughput CDF under single packet dropping attacker.

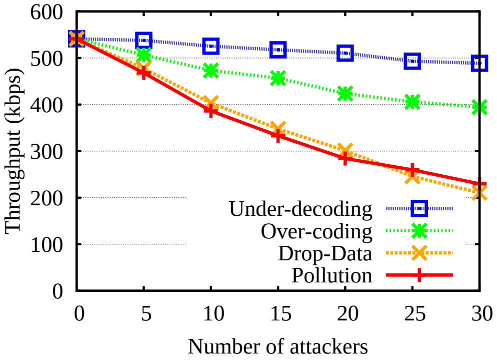


Figure 7: Total network throughput under different number of attackers for an inter-flow coding system.

6.2. Results for Intra-flow Network Coding

Fig. 5 shows the average throughput as we vary the number of packet dropping attackers from 1 to 7. We see that the throughput drops rapidly as the number of attackers increases for all three attack types. In particular, we see that with even a single attacker, both Drop-Data and Drop-ACK attacks lower the throughput to around half of the throughput when there is no attack. Pollution attack causes an even more dramatic degradation of the throughput: A single attacker causes the throughput to drop to less than 10% of the throughput under a benign environment. For all three types of attacks, the impact of the attacks levels out when the number of attackers goes beyond 3. This is an artifact of the random selection of the source and destination for each experiment. When the source and destination selected are immediate neighbors, none of the above attacks can impact the delivery of data. Such cases maintain the average throughput at a constant level, when the attacks cause all the other cases to have zero throughput.

To further analyze the impact of the attacks on different flows, Fig. 6 shows the throughput CDF for the case of a *single* attacker. We first see that the Pollution attack causes over 90% of flows to have zero throughput. This demonstrates the extreme danger of this attack on intra-flow coding systems due to the epidemic effect of packet corruption. For Drop-Data and Drop-ACK attacks, the attack impact is relatively small, but still a significant percentage of flows experience zero throughput: 35% and 50% for Drop-Data and Drop-ACK attacks, respectively. Such cases occur when the single attacker node happens to be the critical forwarder node that every packet has to pass through. Therefore, contrary to the common belief of intra-flow network coding systems being robust to packet dropping attacks, practical systems are actually quite fragile, a side effect of performance optimization algorithms that aim to reduce interference by minimizing the forwarder node set size and forwarding rates. The reason that Drop-ACK inflicts a higher level of damage than Drop-Data is that the ACK packet is delivered via single path routing which is more vulnerable to packet dropping than the data packet delivery which uses multi-path routing.

6.3. Results for Inter-flow Network Coding

Fig. 7 shows the total network throughput for Pollution, Over-coding, Under-decoding and Drop-Data attacks for different number of attackers. We first observe that both Pollution and Drop-Data cause a steady decrease in the network throughput as the number of attackers increases. Unlike in intra-flow network coding where pollution attacks are significantly more damaging than packet dropping attacks, for inter-flow network coding we observe that the effect of pollution attacks is similar to packet dropping attacks. This is because inter-flow network coding is based on single path routing, hence the presence of a packet pollution attacker on a flow causes the same effect as a packet dropping attacker in that they both reduce the throughput of the flow to zero. However, we note that pollution attacks are significantly more challenging to defend against than packet dropping attacks, as it is difficult to verify the correctness of coded packets in inter-flow coding systems, whereas packet dropping attacks can be addressed with monitoring based solutions (*e.g.*, watchdog [40]).

Over-coding and Under-decoding attacks cause a milder degradation of throughput compared to pollution and dropping attacks. This is primarily due to the less number of *effective* attacker nodes in Over-coding and Under-decoding attacks, as only attacker nodes who are also coding nodes (for Over-coding attacks) or decoding nodes (for Under-decoding attacks) execute attack actions, while other selected attacker nodes behave like honest nodes. Comparing Over-coding and Under-decoding attacks, the impact of Under-decoding attacks is even lower than Over-coding attacks. This is because a Under-decoding attacker affects only the flow that the attacker is on, while for Over-coding attacks a malicious coding node can affect multiple flows simultaneously. A surprising result from the figure is the mild impact of Under-decoding attacks even with up to 30 attacker nodes. A careful inspection reveals that most decoding nodes only perform packet decoding occasionally, while performing packet forwarding most of the time. This is consistent with the observation in [35]. Thus, Under-decoding attacks affect only a small number of packets in the network.

7. Conclusion

Through detailed and systematic analysis of current network coding systems, we reveal that both intra-flow and inter-flow network coding systems are vulnerable to a wide range of attacks at various stages of the protocol. The use of coding techniques not only introduces new attacks, but also makes existing attacks more damaging and more challenging to defend against. Except for packet pollution, the security threats for network coding are largely unexplored, hence, provide exciting opportunities for future security research.

References

[1] Global mobile information systems simulation library - glomosim. `http://pcl.cs.ucla.edu/projects/glomosim/`.

[2] MIT roofnet - publications and trace data. `http://pdos.csail.mit.edu/roofnet/doku.php?id=publications`.

[3] A. Adya, P. Bahl, J. Padhye, A. Wolman, and L. Zhou. A multi-radio unification protocol for ieee 802.11 wireless networks. In *Proc. of BroadNets '04*, 2004.

[4] B. Awerbuch, R. Curtmola, D. Holmer, C. Nita-Rotaru, and H. Rubens. ODSBR: An on-demand secure Byzantine resilient routing protocol for wireless ad hoc networks. *ACM Transactions on Information and System Security*, November 2007.

[5] B. Awerbuch, D. Holmer, and H. Rubens. The medium time metric: High throughput route selection in multirate ad hoc wireless networks. *MONET, Spec. Iss. on Internet Wireless Access: 802.11 and Beyond*, 2005.

[6] D. Boneh, D. Freeman, J. Katz, and B. Waters. Signing a linear subspace: Signature schemes for network coding. In *Proc. of PKC '09*, 2009.

[7] N. Cai and R. W. Yeung. Network error correction, part ii: lower bounds. *Commun. Inf. Syst.*, 6(1):37–54, 2006.

[8] S. Chachulski, M. Jennings, S. Katti, and D. Katabi. Trading structure for randomness in wireless opportunistic routing. In *Proc. of ACM SIGCOMM '07*, 2007.

[9] D. Charles, K. Jain, and K. Lauter. Signatures for network coding. In *Proc. of CISS '06*, 2006.

[10] D. S. J. D. Couto, D. Aguayo, J. C. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proc. of ACM MOBICOM '03*, pages 134–146. ACM, 2003.

[11] R. Curtmola and C. Nita-Rotaru. BSMR: Byzantine-resilient secure multicast routing in multi-hop wireless networks. *IEEE Transactions on Mobile Computing*, April 2009.

[12] S. Das, Y. Wu, R. Chandra, and Y. C. Hu. Context-based routing: Technique, applications, and experience. In *Proc. of NSDI '08*.

[13] J. Dong, R. Curtmola, and C. Nita-Rotaru. On the pitfalls of using high-throughput multicast metrics in adversarial wireless mesh networks. In *Proc. of SECON '08*, June 2008.

[14] J. Dong, R. Curtmola, and C. Nita-Rotaru. Practical defenses against pollution attacks in intra-flow network coding for wireless mesh networks. In *Proc. of WiSec '09*, 2009.

[15] Q. Dong, J. Wu, W. Hu, and J. Crowcroft. Practical network coding in wireless networks. In *Proc. of MobiCom '07*, 2007.

[16] R. Draves, J. Padhye, and B. Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *Proc. of MOBICOM '04*. ACM, 2004.

[17] J. Eriksson, S. Krishnamurthy, and M. Faloutsos. Truelink: A practical countermeasure to the wormhole attack in wireless networks. In *Proc. of ICNP '06*, 2006.

[18] K. Fall. A delay tolerant networking architecture for challenged internets. In *Proc. of ACM SIGCOMM*, Aug. 2003.

[19] M. Felegyhazi, J.-P. Hubaux, and L. Buttyan. Nash equilibria of packet forwarding strategies in wireless ad hoc networks. *Mobile Computing, IEEE Transactions on*, 5(5):463–476, May 2006.

[20] C. Gkantsidis, W. Hu, P. Key, B. Radunovic, P. Rodriguez, and S. Gheorghiu. Multipath code casting for wireless mesh networks. In *CoNEXT '07*, 2007.

[21] C. Gkantsidis and P. Rodriguez Rodriguez. Cooperative security for network coding file distribution. In *Proc. of INFOCOM 2006*.

[22] M. Guerrero Zapata and N. Asokan. Securing Ad hoc Routing Protocols. In *WiSe*, 2002.

[23] Z. Guo, P. Xie, J.-H. Cui, and B. Wang. On applying network coding to underwater sensor networks. In *WUWNet '06: Proceedings of the 1st ACM international workshop on Underwater networks*, pages 109–112, New York, NY, USA, 2006. ACM Press.

[24] T. Ho, B. Leong, R. Koetter, M. Medard, M. Effros, and D. Karger. Byzantine modification detection in multicast networks using randomized network coding. In *Proc. of ISIT '04*.

[25] Y.-C. Hu, D. B. Johnson, and A. Perrig. SEAD: Secure efficient distance vector routing for mobile wireless ad hoc networks. In *WMCSA*, 2002.

[26] Y.-C. Hu, A. Perrig, and D. B. Johnson. Packet leashes: A defense against wormhole attacks in wireless ad hoc networks. In *INFOCOM*, 2003.

[27] Y.-C. Hu, A. Perrig, and D. B. Johnson. Ariadne: a secure on-demand routing protocol for ad hoc networks. *Wirel. Netw.*, 11(1-2):21–38, 2005.

[28] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, and M. Medard. Resilient network coding in the presence of byzantine adversaries. In *Proc. of INFOCOM '07*, 2007.

[29] D. B. Johnson, D. A. Maltz, and J. Broch. Dsr: The dynamic source routing protocol for multi-hop wireless ad hoc networks. In *In Ad Hoc Networking, edited by Charles E. Perkins, Chapter 5*, pages 139–172. Addison-Wesley, 2001.

[30] S. Katti, D. Katabi, W. Hu, H. Rahul, and M. Médard. The importance of being opportunistic: Practical network coding for wireless environments. In *In Proc. of Allerton Conference*, 2005.

[31] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft. Xors in the air: practical wireless network coding. In *Proc. of ACM SIGCOMM '06*, 2006.

[32] E. Kehdi and B. Li. Null keys: Limiting malicious attacks via null space properties of network coding. In *INFOCOM*, 2009.

[33] S. Keshav. A control-theoretic approach to flow control. *Proc. of the Conference on Communications Architecture and Protocols*, 1993.

[34] R. Koetter and F. R. Kschischang. Coding for errors and erasures in random network coding. *Information Theory, IEEE Transactions on*, 2008.

[35] D. Koutsonikolas, Y. C. Hu, and C.-C. Wang. An empirical study of performance benefits of network coding in multihop wireless networks. In *INFOCOM*, 2009.

[36] M. Krohn, M. Freedman, and D. Mazieres. On-the-fly verification of rateless erasure codes for efficient content distribution. In *Proc. of Symposium on Security and Privacy*, 2004.

[37] J. Le, J. C. S. Lui, and D. M. Chiu. DCAR: Distributed coding-aware routing in wireless networks. In *Proc. of ICDCS '08*, 2008.

[38] S.-H. Lee, U. Lee, K.-W. Lee, and M. Gerla. Content distribution in vanets using network coding: The effect of disk i/o and processing o/h. In *Proc. of SECON '08*, 2008.

[39] Q. Li, D.-M. Chiu, and J. Lui. On the practical and security issues of batch content distribution via network coding. In *Proc. of ICNP '06*, Nov. 2006.

[40] S. Marti, T. Giuli, K. Lai, and M. Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *Proc. of MOBICOM*, August 2000.

[41] B. Ni, N. Santhapuri, Z. Zhong, and S. Nelakuditi. Routing with opportunistically coded exchanges in wireless mesh networks.

[42] S. Omiwade, R. Zheng, and C. Hua. Practical localized network coding in wireless mesh networks.

[43] K. Sanzgiri, B. Dahill, B. N. Levine, C. Shields, and E. M. Belding-Royer. A secure routing protocol for ad hoc networks. In *ICNP 2002*.

[44] D. Silva, F. Kschischang, and R. Koetter. A rank-metric approach to error control in random network coding. *IEEE Inf. Theory for Wireless Ntwks*, 2007.

[45] D. Wang, D. Silva, and F. R. Kschischang. Constricting the adversary: A broadcast transformation for network coding. In *Allerton 2007*, 2007.

[46] R. W. Yeung and N. Cai. Network error correction, part i: basic concepts and upper bounds. *Commun. Inf. Syst.*, 6(1):19–36, 2006.

[47] Z. Yu, Y. Wei, B. Ramkumar, and Y. Guan. An efficient signature-based scheme for securing network coding against pollution attacks. In *Proceedings of INFOCOM 08*, 2008.

[48] Z. Yu, Y. Wei, B. Ramkumar, and Y. Guan. An efficient signature scheme for securing xor network coding against pollution attacks. In *INFOCOM*, 2009.

[49] X. Zhang and B. Li. DICE: a game theoretic framework for wireless multipath network coding. In *Proc. of Mobihoc 2008*.

[50] X. Zhang and B. Li. Optimized multipath network coding in lossy wireless networks. In *Proc. of ICDCS '08*, 2008.

[51] X. Zhang, G. Neglia, J. Kurose, and D. Towsley. On the benefits of random linear coding for unicast applications in disruption tolerant networks. In *In Proc. of IEEE Network Coding (NETCOD) Workshop*, Apr. 2006.

[52] F. Zhao, T. Kalker, M. Medard, and K. Han. Signatures for content distribution with network coding. In *Proc. of ISIT '07*.

[53] S. Zhong, L. E. Li, Y. G. Liu, and Y. R. Yang. On designing incentive-compatible routing and forwarding protocols in wireless ad-hoc networks. *Wireless Networks*, 13(6):799–816, December 2007.