SPAIS: A Novel Self-checking Pollution Attackers Identification Scheme in Network Coding-based Wireless Mesh Networks

Donghai Zhu^a, Xinyu Yang^a, Wei Yu^b

^a*Dept. of Computer Science and Technology, Xi'an Jiaotong University, Xi'an, China.*

^b*Towson University, Towson, MD 21252, USA.*

Abstract

Pollution attacks refer to ones where attackers modify and inject corrupted data packets into the wireless network with network coding to disrupt the decoding process. In the context of network coding, the epidemic effect of pollution attacks can degrade network throughput significantly because of the mixing nature of network coding. To address this issue, a number of schemes to identify malicious nodes have been developed in the past. Nonetheless, these schemes have their limitations and cannot effectively deal with pollution attacks. In this paper, we propose a novel light-weight Self-checking Pollution Attackers Identification Scheme (SPAIS), which can identify the pollution attackers effectively and efficiently. By making full use of the broadcast nature of wireless media and the insight that a well-behaved node can monitor its downstream neighboring nodes locally by cooperating with other nodes, SPAIS hierarchically organizes the network as levels such that the nodes in the same level can monitor their downstream level nodes

Email addresses: Dr.zhudonghai@stu.xjtu.edu.cn (Donghai Zhu),
xyypd@mail.xjtu.edu.cn (Xinyu Yang), wyu@towson.edu (Wei Yu)

Preprint submitted to Computer Networks

March 16, 2015

cooperatively. Through the combination of theoretical analysis and extensive simulations, our experimental data demonstrates that SPAIS can more effectively identify pollution attackers with a lower cost in comparison with the existing candidate schemes. For example, even if the quality of network connections is not in good condition and the malicious nodes send only one corrupted packet, the pollution attackers can be identified with a high probability.

Keywords: network coding, pollution attack, wireless mesh network, identification.

1. Introduction

Wireless mesh networks (WMNs) have become a promising technology for providing easy wireless Internet access because of highly self-organized and self-configured features of WMNs [1]. Nonetheless, the wireless links between the nodes is lossy due to the temporal and spatial fading of wireless channels, leading to a distributed WMN infrastructure and relatively low performance. To this end, network coding [2, 3], which is a technique that integrates the coding and routing schemes, has been introduced to improve the performance of WMNs. Generally speaking, in contrast to the traditional “store and forward” method, network coding allows relay nodes to mix the information content in packets before forwarding them, which can improve the network throughput significantly. Network coding has been applied to various applications, including the content distribution [4], P2P networks [5], etc. Existing research efforts have shown the great advantage of network coding in wireless networks (e.g., the COPE [6], the MORE [3], etc.)

Network coding offers an appealing feature to construct networks. Nonetheless, because of the mixing nature, network coding is vulnerable to pollution attacks. Through pollution attacks, malicious nodes can alter or forge some corrupted messages and inject them into the network. The impact of pollution attacks can be devastating because with an epidemic propagation, the corrupted packets can pollute the whole network in a short time and degrade the network throughput dramatically [7].

In recent years, a number of authentication schemes have been developed to address pollution attacks against network coding [8–14]. In addition, a number of identification schemes [15–19] have been proposed to identify and locate the pollution attackers. In comparison with the authentication schemes, identification schemes are more proactive in defending pollution attacks. Because authentication schemes are all receivers-based, the detection may be too late as the attack may already consume the bandwidth and significantly disrupt network performance. If attackers can be quickly identified and immediately excluded from the network, the attack consequence can be largely mitigated.

Nonetheless, the existing identification schemes have their limitations and cannot be used to effectively deal with pollution attacks in WMNs where network coding is used. For example, Watchdog scheme [15] relies on a large number of extra trusted nodes to serve as watchdogs, which incurs a high cost to large networks. In MIS [16], a trusted server is responsible for computing the checksum of suspicious packets. The checksum transmission incurs a significant communication overhead. In [18], Algebraic-Watchdog scheme develops the hypothesis testing method to make the decision of identifying

a malicious node. Nonetheless, that scheme needs to collect a large number of detecting samples and its computation cost is high. Similar to MIS [16], SpaceMac [19] relies on a trusted controller to identify the malicious nodes. In addition, SpaceMac and PAI [17] do not consider the lossy wireless environment, which leads to the limited applicability.

In this paper, we propose a novel light-weight Self-checking Pollution Attackers Identification Scheme (SPAIS) to effectively and efficiently identify the pollution attackers against network coding based legacy WMNs where MIMO (Multiple-Input and Multiple-Output) systems and directed antennas are not in use. Our proposed scheme has several novelties. First, unlike existing research efforts, SPAIS does not need any extra nodes to serve as watchdogs or any centralized server or controller. Second, it is self-checking and fully distributed. Third, it is light-weight and incurs only a little bandwidth overhead on the system. **Moreover, network topology can be dynamic. If the identification is triggered and the topology has changed during data transmission, the source can re-determine the network level and re-distribute the random seeds to the corresponding nodes to ensure the secrecy of the random seeds.** Our main contributions are summarized as follows:

- **Novel Scheme.** In our design of SPAIS, we make full use of the broadcast nature of wireless media and adopt the principle that a well-behaved node can monitor its downstream neighbor nodes locally by cooperating with other nodes. To this end, our SPAIS hierarchically organizes the network as levels and the nodes in the same level can cooperatively monitor the nodes in the downstream level. To further

reduce the bandwidth overhead, we develop a homomorphic hash function for network coding.

- **Theoretical Analysis.** We conduct theoretical analysis and prove that our scheme is correct and secure. For example, under typical system settings, our analytical data shows that SPAIS allows only 1 out of 65536 polluted packets passing the verification of the upstream level of the attacker if the network is lossless.
- **Evaluation.** We evaluate the effectiveness of SPAIS in terms of false positive probability and false negative probability through simulation experiments. The simulation data shows that even if in a wireless network where the average link quality is only 0.7 and the attacker sends out only one corrupted packet to launch a pollution attack, the attacker can be located with a probability of 95.62%. In addition, even if there are 30 percent of malicious nodes in the network, the probability that the malicious nodes can disparage a well-behaved node is only 1.45%. We also compare the computation overhead and bandwidth overhead with the existing schemes. The experimental data confirms that our scheme is more efficient than existing schemes.

The remainder of the paper is organized as follows: We introduce the network and threat models in Section 2. We present our proposed scheme in Section 3. We conduct the security analysis of our scheme in Section 4. In Section 5, we show both analytical and experimental results to validate the effectiveness of our proposed scheme in comparison with existing schemes. We discuss the limitation and the potential future work in Section 6. We

review the related work in Section 7 and conclude the paper in Section 8, respectively.

2. System and Threat Models

In this section, we first introduce the system model and then present the threat model.

2.1. System Model

In this paper, we consider a typical multicast scenario in WMNs modeled as a directed acyclic graph $G = (V, E)$, in which a source $\mathcal{S}$ sends a multicast stream of packets to a set of receivers $\mathcal{R}$. We focus on the backbones of legacy WMNs where MIMO systems and directed antennas are not in use, which consists of stationary mesh routers and mesh gateways. For more information of WMNs, please refer to the article [1]. We assume that all nodes in the network perform the generations-based random linear network coding that is well adopted and described in [20].

$\mathcal{S}$ generates a stream of packets, and divides them into generations. Each generation consists of m packets $\underline{\mathbf{x}}_1, \underline{\mathbf{x}}_2, \dots, \underline{\mathbf{x}}_m$, and $\underline{\mathbf{x}}_i$ is represented as a vector of $(\underline{x}_{i,1}, \underline{x}_{i,2}, \dots, \underline{x}_{i,n})$ of finite field $\mathbb{F}_q^n$. Here, q is a prime of a proper size, and all the arithmetic operations are conducted through $\mathbb{F}_q$. In addition, $\mathcal{S}$ generates an augmented packet $\mathbf{x}_i$ by prefixing $\underline{\mathbf{x}}_i$ with the i^{th} unit vector of dimension m and $\mathbf{x}_i$ is denoted as

$$\mathbf{x}_i = \left(\underbrace{0, \dots, 0}_{i-1}, \underbrace{1, 0, \dots, 0}_m, \underline{x}_{i,1}, \underline{x}_{i,2}, \dots, \underline{x}_{i,n} \right). \quad (1)$$

For the random linear network coding, $\mathcal{S}$ sends the linear combination of packets $\mathbf{x}_i$ and the coefficients are randomly selected from the finite field $\mathbb{F}_q$.

That is to say, for packets $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m$ and random coefficients $\alpha_1, \alpha_2, \dots, \alpha_m$, an output packet is represented as

$$\mathbf{y} = \sum_{i=1}^m \alpha_i \mathbf{x}_i = \left(\sum_{i=1}^m \alpha_i x_{i,1}, \dots, \sum_{i=1}^m \alpha_i x_{i,m+n} \right). \quad (2)$$

The first m symbols of $\mathbf{y}$ are referred to as the *global coding vector* $\boldsymbol{\alpha}$. After receiving l ($1 \leq l \leq m$) packets $\mathbf{y}_j, j = 1 \dots l$, forwarders randomly select their coefficients known as the *local coding vector* $\boldsymbol{\beta}$, and perform linear combination operation on the received packets using the local coding vector in a similar way with the source:

$$\mathbf{z} = \sum_{j=1}^l \beta_j \mathbf{y}_j = \left(\sum_{j=1}^l \beta_j y_{j,1}, \dots, \sum_{j=1}^l \beta_j y_{j,m+n} \right). \quad (3)$$

After that, the coded packets $\mathbf{z}$ is transmitted to other forwarders.

The input of the linear combination $\mathbf{y}_j, j = 1, \dots, l$ is referred to as the *input packets* and the output $\mathbf{z}$ is referred to as the *output packet*. Then, a receiver $\mathcal{R}$ can recover the original packets $\mathbf{x}_i, i = 1 \dots m$ using the global coding matrix $\mathbf{G}$, where i^{th} row is the corresponding global coding vector through *Gaussian* eliminations with a high probability after receiving m linearly independent packets $\mathbf{y}_j, j = 1, \dots, m$:

$$[\mathbf{I}, \mathbf{X}] = \mathbf{G}^{-1} \cdot \mathbf{Y}. \quad (4)$$

Here, $\mathbf{I}$ is the identity matrix, $\mathbf{X}$ is the matrix, in which i^{th} row is the corresponding $\mathbf{x}_i$, $\mathbf{G}^{-1}$ is the inverse of the matrix $\mathbf{G}$, and $\mathbf{Y}$ is the matrix, in which i^{th} row is the corresponding $\mathbf{y}_i$. Note that the random network coding is only performed among the packets belonging to the same generation.

2.2. Threat Model

Generally speaking, pollution attacks refer to as ones, in which the attacker modifies and injects corrupted data packets into the network deployed with network coding in order to disrupt the decoding process [12]. In particular, the attacker can compromise some nodes in the network and attempt to inject corrupted packets into the traffic flow or manipulate packets going through nodes that they control, aiming to corrupt data transmission and degrade network performance (e.g., transmission delay, throughput, etc.). In this paper, we focus on the pollution attacks, which are a generic threat to the networks where network coding is used. We do not consider other types of attacks (e.g., traffic analysis and eavesdropping). We also do not consider attacks on the physical or MAC layer. Defending against those attacks is complementary to our work as well.

We assume the source is trusted and is not compromised, following the assumption made by the most existing schemes [8–19]. Nonetheless, the forwarders can be adversarial. Adversarial nodes launch the pollution attack either by injecting bogus packets or by modifying their output packets to contain incorrect data. We say that a packet $\mathbf{z}_j$ is a corrupted packet, if the following equation does not hold:

$$\mathbf{z}_j = \left(\sum_{k=1}^m \alpha_{j,k} x_{k,1}, \dots, \sum_{k=1}^m \alpha_{j,m+n} x_{k,m+n} \right). \quad (5)$$

We assume that the attacker does not have access to the randomness used by the source in order to produce the random seed, which is used for one-way hash chain function. The detail can be found in Section 3.2. We also assume that there is a decoded packet format checking or an end-to-end

message authentication scheme for network coding (e.g., [11]), which allows each receiver $\mathcal{R}$ to efficiently verify the integrity of coded packets.

3. Our Approach

In this section, we first introduce the design goals of our identification scheme. To achieve these goals, we then describe the main idea and detailed approach.

3.1. Main Idea of the SPAIS

The goals of our malicious node identification scheme in wireless network coding are listed as follows:

- *Watchdog-free.* The identification is self-checking. That is to say, the identification shall not need any extra nodes to serve as watchdogs and only relies on the well-behaved nodes inside the network.
- *Distributed and effective identification.* The identification scheme shall not rely on any central server or controller. In terms of the effectiveness, the identification shall find out all malicious nodes.
- *Light-weight operating.* As wireless networks are often resource-restricted, the computation overhead of the identification shall be as low as possible.

To achieve these goals, we present our scheme for malicious nodes identification in the network where network coding is used. The main set of principles behind our scheme are listed as follows: (i) every well-behaved node in the network is responsible for monitoring its downstream neighbor

nodes, and (ii) a light-weight homomorphic hash is designed to verify the packets transmitted by downstream nodes.

By taking the advantage of the broadcast nature of the wireless communication medium, when a node is sending out some packets, both its downstream and upstream nodes can receive these packets. Hence, the upstream nodes can monitor its downstream neighbors locally and verify the integrity of these overheard packets by checking whether the packets are altered by downstream neighbor nodes [18]. Nonetheless, in a random linear network coding, the transmitted packets are the linear combination of several input packets, which may come from one or more different upstream nodes. Hence, the nodes cannot recognize the original packets individually. To deal with this problem, we develop a novel scheme SPAIS, where multiple well-behaved nodes cooperate and efficiently identify the malicious nodes in a wireless network where a random linear network coding is used.

Here, we briefly present the core idea of SPAIS as follows. The network is hierarchically organized as levels [13]. A node is called a level- j node if it is j hops away from the source $\mathcal{S}$. We define the source $\mathcal{S}$ to be the level-0 node and the neighbor nodes of the source as level-1 nodes, and so on. Well-behaved level- i nodes monitor the level- $i+1$ nodes cooperatively. We assume that the source is never compromised. Hence, in the recursive way, every node is monitored by its direct upstream level and monitoring its direct downstream nodes with other well-behaved nodes in this level cooperatively.

More concretely, after overhearing a packet from the downstream level node, one node may not be able to individually determine the integrity of the packet, which is the linear combination of packets coming from other nodes.

Hence, the node shall ask other nodes to help. Instead of transmitting a whole packet, verifying the hash of the packet suffices to confirm whether the packet is corrupted or not. In this way, the communication overhead is significantly reduced.

Our SPAIS is inspired by Algebraic-Watchdog [18], which is based on the hypothesis testing framework. In Algebraic-Watchdog, in order to detect the misbehavior of the downstream nodes, one node needs to collect a larger number of packet samples first, then computes the distribution of the collected samples, and finally identifies the malicious nodes by comparing the difference of the distribution between the well-behaving and adversarial nodes.

3.2. Detailed Description of SPAIS

As a complement to the work of defending pollution attacks, which aims at detecting and filtering corrupted packets in the network such as [8–14], our scheme SPAIS aims at identifying the malicious nodes. We let each destination node detect the corrupted packets by either adopting a certain authentication scheme such as [11] or checking whether the decoding result matches with the specific format. Once corrupted packets are detected by destination nodes, our identification scheme is triggered by the alert message from the destination nodes.

We now describe our identification scheme in three stages, which will be detailed below. For convenience, main notations used in this paper are listed in Table 1.

Table 1: Notation Table	
L :	Maximum network level
C_i :	Coding information in the i -th packet
s_i :	Random seed generated by source for level- i nodes
$\mathbf{z}_j$:	j -th packets overheard from downstream nodes
$\mathbb{L}_i^j$:	Set of target nodes for identifying packet $\mathbf{z}_j$ in level i
$\mathbf{t}_k^j$:	k -th target packet of $\mathbf{z}_j$
$\mathbb{T}^j$:	Set of target packets of $\mathbf{z}_j$
H_{s_i} :	Random matrix generated from random seed s_i
$HS_{s_i}(\mathbf{z})$:	Hash of packet $\mathbf{z}$ computed from s_i

3.2.1. Stage 1: System Setup

In the setup stage, the source $\mathcal{S}$ determines the network level L and disseminates the random seed s_i for each level- i nodes.

Determining network level L . Similar to [13], for each node $v \in V - \mathcal{S}$, its level $l(v)$ on the coding graph $G = (V, E)$ is defined as the hop count of the shortest path from $\mathcal{S}$ to v . Network level L is defined as

$$L \triangleq \max_{v \in V - \mathcal{S}} l(v). \quad (6)$$

As stated in [13], L can be computed in $O(|V| \log |V| + |E|)$ time when no attacker exists. In the presence of attackers, L can be computed by using passive network tomography schemes [21]. More importantly, this stage can be performed offline when the wireless mesh network is deployed.

One-way hash chain. For each level- i node, $\mathcal{S}$ generates a sequence of random seeds s_i and distributes the random seed to the level- i node in advance. To be specific, we use a one-way pseudo-random function f to generate the sequence. First, $\mathcal{S}$ generates a random number s_0 as the base value, $\mathcal{S}$ then computes the sequence of seeds recursively with $s_{i+j} = f^j(s_i)$, in which $0 \leq i \leq L-1$ and $f^j(\cdot)$ means j consecutive applications of f . After encrypting with the shared key to ensure the confidentiality and signing the s_i with its private key using a common signature scheme such as RSA or DSA to ensure the authenticity, $\mathcal{S}$ disseminates the seeds s_i to the corresponding nodes of level- i .

With such a one-way hash chain, the upstream level- i can compute the random seed of downstream level- $i+1$ with its s_i as $s_{i+1} = f(s_i)$ whereas level- $i+1$ nodes cannot obtain the s_i of level- i .

3.2.2. Stage 2: Data Packet Transmission

After the system setup stage, the next stage is the data packet transmission stage, which is almost the same as the conventional random linear network coding [3]. The nodes receive packets from their upstream nodes, compute the linear combination of packets and send the coded packets to their downstream nodes. Unlike the existing defense schemes against pollution attacks, we do not append any authentication tags (e.g., homomorphic MACs, signatures or hashes), which are used for verifying the integrity of the corresponding packets. Instead, to detect the corrupted packets and to identify the malicious nodes in the network if the pollution attack really exists, all the nodes should store the overheard downstream packets in the buffer. In some cases, if the storage is limited, the hashes of the overheard downstream

Figure 1: A valid packet p_i sent by a well-behaved node

packets can be computed in advance and stored to save the storage overhead. The detailed hash computation is described in Section 3.3. Moreover, we append the coding information. Specifically, each data packet $\mathbf{z}_i$ consists of four components as follows:

- i corresponds to the packet index, which is the unique identifier for every packet.
- α_i corresponds to the global coding vector of packet i .
- $\mathbf{d}_i$ corresponds to the data vector of packet i .
- C_i corresponds to the coding information and it consists of m_i , the number of upstream packets that are used to generate the coding vector, indexes of these upstream packets, and β_i , the corresponding local coding coefficients, which are randomly picked by the upstream nodes of packet i .

Fig. 1 shows the valid packet p_i sent by a well-behaved node. Note that the prefix packet index i and the postfix coding information C_i do not participate in the process of linear combination and is appended to the packet after the linear combination.

3.2.3. Stage 3: Malicious Nodes Identification

If one of the destination nodes detects corrupted packets in the Stage 2, it sends an alert message to the source. If the source $\mathcal{S}$ receives the alert message from a destination node, it means the pollution attackers exist. For each overheard packet from level-1, $\mathcal{S}$ computes the linear combination of the input of the overheard packets, which is stored in its output buffer, and then compare the result with the overheard packet. For an overheard packet $\mathbf{z}$, which is the linear combination of $\mathbf{x}_i$, $\mathcal{S}$ checks whether the following condition is true or not:

$$\mathbf{z} \stackrel{?}{=} \sum_{i=1}^m \alpha_i \mathbf{x}_i. \quad (7)$$

If the condition in Equation (7) holds, the sender of the packet $\mathbf{z}$ is non-malicious; otherwise, it is a malicious node and will be excluded from the network. Then, the source sends an alert message to the well-behaved level-1 nodes to continue the identification process. We assume that the source is not compromised, the security of level-1 can be assured by level-0 (i.e., the source).

For level- i nodes, in some cases, the node, say R_i , is not able to verify the overheard packet $\mathbf{z}_j$ from its downstream node of level- $i+1$, say R_{i+1} , because the input of $\mathbf{z}_j$ comes from not only R_i , but also other nodes of level- i . Under those circumstances, the well-behaved nodes of level- i shall cooperatively verify the packets from the nodes in the downstream level, after the identification is triggered. We define a node, which is the sender of the input packet of $\mathbf{z}_j$ in level- i , as a target node, and denote the set of target nodes as $\mathbb{L}_i^j$, which can be obtained from the coding information c_j of the overheard packet $\mathbf{z}_j$. In addition, we define a packet, which is the

input of $\mathbf{z}_j$ as a target packet $\mathbf{t}_k^j, k = 1, \dots, m_j$, and denote the set of target packets as $\mathbb{T}^j$. Subsequently, R_i broadcasts request messages to the nodes $v \in \mathbb{L}_i^j$. The request message consists of the identifiers of target nodes and the identifiers of the target packets, and is attached with an MAC (Message Authentication Code) in order to assure the integrity of the request message. After receiving the request, each node $v \in \mathbb{L}_i^j$ computes the hashes of the requested packets, attaches the MAC to the messages, and replies the hashes back to R_i . With the replied hashes, R_i can verify the overheard packets. The detailed procedure of level- i nodes to identify packets from downstream nodes is shown in Algorithm 1. The detailed hash computation and packet verification will be discussed in subsequent subsection.

Intuitively, to establish such a “chain of trust”, the identification procedure should be level-by-level. This means that the identification of level- $i+1$ will be waiting until identification of level- i is finished. We introduce two following identification functions to adapt to different network scenarios.

Level-by-level Identification. As stated, after the source has checked all the packets overheard from the level-1 nodes, it will exclude the malicious nodes if they exist, and send signed alert messages to level-1 nodes to trigger the identification of level-1. Then, the identification carries on level-by-level. That is, to find all the malicious nodes in the network, for all the $1 \leq i \leq L$, the identification of level- i is triggered by the alert message from level- $i-1$, which triggers the identification of level- $i+1$.

Note that in the level-by-level identification function, the nodes begin to check the downstream level nodes only if this level is ensured that there is no malicious nodes. Therefore, the nodes which are far from the source will

wait before being checked.

Level-independent Identification. In the level-by-level identification, all the valid nodes are sequentially checked and the identification process is unnecessarily delayed. To speed up the level-by-level identification, the identification process can be performed concurrently and individually in different levels. Specifically, when the source is verifying the packets of level-1 nodes, the level- i nodes can also verify the level- $i+1$ nodes simultaneously rather than waiting for alert messages from level- $i-1$. The level- i identification is valid if and only if there is no malicious nodes of level- $i-1$ participating the identification process. When one or more malicious nodes in the level- i are identified by a valid identification, the malicious nodes will be excluded from the network and level- $i+1$ identification shall be restarted if those malicious nodes participated.

If there is only a few malicious nodes in the network, the level-independent identification function is more suitable than the level-by-level identification, because the former can avoid the unnecessary delay, which occurs in the latter. While if there are a lot of malicious nodes in the network, the level-by-level identification is better because it avoids the frequent re-identification in the level-independent identification function. Nonetheless, both of them are equally effective to find all the malicious nodes in the network.

3.3. Hash Computation and Packet Verification

We now describe how the hashes are computed and how the overheard packets are verified in detail.

As mentioned in the system model 2.1, we denote the generation size of network coding as m . Without loss of generality, we denote the overheard

packet as $\mathbf{z}_j$. Denote $g : \{0, 1\}^\kappa \times \{0, 1\}^{\log_2(b) + \log_2(m+n)} \rightarrow \mathbb{F}_q$ as a pseudo-random function, where κ is the size of the random seed s_i and b controls the size of the hashes.

Hash Computation. Each level- i node generates a random $b \times (m+n)$ matrix $H_{s_i} = [h_{j,k}]$ using the pseudo-random function g and the seed s_i , where $h_{j,k} = g_{s_i}(j \parallel k)$. Here, $j \parallel k$ means that the concatenation of the binary representation of j and k . Then, if node $v \in \mathbb{L}_i$ receives the request for the target packet $\mathbf{t}_k^j$, the hash $HS_{s_i}(\mathbf{t}_k^j)$ of $\mathbf{t}_k^j$ can be computed by

$$HS_{s_i}(\mathbf{t}_k^j) = H_{s_i} \cdot [\mathbf{t}_k^j]^T. \quad (8)$$

Here, $[\cdot]^T$ is the matrix transpose operation. Since H_{s_i} is a $b \times (m+n)$ matrix and $[\mathbf{t}_k^j]^T$ is a $(m+n) \times 1$ matrix, the $HS_{s_i}(\mathbf{t}_k^j)$ is a $b \times 1$ matrix. Then, node v replies the hash $HS_{s_i}(\mathbf{t}_k^j)$ to the node R_i . If v is requested for multiple target packets, the corresponding multiple hashes should be assembled into one message to reduce the communication overhead.

Packet Verification. Given all the authentic hashes HS , the node R_i checks the validity of the overheard packet $\mathbf{z}_j$ by checking whether the following equation holds or not:

$$\sum_{k=1}^{m_j} \beta_{j,k} \cdot HS_{s_i}(\mathbf{t}_k^j) \stackrel{?}{=} HS_{s_i}(\mathbf{z}_j). \quad (9)$$

Here, m_j and $\beta_{j,k}$ are parts of the coding information c_j and c_j comes from the $\mathbf{z}_j$. If Equation (9) holds, the sender of the packet is non-malicious; otherwise, it is a malicious node and will be excluded from the network. Then, the source sends an alert message to the well-behaved next level nodes to inform that the sender of the corrupted packet is excluded and to continue the identification

of the next level. Because the node R_i and other nodes $v \in \mathbb{L}_i$ are monitored by their upstream level- $i-1$ nodes, which are also monitored, the security of level- $i+1$ can be assured by level- i . More concretely, level- $i-1$ nodes will check the hashes from $v \in \mathbb{L}_i$ in a similar way through the packet verification process and make sure that any $v \in \mathbb{L}_i$ indeed sends the correct hashes out to prevent the malicious nodes collusion, which will be further analyzed in Section 4. In such a recursive and collaborative manner, a “chain of trust” is established.

The correctness of the scheme can be proofed by the following theorem.

Theorem 1 (The correctness of our homomorphic hash). *The hash value of encoded packet $\mathbf{z}_j = \sum_{k=1}^{m_i} \beta_{j,k} \mathbf{t}_k^j$ can be computed by the hash of the input packets $\mathbf{t}_k^j$.*

Proof. We have

$$\begin{aligned}
HS_{s_i}(\mathbf{z}_j) &= HS_{s_i} \cdot [\mathbf{z}_j]^T = HS_{s_i} \cdot \left[\sum_{k=1}^{m_j} \beta_{j,k} \mathbf{t}_k^j \right]^T \\
&= \sum_{k=1}^{m_j} \beta_{j,k} \cdot (HS_{s_i} \cdot [\mathbf{t}_k^j]^T) \\
&= \sum_{k=1}^{m_j} \beta_{j,k} \cdot HS_{s_i}(\mathbf{t}_k^j).
\end{aligned} \tag{10}$$

□

3.4. Batch Verification

In order to reduce the communication overhead further and speed up the packet verification process, we introduce the batch verification approach.

Our scheme supports the packet batch verification approach, which can verify multiple packets that are sent from the same node in one process.

Because of the mixing nature of networks coding, the input of these packets will be very repetitive. If the request node R_i requests all the hashes, it is a tremendous waste of bandwidth. To this end, if R_i receives multiple packets from the same downstream level node, before sending the request messages for each packet, R_i selects the distinct target packet index from all the necessary packets. In addition, when verifying the hashes, R_i randomly selects u coefficients (here, u is the number of the overheard packets from the same downstream node), generates a new packet, which is the linear combination of the overheard packets with these u coefficients, and verify the new packet in a similar way with Equation (9). For example, assume that an overheard packet p_1 is the linear combination of packet p_a , p_b and p_c , another packet p_2 is from p_a and p_d . Without batch verification, each input packet (p_a, p_b, p_c of p_1 and p_a, p_d of p_2) needs 1 request and the hash of each packet shall be computed once, 5 requests and 7 hash computations are needed totally. With batch verification, the node randomly selects two coefficients γ_1, γ_2 , and generates a new packet $p = \gamma_1 p_1 + \gamma_2 p_2$, and then sends out four hash requests (once for p_a, p_b, p_c and p_d , respectively). After the hashes are computed and returned to the node, it computes $\gamma_1 \cdot HS(p_1) + \gamma_2 \cdot HS(p_2)$ and compares the result with $HS(p)$. As we can see, only 4 requests and 5 hash computations are enough to verify these 2 packets.

If the new packet passes verification, all the input packets are valid and the sender of these packets passes the identification. Otherwise, at least one packet must be polluted and the sender is a malicious node as well. This tells us that verifying only one new linear combination of those packets is enough to identify one downstream level node, which will significantly reduce

the bandwidth overhead and computation overhead.

4. Security Analysis

In this section, we first discuss the condition that malicious nodes in the network are independent. We then consider the more complex case, where attackers can collude to hide themselves and conduct the pollution attack. We say a packet is secure if its sender is monitored by the secure upstream level. This means that if a malicious node sends out a corrupted packet and the corrupted packet is received by its upstream level nodes, the corrupted packet will be detected and the malicious node will be identified with a high probability. Hence, we say that a node level is secure if the packets sent from of this level are overheard and verified by its upstream level.

Independent Malicious Nodes. Theorem 2 quantifies the security level of our identification scheme with respect to two parameters, q and b , where q is the finite field size of network coding and b is the security parameter described in Section 3.3.

Theorem 2. *In our scheme, the probability of a polluted packet can pass the verification at a secure level is at most $\frac{1}{q^b}$.*

Proof. Let $\tilde{\mathbf{z}}_j$ be a polluted packet, i.e., $\tilde{\mathbf{z}}_j \neq \sum_{k=1}^{m_i} \beta_{j,k} \mathbf{t}_k^j$, sent from level- $i+1$, which is overheard by a level- i node R_i . The probability of the event that $\tilde{\mathbf{z}}_j$ passes the verification (i.e., Equations (7) or (9) holds) is computed as follows:

Denote $\mathbf{z}_j$ as the corresponding correct packet, that is, $\mathbf{z}_j = \sum_{k=1}^{m_i} \beta_{j,k} \mathbf{t}_k^j$. We also have $\mathbf{u}_j = \tilde{\mathbf{z}}_j - \mathbf{z}_j = (u_1, u_2, \dots, u_{m+n})$. Obviously, $\tilde{\mathbf{z}}_j \neq \mathbf{z}_j$. Hence,

there exists some $i, 1 \leq i \leq m+n$, such that $u_i \neq 0$. As assumed, we have $HS_{s_i}(\tilde{\mathbf{z}}_j) = HS_{s_i}(\mathbf{z}_j)$, which means that $HS_{s_i}(\mathbf{u}_j) = 0$. The above equation can be rewritten as $\mathbf{hs}_1 u_1 + \dots + \mathbf{hs}_i u_i + \dots + \mathbf{hs}_{m+n} u_{m+n} = 0$, where $\mathbf{hs}_i$ is the i^{th} vector of $\mathbf{HS}$. Because $u_i \neq 0$, u_i has a unique multiplicative inverse v_i in $\mathbb{F}_q$ such that $u_i v_i = 1$. Hence, we have

$$\begin{aligned} \mathbf{hs}_i = & -v_i(\mathbf{hs}_1 u_1 + \dots + \mathbf{hs}_{i-1} u_{i-1} \\ & + \mathbf{hs}_{i+1} u_{i+1} + \dots + \mathbf{hs}_{m+n} u_{m+n}). \end{aligned} \quad (11)$$

Nonetheless, as the network is hierarchically organized, the downstream level is not able to derive the random seed of upstream level and cannot obtain the HS . In this way, although the malicious node can control the values of u_i , the probability that Equation (11) holds is equal to the probability that a random vector with the size of b can satisfy the equation without HS . Therefore, the probability of a polluted packet can pass the verification is $\frac{1}{q^b}$. $\square$

As we can see, we can reduce the probability that a polluted packet can pass the verification $\frac{1}{q^b}$ by using a larger field size q or larger security parameter b , which will both introduce more communication overhead. In addition, the probability $\frac{1}{q^b}$ can be achieved when all the hashes are received by R_i .

Moreover, if some of the hashes are lost because of the lossy wireless links, the overheard packet cannot be verified and it will be marked as an uncertain packet. To this end, Theorem 3 quantifies the probability that a packet will be marked as an uncertain packet with respect to the following parameters, p , r and $k_R, k_j, j = 1, \dots, r$, where r is the number of the requested nodes

for a certain packet, k_R and k_j are the maximum retransmission number of requesting node R and requested nodes $j, j = 1 \dots r$, respectively and p is the average link quality of nodes in the same level. **In this paper, the link quality between two nodes A and B is defined as the delivery probability of sending packets from node A to node B . In practical scenarios, it can be measured by dividing the number of packets received by B by the total number of packets sent from A .** All the uncertain packets will trigger another round of verification in order to avoid any corrupted packets passing the verification.

Theorem 3. *An overheard packet will be marked as an uncertain packet with a probability, denoted as P_{up} , no larger than $1 - (1 - (1 - p)^{k_R})^r \cdot \prod_{j=1}^r (1 - (1 - p)^{k_j})$.*

Proof. If R_i does not receive enough hashes, which means either the request message from R_i or one of the reply messages from $v \in \mathbb{L}_i$ is lost due to the lossy wireless links, the overheard packet will be marked as uncertain. The probability of the event that the request messages from R_i are received by all the $v \in \mathbb{L}_i$ is $(1 - (1 - p)^{k_R})^r$, and the probability of all the reply messages are received by R_i is $\prod_{j=1}^r (1 - (1 - p)^{k_j})$. When these two events occur, R_i receives enough hashes and can verify the overheard packet. Hence, the probability of an overheard packet will be marked as an uncertain packet is $1 - (1 - (1 - p)^{k_R})^r \cdot \prod_{j=1}^r (1 - (1 - p)^{k_j})$. $\square$

Moreover, a malicious node X may disparage innocent nodes by replying a wrong hash. Nonetheless, as stated in Section 3.3, the hash reply messages are also monitored by the upstream level of the malicious node. The mis-

behavior will be discovered as long as the wrong hash message is received with a probability of $1 - \prod_{i \in \mathbb{U}_X} (1 - p_i)^k$, where $\mathbb{U}_X$ is set of nodes in the upstream level of X , k is the maximum retransmission number of X , and p_i is the link quality between X and i . For example, when $k = 5$, and there are only two nodes in $\mathbb{U}_X$, where link qualities are both 0.5, the probability is $1 - 9.7 \cdot 10^{-4}$.

In summary, Theorem 2 shows the probability that a packet can pass the verification of the secure upstream level (i.e., the links are lossless) and Theorem 3 shows the probability that the packet will be marked as an uncertain packet in a wireless lossy environment. If the malicious nodes in the network are independent to launch pollution attacks, the misbehavior will be detected and the malicious nodes will be identified with a high probability.

Collaborative Malicious Nodes. We consider multiple malicious nodes collude to hide themselves or disparage innocent nodes. If a colluding malicious node X is in the upstream level node of another malicious node Y and is requested for the hash to verify a corrupted packet from Y , X may choose not to reply the hash to hide Y . Nonetheless, as X is monitored by its upstream level, the inaction of X can be detected as well and be reported to the source.

In addition, malicious nodes may collude to disparage innocent nodes (say Z). That is, some malicious nodes, which are in the same level, say X and Y , collude to falsely accuse an innocent node Z , which is in the downstream level of X and Y after receiving the correct packet from Z . Under this condition, our previous work [22], which can filter injected false data effectively and achieve a high resilience to the number of compromised

nodes without relying on static routes and node localization, can be applied to detect such collusion of malicious nodes.

In our previous work [22], the polynomials instead of MACs are adopted for endorsing the packet reports to achieve the resilience to the false packet report. The corresponding authentication polynomial and check polynomial are allocated and assigned to each sensor node. These polynomials are stored in nodes, bundled with node ID and derived by the primitive polynomials assigned from a primitive polynomial pool. In our case, different primitive polynomials will be used in different levels through the level-based primitive polynomial assignment. The authentication polynomial stored in each node is used to endorse the report of local packet report, while the check polynomial is used to validate the received reports. With a T -authentication framework, i.e., a legitimate report shall be authenticated by T nodes from the same level, the forwarding node of other levels could verify the report only if it shares the authentication information with the report node.

5. Performance Evaluation

In this section, we define metrics and conduct experiments to show the effectiveness of our proposed identification scheme. We use the false negative probability and false positive probability to quantify the accuracy of the scheme and the bandwidth and computation overhead to quantify the efficiency of the scheme. In the following, we first present the simulation setup and then present the evaluation results.

5.1. Evaluation Setup

In our evaluation setting, we consider a random graph topology that consists of 30 nodes in a wireless network. The positions of these nodes were chosen randomly in a square of side 400 units. The communication range of each node is set to 150 units. The source node, destination node, and the malicious nodes are selected randomly as well. Each simulation is repeated 1000 times and the results are averaged as illustrated below. Fig. 2 shows one example of randomly generated topology. The color of the nodes shows the level of nodes, and the dash line between the nodes means the nodes are in the communication range with each other. **What's more, we use level-by-level identification in the evaluation.**

Figure 2: An Example of Random Topology

5.2. Probability of packets become uncertain

Recall that if some of the hashes are lost because of the lossy wireless links, the overheard packet cannot be verified and will be marked as an uncertain packet. Denote P_{up} as the probability of packets that becomes uncertain. **In this subsection, the definition of parameters p , r , k_R and k_j is the same as the one in Section 4. That is, r is the number of the requested nodes for a certain packet, k_R and k_j is the maximum retransmission number of requesting node R and requested nodes j , where $j = 1 \dots r$, respectively and p is the average link quality of nodes in the same level.** To simplify the evaluation of the probability P_{up} in a practical wireless network, we set k_R and $k_j, j = 1, \dots, r$ are all equal to a predetermined value k .

Fig. 3 shows the probabilities of packets become uncertain under several different sample parameter selections. We can see that the P_{up} increases with r , and decreases as p and k increase. Even if under a relatively worse condition where $p = 0.7, k = 5, r = 3$, the probability is at most 1.45%. If the retransmission number k increases linearly, the communication overhead that is proportional to the k also increases linearly, while the probability P_{up} decreases much rapidly.

5.3. False Negative Probability

Recall that the false negative probability P_{fn} is defined as the probability of a malicious node, which sends one or more corrupted packets without being identified. As we apply our previous study PCREF [22] to prevent the malicious nodes colluding, the event occurs if the corrupted packet is not overheard by T upstream level nodes as we stated in Section 4. In our

Figure 3: The probability of packets become uncertain under different settings

simulation, we set the ratio n_{un} of the threshold T to the number of nodes in the certain level as a parameter.

Fig. 4 shows the relationship between the false negative probability P_{fn} and the average link quality for different c , which is defined as the number of corrupted packets sent by malicious nodes. As we can see, the false negative probability declines rapidly as the link quality increases. For example, if the average link quality is above 0.8, even if a malicious node injects only one packet into the network, it will be identified with a probability of 97.25%(1 – 2.75%). In addition, if the malicious node sends more corrupted packets, the probability that it can succeed in escaping the identification will be much smaller. For example, if the malicious node injects 2 and 4 packets into the network respectively and the average link quality is 0.8, it will be identified with a probability of 99.06%(1 – 0.94%) and 99.93%(1 – 0.07%), respectively.

We also compare the false negative probability of SPAIS with

Figure 4: The false negative probability under different settings (c is the number of corrupted packets sent by malicious nodes)

that of other schemes in the same scenario. Fig. 5 shows the relationship between the false negative probability P_{fn} and the average link quality for SPAIS, MIS, SpaceMac and PAI. For SPAIS, we choose $T = \frac{1}{2}n_{un}$. For PAI, we choose the number of rounds $n = 60$ since it is the median value in their evaluation. For MIS and SpaceMac, the central server is randomly selected in each simulation. What's more, in all the compared schemes, we choose the number of corrupted packets sent by malicious nodes $c = 1$. As we can see, when the link qualities in the network are low (less than 0.53), SPAIS performs better than the other three schemes, while the link quality is above 0.56, both MIS and SpaceMac performs slightly better than SPAIS because they are central server-based schemes.

Figure 5: The false negative probability of SPAIS, MIS, SpaceMac and PAI in same scenario

5.4. False Positive Probability

Recall that the false positive probability P_{fp} is defined as the probability that a well-behaved node is considered as a malicious node. As stated in Section 4, this will not happen if there is no collaborative attackers. Under the condition that the attackers can collude, malicious nodes can succeed in disparaging a well-behaved node, only if they take over T upstream level neighbors of the well-behaved node; otherwise the corrupted packet report to the source will be filtered and the disparagement will not succeed.

Fig. 6 shows the relationship between the false positive probability P_{fp} and the number of the malicious nodes in terms of different T . We can observe that with the increasing number of malicious nodes, the false positive probability increases rapidly. **For example, if T is set as $1/2 * n_{un}$ that means only if malicious nodes take over half of upstream nodes, they can**

Figure 6: The false positive probability under different settings

succeed in disparaging a well-behaved node, the false positive probability is only 4.2% when there are 9 malicious nodes (about 30%) in the network. Moreover, if there are more adversarial nodes, say 15 (half of the nodes), in the network, the probability will be 14.02%.

We also compare the false positive probability of SPAIS with that of other schemes in the same scenario. Fig. 7 shows the relationship between the false positive probability P_{fp} and the number of the malicious nodes for SPAIS, MIS, SpaceMac and PAI. The parameters for the compared schemes are selected same as the false negative probability simulation. As we can see, SPAIS performs better than the other three schemes in most cases. That is to say, compared with other schemes, SPAIS can prevent the well-behaved node being disparaged by the colluded malicious nodes

Figure 7: The false positive probability of SPAIS, MIS, SpaceMac and PAI in same scenario

with a higher probability.

5.5. Computation Overhead

We now compare the computation overhead of our identification scheme SPAIS with other schemes. We implemented these schemes on top of software package *MIRACL* [23] and tested them on an Intel Core2 6320 (1.86GHz) Linux machine. We run these schemes to obtain the time taken for conducting hash and verification per packet for different packet sizes. We ignored the computation overhead of the system setup phase because it can be performed offline. In our experiments, the computation overhead is measured in the actual running time.

For each packet, when one node shall verify an overheard packet from its downstream level, the node requests the hash of the packet from other nodes in the same level. To reply the requested hash, these nodes will compute l

Figure 8: The total time (hash computation + verification time) versus packet size

Table 2: The total time (hash computation + verification time) of MacSig and ESBS

packet size	128	256	512	1024
MacSig	281 ms	290 ms	311 ms	348 ms
ESBS	871 ms	1217 ms	1900 ms	3775 ms

hashes totally. Therefore, the computation overhead in this stage is $l \cdot b \cdot (m + n)$ multiplications per packet because each hash needs $b \cdot (m + n) \cdot 1$ multiplications. After the hashes are replied, $b \cdot (m + n) + l$ multiplications are required to compute the hash of the original overheard packet and combine the hashes together. Hence, the total computation overhead is $l \cdot b \cdot (m + n) + b \cdot (m + n) + l$ multiplications.

Fig. 8 and Table 2 shows the total time (hash computation time plus verification time) versus packet size. As we can see, the computation overhead increases linearly as the packet size grows from 128 to 1024. For a 1024 byte

packet, the total verification can be finished in only 565us, which incurs little delay. Although the computation overhead of our scheme is slightly larger than the SpaceMac, we shall note that the verification of our scheme is distributed, which makes our scheme work better. Note that some public-key based crypto-systems (e.g., [14, 9]) incur much longer delays.

Note that the Algebraic-Watchdog scheme is not a cryptographic approach, and the evaluation of Algebraic-Watchdog itself is performed in MATLAB, while in our computation overhead evaluation, all the compared schemes are cryptographic approaches and are implemented on top of software package MIRACL which is a C/C++ library. Moreover, the MIRACL is not appropriate for implementing Algebraic-Watchdog. We believe that it is unfair to compare the computation overhead of Algebraic-Watchdog obtained from MATLAB with that of other schemes obtained from C language. Because of all these reason, the computation overhead of Algebraic-Watchdog is not presented.

5.6. Bandwidth Overhead

We also compare the bandwidth overhead of our identification scheme SPAIS with other schemes. For a similar reason, we do not consider the bandwidth overhead in the system setup stage.

The online bandwidth overhead per packet only includes the packet index and the coding information as stated in Section 3.2.2. The coding information consists of l node ID that has 2 Bytes, l packet indexes that has 2 Bytes, and the local coding vector that is l symbols over $\mathbb{F}_q$. Hence, the bandwidth overhead per packet is $\frac{(2+2) \cdot 8 \cdot l / |q| + l}{m+n}$, where $|q| = \lceil \log_2 q \rceil$. In our

Figure 9: The bandwidth overhead versus packet size

experiments, we set $l = 7.16$, which is the average number of combined packets obtained from our simulation results and $|q| = 8$, which is typical value in most schemes.

Fig. 9 shows the bandwidth overhead per packet versus packet size. As we can see, the bandwidth overhead declines as the packet size grows from 256 to 1024. The bandwidth overhead of our scheme is a bit larger than MIS and SpaceMac. The overhead of DART is smallest as shown in the figure, but it is not true in practice because the checksum matrix of DART will be multicasted to multiple nodes while we only consider it once here because the actual transmission number depends on the practical network topology. ESBS [9] is a typical representative of the public-key based network coding authentication schemes and its bandwidth overhead is relatively larger. In addition, the bandwidth overhead of MacSig, as a hybrid-key based network coding authentication scheme, is the largest as it includes a large number of

MACs.

6. Discussion

From the evaluation results, we can say that our scheme works better under the condition that the communication quality is higher and the number of malicious nodes in the network is smaller. In extreme, if our scheme is applied into a lossless environment, all the malicious nodes can be identified. Nonetheless, if the link qualities are not in good condition and there are more malicious nodes in the network, the performance of our scheme can still achieve fairly good performance. **In a practical network, the average link quality between neighboring nodes will not be in an ideal condition, while malicious nodes in the network may occupy the half of total nodes. Based on reported in [25], half of transmission links have a loss probability greater than 30%. Also as shown in [12], the wireless environment makes the threat of pollution attacks particularly severe. In comparison with existing schemes, SPAIS performs better under such conditions.** As shown in the evaluation results, the malicious nodes can be identified with a high probability in one round. In addition, if the malicious nodes escape the current identification and still try to disrupt the subsequent transmission, additional round of identification will be triggered until all malicious nodes are identified.

From the hash computation, we can see that the hash of each overheard downstream packet is a $b * 1$ matrix, which incurs very small storage overhead. Assume that the node has n downstream neighbor nodes, and overhears m_p packets from each neighbor nodes.

The storage overhead of hashes is $b * m_p * n$ bytes. In practical scenarios, the number of downstream neighbor nodes n will not exceed 10 that is observed from our simulation topology, the number of overheard packets m_p will not exceed the generation size of network coding m , which can be set as 32, and b is set as 2 in our scheme. Therefore, the upper bound of storage overhead is 640 bytes.

Note that in our SPAIS, when the identification is triggered, the source should know the level of other nodes in advance, and then distribute the corresponding random seeds to them. In addition, the nodes in network should also know the level of their neighbors to cooperatively verify the overheard packets. Therefore, our scheme is designed for WMNs where the mesh routers are static. Nonetheless, if the network topology is dynamic, our scheme can still work with some modification. Specifically, when the identification is triggered and the topology changes during the data transmission, the source shall re-distribute the random seeds to the corresponding nodes to assure the secrecy, and when verifying the overheard packets, the node shall ask the right nodes which may not be the neighbor now for the hashes. To this end, maintaining the topology, distributing the random seeds and the multi-hop hash requesting will introduce extra bandwidth overhead, and the false positive probability will be higher too. We plan to deal with these issues in our future work.

Note that in SPAIS as well as the malicious nodes identification scheme Algebraic-Watchdog, even if all of the nodes in the same level are compromised by the attacker, even there is no way to recover the corrupted packets, these malicious nodes can be

identified by their upstream level nodes as long as there is enough well-behaved upstream nodes. In this case, if the min-cut to the destination node is dominated by attackers, other complementary authentication schemes such as MacSig [14] can be leveraged to detect the corrupted packets and keep the correct ones.

7. Related Work

As a complement to the work of defending pollution attacks, which aims at detecting and filtering corrupted packets in the network (e.g., ESBS [9] as one of public-key cryptographic schemes, DART [12] as one of symmetric-key cryptographic schemes, and [14] as a hybrid-key scheme), our SPAIS aims at identifying the malicious nodes. In the following, we will review the existing schemes, which are most relevant to the problem of identifying pollution attackers, which is the main focus of our paper. The detailed study of network coding authentication schemes can be found in [24].

Liang *et al.* [15] proposed a lightweight misbehavior detection scheme that integrates the idea of watchdogs and error detection coding in wireless networks. Nonetheless, this mechanism needs extra trusted nodes to serve as watchdogs. Obviously, it is not economical to deploy a larger number of watchdogs in a large wireless network. Wang *et al.* [16] proposed a lightweight scheme, namely MIS, to limit network-coding pollution attacks by identifying malicious nodes in the peer-to-peer streaming system. Nonetheless, using the scheme in [16], a centralized server is necessary to compute the checksum for a suspect node to detect a corrupted packet sent from its upstream nodes. In addition, the communication overhead is significant as the

checksum needs to transmit to all the nodes in the network. Le *et al.* [19] proposed a defense scheme namely SpaceMac to combat pollution attacks for intra-session network coding. Under a lossless and centralized condition, their developed scheme can identify all the pollution attackers with a low computation overhead and bandwidth overhead. Rossano *et al.* [26] proposed a belief propagation-based technique to identify malicious nodes in a peer-to-peer streaming architecture, where each peer is assigned to one monitor node. In summary, all of these schemes need extra nodes to server as either watchdogs or centralized servers.

Li *et al.* [17] focused on identifying pollution attackers in WMNs by implementing the idea of shrinking the suspicious set. Their developed mechanism PAI is randomized and fully distributed. Nonetheless, the PAI mechanism is based on the network coding batch verification, while a detailed batch verification scheme to effectively detect the corrupted packets was not considered. In addition, the packet loss in WMNs was not considered as well as the work [19]. In summary, these schemes may be only limited to applications such as wired peer-to-peer streaming system and file distribution system. In addition, in Algebraic-Watchdog [18], nodes can verify their neighbors probabilistically and police them locally by the means of overheard messages in a coded network. Nonetheless, different from other schemes, Algebraic Watchdog uses the hypothesis testing method to make the decision to identify a malicious node, which needs to collect a larger number of detecting samples and occurs a large computation overhead.

8. Conclusion

In this paper, we proposed a novel pollution attacker identification scheme to identify the pollution attackers effectively and efficiently. By making full use of the broadcast nature of wireless media and the principle that a node can monitor its downstream neighbor nodes locally by cooperating with other nodes, our developed scheme hierarchically organizes the network as levels and the nodes in the same level can monitor their downstream level nodes cooperatively. To further reduce the bandwidth overhead incurred by the communication associated with the same level nodes, we developed a homomorphic hash function for network coding. Through a combination of theoretical analysis and extensive simulation study, our data shows that SPAIS can more effectively identify pollution attackers with a lower cost in comparison with the existing schemes.

Acknowledgement

The work was supported in part by the following funding agencies in China: the New Century Excellent Talents in Universities of China under grant NCET-09-0649, the Ph.D. Program Foundation of Ministry of Education of China under grant 20100201110027, the Program for Industrial Development of Shaanxi under grant 2011K06-40, and in part by the National Science Foundation of China (NSFC) under Grant 61373115 and Grant 61402356. This work was also supported in part by US National Science Foundation (NSF) under grants: CNS 1117175 and 1350145. Any opinions, findings and conclusions or recommendations expressed in this material are

those of the authors and do not necessarily reflect the views of the funding agencies.

References

- [1] I. Akyildiz, X. Wang, A survey on wireless mesh networks, *Communications Magazine*, IEEE 43 (9) (2005) S23–S30.
- [2] R. Ahlswede, N. Cai, S.-Y. Li, R. Yeung, Network information flow, *Information Theory*, IEEE Transactions on 46 (4) (2000) 1204–1216.
- [3] S. Chachulski, M. Jennings, S. Katti, D. Katabi, Trading structure for randomness in wireless opportunistic routing, in: *Proc. of ACM SIGCOMM’ 2007*, August 2007.
- [4] C. Gkantsidis, P. Rodriguez, Network coding for large scale content distribution, in: *Proc. of IEEE INFOCOM*, 2005.
- [5] Z. Liu, C. Wu, B. Li, S. Zhao, Uusee: Large-scale operational on-demand streaming with random network coding, in: *Proc. of IEEE INFOCOM*, 2010.
- [6] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, J. Crowcroft, Xors in the air: practical wireless network coding, in: *Proc. of ACM SIGCOMM*, 2006.
- [7] Secure network coding for wireless mesh networks: Threats, challenges, and directions, *Computer Communications* 32 (17) (2009) 1790–1801.

- [8] C. Gkantsidis, P. Rodriguez Rodriguez, Cooperative security for network coding file distribution, in: Proc. of 25th IEEE International Conference on Computer Communications(INFOCOM), 2006.
- [9] Z. Yu, Y. Wei, B. Ramkumar, Y. Guan, An efficient signature-based scheme for securing network coding against pollution attacks, in: Proc. of IEEE INFOCOM, 2008.
- [10] Z. Yu, Y. Wei, B. Ramkumar, Y. Guan, An efficient scheme for securing XOR network coding against pollution attacks, in: Proc. of IEEE INFOCOM, April 2009.
- [11] S. Agrawal, D. Boneh, Homomorphic macs: Mac-based integrity for network coding, in: 2009 Applied Cryptography and Network Security, pp. 292–305.
- [12] J. Dong, R. Curtmola, C. Nita-Rotaru, Practical defenses against pollution attacks in intra-flow network coding for wireless mesh networks, in: Proc. of the second ACM conference on Wireless network security, 2009.
- [13] Y. Li, H. Yao, M. Chen, S. Jaggi, A. Rosen, RIPPLE authentication for network coding, in: Proc. of IEEE INFOCOM, March 2010.
- [14] P. Zhang, Y. Jiang, C. Lin, H. Yao, A. Wasef, X. Shen, Padding for orthogonality: Efficient subspace authentication for network coding, in: Proc. of IEEE INFOCOM, April 2011.
- [15] G. Liang, R. Agarwal, N. Vaidya, When watchdog meets coding, in: Proc. of IEEE INFOCOM, 2010.

- [16] Q. Wang, L. Vu, K. Nahrstedt, H. Khurana, Mis: malicious nodes identification scheme in network-coding-based peer-to-peer streaming, in: Proc. of IEEE INFOCOM, IEEE, 2010.
- [17] Y. Li, J. C. Lui, Identifying pollution attackers in network-coding enabled wireless mesh networks, in: Proc. of 20th International Conference on Computer Communications and Networks (ICCCN), IEEE, 2011.
- [18] M. Kim, M. Médard, J. Barros, Algebraic watchdog: mitigating misbehavior in wireless network coding, *Selected Areas in Communications, IEEE Journal on* 29 (10) (2011) 1916–1925.
- [19] A. Le, A. Markopoulou, Cooperative defense against pollution attacks in network coding using spacemac, *Selected Areas in Communications, IEEE Journal on* 30 (2) (2012) 442–449.
- [20] P. A. Chou, Y. Wu, K. Jain, Practical network coding, in *Proc. of the Allerton Conference* (October 2003).
- [21] H. Yao, S. Jaggi, M. Chen, Network coding tomography for network failures, in: Proc. of INFOCOM, IEEE, 2010.
- [22] X. Yang, J. Lin, P. Moulema, W. Yu, X. Fu, W. Zhao, A novel en-route filtering scheme against false data injection attacks in cyber-physical networked systems, in: Distributed Computing Systems (ICDCS), IEEE 32nd International Conference on, 2012, pp. 92–101.
- [23] M. Scott, MIRACL library, <http://www.shamus.ie>, 2011.

- [24] D. Zhu, X. Yang, W. Yu, Towards effective defense against pollution attacks on network coding, in: Communications (ICC), IEEE International Conference on, IEEE, 2012, pp. 830–834.
- [25] D. Aguayo, J. Bicket, S. Biswas, G. Judd, R. Morris, Link-level measurements from an 802.11b mesh network, in: Proc. of ACM SIGCOMM'2004, 2004, pp. 121–132.
- [26] R. Gaeta, M. Grangetto, Identification of malicious nodes in peer-to-peer streaming: A belief propagation-based technique, Parallel and Distributed Systems, IEEE Transactions on 24 (10) (2013) 1994–2003.

1 Identifying Procedure of level- i Nodes

R_i pick a packet $\mathbf{z}_j$ from downstream level nodes;

if $\mathbf{z}_j$ can be verified by R_i individually **then**

R_i verifies $\mathbf{z}_j$ using Equation 8 and Equation 9;

else

$\mathbb{T}^j = \{input(\mathbf{z}_j)\};$

$\mathbb{L}_i^j = \{sender(\mathbb{T}^j)\};$

R_i sends request messages to $\mathbb{L}_i^j$ and waits the reply messages;

for each node $v \in \mathbb{L}_i^j$ **do**

if v receives a request message from R_i **then**

v computes the hash $HS_{s_i}(\mathbf{t}_k^j)$ using Equation 8;

v replies the $HS_{s_i}(\mathbf{t}_k^j)$ to R_i ;

end if

end for

if R_i receives all the requested hashes **then**

R_i verifies the packet $\mathbf{z}_j$ using Equation 9;

if $\mathbf{z}_j$ is corrupted **then**

R_i reports the $sender(\mathbf{z}_j)$ is malicious;

end if

end if

end if

Donghai Zhu received the B.S degree in the Department of computer science and technology from Xi'an Jiaotong University, in 2010. He is currently a PhD candidate in Department of Computer Science and Technology from Xi'an Jiaotong University. His current research interests include wireless ad hoc/mesh networks, network coding, and network security.

Xinyu Yang received the Diploma in Computer Science and Technology from the Xi'an Jiaotong University of China in 2001 and the Bachelor, Master, and PhD degrees from Xi'an Jiaotong University in 1995, 1997, and 2001. He has held positions with the Department of Computer Science and Technology at the Xi'an Jiaotong University, where he is a professor teaching in the Department of Computer Science and Technology. His main technical interests lie in the areas of network security, wireless communication and mobile ad hoc networks.

Wei Yu received the B.S. degree in electrical engineering from Nanking University of Technology, Nanjing, China, in 1992, the M.S. degree in electrical engineering from Tongji University, Shanghai, China in 1995, and the Ph.D. degree in computer engineering from Texas A&M University, College Station, in 2008. He is currently an Assistant Professor with the Department of Computer and Information Sciences, Towson University, Towson, MD. Before that, he was with Chico Systems Inc. for almost nine years. His research interests include cyberspace security, computer networks, and distributed systems.

Donghai Zhu

Xinyu Yang

Wei Yu