An Efficient Null Space-Based Homomorphic MAC Scheme Against Tag Pollution Attacks in RLNC

Alireza Esfahani, *Student Member, IEEE*, Georgios Mantas, *Member, IEEE*, and Jonathan Rodriguez, *Senior Member, IEEE*

***Abstract*—This letter proposes an efficient null space-based homomorphic message authentication code scheme providing resistance against tag pollution attacks in random linear network coding, where these attacks constitute a severe security threat. In contrast to data pollution attacks, where an adversary injects into the network corrupted packets, in tag pollution attacks the adversary corrupts (i.e. pollutes) tags appended to the end of the coded packets to prevent the destination nodes from decoding correctly. Our results show that the proposed scheme is more efficient compared to other competitive tag pollution immune schemes in terms of computational complexity.**

***Index Terms*—Network coding, random linear network coding, tag pollution attacks, homomorphic MACs.**

I. Introduction

RANDOM Linear Network Coding (RLNC) was proposed by Ho et al. in [1] as a fully distributed approach for performing Network Coding (NC) [2]. In RLNC, each node selects randomly a set of coefficients and uses them to make linear combinations of the incoming packets. Furthermore, it is worthwhile to mention that RLNC achieves the same capacity as that achieved by the Max-flow Min-cut theorem [3]. However, RLNC is susceptible to data pollution attacks, where an adversary (e.g., compromised intermediate node) injects into the network corrupted packets that prevent the destination nodes from decoding correctly. This has as a result not only network resource waste but also energy waste at the nodes [4]. Hence, during the past few years, several schemes have been presented to provide resistance against data pollution attacks. Among them, those based on homomorphic Message Authentication Codes (MACs) are considered as a low-complexity solution for data pollution attacks [5]–[7]. However, most of the homomorphic MAC schemes are vulnerable to tag pollution attack, which was first defined by Li et al. in [8]. In tag pollution attacks, the adversaries modify tags (i.e., homomorphic MACs) appended to the end of the coded packet. A tag is a piece of information appended to the end of the coded packet to ensure integrity of the transmitted packet data. Each tag has a fixed position in the coded packet and thus, it is easy for an adversary to control its pollution.

This work was supported by FP7-PEOPLE-2011-IAPP-CODELANCE (285969) and FCT-SFRH/BD/102029/2014. The associate editor coordinating the review of this paper and approving it for publication was H. Otrok.

The authors are with the Instituto de Telecomunicações - Pólo de Aveiro, Aveiro, Portugal (e-mail: alireza@av.it.pt; gimantas@av.it.pt; jonathan@av.it.pt).

Therefore, in this letter, we propose an efficient homomorphic MAC scheme, where each tag has not a fixed location in the coded packet, providing resistance against tag pollution attacks. In the proposed scheme, the tags are calculated based on null space properties and then, they are secretly swapped with the packet data [9]. Hence, they are not appended to the end of the coded packet and thus, an adversary is not able to distinguish them from the packet data. Our results show that the proposed scheme is more efficient compared to the schemes proposed in [7] and [10], in terms of computational complexity without incurring additional communication overhead. To the best of our knowledge, the proposed schemes in [7] and [10] are the most competitive schemes in the literature for providing resistance against tag pollution attacks in RLNC.

II. System Model

A. RLNC Model

We consider that the RLNC model, where our proposed scheme is applied, is based on a directed multigraph (S, I, E) which consists of the source node S, the non-source node set I (i.e., $I = \{I_1, \ldots, I_w\}$), and the link set E ($i.e.$, $E = \{e_1, \ldots, e_j\}$), as it is shown in Fig. 1. The source node S transmits its packets to the destination nodes (i.e., I_{w-1} and I_w) through a number of intermediate nodes. We consider two types of packets: native packets and coded packets. The native packets are packets generated at the source node. On the other hand, the coded packets are packets encoded and recoded at the source and intermediate nodes. At the setup phase, the source divides each message into a sequence of native packets and partitions them into generations. Thus, we consider that each generation consists of m native packets denoted as $\underline{\mathbf{u}}_1, \underline{\mathbf{u}}_2, \ldots, \underline{\mathbf{u}}_m$. Each native packet $\underline{\mathbf{u}}_i$, for $1 \leq i \leq m$, is represented as a vector $\underline{u}_1, \underline{u}_2, \ldots, \underline{u}_n$ in the finite field $\mathbb{F}_p^n$, where $p = 2^8$ and it denotes the size of the finite field. In [11], it has been shown that 2^8 is usually sufficient for practical use and convenient for computation. Then, the source S generates a coded packet $\mathbf{u}_i$ for each native packet $\underline{\mathbf{u}}_i$ by prefixing $\underline{\mathbf{u}}_i$ with the i^{th} unit vector of dimension m. The coded packet is represented as a row vector in the finite field $\mathbb{F}_p^{m+n}$ as follows:

$$\mathbf{u}_i = (\overbrace{\underbrace{0, \cdots, 0}_{i-1}, 1, 0, \cdots, 0}^{m}, \underline{u}_{i,1}, \cdots, \underline{u}_{i,n}) \in \mathbb{F}_p^{m+n} \quad (1)$$

For simplicity, Equation 1 can also be written as follows:

$$\mathbf{u}_i = (u_{i,1}, \cdots, u_{i,m+n}) \in \mathbb{F}_p^{m+n} \quad (2)$$

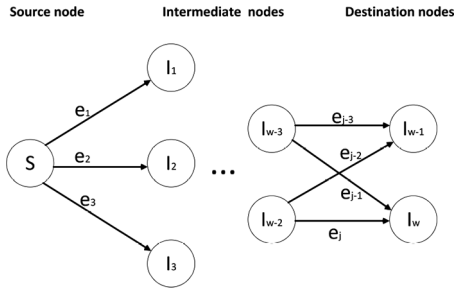

Fig. 1. RLNC model where the proposed scheme is applied.

After that, the source S transmits the coded packets to its neighbor nodes. Each intermediate node buffers its received packets $\mathbf{u}_i$ temporarily and creates a coded packet y, which is a linear combination of a number of h coded packets $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_h$ belonging to the same generation. A coded packet is represented as follows:

$$y = \sum_{i=1}^{h} \alpha_i \mathbf{u}_i \tag{3}$$

where α_i is randomly selected from $\mathbb{F}_p$, and all the arithmetic operations are done over the finite field $\mathbb{F}_p$. A coded packet y is considered to be valid if it is in the linear subspace spanned by the coded packets generated at the source. This is denoted as $y \in span\{\mathbf{u}_1, \cdots, \mathbf{u}_m\}$. In fact, when y is valid, the linear combination coefficients are the first m symbols of the packet y. Otherwise, y is invalid and it is denoted as $y \notin span\{\mathbf{u}_1, \cdots, \mathbf{u}_m\}$, which may be caused due to transmission errors or pollution attacks. At the destination, when a sink node obtains m linearly independent coded packets, it can decode them by using Gaussian eliminations [1].

B. Adversary Model

In our adversary model, the adversary aims to corrupt (i.e., pollute) tags of coded packets belonging to the same generation, and not to multi-generations [12], so that coded packets with legitimate content will be discarded later from next intermediate nodes due to the corrupted tags. It is possible for a coded packet with legitimate content and corrupted tags to travel multiple hops if the intermediate nodes hold the same keys with the adversary or they do not hold those keys that can verify the coded packet and detect the corrupted tags. Otherwise, if the next intermediate nodes hold the keys that verify the coded packet, then the corrupted tags will be detected immediately and the coded packet will be discarded. Moreover, we assume the source node and the destination nodes are trustworthy and secure. Finally, the process of key distribution is considered secure.

III. Null Space-Based Homomorphic MAC Scheme

A. Construction

The proposed homomorphic MAC scheme is based on null space properties [6] for tag generation and verification. The proposed scheme consists of the following five steps:

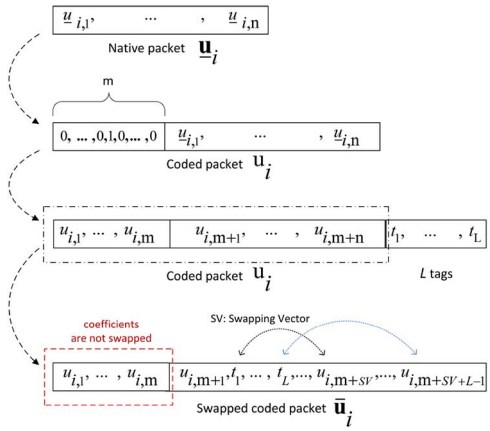

Fig. 2. Swapping process in the proposed null space-based homomorphic MAC scheme.

1) Key Distribution to Source Node: A Key Distribution Center (KDC) distributes L key vectors $\mathcal{K}_1, \mathcal{K}_2, \ldots, \mathcal{K}_L$ to the source node. Each of them is represented in the finite field $\mathbb{F}_{\mathrm{p}}^{m+n+L}$.

2) Tag Generation: The source node S uses the L key vectors $\mathcal{K}_1, \mathcal{K}_2, \ldots, \mathcal{K}_L$ to produce L tags for each coded packet consisting of $m+n$ symbols. These L tags $(t_1, t_2, \ldots, t_L$, where $t_i \in \mathbb{F}_{\mathrm{p}})$ are calculated according to the following formula:

$$\begin{bmatrix} \mathcal{K}_{1,1} & \ldots & \mathcal{K}_{1,m+n} \\ \vdots & \vdots & \vdots \\ \mathcal{K}_{L,1} & \ldots & \mathcal{K}_{L,m+n} \end{bmatrix}_{L*(m+n)} * \begin{bmatrix} u_{i,1} \\ u_{i,2} \\ \vdots \\ u_{i,m+n} \end{bmatrix}_{(m+n)*1} + \begin{bmatrix} \mathcal{K}_{1,m+n+1} & \ldots & \mathcal{K}_{1,m+n+L} \\ \vdots & \vdots & \vdots \\ \mathcal{K}_{L,m+n+1} & \ldots & \mathcal{K}_{L,m+n+L} \end{bmatrix}_{L*L} * \begin{bmatrix} \mathbf{t}_1 \\ \mathbf{t}_2 \\ \vdots \\ \mathbf{t}_L \end{bmatrix}_{L*1} = 0 \tag{4}$$

Afterwards, the source node appends the L calculated tags to the end of the coded packet $\mathbf{u}_i$, which is created by prefixing the native packet $\underline{\mathbf{u}}_i$ with m coefficients, as it is shown in Fig 2.

3) Swapping: To avoid tag pollution attacks, the L tag symbols of the coded packet $\mathbf{u}_i$ are swapped with only L out of the n symbols of the coded packet $\mathbf{u}_i$. It is worthwhile to mention that the coefficients of the coded packet do not participate in the swapping process so that the destination nodes can decode correctly. Particularly, the swapping process is based on a secret value SV (i.e., positive integer) that plays the role of the swapping vector. This secret value is generated randomly by the KDC, through a pseudorandom function, and it is known to the source node and all the destination nodes. However, it is unknown to the intermediate nodes. The result of this swapping process is a swapped coded packet $\overline{\mathbf{u}}_i$, where the L tags symbols are mixed with the n symbols of the coded packet $\mathbf{u}_i$, as it is shown in Fig 2, where $SV = 2$. Each swapped coded packet is represented as follows:

$$\overline{\mathbf{u}}_i = Swap(\mathbf{u}_i)_{SV} \tag{5}$$

On the other hand, at the destination nodes, an inverse swapping is required, before RLNC-decoding, to obtain the native packet.

4) Key Distribution to Intermediate and Destination Nodes: Based on the swapping vector SV, the KDC creates new key vectors $\mathcal{K}'_1, \mathcal{K}'_2, \ldots, \mathcal{K}'_L$ by swapping accordingly the symbols of each key vectors $\mathcal{K}_1, \mathcal{K}_2, \ldots, \mathcal{K}_L$, which have been used by the source node for tag generation. Each new key vector is represented as follows:

$$\mathcal{K}'_i = Swap(\mathcal{K}_i)_{SV} \tag{6}$$

The KDC of our scheme adopts a key distribution model, based on the cover free set systems [13], in order to provide resistance against c compromised nodes. In this model, the maximum number of key vectors that should be assigned to each intermediate and destination node cannot be more than $R = e*ln(1/q)$, where q is a security parameter (usually $q = 10^{-3}$). In our proposed scheme, this assumption is satisfied since only one key vector is required to be assigned by the KDC to each intermediate and destination node. This is why each key vector is orthogonal to the swapped coded packet, and thus the intermediate and destination nodes require only one key vector to verify the swapped coded packet.

5) Verification: Given that each intermediate and destination node holds a key vector, $\mathcal{K}'_i$, they verify a swapped coded packet $\overline{\mathbf{u}}_i$ based on the following formula:

$$\delta = Swap(\mathcal{K}_i)_{SV} * Swap(\mathbf{u}_i)_{SV} = \sum_{j=1}^{m+n+L} \mathcal{K}'_{i,j} * \overline{\mathbf{u}}_{i,j} \tag{7}$$

If $\delta = 0$, then the swapped coded packet $\overline{\mathbf{u}}_i$ is accepted and transmitted to the next nodes. Otherwise, it is discarded.

B. Correctness

Theorem 1 (Correctness) The proposed scheme is correct.

Proof: (Poof by Contradiction) We assume to the contrary that the proposed scheme is not correct. If this is the case, from Equation 7 we have that $\delta \neq 0$. We consider a coded packet $x^i = (x^i_1, \ldots, x^i_{m+n})$ and L key vectors $K = (\mathcal{K}_1, \ldots, \mathcal{K}_L)$, to generate L tags $t^i = (t^i_1, \ldots, t^i_L)$ based on Equation 4. According to Equation 7 and by considering $SV = 1$, we have the following:

$$\begin{aligned}
&Swap(\mathcal{K})_{SV} * Swap(\overline{x}^{i^T})_{SV} \\
&= \begin{bmatrix} \mathcal{K}_{1,1} \ldots \mathcal{K}_{1,m+n+1} \ldots \mathcal{K}_{1,m+n+L} & \mathcal{K}_{1,m+1} \ldots \mathcal{K}_{1,m+L} \\ \vdots & \\ \mathcal{K}_{L,1} \ldots \mathcal{K}_{L,m+n+1} \ldots \mathcal{K}_{L,m+n+L} & \mathcal{K}_{L,m+1} \ldots \mathcal{K}_{L,m+L} \end{bmatrix} \\
&\quad * \begin{bmatrix} \mathbf{x}^i_1 \ \ldots \ t^i_1 \ \ldots \ t^i_L \ \ldots \ \mathbf{x}^i_{m+n} & \mathbf{x}^i_{m+1} \ \ldots \ \mathbf{x}^i_{m+L} \end{bmatrix}^T \\
&= \begin{bmatrix} \mathcal{K}_{1,1} * \mathbf{x}^i_1 + \ldots \mathcal{K}_{1,m+n+L} * t^i_L \\ \vdots \\ \mathcal{K}_{L,1} * \mathbf{x}^i_1 + \ldots \mathcal{K}_{L,m+n+L} * t^i_L \end{bmatrix}_{L*1}
\end{aligned} \tag{8}$$

However, the vector $\overline{x}^i = [x^i_1, \ldots, x^i_{m+n}, t^i_1, \ldots, t^i_L]$ is orthogonal to each of the L key vectors according to Equation 4. Thus, we have the following:

$$\begin{aligned}
K * \overline{x}^{i^T} &= \begin{bmatrix} \mathcal{K}_{1,1} & \ldots & \mathcal{K}_{1,m+n+L} \\ \vdots & \vdots & \vdots \\ \mathcal{K}_{L,1} & \ldots & \mathcal{K}_{L,m+n+L} \end{bmatrix} * \begin{bmatrix} \mathbf{x}^i_1 \\ \mathbf{x}^i_2 \\ \vdots \\ t^i_L \end{bmatrix} \\
&= \begin{bmatrix} \mathcal{K}_{1,1} * \mathbf{x}^i_1 + \ldots \mathcal{K}_{1,m+n+L} * t^i_L \\ \vdots \\ \mathcal{K}_{L,1} * \mathbf{x}^i_1 + \ldots \mathcal{K}_{L,m+n+L} * t^i_L \end{bmatrix}_{L*1} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}_{L*1}
\end{aligned} \tag{9}$$

From Equations 8 and 9, it is obvious that $\delta = 0$. However, this contradicts the original assumption that $\delta \neq 0$. Thus, the proposed construction is correct. According to the inductive reasoning, it can also be proved that the proposed scheme is correct for each SV ($1 \leq SV \leq n$). ■

IV. Security Analysis

We consider that an adversary can wiretap all the swapped coded packets (i.e., $\overline{\mathbf{u}}_i$) of the network and obtain the key vector $\mathcal{K}'_i$ of each compromised node. In this section, we firstly calculate the probability of the adversary to guess the swapping vector SV of the source and then, we calculate the probability of the adversary to launch a tag pollution attack.

A. Probability of Guessing the Swapping Vector SV

SV cannot be obtained from compromised intermediate nodes, since it is known only to the source node and the destination nodes which are considered trustworthy and secure. Also, SV cannot be derived from the transmitted swapped coded packets $\overline{\mathbf{u}}_i$ because they are already swapped and RLNC-encoded. Thus, the adversary can only take a random guess, which has a probability of $\frac{1}{n}$ to be correct. Moreover, the adversary is not able to confirm whether the random guess is correct or not.

B. Probability of Launching a Tag Pollution Attack

To launch a tag pollution attack, an adversary has to corrupt a swapped coded packet and send it to a neighbor node, where it should pass the verification step. According to Equation 7, the corrupted packet will pass the verification in a neighbor node, if this node keeps the same key vector with the adversary. Otherwise, the corrupted packet will be detected and discarded by the neighbor node. According to our key distribution model, each intermediate and destination node is assigned only one key vector out of the L source key vectors. Thus, in the case that the adversary has only one neighbor node, the probability that the adversary has the same key with its neighbor node is not greater than $1/L$. In other words, in this case, the probability that the adversary can launch a tag pollution attack is not greater than $1/L$. However, according to the RLNC model that we have considered for the proposed scheme, the case that the adversary has only one neighbor node is not possible at all. Indeed, it is only a theoretical case that we take into consideration for the sake of completeness.

On the other hand, in the case that the adversary has d neighbor nodes, the probability that the adversary has the same key vector with one of its d neighbor nodes is not greater than $1/L^d$. Consequently, the probability of launching a tag pollution attack in this case is not greater than $1/L^d$, which is negligible. As a result the corrupted packet will be detected

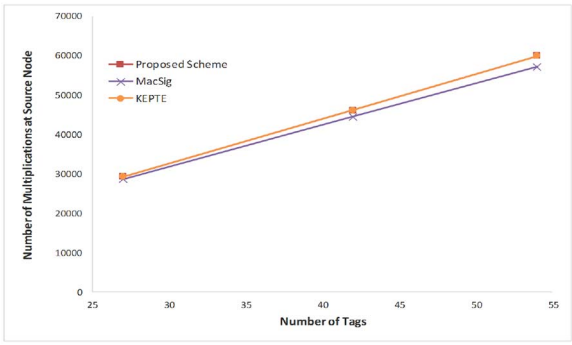

Fig. 3. The number of multiplications required for tag generation at the source node.

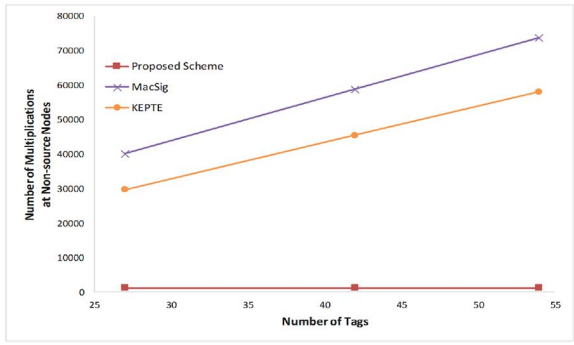

Fig. 4. The number of multiplications required for tag verification at each non-source node.

immediately. This case is realistic according to the RLNC model considered for the proposed scheme.

V. Performance Evaluation

We analyze the performance of our proposed null space-based homomorphic MAC scheme in terms of computational complexity and communication overhead. We follow the settings defined in [7] and thus, we set $p = 2^8$, $n = 1024$, and $m = 32$. We compare our proposed scheme with MacSig [7] and KEPTE [10] which are the most competitive schemes in the literature for providing resistance against tag pollution attacks in RLNC.

A. Computational Complexity

The computational complexity of our proposed null space-based homomorphic MAC scheme is considered for the source and non-source nodes. For the source node, we calculate the complexity of tag generation. For non-source nodes, we calculate the complexity of the verification step.

At source node: The number of tags which is generated at the source node is equal to L. According to Equation 4, $L * (m + n + L)$ finite field multiplications are required for generating the L tags. From Figure 3, we observe that our proposed scheme requires almost the same number of multiplications for tag generation as both the MacSig and KEPTE schemes.

At Non-source nodes: According to Equation 7, the verification step requires $m + n + L$ finite field multiplications. From Figure 4, we observe that our proposed scheme requires much less number of multiplications for tag verification compared to both the MacSig and KEPTE schemes.

B. Communication Overhead

To calculate the communication overhead of our proposed scheme, we take into consideration the number of the tags appended to the end of each coded packet. However, each tag is one symbol of the finite field $\mathbb{F}_p^n$, where $p = 2^8$, and the total number of tags is $L = \frac{1}{1-\delta} e(c+1) \ln(\frac{1}{q})$, as it is shown in [7]. Consequently, the communication overhead of our proposed scheme is $L * \lceil \log_2 p \rceil$ bits per packet. Moreover, for the three different values of L given in [7] (i.e. $L = 27, 42$ and 54), our proposed scheme incurs 2% to 5% communication overhead, which is similar to the one incurred by the KEPTE scheme [10], but half of the communication overhead incurred by the MacSig scheme (i.e., 5% to 10%) [7].

VI. Conclusion

In this letter, we have proposed a null space-based homomorphic MAC scheme providing resistance against tag pollution attacks in RLNC. The performance evaluation of our scheme shows that it is more efficient compared to the MacSig and KEPTE schemes, which are the most competitive tag pollution immune schemes, in terms of computational complexity without incurring additional communication overhead.

References

[1] T. Ho *et al.*, "A random linear network coding approach to multicast," *IEEE Trans. Inf. Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.
[2] R. Ahlswede, N. Cai, S.-Y. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, Jul. 2000.
[3] B. Bollobás, *Graph Theory*. Amsterdam, The Netherlands: Elsevier, 1982.
[4] M. Kim *et al.*, "On counteracting byzantine attacks in network coded peer-to-peer networks," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 5, pp. 692–702, Jun. 2010.
[5] S. Agrawal and D. Boneh, "Homomorphic MACs: MAC-based integrity for network coding," in *Applied Cryptography and Network Security*. Network, NY, USA: Springer, 2009, pp. 292–305.
[6] E. Kehdi and B. Li, "Null keys: Limiting malicious attacks via null space properties of network coding," in *Proc. IEEE INFOCOM*, 2009, pp. 1224–1232.
[7] P. Zhang, Y. Jiang, C. Lin, H. Yao, A. Wasef, and X. Shen, "Padding for orthogonality: Efficient subspace authentication for network coding," in *Proc. IEEE INFOCOM*, 2011, pp. 1026–1034.
[8] Y. Li, H. Yao, M. Chen, S. Jaggi, and A. Rosen, "Ripple authentication for network coding," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.
[9] P. Zhang, Y. Jiang, C. Lin, Y. Fan, and X. Shen, "P-coding: Secure network coding against eavesdropping attacks," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.
[10] X. Wu, Y. Xu, C. Yuen, and L. Xiang, "A tag encoding scheme against pollution attack to linear network coding," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 1, pp. 33–42, Jan. 2014.
[11] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proc. of 41st Annu. Allerton Conf. Communication Control and Computing*, Oct. 2003.
[12] C. Cheng, J. Lee, T. Jiang, and T. Takagi, "Security analysis and improvements on two homomorphic authentication schemes for network coding," *IEEE Trans. Inf. Forensic Secur.*, vol. 11, no. 5, pp. 993–1002, Jan. 2016.
[13] A. Esfahani, D. Yang, G. Mantas, A. Nascimento, and J. Rodriguez, "Dual-homomorphic message authentication code scheme for network coding-enabled wireless sensor networks," *Int. J. Distrib. Sensor Netw.*, vol. 2015, p. e510251, Feb. 2015.