International Journal of Power Control Signal and Computation(IJPCSC)
Vol3. No1. Jan-Mar 2012 ISSN: 0976-268X
www.ijcns.com

Towards privacy preservation Against traffic analysis in Wireless networks

[1]Ms.T.E.Janani Devi, [2]Dr.S.Nagarajan
[1]M.Tech (IT)-Student, Hindustan University
E-mail: jananidevi2010@gmail.com
[2] HOD-IT Dept , Hindustan University
E-mail: nagarajan@hindustanuniv.ac.in

Abstract

Due to the open wireless medium, Multi-hop Wireless Networks MWNs are susceptible to various attacks,such as traffic analysis and flow tracing can be launched by the malicious adversaries. Different from previous schemes, we investigate the privacy issue from a brand new perspective using network coding to achieve privacy preservation, since the coding/mixing operation is encouraged at intermediate nodes network coding has the potential to thwart these attacks. However, the simple deployment of network coding cannot achieve the goal once enough packets are collected by the adversaries. In this paper, we propose a novel network coding based privacy-preserving scheme against traffic analysis in multihop wireless networks. With symmetric key encryption ,Advanced encryption standard (AES) on Global Encoding Vectors (GEVs), the proposed scheme offers two significant privacy-preserving features, packet flow untraceability and message content confidentiality, for efficiently thwarting the traffic analysis attacks. By inverting the GEVs, the proposed scheme keeps the random coding feature, and each sink can recover the source packets with a very high probability .Theoretical analysis demonstrate the validity and efficiency of the proposed scheme.

Index Terms—Network coding, Advanced encryption standard, Global Encoding Vectors, traffic analysis

1. INTRODUCTION

Wireless networks are prevalent everywhere and are commonly implemented because of the ease of deployment and their ability to provide network access to areas where running cable is not an option and also due to their convenience, portability,and low cost.. However, they still suffer inherent shortcomings such as limited radio coverage, poor system reliability, and lack of security and privacy. Multi-hop Wireless Networks are regarded as a highly promising solution for extending the radio coverage range of the existing wireless networks, and they can also be used to improve the system reliability through multi-path packet forwarding. MWNs are susceptible to various attacks, such as eavesdropping, data modification/injection, and node compromising. Since wireless networks do not have defined borders and air waves can penetrate into unintended areas allowing attackers to bypass perimeter firewalls, sniff sensitive information, access the internal network or attack wireless hosts without direct access to the network. These attacks may breach the security of MWNs, including confidentiality,integrity, and authenticity. In addition, some advanced attacks,such as traffic analysis and flow tracing, can also be launched by a malicious adversary to compromise users' privacy, including source anonymity and traffic secrecy. Among all these threats, privacy (especially source anonymity) is of special interest since it cannot be fully addressed by traditional security mechanisms such as encryption and authentication.

Consider a simple example of multicast communication in military ad hoc networks, where nodes can communicate with each other through multi-hop packet forwarding. If an attacker can intercept packets and trace back to the source through traffic analysis, it may disclose some sensitive information such as the location of critical nodes (e.g., the commanders) and then further it may impair the location privacy. Subsequently, the attacker can take a series of actions to launch the so called Decapitation Strike to destroy these critical nodes . Another example of event reporting in sensor networks . When a sensor detects an event, it sends a message including event related information to the base station. If an attacker (the hunter here) can intercept the message, it may know such sensitive information as whether, when and where a concerned event has happened, e.g., the appearance of an endangered animal in a monitoring sensor network . Following this, the attacker can take some action to capture/kill the animal.

In wireless ad hoc networks, passive attacks such as eavesdropping and traffic analysis arise so any malicious entity can sniff the traffic of the victim network. Several privacy-preserving solutions were proposed, such as proxy based schemes, Chaum's mix-based schemes and onion-based schemes, may either require a series of trusted forwarding proxies or result in severe performance to combat the traffic analysis and flow path tracing attacks. Its very challenging task to efficiently thwart traffic analysis and packet flow untraceability and provisioning source anonymity by

providing enhanced privacy in wireless networks.In this paper a novel method to efficiently thwart all these attacks is proposed by network coding. In today's practical communication networks such as the Internet, information delivery is performed by routing. And also a promising generalization of routing is network coding. The potential advantages of network coding over routing include resource (e.g., bandwidth and power) efficiency, computational efficiency, and robustness to network dynamics.

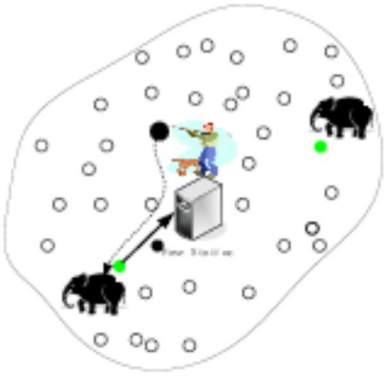


Figure. 1. An application of sensor networks for animal monitoring.

Network coding was first introduced by Ahlswede et al. And has been proposed in the seminal paper to achieve the multicast capacity of the network and has received considerable attention at the theoretical level, Subsequently there are two key techniques, random coding and linear coding, which further promoted the development of network coding. The random coding makes network coding more practical, while the linear coding is proven to be sufficient and computationally efficient for network coding. . Recently, there has been growing interest in exploring the benefits and potential tradeoffs of network coding in practical scenarios.

Network coding has shown higher throughput than conventional multicast theoretically and experimentally. Compared with conventional packet forwarding technologies, network coding offers, by allowing and encouraging coding/mixing operations at intermediate forwarders. network coding provides an intrinsic message mixing mechanism, which implies that privacy preservation may be efficiently achieved in a distributed manner.

Moreover, the unlinkability between incoming packets and outgoing packets, which is an important privacy property for preventing traffic analysis/flow tracing, can be achieved by mixing the incoming packets at intermediate nodes. However, the privacy offered by such a mixing feature is still vulnerable, since the linear dependence between outgoing and incoming packets can be easily analyzed. A simple deployment of network coding cannot prevent traffic analysis/flow tracing since the explicit Global Encoding Vectors (GEVs, also known as tags) prefixed to the encoded messages provide a back door for adversaries to compromise the privacy of users. Once enough coded packets are collected, adversaries can easily recover the original packets and then conduct the attacks based on these packets.Studies show that the potential of homomorphic encryption along with network coding to combat traffic analysis attacks. Homomorphic Encryption is one type of encryption where the arithmetic operations that takes place on cipher text is reflected on the plain text. It hinges on the fact that there is a difference between the number of input packets and the number of output packets attributed to network coding. In existing system they used GEVs with HEF. Homomorphic encryption function offers attractive features like **1**) Enhanced Privacy against traffic analysis and flow tracing, 2) Efficiency, 3) High Invertible Probability even though there are so many advantages their problem is if GEVs rate gets increases HEF will not be efficient**.** Because the key size of HEF is small so large amount of data processing is not possible in HEF with GEV is not possible .Therefore we go for symmetric key algorithm, here we use AES with GEV. In addition, network coding processing introduces delays that differ according to the number of packets encoded. The aforementioned scheme confuse the attacker and protect the network against timing attacks. Hence, protect the privacy of the communicating nodes.

2. NETWORK CODING BASED PRIVACY PRESERVATION SCHEME OVERVIEW OF NETWORK CODING

Each message on an output link must be a *copy* of a message that arrived earlier on an input link. Network coding, in contrast, allows each node in a network to perform some computation. Therefore, in network coding, each message sent on a node's output link can be some *function* or "*mixture*" of messages that arrived earlier on the node's input links, as illustrated in Fig 1. Thus, network coding is generally the transmission, mixing (or encoding), and re-mixing (or re-encoding) of messages arriving at nodes *inside* the network, such that the transmitted messages can be unmixed (or decoded) at their final destinations.

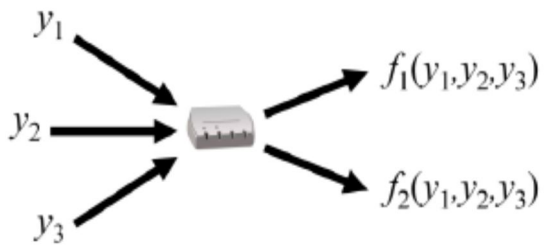


Figure.2 Network Coding: Network nodes can compute functions of input messages

Currently, network coding has been widely recognized as a promising information dissemination approach to improve network performance. Primary applications of network coding include file distribution and multimedia streaming on P2P overlay networks, data transmission in sensor networks , tactical communications in military networks , etc. Compared with conventional packet forwarding technologies, network coding offers, by allowing and encouraging coding/mixing operations at intermediate forwarders .In addition, network coding can work as erasure codes to enhance the dependability of a distributed data storage system .The deployment of network coding in MWNs can not only bring the above performance benefits, but also provide a feasible way to efficiently thwart the traffic analysis/flow tracing attacks since the coding/mixing operation is encouraged at intermediate nodes. several significant advantages such as potential throughput improvement , transmission energy minimization , and delay reduction .

Advantage #1: Maximizing Throughput: Network coding has several advantages over routing. The first is the potential of network coding to improve throughput. Consider the following situation. Two streams of information, both at bit rate B bits per second, arrive at a node, contending for an output link, having capacity B bits per second. With network coding, it may be possible to increase throughput by pushing both streams through the bottleneck link at the same time.

Advantage #2: Minimizing Energy per Bit

There are advantages to network coding beyond maximizing throughput. In particular, network coding can minimize the amount of energy required per packet (or other unit) of information multicast in a wireless network.

Advantage #3: Minimizing Delay

Network coding can also minimize the delay, as measured, for example, by the maximum number of hops for a packet to reach a receiver.

3. RANDOM CODING (MIXING) AT INTERMEDIATE NODES

Unlike other packet-forwarding systems, network coding allows intermediate nodes to perform computation on incoming messages, making outgoing messages be the mixture of incoming ones. This elegant principle implies a plethora of surprising opportunities, such as random coding . As shown in Fig. 2, whenever there is a transmission opportunity for an outgoing link, an outgoing packet is formed by taking a random combination of packets in the current buffer. In practical network coding, source information should be divided into blocks with *h* packets in each block. All coded packets related to the *k*th block belong to generation *k* and random coding is performed only among the packets in the sagime generation. Packets within a generation need to be synchronized by buffering for the purpose of network coding at intermediate nodes.

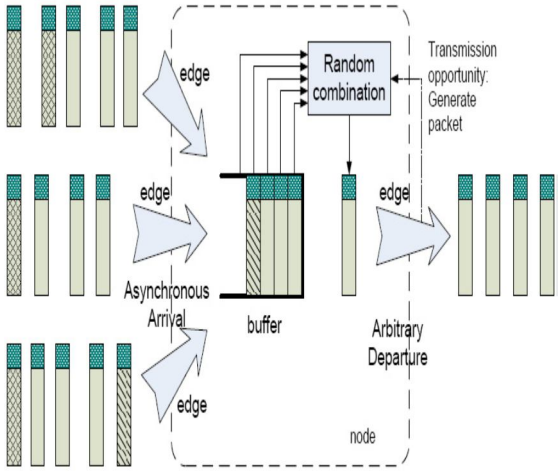


Figure.3. Random coding (mixing) at intermediate nodes.

4. GLOBAL ENCODING VECTORS(GEVs):

Consider an acyclic network (V, *E*, *c*) with unit capacity,i.e., *c*(E) = 1 for all e € *E*, meaning that each edge can carry one symbol per unit time, where *V* is the node set and *E* is the edge set. Assume that each symbol is an element of a finite field F*q*. Consider a network scenario with multicast sessions, where a session is comprised of one source *s*€*V* and a set of

sinks $T \subseteq V$ (or one single sink t $\in V$). Let $h = MinCut(s, T)$ be the multicast capacity, and $x1, \cdots, xh$ be the h symbols to be delivered from s to T . For each outgoing edge e of a node v, let $y(e) \in Fq$ denote the symbol carried on e, which can be computed as a linear combination of the symbols $y(e')$ on the incoming edges e' of node v, i.e., $y(e) = \Sigma e' \beta e'(e)\, y(e')$ The coefficient vector $\beta(e) = [\beta e'(e)]$ is called *Local Encoding Vector* (LEV). By induction, the symbol $y(e)$ on any edge $e \in E$, can be computed as a linear combination of the source symbols $x1, \cdots, xh$ i.e., $y(e) = \Sigma h\, i{=}1\, gi(e)xi$. The coefficients form a *Global Encoding Vector* (GEV) $\mathbf{g}(e) = [g1(e), \cdots, gh(e)]$,which can be computed recursively as $\mathbf{g}(e) = \Sigma e' \beta e'(e)\, g(e')$ using the LEVs $\beta(e)$. Suppose that a sink t $\in T$ receives symbols $y(e1), \cdots, y(eh)$, which can be expressed in terms of the source symbols as

$$\begin{bmatrix} y(e_1) \\ \vdots \\ y(e_h) \end{bmatrix} = \begin{bmatrix} g_1(e_1) & \cdots & g_h(e_1) \\ \vdots & \ddots & \vdots \\ g_1(e_h) & \cdots & g_h(e_h) \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_h \end{bmatrix} = G_t \begin{bmatrix} x_1 \\ \vdots \\ x_h \end{bmatrix} \quad (1)$$

where Gt is called *Global Encoding Matrix* (GEM) and the ith row of Gt is the GEV associated with y(ei). Sink t can recover the h source symbols by inverting Gt and then applying the inverse to $y(e1), \cdots, y(eh)$.

In general, each packet can be considered as a vector of symbols ., $y(e) = [y1(e), \cdots, yN(e)]$.By likewise grouping the source symbols into packets $\mathbf{x}i = [xi,1, \cdots, xi,N,]$,the above algebraic relationships carry over to packets. To facilitate the decoding at the sinks, each message should be tagged with its GEV $\mathbf{g}(e)$, which can be easily achieved by prefixing the ith source packet xi with the ith unit vector $\mathbf{u}i$. Then, each packet is automatically tagged with the corresponding GEV, since

$$[\mathbf{g}(e), y(e)] = = \Sigma e' \beta e'(e)\, [\mathbf{g}(e'), y(e')] \quad (2)$$

The benefit of tags is that the GEVs can be found within the packets themselves, so that the sinks can compute Gt without knowing the network topology or packet-forwarding paths. Nor is a side channel required for the communication of Gt .Actually, the network can be dynamic, with nodes and edges being added or removed in an ad hoc way. The coding arguments can be time varying and random.

5. ADVANCED ENCRYPTION STANDARD(AES)

The algorithm described by AES is a symmetric-key algorithm, meaning the same key is used for both encrypting and decrypting the data. AES is based on a design principle known as a Substitution permutation network. It is fast in both software and hardware. Unlike its predecessor, DES, AES does not use a Feistel network.AES has a fixed block size of 128 bits and a key size of 128, 192, or 256 bits, whereas Rijndael can be specified with block and key sizes in any multiple of 32 bits, with a minimum of 128 bits. The blocksize has a maximum of 256 bits, but the keysize has no theoretical maximum.AES operates on a 4×4 column-major order matrix of bytes, termed the *state* with a larger block size have additional columns in the state.

Most AES calculations are done in a special finite field.The AES cipher is specified as a number of repetitions of transformation rounds that convert the input plaintext into the final output of ciphertext. Each round consists of several processing steps, including one that depends on the encryption key. A set of reverse rounds are applied to transform ciphertext back into the original plaintext using the same encryption key. The main loop of the AES encryption algorithm performs four different operations on the State matrix, called SubBytes, ShiftRows, MixColumns, and AddRoundKey in the specification. The general consensus is that it is the most secure encryption algorithm available.

AES has been subjected to more scrutiny than any other encryption algorithm to date. On both a theoretical and practical basis, AES is considered "secure" in the sense that the only effective way to crack it is through a brute-force generation of all possible keys. With a key size of 256 bits, no known brute-force attack can break AES in a reasonable amount of time (it would take years even on the fastest systems available).

Note that the most likely successful attack on an AES cipher results from a weak implementation that allows what is called a timing attack. The attacker uses different keys and precisely measures the time the encryption routine requires. If the encryption routine is carelessly coded so that execution time depends on the key value, it is possible to deduce information about the key. In AES, this is most likely to occur in the MixColumns routine because of field multiplication. Two safeguards against this attack are to insert dummy instructions so that all multiplications require the same number of instructions, or to implement field multiplication as a lookup table.

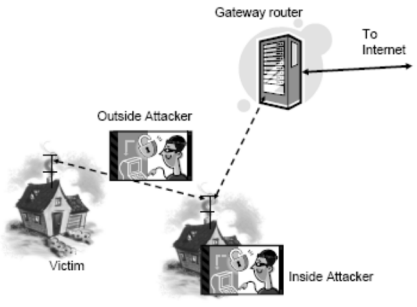
6. THREAT MODELS

We consider the following two attack models.

Outside Attacker: An outside attacker can be considered as a global passive eavesdropper who has the ability to observe all network links as shown in Fig 3(i)(a). An outside attacker can examine the tags and message content, and thus link outgoing packets with incoming packets. Further, even if end to end encryption is applied to messages at a higher layer, itis still possible for a global outside attacker to trace packets by analyzing and comparing the message ciphertext.

Inside attacker: An inside attacker may compromise several intermediate nodes as shown in Fig3(ii)(b). Link-tolink encryption is vulnerable to inside attackers since they may already have obtained the decryption keys and thus the message plaintext can be easily recovered. Both inside and outside attackers may perform more advanced traffic analysis/flow tracing techniques, including size correlation, time correlation, and message content correlation.Adversaries can further explore these techniques to deduce the forwarding paths and thus to compromise user privacy.

Without loss of generality, we assume that an anonymous secure routing protocol is deployed to assist network nodes to determine forwarding paths. The generation number of a packet can be hidden in the secure routing scheme through link-to-link encryption. In this way, attackers cannot find the generation number of a packet for their further analysis. Notice that secure routing paths are only required to be established at the beginning of each session; during the packet transmission, secure routing paths are not required to change or re-established for each new generation.



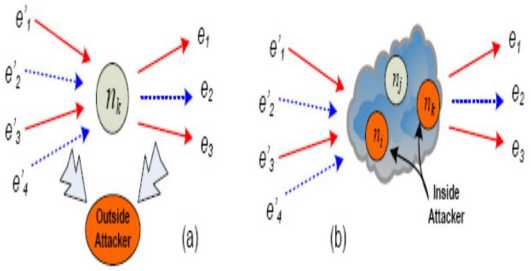


Figure.4 (i)Attack Model Figure.4(ii). (a) outside attacker; (b) inside attacker.

7. RELATED WORK

Several privacy-preserving schemes have been proposed, and they can be classified into three categories: proxy-based schemes, chaum's mix-based schemes, and onion-based schemes. Proxy-based schemes include Crowds and Hordes .The properties offered by Crowds is different from those offered by mixes.Crowds provide (probable innocence) sender anonymity against collaborating crowd members. In contrast, in the closest analog to this attack in typical mix systems[i.e., a group of collaborating mix servers]mixes do not provide sender anonymity but do ensure sender and receiver unlinkability. Another difference is that mixes provide sender and receiver unlinkability against a global eavesdropper.

Crowds does not provide anonymity against global eavesdroppers. However, here is a crowd to spanned multiple administrative domains, where the existence of a global eavesdropper is unlikely. Another difference is that mixes typically rely on public key encryption,the algebraic properties of which have been exploited to break some implementations .

A new protocol for providing anonymous communication on the internet called Hordes, has been proposed which provides a level of comparable anonymity to recent protocols while reducing the amount of work required of participants, as well as significance reducing the latency of data delivery and the link utilization. Hordes achieves these reductions by making use of multicast communication, and is the first protocol designed to provide anonymity that does so. The common characteristic of these schemes is to employ one or more network nodes to issue service requests on behalf of the originator.

Chaum's mix based schemes include MorphMix and Mixminion.And the concept of a MIX-net has been introduced which has been the promising solution for for anonymous communications in the internet. A MIX-net consists of a group of servers, called MIXes (or MIX nodes), each of which is associated with a public key. Each MIX receives

encrypted messages, which are then decrypted, batched, reordered, and forwarded on without any information identifying the sender. And also it has been proved the security of MIXes against a passive adversary who can eavesdrop on all communications between MIXes but is unable to observe the reordering inside each MIX. Recent research on MIX-nets includes stop-and-go MIX-nets , distributed flash MIXes and their weaknesses , and hybrid MIXes .One type of MIX hierarchy is a cascade. In a cascade network, users choose from a set of mixed paths through the MIX-net. Morph Mix is still very much work in progress and has some limitations in its current state.

As anonymous tunnels can fail at any time, the system is best suited for applications making use of several short-lived end-to-end communications such as web browsing. These schemes commonly apply techniques such as shaping, which divides messages into a number of fixed-sized chunks, and mixing, which caches incoming messages and then forwards them in a randomized order. These two techniques can be used to prevent attacks such as size correlation and time correlation. Onion-based schemes include Onion Routing and Onion Ring . The common feature of these schemes is to chain onion routers together to forward messages hop by hop to the intended recipient. Therefore, every intermediate onion router knows only about the router directly in front of and behind itself, respectively, which can protect user privacy if one or even several intermediate onion routers are compromised.

Network coding has privacy-preserving features, such as shaping, buffering, and mixing. However, network coding suffers from two primary types of attacks, *pollution attacks* and *entropy attacks.* The applications built on top of network coding are vulnerable to *pollution attacks*, in which the compromised forwarders can intentionally pollute the transmitted messages or inject the forged messages into networks. These attacks prevent the sinks from recovering the source messages correctly.

A more severe problem is pollution propagation That is, even a small number of polluted messages can quickly propagate into the networks and infect a large proportion of nodes, because each polluted message can be used by all downstream nodes. Therefore, the polluted messages should be detected and filtered as early as possible. *Pollution attacks* can be launched by untrusted nodes or adversaries through injecting faked messages or modifying authentic messages, which are fatal to the whole network due to the rapid propagation of pollution. In *entropy attacks*, adversaries forge non-innovative packets that are linear combinations of “stale” ones, thus reducing the overall network throughput. The vulnerabilities of inter/intra flow network coding frameworks are identified, and general guidelines are provided to achieve the security objectives of network coding systems.

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[12,13],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[12] in has a high computational overhead, but this limitation is overcome in[13] by using probabilistic batch verification in conjunction with a cooperative detection mechanism.

The work in[14] 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[15,16,17,18] 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 [19] which achieves constant-size public key at the cost of using expensive bilinear maps. The scheme presented in [20] 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 [21], or use a distributed protocol which allows receivers to tolerate pollution and recover native packets [22]. 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 – z0, where C is the network capacity from the source to the receiver and z0 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. [23] 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 . 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.

In summary, existing studies on secure network coding mainly focus on detecting or filtering out polluted messages . Little attention has been paid to the privacy issues,especially to protect the encoded messages from tracking or traffic analysis.

8. PERFORMANCE EVALUATION

Computational overhead:

The computational overhead of the proposed scheme can be investigated respectively from three aspects: source encoding,intermediate recoding, and sink decoding. Since the computational overhead of the proposed scheme we will take the Paillier cryptosystem as the encryption method when necessary. Note that the computational overhead is counted independent of the underlying network coding framework.

Source Encoding Overhead: Consider h GEVs with h elements in each GEV, which form an $h \times h$ GEM. After source encoding, every element in the GEM is encrypted one by one. Thus, the computational overhead is $O(h2)$ in terms of encryption operations. Every encryption operation requires 2 exponentiations, 1 multiplication, and 1 modulus operation in the Paillier cryptosystem. Therefore, the computational complexity is $O(h2 \cdot \log n)$ in terms of multiplication operations.

Intermediate Recoding Overhead: In intermediate nodes, linear transformation on the elements of GEVs can be performed only by manipulating the ciphertext of these elements because intermediate nodes have no knowledge of decryption keys. According to Eq. (6), the computational complexity of producing one element in new GEVs is h exponentiations and $h-1$ multiplications on the ciphertext, which is $O(h \cdot \log n)$ in terms of multiplications together. Thus, the computational complexity is $O(h2 \cdot \log n)$ for a GEV and $O(h3 \cdot \log n)$ for a GEM with h GEVs in terms of multiplication.

Sink Decoding Overhead: After receiving an encoded message, a sink can decrypt the elements in the GEV. According to the Paillier cryptosystem, decrypting an element requires1 exponentiation, 1 multiplication, and 1 division operation.
Therefore, the computational complexity of decrypting a GEV is $O(h \cdot \log n)$ in terms of multiplication operations. Thus, for a whole GEM with h GEVs, the computational overhead is $O(h2 \cdot \log n)$ in terms of multiplication.

Communication overhead:

Let h messages be generated, and each message is of length $\cdot$ bits. For source encoding, each message is prefixed with h codewords from a ring of size $\cdot$. Considering the ciphertext expansion of the Paillier cryptosystem, we can calculate the communication overhead as $2h \cdot \log n / l$.

Performance optimization:

As described in the previous subsections, the invertible probability and computational overhead of the proposed scheme are $1- p(p^{-1} + q^{-1})$ and $O(h3 \cdot \log n)$, respectively. Thus, the statistical computational overhead for a GEM can be expressed in terms of multiplications as follows:

$$CO = \frac{h^3 \cdot \log n}{1 - t(p^{-1} + q^{-1})} \quad (3)$$

From Eq., we can see that the computational overhead of the proposed scheme is a monotonically increasing function of h, i.e., the length of a GEV, for any given n and t. As discussed in Section IV, the security of the proposed scheme is also monotonically increasing with the increase of h. Thus, a tradeoff between the security and the computational overhead should be considered in practical deployment. A typical way to deal with this tradeoff is to set the security requirements first and then choose the minimum h to meet the requirements. In this way, the minimum computational overhead can be achieved. On the other hand, noticing that $n = pq$ and $|p| \approx |q|$, we can approximate Eq. for any given h and formulate it as the following optimization problem:

$$\begin{aligned} Minimize \quad & g(n,t) = \frac{log\ n}{1-2t/\sqrt{n}} \\ subject\ to: \quad & n \geqslant 4,\ n\ is\ an\ integer, \\ where \quad & t \geqslant 1,\ t\ is\ an\ integer. \end{aligned} \tag{4}$$

By solving the ordinary differential equation $\partial g/\partial n = 0$,we have: $n1 = 4t\hat{}2\ LambertW\hat{}2(-(2e \cdot t)-1)$ and $n2 = 4t\hat{}2\ LambertW\hat{}2((2e \cdot t)-1)$ .Since the *Lambert-W* function has infinite branches in the complex plane, we only consider the branches which have real-valued solutions with real arguments. In addition, since $t \geq 1$(t is the total coding time) and the *Lambert-W* function is single-valued in the real plane for a positive argument, we can determine that $t2 < 4t\hat{}2 \cdot ((2e \cdot t)\hat{}-1)\hat{}2 = e\hat{}-2$, which is in conflict with the condition $n \geq 4$. Thus, n2 can be excluded for further consideration. $n1$ is double-valued in the real plane since the argument of the function LambertW(x),x $= -(2e \cdot t)\hat{}-1$, is in the region of $(-1/e, 0)$. We denote the double-valued results as

$$n_1(k,t) = 4t^2 LambertW^2(k, \frac{-1}{2e \cdot t}),\ k = -1, 0, \tag{5}$$

where k can be any integer in the complex plane. For a realvalued solution of n1, k can only be 0 and -1. Similarly, we can determine that $n1(0, t) < n1(0, t)|t=1 = 0.215$ and this principal value is not in accord with the prescribed condition $n \geq 4$ Finally, we can determine that the result $n1(-1, 5)$ is the point where the objective function $g(n,t)$ achieves its minimum for any given parameter t. For example, given $t = 5$, we can get three real-valued results as follows: $n1(-1, 5) = 2390.936$, $n1(0, 5) = 0.146$, and $n2(0, 5) = 0.126$, where only $n1(-1, 5)$ meets the prescribed condition $n \geq 4$. After obtaining the minimum point, we can find the closest positive integer n which is the product of two primes p and q, i.e., n = pq. The integer n can then be substituted into Eq. to achieve the minimum computational overhead.

9. CONCLUSION:

In this paper, we have proposed an efficient network coding based privacy-preserving scheme against traffic analysis and flow tracing in multi-hop wireless networks. With the Advanced encryption standards(AES) on Global Encoding Vectors (GEVs), the proposed scheme offers two significant privacy preserving features, packet flow untraceability and message content confidentiality, which can efficiently thwart traffic analysis/flow tracing attacks.The proposed scheme keeps the essence of random linear network coding, and each sink can recover the source messages by inverting the GEVs with a very high probability.The quantitative analysis and simulative evaluation on privacy enhancement and computational overhead demonstrate the effectiveness and efficiency of the proposed scheme.

REFERENCES

[1] M. Shao, Y. Yang, S. Zhu, and G. Cao, "Towards statistically strong source anonymity for sensor networks," in *Proc. IEEE INFOCOM'08*, pp. 51-55, 2008.

[2] M. K. Reiter and A. D. Rubin, "Crowds: anonymity for web transactions,"*ACM Trans. Inf. and System Security*, vol. 1, no. 1, pp. 66-92,Nov. 1998.

[3] C. Shields and B. N. Levine, "A protocol for anonymous communication over the Internet," in *Proc. ACM CCS'00*, pp. 33-42, 2000.

[4] M. Rennhard and B. Plattner, "Introducing MorphMix: peer-to-peer based anonymous Internet usage with collusion detection," in *Proc. ACM Workshop on Privacy in the Electronic Society*, pp. 91-102, 2002.

[5] G. Danezis, R. Dingledine, and N. Mathewson, "Mixminion: design of a type III anonymous remailer protocol," in *Proc. IEEE Symposium on Security and Privacy*, pp. 2-15, May 2003.

[6] D. Goldschlag, M. Reed, and P. Syverson, "Onion routing for anonymous and private Internet connections," *Commun. ACM*, vol. 42, no. 2,pp. 39-41, Feb. 1999.

[7] X. Wu and N. Li, "Achieving privacy in mesh networks," in *Proc. ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN'06)*, pp.13-22, 2006.

[8] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204-1216, July 2000.

[9] J. Dong, R. Curtmola, and C. Nita-Rotaru, "Secure network coding for wireless mesh networks: threats, challenges, and directions," *Computer Commun. (Elsevier)*, vol. 32, no. 17, pp. 1790-1801, Nov. 2009.

[10] N. Cai and R. W. Yeung, "Secure network coding," in *Proc. IEEE ISIT'02*, 2002.

[11] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, M. Medard, and M. Effros, "Resilient network coding in the presence of Byzantine adversaries," *IEEE Trans. Inf. Theory*, vol. 54, no. 6, pp. 2596-2603, 2008.

[12] 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.

[13] C. Gkantsidis and P. Rodriguez Rodriguez. Cooperative security for network coding file distribution. In Proc. Of INFOCOM 2006.

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

[15] D. Charles, K. Jain, and K. Lauter. Signatures for network coding. In Proc. of CI ’06, 2006.
[16] 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.
[17] F. Zhao, T. Kalker, M. Medard, and K. Han. Signatures for content distribution with network coding. In Proc. Of ISIT ’07.
[18] 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.
[19] D. Boneh, D. Freeman, J. Katz, and B.Waters. Signing a linear subspace: Signature schemes for network coding. In Proc. of PKC ’09, 2009.
[20] Z. Yu, Y.Wei, B. Ramkumar, and Y. Guan. An efficient signature scheme for securing xor network coding against pollution attacks. In INFOCOM, 2009
[21] 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.
[22] 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.
[23] D. Wang, D. Silva, and F. R. Kschischang. Constricting the adversary: A broadcast transformation for network coding. In Allerton 2007, 2007.