© 2013 IEEE. Reprinted, with permission, from S. Pfennig and E. Franz, **Secure Network Coding: Dependency of Efficiency on Network Topology**, in *IEEE International Conference on Communications (ICC 2013)*, pp. 2100-2105, 2013 June 9-13.

This material is posted here with permission of the IEEE. Such permission of the IEEE does not in any way imply IEEE endorsement of any of the products or services of Technical University Dresden. Internal or personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution must be obtained from the IEEE by writing to pubs-permissions@ieee.org. By choosing to view this document, you agree to all provisions of the copyright laws protecting it.

Secure Network Coding: Dependency of Efficiency on Network Topology

Stefan Pfennig and Elke Franz

Technische Universität Dresden, Faculty of Computer Science

01062 Dresden, Germany

Email: {stefan.pfennig|elke.franz}@tu-dresden.de

Abstract—Network Coding is a new possibility to transmit data through a network. By combining different packets instead of simply forwarding, network coding offers the opportunity to reach the Min-Cut/Max-Flow capacity in multicast data transmissions. However, the basic schemes are vulnerable to so-called pollution attacks, where an attacker can jam large parts of the transmission by infiltrating only one bogus message. In the literature we found several approaches which aim at handling this kind of attack with different amounts of overhead. Though, the cost for a specific secure network coding scheme highly depends on the underlying network. The goal of this paper is on the one hand to describe which network parameters influence the efficiency of a certain scheme and on the other hand to provide concrete suggestions for selecting the most efficient secure network coding scheme considering a given network. We will illustrate that there does not exist "the best" secure network scheme concerning efficiency, but all selected schemes are more or less suited under certain network topologies.

I. INTRODUCTION

Network Coding as introduced by a Ahlswede et al. [2] is a simple and elegant way to increase throughput in an arbitrary network for a multicast scenario. It was shown by Li et al. that linear network coding suffices to achieve the Min-Cut/Max-Flow bound, which is the maximum for a single flow [16]. However, Network Coding also has its drawbacks. Due to the design, it is highly vulnerable to so-called pollution-attacks, where an active attacker can jam large parts of the network by sending only one bogus message.

Because of this fact, there has been a lot of research into security of Network Coding in the past few years. Whereas some analysis deal with confidentiality [19], the vast majority deals with preventing pollution attacks and keeping integrity and availability of messages [3], [10], [12], [18]. Common cryptographic solutions do not work, since data packets are combined and therewith digital signatures or similar things become invalid. That is the reason for using homomorphic cryptographic functions, like homomorphic signatures [8], [13], homomorphic message authentication codes (MACs) [1], [21], or homomorphic hashes [9], [14]. Another possibility to cope with secure network coding is to utilize time asymmetry or more precisely to delay the release of data necessary for verifying a data packet [5], [15], [17].

This work is supported by the German Research Foundation (DFG) in the Collaborative Research Center 912 "Highly Adaptive Energy-Efficient Computing".

Considering the variety of existing network coding schemes preventing pollution attacks, the question arises which of these schemes should be used. There are two important factors that influence the answer: security and efficiency. Within this paper, we focus on the efficiency aspect.

In the past we did some analysis of existing schemes and compared the efficiency parameters of some prototypic schemes. Although we only analyzed 3 specific network graphs, the evaluations have shown that the efficiency highly depends on the network [7]. The contribution of this paper is to determine which scheme is best suited concerning efficiency for a chosen network characterized by few parameters.

Within our analysis we focus on the dependency on the network topology. Thus, we make simplifying assumptions regarding the implementation on a specific machine.

Section II gives an overview on Network Coding and introduces all schemes considered in the evaluation of this paper. In Section III, we present several parameters for describing efficiency and the network. The results of our analysis are presented in Section IV. Section V concludes and gives an outlook on further work.

II. OVERVIEW ON NETWORK CODING

A. Random Linear Network Coding

Network Coding is applied to a network given as a finite, directed, acyclic graph $G = (V, E)$ with a sending node (source) $s \in V$, forwarding nodes (relays) $F \subset V$ and receiving nodes (sinks) $R \subset V$. For simplicity, we assume each channel $e \in E$ to be of unit capacity. Data packets $\mathbf{x}_i$ sent through the network consist of n symbols $x_{i,j} \in \mathbb{F}_q$ each.

Unlike traditional routing, where a node simply passes a data packet $\mathbf{x}_i$ to the next node, a node computes various combinations of its incoming data packets and sends them to next nodes. On receiving l incoming messages, a forwarding node randomly selects coefficients $\alpha_i \in \mathbb{F}_q$, $i = (1, \dots, l)$ and computes a linear combination of the received messages

$$\mathbf{x}_j = \sum_{i=1}^l \alpha_i \cdot \mathbf{x}_i. \quad (1)$$

that will be sent on its outgoing edges j .

For a practical implementation, we refer to Practical Network Coding (PNC) [4]. PNC is a framework that enables all receiving nodes R to decode the data packets without knowing

the coefficients α_i randomly chosen by the forwarding nodes F. The sender s divides the data into portions $\bar{p}_i \in \mathbb{F}_q^m$ of m symbols each. Furthermore, s constructs global encoding vectors $\beta_i = (\beta_{i,1}, \beta_{i,2}, \dots, \beta_{i,h}) \in \mathbb{F}_q^h$, where $\beta_{i,j} = 1$ if $i = j$ and 0 otherwise. The data portions $\bar{p}_i$ amended by the global encoding vector β_i establish the native data packets $x_i = [\beta_i, \bar{p}_i]$. The value h represents the network capacity; h data packets can be transmitted at once through the network using network coding. These h data packets form a generation $G = [B, P]$ with a unique identifier gid . All data packets of one generation G can be arbitrarily combined according to equation (1) on the flow to the receiving nodes R.

At the end of a transmission, all receiving nodes $r \in R$ will have at least h data packets which form a matrix. If the rank of this matrix is h , the receiving nodes can successfully decode the message by solving a system of linear equations. Although full rank at the receiving nodes cannot be guaranteed, the probability for decodability depends on the field size q . For normal PNC it is sufficient to use at least $q = 2^8$ [11].

B. Selected Schemes Secure against Pollution Attacks

The solely implementation of PNC is vulnerable to pollution attacks. All data packets of a generation G span a linear subspace which is left if even one data packet is polluted and does not lie in this subspace anymore. Thus, the receiving nodes cannot decode messages. This necessitates the forwarding nodes F to recognize corrupted packets.

There are plenty of different schemes developed which cannot entirely be addressed here, thus, we focused on 4 prototypic and substantially different schemes which should represent main approaches. In the following, the selected schemes are shortly explained to get a rough idea how they work. For a more detailed explanation, we refer to the given sources.

a) *Homomorphic Hashes* [9]: The main idea is to use a homomorphic hash function and distribute digitally signed packets with hashes of the native data packets a priori. Hence, the sender computes for each native data packet $\bar{p}_i$ a hash value $h(\bar{p}_i)$. Forwarding nodes can verify the combined data packets $x_i = [\beta_i, \bar{p}_i]$ by checking

$$h(\bar{p}_i) \stackrel{?}{=} h(\bar{p}_1)^{\beta_{i,1}} \cdot h(\bar{p}_2)^{\beta_{i,2}} \cdot \dots \cdot h(\bar{p}_h)^{\beta_{i,h}} \bmod r.$$

The hashes are 128 byte each, whereas the field size of the codewords is about $q \approx 2^{256}$.

b) *DART* [5]: DART uses time asymmetry to deliver delayed checksums of a generation. The network coding works as in PNC. Additionally, the sender periodically computes and broadcasts checksum packets containing a checksum $\text{chk}_s(G)$, a seed s , and a timestamp t for the current generation G . The checksum is computed based on a pseudo random matrix H_s that can be derived from the seed s . Each node buffers received data packets until a checksum packet arrives. Only data packets received before the creation of a checksum can be verified by this checksum. For verification, the node generates H_s and checks if the product of checksum and global encoding vector

equals the product of random matrix and encoded data

$$\beta_i \text{chk}_s(G) \stackrel{?}{=} \bar{p}_i H_s.$$

c) Homomorphic MACs based on Time Asymmetry [20]:

The concept of this scheme is to utilize symmetric authentication by means of homomorphic MACs (Message Authentication Codes). The necessary key exchange is realized by means of delayed key release. The MACs are integrated into the data packets. The sender computes a chain of seed values that are the basis for computing the keys necessary for computing the MACs. The final value of this chain, which is important for checking the validity of the other seed values, is digitally signed and broadcasted to all nodes.

The remaining seed values are periodically broadcasted to the other nodes for verifying the data packets. After checking the seed value, the node computes the necessary keys and checks the MACs. The data packets with MACs are combined according to formula (1). For security reasons, the size of a symbol is increased to $q = 2^{128}$.

d) *RSA-based scheme* [8]: This scheme takes advantage of the homomorphic property of the basic RSA signature scheme. The sender computes a digital signature for each data packet and integrates it into the particular packet.

Forwarder and receiver can verify signatures with the sender's public test key. Due to the homomorphic property, nodes can combine packets and compute valid signatures for the combined data packets without knowing the private signing key. The signature is computed modulo a composite number N of 2048 bit. In contrast to the other schemes, the data symbols are integers instead of finite field elements. Thus, every data packet will grow by the number of hops k and the number of ingoing edges, which we roughly substitute with the multicast capacity h .

III. INFLUENCE OF NETWORK CHARACTERISTICS ON EFFICIENCY

A. Parameters for Describing Efficiency

Efficiency is a rather vague term. In this context, efficiency considers additional operations, memory and communication overhead in comparison to ordinary Network Coding. However, computational time, memory usage or latencies highly depend on the implementation and on the underlying system. Since we aim at describing the influence of network topologies on efficiency, we selected the 3 following parameters according to [7].

The *relative payload* P describes the ratio of the payload data which can be decoded at all receiving nodes R to all data sent by the sending node s including also additional data, e.g., hashes, checksums, MACs, or digital signatures. This data overhead has to be sent either in extra data packets or is integrated into existing data packets. Depending on the network topology, it is often necessary to send data packets or extra packets multiple times on different links, which increases the overhead again. Hence, this parameter should assess the overall effort for the sender.

For comparing the time needed for an transmission we evaluate the *number of ticks* T necessary for sending the messages to all receiving nodes R . We define that within one tick (time slice) a node can process all necessary packets and send on all outgoing edges concurrently.

Comparing the number of ticks necessary for transmitting one generation does not yield meaningful results since the payload per data packet strongly depends on the applied scheme. Hence, it is more reasonable to evaluate the number of ticks needed for a certain amount of data B .

For all schemes, we did not consider any pipelining methods, but assume sending generations sequentially. Hence, we need to divide B by the payload per packet to get the number of generations and multiply this with the number of ticks per generation. The result should highly correlate with the actual delay and the time necessary for transmission.

Another method to describe the communication overhead is simply to count all *send operations* O in the network for the transmission of all necessary data. The number of sending operations O mainly depends on the number of edges $|E|$. However, this is a constant factor for all schemes so we can set them to any positive integer and the difference between the schemes will stay proportionally. For the same reason like in number of ticks, we used a special amount of data B to ensure comparability.

Considering a constant packet size, the number of send operations estimates the communication overhead more realistic than the relative payload. Despite this, the number of send operations rates the overall effort within the whole network.

B. Parameters for Characterizing the Network

A network is characterized by many parameters, like numbers of nodes, edges, receivers, incoming edges, etc. Based upon the results in [7], these parameters may influence the efficiency of a secure network coding schemes to a different degree. Thus, we can reduce the parameters for characterizing the network to the most significant ones for keeping the evaluation manageable.

At the end, we only analyzed the maximum path length from the sending node s to any receiving node r_i , i.e., the number of hops k and the multicast capacity h , which can be determined via a Min-Cut/Max-Flow algorithm. Other network parameters, e.g., number of edges $|E|$ like mentioned before surely have an impact on the efficiency parameters, too. However, their influence can be described by a constant factor so the relation between the schemes is not changed.

For the RSA-based scheme it is also important to know the average number of incoming edges for each node. We can roughly substitute this value with the multicast capacity h . The same applies to the number of outgoing edges of the sending node that is also approximated by h .

All in all, we have only two main parameters to consider when we would like to know which of the secure network coding schemes has the best efficiency regarding a certain network topology. Although these assumptions are reasonable, they have to be checked before applying to a given network.

C. General Assumptions

The packet size n is of high importance for all schemes because the overhead within one packet or transmission stays static in most cases so that the packet size directly influences the payload per data packet. But setting the packet size too large will introduce too high delays as well. Thus, we assumed $n = 1500$ byte according to the size of an IP packet.

The generation id gid must be large enough to prevent replay attacks (infiltrating an old data packet into a new generation to jam the network) but also small enough to keep the overhead acceptable. Therefore, we assume a size of 8 bytes for an ascending gid . Further details about the formats of the packets can be found in [6].

Furthermore, some parameters are responsible for the security level of the network coding schemes. We assume a size of 128 byte for the digital signatures as reasonable. For the RSA-based scheme we use a composite modulus of size 2048 bit. Other scheme specific parameters, e.g. size of the finite field, the hashes, the checksum, etc., are set to those values recommended in the original papers.

IV. EVALUATING THE SELECTED SCHEMES

A. Dependencies

We analyzed all selected schemes in order to derive formulas describing the efficiency parameters dependent on the network parameters k and h . We will shortly explain how we derived the formulas by the example of the Homomorphic Hash scheme. The other formulas are derived in a similar manner; they are given in the appendix.

The relative payload depends on the payload per generation and the amount of data to be sent for the sender s at all. A data packet for the homomorphic hash scheme contains the gid (8 byte), the global encoding vector (32 byte $\cdot h$ since $q \approx 2^{256}$), and the payload data. One generation contains h data packets. Additionally, we need h (for each direct subsequent node) extra packets with hashes (128 byte $\cdot h$), and a signature plus gid for each extra packet (136 byte). The number of extra packets can be computed by dividing the space for the hashes by the packet size less the signature plus gid . The corresponding formula for relative payload P is:

$$P_{HH} = \frac{h(n - 32h - 8)}{h(n + 128h + \left\lceil \frac{128h}{n-136} \right\rceil * 136)}$$

The number of ticks T mainly depends on the number of hops k . For homomorphic hashes, we have to transmit at least one hash packet in advance. Thus, we have to sum up k and the number of hash packets. Additionally, we need to get the number of generations by dividing the amount of data we want to transmit by the payload data per data packet:

$$T_{HH} = \left\lceil \frac{B}{(n - 32h - 8)} \right\rceil * \left(k + \left\lceil \frac{128h}{n - 136} \right\rceil \right)$$

The number of sending operations O has three factors: the number of generations, the number of sending operations per edge in one generation, and the number of edges $|E|$. Each

Fig. 1. Results for relative payload P . Figures (a-d) show the reachable payload for the individual schemes. Figure (e) shows which schemes are best suited for a pair (h, k) . Figure (f) shows the corresponding payload when using the best suited scheme.

node has to send one data packet and at least one hash packet per generation. Hence, we get:

$$O_{HH} = \left\lceil \frac{B}{(n - 32h - 8)} \right\rceil * \left(1 + \left\lceil \frac{128h}{n - 136} \right\rceil \right) * |E|$$

Based on these formulas, we computed the efficiency parameters for $h \in [1, 50]$ and $k \in [1, 200]$. The corresponding diagrams are displayed in Figure 1 (relative payload), Figure 2 (number of ticks) and Figure 3 (send operations).

Thereby, the first 4 diagrams always show the individual results for each scheme. Further, we evaluated for each pair of parameters (h, k) which scheme is the most efficient and mapped it to the diagram (e). At the end, we combined all diagrams and show what efficiency level is achievable when always using the best scheme.

Generally, increasing values for k and h imply an increased number of send operations as well as an increased number of ticks, but a decreased relative payload. To assure a good visibility of the results, the axis of the diagrams are arranged differently in Figure 1 and Figure 2 and 3. Furthermore, we chose for the latter $B = 20000$ Bytes to compute the absolute values. However, increasing or decreasing B will not change the shape of the diagrams or the map of best schemes, but only the absolute values for number of ticks and send operations.

B. Discussion

As to be expected, there is no scheme that achieves the best relative payload P in all network topologies. The payload

decreases for increasing h at Homomorphic Hashes, whereas the maximum number of hops k does not have any influence. In contrast to this, the payload of the DART scheme is highly dependent on k , so that h has minor importance. For Homomorphic MACs and the RSA-based scheme both increasing of k and h decreases the payload. In addition, we found out that there exists combinations of h and k where both schemes are unable to communicate, i.e., the payload is zero (For larger values h and k this could also happen to Homomorphic Hashes and DART). Surprisingly, all 4 schemes have their right to exist, since each of them is superior to the other schemes for certain pairs (h, k) . This fact produces an interesting diagram for the overall best, where both network parameters influence the payload. Though, using the best scheme will result in a small relative payload P for large h and k anyway. Certainly, we have to consider that a higher h also means more data at the recipients, which compensates the low relative payload.

The different suitability of the schemes for varying h and k continues with number of ticks T . For Homomorphic Hashes, again mainly depends on the multicast capacity h ; but in contrast to the relative payload, the number of hops k has also a moderate influence. In DART, the number of hops k is the main influence on T again. Homomorphic MACs and the RSA-based scheme are influenced by both network parameters h and k , again. However, for Homomorphic MACs the multicast capacity is more relevant for the ability to send than for amount of ticks, because we have combinations of

Fig. 2. Results for number of ticks T ($B = 20000$ byte). Figures (a-d) show the number of ticks for the individual schemes. Figure (e) shows which schemes are best suited for a pair (h, k) . Figure (f) shows the corresponding number of ticks when using the best suited scheme.

Fig. 3. Results for send operations O ($B = 20000$ byte). Figures (a-d) show the number of send operations for the individual schemes. Figure (e) shows which schemes are best suited for a pair (h, k) . Figure (f) shows the corresponding number of send operations when using the best suited scheme.

h and k again where both schemes fail to work, since the overhead data would exceed the packet size. Altogether, this implies for the map of the best suitability of the schemes that we have only 3 groups and the Homomorphic MAC scheme does not show best performance for any combination of h and k . At the end, we can show a diagram of the overall best which mainly is a combination of Homomorphic Hashes and DART regardless of a small area ($k < 10$ and large $h > 10$) where the RSA-based scheme performs best.

Except for minor changes, the main statement remains for the send operations O . Whereas Homomorphic Hashes are still mostly dependent on the multicast capacity h , the send operations for DART are mainly influenced by k . Homomorphic MACs and the RSA-based scheme will also not work for certain values of h and k and both schemes are dependent on both parameters. The diagram of the best suited scheme shifts slightly. Now DART's performance is only better than the performance of Homomorphic Hashes for $h > 40$. A small area where the RSA-based scheme suits best still remains.

V. CONCLUSION

To conclude, the maps of best suited schemes are similar for all efficiency parameters. For a moderate multicast capacity h and a high number of hops k , the Homomorphic Hashes seems to be best suited. For the contrary, i.e., large multicast capacity h and low number of hops k , one should prefer the RSA-based scheme. And for both large h and k DART is the best solution. However, even Homomorphic MACs can provide best efficiency if you mainly consider the relative payload and adopt it for a network with h and k around 30.

For using the schemes in practice, a deeper security analysis is necessary. A further topic of future work is an efficiency analysis that takes computational time or memory space into account. We are currently working on a network simulator that is necessary for these evaluations.

APPENDIX

a) Relative payload:

$$P_{\text{DART}} = \frac{h(n - h - 8)}{h(n + k(142 + 2h))}$$

$$P_{\text{HMAC}} = \frac{h(n - 16(h + k) - 8)}{h(n + 24k + 152)}$$

$$P_{\text{RSA}} = \frac{(n - 264 - k(h + 5)(1 + \frac{\lceil \text{ld}(h) \rceil}{8}))}{n}$$

b) Number of ticks:

$$T_{\text{DART}} = \left\lceil \frac{B}{(n - h - 8)} \right\rceil * (3k - 1)$$

$$T_{\text{HMAC}} = \left\lceil \frac{B}{(n - 16(h + k) - 8)} \right\rceil * (0.5k^2 + 1.5k)$$

$$T_{\text{RSA}} = \left\lceil \frac{B}{(n - 264 - k(h + 5)(1 + \frac{\lceil \text{ld}(h) \rceil}{8}))} \right\rceil * k$$

c) Send operations:

$$O_{\text{DART}} = \left\lceil \frac{B}{(n - h - 8)} \right\rceil * (k + 1) * |E|$$

$$O_{\text{HMAC}} = \left\lceil \frac{B}{(n - 16(h + k) - 8)} \right\rceil * (k + 2) * |E|$$

$$O_{\text{RSA}} = \left\lceil \frac{B}{(n - 264 - k(h + 5)(1 + \frac{\lceil \text{ld}(h) \rceil}{8}))} \right\rceil * |E|$$

REFERENCES

- [1] S. Agrawal and D. Boneh, "Homomorphic MACs: MAC-based integrity for network coding," in *Applied Cryptography and Network Security*, 2009, pp. 292 – 305.
- [2] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.
- [3] N. Cai and R. W. Yeung, "Secure Network Coding," in *Proc. IEEE Int. Symp. on Information Theory*, 2002.
- [4] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proc. Annual Allerton Conference on Communication, Control, and Computing*, 2003.
- [5] J. Dong, R. Curtmola, and C. Nita-Rotaru, "Practical Defenses Against Pollution Attacks in Intra-Flow Network Coding for Wireless Mesh Networks," in *Proc. WiSec*, 2009.
- [6] E. Franz, S. Pfennig, and A. Fischer, "Communication overhead of network coding schemes secure against pollution attacks," TU Dresden, Tech. Rep. TUD-FI12-07-Mai2012, 2012.
- [7] —, "Efficiency of secure network coding schemes," in *Proc. Communications and Multimedia Security*, 2012, pp. 145–159.
- [8] R. Gennaro, J. Katz, H. Krawczyk, and T. Rabin, "Secure Network Coding over the Integers," in *Proc. PKC 2010*, 2010, pp. 142–160.
- [9] C. Gkantsidis and P. R. Rodriguez, "Cooperative Security for Network Coding File Distribution," in *Proc. IEEE Int. Conf. on Computer Communications*, 2006.
- [10] T. Ho, B. Leong, R. Koetter, M. Médard, M. Effros, and D. R. Karger, "Byzantine Modification Detection in Multicast Networks with Random Network Coding," *IEEE Trans. on Information Theory*, vol. 54, no. 6, pp. 2798–2803, 2008.
- [11] T. Ho, R. Koetter, M. Médard, D. R. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," in *Proc. of the IEEE International Symposium on Information Theory*, 2003.
- [12] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, and M. Médard, "Resilient network coding in the presence of byzantine adversaries," in *Proc. 26th Annual IEEE Conf. on Computer Commun., INFOCOM*, 2007, pp. 616–624.
- [13] M. Kim, L. Lima, F. Zhao, J. Barros, M. Mdard, R. Koetter, T. Kalker, and K. J. Han, "On counteracting byzantine attacks in network coded peer-to-peer networks," *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 5, pp. 692–702, 2010.
- [14] M. Krohn, M. Freedman, and D. Mazières, "On-the-Fly Verification of Rateless Erasure Codes for Efficient Content Distribution," in *Proc. IEEE Symp. on Security and Privacy*, 2004.
- [15] A. Le and A. Markopoulou, "TESLA-based defense against pollution attacks in p2p systems with network coding," in *International Symposium on Network Coding (NetCod)*, 2011.
- [16] S.-Y. R. Li and R. W. Yeung, "Linear network coding," *IEEE Trans. on Information Theory*, vol. 49, no. 2, pp. 371–381, 2003.
- [17] Y. Li, H. Yao, M. Chen, S. Jaggi, and A. Rosen, "RIPPLE authentication for network coding," in *Proc. of INFOCOM'10*, 2010, pp. 2258 – 2266.
- [18] L. Lima, J. P. Vilela, P. F. Oliveira, and J. Barros, "Network coding security: Attacks and countermeasures," *CoRR*, vol. abs/0809.1366, 2008.
- [19] J. P. Vilela, L. Lima, and J. Barros, "Lightweight security for network coding," in *Proc. IEEE Int. Conf. on Communications*, 2008.
- [20] Y. Wang, "Insecure "Provably Secure Network Coding" and Homomorphic Authentication Schemes for Network Coding," IACR Eprint archive, 2010.
- [21] Z. Yu, Y. Wei, B. Ramkumar, and Y. Guan, "An efficient scheme for securing XOR network coding against pollution attacks," in *Proc. IEEE INFOCOM*, 2009.