Weak Data Secrecy via Obfuscation in Network Coding Based Content Distribution

Roberto G. Cascella*, Zhen Cao†, Mario Gerla†, Bruno Crispo* and Roberto Battiti*

*Dipartimento di Ingegneria e Scienza dell'Informazione, Università di Trento, Via Sommarive 14, I-38100 Trento
†Computer Science Department, University of California Los Angeles, 3732F Boelter Hall, CA 90095 Los Angeles
Email: {cascella, crispo, battiti}@disi.unitn.it, {caozhen@nrl., gerla@ }cs.ucla.edu

***Abstract*—Content distribution systems benefit from network coding in terms of error/loss protection and faster dissemination at the cost of exposing the data to intermediate nodes that are not the intended recipients. In these systems data secrecy is at risk and it can be guaranteed by using encryption as a straightforward method. However, this inevitably increases the communication costs and decreases the willingness to cooperate of intermediate nodes, which have no expectation of recovering the file.**

In this paper we propose an incentive secrecy preserving scheme based on obfuscation for time-sensitive data. The source intentionally obscures the original file and informs only the intended recipients which file blocks are corrupted. This scheme, combined with network coding, preserves the data secrecy and, in the meantime, fosters the cooperation of intermediate nodes. The evaluation of our solution shows that the proposed scheme is more efficient than encryption based secrecy preserving methods. In particular, the distribution of content in term of downloading speed is greatly enhanced.

I. Introduction

The widespread of mobile technology has changed the communication paradigm leading to a new service centric architecture where users are both data consumers and producers. To enhance this type of communication, P2P networks have emerged as solutions to use efficiently the end-point available bandwidth, thus, providing cost-effective content dissemination. In fact, several P2P applications have been deployed for mobile users to enable both communication and content gathering networks. Virtual communities or groups of interest can now spontaneously mushroom to exchange content or share videos without the need of a specific infrastructure. As an example, vehicles can disseminate road conditions or other information, such as safety or urban monitoring videos, which can be of interest to the community [1].

In this context, *network coding* [2] has emerged as a technique to potentially increase the performance of P2P networks, in terms of throughput and latency while delivering large files [3]. Network coding enables intermediate nodes to produce new codewords for the information by reducing the need for nodes' co-ordination in dynamic environments and, at the same time, by sustaining the scalability of the system [2]. The effectiveness of network coding, and in general the P2P paradigm applied to content sharing applications, is subject to nodes' cooperation which cannot be assumed in real systems without a centralised authority unless the nodes are not interested in the content or an incentive mechanism is in place.

If the information is sensitive, or the source wants to distribute it to a small subset of nodes, the data need to be encrypted. As a result, intermediate nodes will not be able to recover the original content and it will also cause their lack of cooperation, which will drastically reduce the benefit of network coding.

In this paper, we target time-sensitive data and, by exploiting the network coding construction, present a lightweight solution for guaranteeing weak secrecy, which represents an alternative to data encryption. As opposed to strong data secrecy, where the information can be only recovered with the proper cryptographic material, we refer to weak secrecy as the capability of the source to retain control over the information for a bounded time: the content can be revealed to other nodes after it is not considered sensitive.

The proposed approach consists of transmitting the information in clear text and exploiting a vulnerability of network coding, specifically the fact that data might be easily polluted so that the reconstruction of the file is not possible [4]. The source obfuscates the real information and uses a secure channel to notify the authorised destinations which blocks have been intentionally corrupted, so that encryption is limited to a small portion of the entire file.

The application of this solution has two important features. It reduces the processing overhead, which can be cumbersome for real-time applications, and it can be used to retain the cooperation of intermediate nodes to speed up the dissemination phase in a dynamic system, where the group membership changes frequently.

The rest of the paper is organised as follows. Sec. II discusses the related works. Sec. III presents an application scenario and the threat model for network coding based content distribution networks. Sec. IV details the proposed solution and Sec. V analyses its feasibility respectively. Sec. VI evaluates the approach and Sec. VII concludes the paper.

II. Related works

Network coding is a linear transformation over the n blocks f_i of a file F. These blocks of length b are represented by

Work partially supported by the project DAMASCO funded by the Italian Ministry of Research. R. Cascella, B. Crispo and R. Battiti also wish to acknowledge the project BIONETS (FP6-027748) funded by the FET program of the European Commission.

$m = \lceil b/q \rceil$ symbols, defined in the field F_{2^q}, i.e., $f_i = [f_{i,1}, \ldots, f_{i,m}]$. Before transmitting, each intermediate node creates a new packet, which is the linear combination of the blocks locally available, by selecting randomly n coefficients which constitute the coefficient vector c, with $c_i \in F_{2^q}$. Thus, a linear combination of blocks is $y = \sum_{i=1}^{n} c_i f_i = (\sum_{i=1}^{n} c_i f_{i,1}, \cdots, \sum_{i=1}^{n} c_i f_{i,m})$, and the potential number of unique blocks is only limited by the field from which the coefficients are picked. In its general form, network coding requires n linearly independent encoding vectors to completely decode a file. But, if the coefficient vector is not transmitted correctly or if a block has been corrupted, the reconstruction of the original file cannot be made at the destination.

Few solutions based on homomorphic hash functions have been proposed to enable the verification of the integrity of the content at the receivers [5]. For instance, [6] proposes a signature scheme for secure network coding to identify corrupted blocks and detect pollution attacks. [4] also presents an efficient signature-based scheme, that leverages the RSA signature and does not need of any secure channel, to detect and filter pollution attacks for applications that adopt linear network coding techniques.

To provide confidentiality, data encryption is the traditional solution to avoid the information being accessed by unauthorised parties. But, in the targeted mobile environment, the use of symmetric group keys to encode the information requires the group membership to be stable and the shared key to be often refreshed, as this key can be crypto-analysed over a large amount of data. In [7], the authors present a general protocol to detect nodes who transmit corrupted data. However, this solution heavily relays on a public key based broadcast encryption that uses public cryptography to distribute to nodes the symmetric keys used for ciphering the data. Moreover, the verification of the block is done by using an incremental hash function which cannot be directly applied in network coding as intermediate nodes generate new information before transmitting the packets.

The cost of cryptographic operations is paid in terms of computational and processing time, which should be reduced at minimum, because we cannot expect mobile nodes to be capable of decrypting and processing large amount of data. This overhead causes higher cost if the value of the content decreases exponentially with the time: the utility of the data is bounded to the time frame close to the instant the content is gathered.

In the literature, little attention has been devoted to investigate solutions that go behind encryption of the entire file to protect sensitive information from unwanted disclosure. If only *weak* protection of the information is required, network coding already offers the possibility to disseminate different blocks of data by using multiple paths so as the reconstruction of the information is only guaranteed if a node has access to all these flows [8]. However, in real applications multiple disjoint paths might not be always available, and this requires the completely knowledge of the links connecting intermediate nodes.

An efficient technique to provide confidentiality without encryption is *chaffing and winnowing* [9]. The idea consists in distributing more information than required in clear text; then the source produces Message Authenticating Codes (MACs) to notify the destination which packets must be removed to recover the original data. The security of the scheme is based on the difficulty of the attacker to identify the informative packets. A more secure approach of this scheme consists of processing the data before the transmission by applying a linear transformation, i.e., *all or nothing*, which causes that all informative blocks must be received to reveal the original message [10] [11]. This avoids that an adversary might guess with a certain degree the good packets by possible correlation of the information.

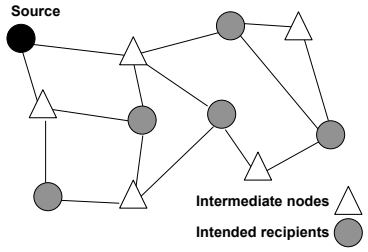

Fig. 1. Content distribution system: illustration.

Our approach differentiates from previous works as it defines a mechanism that obscures sensitive information within a bounded time frame, i.e., weak secrecy guarantee. We exploit pollution attacks on network coding and leverage the chaffing and winnowing technique proposed by Rivest to define a basic *all or nothing* scheme. Our approach makes use of encryption only to protect the partial information the source sends to the destinations to notify the corrupted blocks. When the content is polluted at the source, intermediate nodes are not able to identify useless blocks and *winnow* them in real-time.

III. System objective

In this section, we define the system objective by discussing an application scenario that can benefit from the use of network coding for content distribution and, at the same time, can require protection of the information. Then, we discuss the adversarial model.

We envision the presence of mobile devices that record data and timely share content within a community of interest. The type of information is real-time and the interest of the community decreases as time goes. Thus, the source must disseminate the content immediately after the event and it must ensure that only authorised users, or those subscribed to the service, can have access; others can obtain the content later.

As illustrated in Fig.1, the source node splits the original file into blocks and distributes them by using linear network coding, as described in Sec. II. In Fig.1, the grey round nodes are the intended recipients of the information while the triangles are the intermediate nodes. The purpose of our system is to deliver the information to the gray nodes while keeping the triangle nodes unaware of the true data.

In this research, we argue that encryption has several drawbacks in our settings. Indeed, it is computational expensive

and it is difficult to distribute and maintain the symmetric or group keys among the intended recipients. Moreover, the intermediate nodes, who are not expected to decrypt the data, would not like to cooperate unless a proper incentive mechanism is adopted, e.g., monetary exchange. Losing the benefit of codewords contributed by those intermediate nodes, the intended recipients will not be able to recover the file as quickly as when the file is not encrypted. Hence, if on one hand network coding is effective in reducing the time for distribution [1], on the other end encryption of the entire file can take over this benefit.

To this end, this paper proposes a mechanism which not only provides data secrecy to the intended recipients, but also retains the cooperation of intermediate nodes. We assume that intermediate nodes are willing to cooperate if they have an expectation of recovering the file after a full download. Otherwise they become "lazy" in encoding and contributing codewords to the others in the network.

The basic idea of achieving this system objective is the use of obfuscation: before the file distribution starts, the source node pollutes the original file and informs the authorised destinations which blocks have been corrupted. In this way, the source node ensures that the intended recipients of the file can reconstruct the original file after decoding. While, intermediate nodes are only able to get the corrupted file because they do not have the hints for discarding the polluted part timely.

A. Adversarial model

We consider a self-organized system populated by two types of attackers: malicious nodes and eavesdroppers. Malicious nodes inject false content or they produce fake encoded blocks to limit the reconstruction of the original file. Eavesdroppers are unauthorized nodes, interested in the data, who are assumed to contribute to the data streaming in a fair manner, i.e., they share the blocks they host without injecting corrupted information. Several adversarial attacks can be studied for the resulting overlay network and the distribution process. We only focus on pollution attacks and unauthorised data access.

Corrupted information cannot be identified by using security mechanisms based on simple hash functions as network coding enables re-coding at the intermediate nodes. Indeed, the block, that each node sends, is never the same: the information is always updated when receiving new blocks and the node randomly chooses the coefficients over the finite field. In this setting, a corrupted block can be recombined with good blocks, thus, the percentage of useless content significantly increases.

In these settings, we must ensure that the receivers should be able to eliminate possible fake blocks to recover the original file and the source should still retain the control to which nodes the data will be revealed by means of intentional pollution. These operations should also be accomplished in real-time and intermediate nodes should have incentives to cooperate.

IV. Our Scheme: Weak secrecy via obfuscation

In this section we describe our mechanism to provide weak secrecy for real time sensitive data in a dynamic environment. We exploit the attack model of malicious nodes in the sense that the source pollutes voluntarily the content and distributes only to authorised nodes the information required to recover the original data by using a secure channel.

A. Secure Random Checksum

We base our weak secrecy method on the use of the Secure Random Checksum (SRC) for the identification of corrupted blocks. In [5], the SRC is used as a lightweight alternative to prevent pollution attacks to network coding based content distribution systems.

We now give a high level explanation of how the SRC works. The server produces m random elements (numbers in the encoding field) as many as the symbol elements in each block. Then, it performs a pairwise multiplication of the vector of random elements with the vector of block elements and adds the results. For example, let assume the symbol elements of the block to be $f_i = [f_{i,1}, \ldots, f_{i,m}]$ and the random numbers for a particular node to be $t = [t_1, \ldots, t_m]$ (different nodes have different sets of random numbers), then, the SRC of block f_i is $SRC_i = \sum_{j=1}^{m} t_j f_{i,j}$. The same process is repeated for all the blocks $f_1, f_2, \ldots, f_n$. Thus, the mask based checksum for the original file F is $(SRC_1, \ldots, SRC_n)$ which is distributed with $\vec{t}$ over a secure channel.

The verification of SRCs can be performed each time a node receives an encoded block $y = \sum_{i=1}^{n} c_i f_i$, as defined in eq. 1.

$$\begin{aligned} SRC(y) &= \sum_{k=1}^{m} t_k y_k = \sum_{k=1}^{m} t_k (\sum_{i=1}^{n} c_i f_{i,k}) \\ &= \sum_{i=1}^{n} c_i (\sum_{k=1}^{m} t_k f_{i,k}) = \sum_{i=1}^{n} c_i SRC_i \end{aligned} \tag{1}$$

A node can check if any block has been modified by applying the random mask to the encoded block and comparing this mask with a combination of the mask-checksums weighted by the coefficient vector. This results in fast identification of corrupted data as the SRCs are linear operations which can be computed very efficiently.

As discussed in Sec. II, homomorphic hash functions can be used for the verification of corrupted blocks, but even if more reliable than SRCs, they are also computationally expensive. Indeed in our implementation, the SRC generation rate on a Intel Pentium 1.73GHz machine is as fast as 96MB/s, which is faster than the rate of homomorphic hashes 20KB/s.

B. Weak secrecy via Obfuscation

In order to provide weak secrecy protection for the content, the source must inform the intended recipients which portion of the file is intentionally corrupted and should be discarded after the recovery phase. We exploit the idea of using SRCs not only for filtering corrupted content but also for the identification of intentionally polluted blocks.

The source produces valid SRCs for each node in the system and those SRCs that correspond to the corrupted part will be modified to alert only the intended recipients. When the file is reconstructed, the destination discards those blocks that

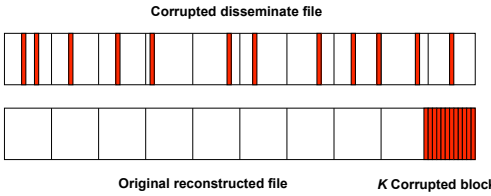

Fig. 2. File reconstruction: a legitimate destination discards the k corrupted blocks for the reconstruction of the file.

correspond to wrong SRCs. The details are described below and shown in Fig. 2.

Let suppose that the original file F is divided into n blocks $f = [f_1, f_2, \ldots, f_n]$ composed of m symbols $f_i = [f_{i,1}, f_{i,2}, \ldots, f_{i,m}]$ and the coefficients $t_1, t_2, \ldots, t_m$ chosen in the field F_{2^q}. For each file, we introduce a number of $k = \alpha \cdot n$, with $\alpha \in]0, 1[$, corrupted blocks into the original file in order to mask the sensitive information in it. Then, the message distributed using the network coding scheme is $f' = [f'_1, f'_2, \ldots, f'_{n+k}]$. The server generates the SRC for each block and intentionally replaces the SRCs using a random wrong value for the k corrupted blocks. The altered file is then distributed using a network coding scheme.

After the legitimate users recover the whole file, they can check whether the SRC for each block is correct and drop those blocks that do not match with their SRC values. The legitimate users can thus decode the original file as expected because the SRCs for the corrupted part are intentionally wrong. Whereas, the intermediate untrustworthy nodes, that receive valid SRCs also for the corrupted blocks, are "fooled" with the incorrect information.

V. Analysis of the solution

The security of the proposed mechanism consists of the impossibility of an attacker to guess which blocks contain good data. Thus, we must ensure that the data, transmitted in clear text, do not contain sufficient information to recover the original file or one of its portion. To guarantee this property, we use an *all or nothing* transform which states that it is sufficient for the source to operate on the data in such a way that the result of an initial linear transformation can be used to recover the file input only if all blocks are available [10].

This means that the output blocks except one do not give any information to reconstruct any part of the input unless all blocks are used: the entropy of one block of the input does not change if more output blocks are available, while the conditional entropy of the input given the output is 0.

Stinson formalizes an *all or nothing* transform that is unconditionally secure [11]. This result can be applied in network coding, as it is shown that if there exists an invertible matrix M with non 0 entries in the finite field F_{2^q}, then $\phi(x) = xM^{-1}$ is a linear *all or nothing* transformation, where x is the data.

Thus, if we encrypt only one output block and transfer this block over a secure channel, only the intended destination can recover the entire data. Similar findings have also been applied to rateless codes where few blocks are encrypted to provide confidentiality; in this case only the source generate packets [12]. This solution applied to network coding is effective to provide confidentiality at low cost but it does not give incentives to intermediate nodes for cooperation, unless a monetary incentive scheme is used.

As discussed in Sec. IV, our solution consists of sending over a secure channel the SRCs, which identify corrupted data, so that the destinations can obtain all valid blocks that can be used to recover the original file. To make intermediate nodes cooperate, the source divides the file in pieces, and processes and distributes each piece separately by means of network coding. The non-sensitive information will be made accessible to intermediate nodes, while the rest, i.e., few pieces will be polluted to obscure the content. With this setting, intermediate nodes will have the necessary information to retrieve the whole file, but they do not know the valid combination of blocks.

This scheme only provides *weak* secrecy as with an exhaustive search intermediate nodes might succeed to reconstruct the file. The probability of recovering the file is a function of the number of the original blocks (n) and the fraction of corrupted ones (k) and it is inversely equal to $\binom{n+k}{k}$. Thus, the intermediate nodes might succeed only when the information is not anymore valued as sensitive

In our scheme we define the SRCs as a way to identify data that must discarded. This changes their initial objective, i.e., enabling the detection of possible packets' manipulation. But, the intermediate nodes can still continue to use SRCs as defined in [5], while, the destination nodes can distinguish packets corrupted by the source or by the attackers by applying homomorphic hash functions and signature schemes to detect pollution attacks, as presented in Sec. II.

VI. Evaluation

In this section, we compare our approach that uses the SRCs to provide weak secrecy with the common method that uses encryption to provide strong secrecy. In this first set of experiments, we do not consider the cooperation level of intermediate nodes, i.e., simply assuming that their participation in the content distribution system is independent of whether or not the file block is encrypted. In this scenario, we examine the performance enhancement of our weak secrecy mechanism compared with the schemes that use encryption.

In our setting, each symbol $f_{i,j}$ in the file block f_i is defined in the Galois field F_{2^8}, as well as the random coefficients for the SRCs. Let consider the SRC generation $SRC_i = t_1 f_{i,1} + t_2 f_{i,2} + \ldots + t_m f_{i,m}$. For each symbol $f_{i,j}$ it requires a pair of multiplication (i.e., $t_j f_{i,j}$) and addition ($+ = t_j f_{i,j}$). Let δ be the time of executing the pair of operations, then the per-block encoding time is proportional to the block size m, i.e., $m \cdot \delta$. The rate of Secure Random Checksum is $R_{SRC} = \frac{1}{T} = \frac{1}{\delta \cdot m}$.

We measure δ on a machine of Intel Pentium 1.73 GHz and get $\delta = 10.42 \times 10^{-9} s$. This measure is averaged over $1,000$ runnings and the resulting rate for the SRCs' generation is 96 MB/s. In our measurement on the same machine, we use the OpenSSL version 0.9.8a to test the speed of the most commonly used symmetric encryption algorithm Advanced

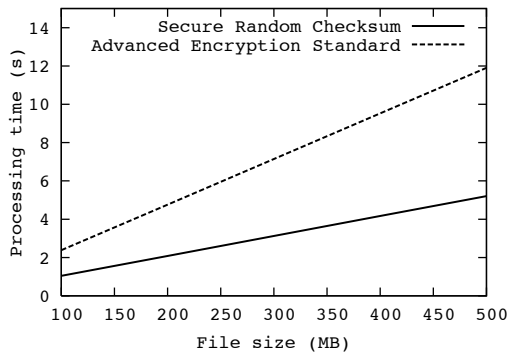

Fig. 3. File processing time versus file size with SRC and AES

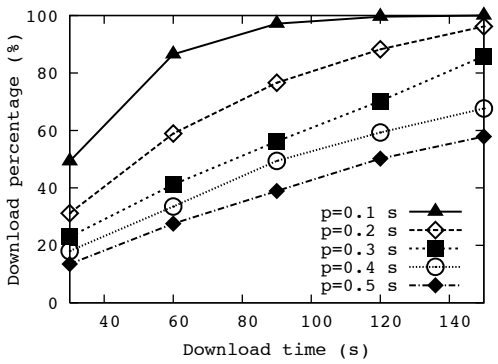

Fig. 4. File download rate versus time elapsed for different cooperation rates of the intermediate nodes.

Encryption Standard (AES). The rate for the AES with 256-bit key is 42 MB/s. Fig.3 plots the time of the encryption and of the generation of SRCs with respect to the file size; it shows that the SRC generation is more efficient than encryption.

Then, we consider the cooperation level for each intermediate node. This determines its willingness to participate into the content distribution and represents the packet sending rate of the nodes. In our weak secrecy method the intermediate nodes averagely have higher cooperation levels (i.e., more willing to contribute), since the original file is not encrypted and they have an expectation to recover the original data.

We simulate 50 nodes uniformly distributed in an area of $2,400\text{m} \times 2,400\text{m}$ which communicate via $802.11b$ with a range of 300m. These nodes cooperate in a network coding scheme to download from a server a file of 256 blocks, each of size $1KB$. In our simulation, we consider 90% of legitimate nodes who contribute a packet at the cooperation rate (every $p = 0.1\text{s}$) and 10% of intermediate recipients which contribute at a smaller rate.

Fig.4 depicts the download rate of legitimate nodes when simulation time elapses. We simulate different cooperation levels p and we expect that if our weak secrecy scheme is used intermediate nodes will forward packets at a high cooperation rate, such as every 0.1s, as they have the expectation to recover the file. On the contrary if encryption is used to provide strong data secrecy, we assume that some "lazy" nodes, who do not have the decryption key, will have a smaller cooperation level ($p > 0.1\text{s}$), which results in worst download rate.

VII. Conclusion

This paper presents a weak-secrecy preserving method for network coded time-sensitive data distribution. Our approach relies on obfuscation, i.e., a source "obscures" the original file and notifies the legitimate recipients the corrupted blocks. The destination nodes then use the Secure Random Checksums (SRCs) to identify polluted blocks, while other users are "fooled" with corrupted data. We analyse the security of our approach and show that data secrecy of the original file can be more efficiently achieved by using an *all or nothing* transform.

Our solution combines secrecy of the content with cooperation of intermediary nodes. We stimulate cooperation using time-bounded secrecy which gives to intermediate nodes an expectation to read after some delay the content, they previously forwarded, thus, incentivating them to participate to the scheme. Indeed, we use the term weak because unauthorised users can recover the data only with an exhaustive search to identify corrupted blocks or if the source sends them the SRCs to eliminate corrupted blocks. For instance, this can be done when the information is not considered sensitive anymore or in accordance to a business model, currently under investigation, which can use our solution to foster cooperation and distribute content to consumers with different service subscriptions.

The system also performs more efficiently than secrecy preserving methods that use encryption because only the SRCs are encrypted (as opposed to the entire file) and transmitted over a secure channel. As part of future work, we are investigating solutions to establish this secure channel and to manage the keys for the encryption and distribution of SRCs.

Finally we evaluate our solution and show that the efficiency of content distribution in terms of downloading speed is enhanced when the intermediate nodes are willing to cooperate.

References

[1] U. Lee, J.-S. Park, J. Yeh, G. Pau, and M. Gerla, "Code torrent: content distribution using network coding in vanet," in *MobiShare*, Sep. 2006.
[2] A. Al Hamra, C. Barakat, and T. Turletti, "Network coding for wireless mesh networks: A case study," in *WOWMOM*, Jun. 2006.
[3] C. Gkantsidis and P. Rodriguez, "Network Coding for Large Scale Content Distribution," in *IEEE INFOCOM*, Mar. 2005.
[4] Z. Yu, Y. Wei, B. Ramkumar, and Y. Guan, "An efficient signature-based scheme for securing network coding against pollution attacks," in *IEEE INFOCOM*, Apr. 2008.
[5] C. Gkantsidis and P. Rodriguez, "Cooperative security for network coding file distribution," in *IEEE INFOCOM*, Apr. 2006.
[6] F. Zhao, T. Kalker, M. Medard, and K. J. Han, "Signatures for content distribution with network coding," in *ISIT*, Jun. 2007.
[7] W. Ahmad and A. Khokhar, "SAMcast - a scalable, secure and authenticated multicast protocol for large scale P2P networks," in *IEEE ICC*, Jun. 2007.
[8] L. Lima, M. Médard, and J. Barros, "Random linear network coding: A free cipher?" *CoRR*, vol. abs/0705.1789, May 2007.
[9] R. L. Rivest, "Chaffing and winnowing: Confidentiality without encryption," *CryptoBytes (RSA Laboratories)*, vol. 4, no. 1, pp. 12–17, 1998.
[10] R. L. Rivest, "All-or-nothing encryption and the package transform," in *FSE*, ser. LNCS, vol. 1267, Jan. 1997, pp. 210–218.
[11] D. R. Stinson, "Something about all or nothing (transforms)," *Design, Codes and Cryptography*, vol. 22, no. 2, pp. 133–138, 2001.
[12] J. Byers, J. Considine, G. Itkis, M. C. Cheng, and A. Yeung, "Securing bulk content almost for free," *Computer Communications*, vol. 29, pp. 280–290, 2006.