An Integrated System Solution for Secure P2P Content Distribution Based on Network Coding

Heng He[1,2], Ruixuan Li[1+], Guoqiang Gao[1], Zhiyong Xu[3], Weijun Xiao[4]
[1]School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, P.R.China
[2]School of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan, P.R.China
[3]Department of Mathematics and Computer Science, Suffolk University, Boston, USA
[4]Department of Electrical and Computer Engineering, University of Minnesota, Twin Cities, USA
E-mail: willam1981@gmail.com, rxli@hust.edu.cn, ggq@smail.hust.edu.cn, zxu@mcs.suffolk.edu, wxiao@umn.edu

Abstract—**Network coding has been demonstrated to be able to improve the performance of P2P content distribution. However, it is vulnerable to pollution attacks, leading to substantial performance degradation. Moreover, existing corruption detection schemes for network coding are not applied well to P2P systems. More efficient scheme based on attacker identification is required to thwart such attacks. In this paper, we propose an integrated system solution for secure P2P content distribution based on network coding, referred to as ISNC. In ISNC, we first design our system architecture based on extended uniform bipartite networks that can achieve high throughput with network coding. Based on the architecture, we present a secure network coding signature scheme and an identity-based malicious peer identification scheme. The two schemes can cooperate to thwart pollution attacks effectively in P2P network, not only detecting corrupted blocks, but also identifying all the malicious peers. Simulation results show that ISNC can effectively limit the pollution spread and identify malicious peers quickly, even when they collude to launch attacks. Compared with existing related schemes, ISNC is especially applicable for P2P content distribution, and can achieve both high security and overall efficiency.**

Keywords-P2P content distribution; network coding; pollution attacks; homomorphic hash; attacker identification

I. INTRODUCTION

Network coding has been first proposed in the field of information theory to improve throughput in a multicast session [1], and has since received extensive research attention. The essence of network coding is that information to be transmitted in a communication session can be encoded rather than simply forwarded by the participating peers. It is a well known result that network coding may achieve better throughput in certain network topologies [2]. Recent studies [3, 4] have demonstrated that network coding is also beneficial for content distribution in P2P networks, since it can improve the resilience to peer churn, leading to shorter downloading time. Network coding distributes encoded blocks rather than original blocks, and all encoded blocks are equivalent to any peer, thus the need for content reconciliation and location of rare original blocks is eliminated. However, these main advantages of network coding are only applicable in P2P networks consisting of trustworthy peers. Because any malicious peer may generate corrupted encoded blocks, and other peers may unintentionally use them to create new blocks that are also corrupted. As a result, a single corrupted block can rapidly pollute the network and cause the significant performance degradation of the system. Therefore, it's necessary to check the encoded blocks on-the-fly to verify their validity before using them for encoding. Unfortunately, the traditional digital signatures and hashes can not protect encoded blocks from being corrupted, since each peer generates unique encoded blocks that cannot be signed by the server individually.

The attacks that corrupt valid encoded blocks in a network coding system are called pollution attacks [4, 5], and the corruption detection schemes have been well studied. Homomorphic hash functions have first been introduced by Krohn et al. [6] to allow intermediate peers to check blocks on-the-fly that are encoded at the source using rateless codes. However, homomorphic hash functions are computationally expensive. Gkantsidis et al. [4] introduced the homomorphic hash technology for network coding into P2P networks and proposed a cooperative security scheme. The scheme reduces the computational time of the homomorphic hash by only requiring peers to check blocks probabilistically and makes peers cooperate to protect themselves against malicious peers by alerting affected peers when corrupted blocks are found. However, the scheme cannot identify and remove malicious peers, which means attackers can continuously generate corrupted blocks that will cause more overhead. Several other schemes [5, 7-10] have also been proposed to solve the problem of pollution attacks, but they are generally rather expensive or not applicable to P2P systems. Compared to the researches on corruption detection schemes, identifying attackers has received much less attention. However, attacker identification is a more desirable and efficient approach as it prevents network resources from being continually wasted on handling corrupted blocks from malicious peers.

[+] Corresponding author. E-mail: rxli@hust.edu.cn

In this paper, we propose a novel and Integrated system solution for Secure P2P content distribution based on Network Coding, referred to as ISNC. In ISNC, we first present our system architecture which is based on extended uniform bipartite networks that can achieve high throughput with network coding. The architecture can provide high reliability and resilience to peer churn. Based on the architecture, we then present a secure network coding signature scheme and an identity-based malicious peer identification scheme. The two schemes can cooperate to thwart pollution attacks effectively in P2P network, not only detecting corrupted blocks, but also identifying malicious peers quickly. ISNC can effectively limit the pollution spread and deal with collusion attacks of multiple malicious peers, identifying all of them.

The rest of the paper is organized as follows. Section II describes our system architecture. Section III presents the secure network coding signature scheme. Section IV presents the identity-based malicious peer identification scheme. Section V gives the evaluation results of the performance and security of ISNC. Section VI describes the related works. Section VII concludes the paper.

II. THE SYSTEM ARCHITECTURE

P2P network is a perfect place to apply network coding, for it is easy to tailor the topology of the overlay to facilitate network coding. In this section, we present our system architecture for P2P content distribution based on network coding. We will describe the system architecture from two main aspects, including system topology and content distribution with network coding over the topology.

A. System Topology

We utilize and extend the uniform bipartite network as the topology of the system architecture, making it applicable to P2P content distribution, also achieving high throughput with network coding. The system topology is also the infrastructure of the following secure schemes. Figure 1 shows an example of the extended uniform bipartite network.

A uniform bipartite network C_n^k is a regular graph which adheres to the following rules:

1) The first layer contains one source peer which generates messages;

2) The second layer contains n relay peers which receive messages from the source peer and relay them to the receiver peers;

3) The third layer contains C_n^k receiver peers which receive messages from the relay peers.

There are n links connecting the source peer to the n relay peers respectively. For every k peers out of the n relay peers, there are k links connecting them to a receiver peer. Since there are a total of C_n^k different combinations, the number of receiver peers is C_n^k.

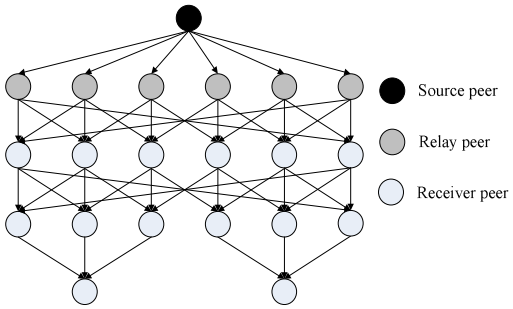

Figure 1. The extended uniform bipartite network

In Figure 1, the above three layers constitute a uniform bipartite network with n=6 and k=3, for the network coding gain achieved by the system reaches its maximum when $k=n/2$ [11]. Each peer in the fourth layer and below connects with k peers in the upper layers except the second relay layer.

B. Content Distribution with Network Coding

Before the source peer sends out the file, it first divides the file into multiple groups, each of which is further divided into k blocks. Then each block is subdivided into r codewords, and each codeword is an element of modular subgroups of Z_q.

We utilize random linear network coding scheme [3] to encode every group of the file instead of the whole file, which can reduce coding complexity effectively. Assume group g has original blocks $(b_1,\ldots, b_k)$. To code a new coded block B of group g, the source peer first randomly chooses a coefficient vector C, which contains k elements $(c_1,\ldots, c_k)$ of Z_q. It then produces B as $B=\sum_{i=1}^{k} c_i \cdot b_i$. After that, C is associated with B and both of them are sent out. Coding operation is not limited to the source. In Figure 1, if a peer which is not a relay peer has m ($m \le k$) coded blocks of group g, when serving another peer, it also independently and randomly chooses a coefficient vector from Z_q, and then produces a coded block B' as a linear combination of the m coded blocks. The coding coefficients used to encode original blocks to B' are associated with B'. A peer can reconstruct the original blocks of group g as soon as receiving k coded blocks for which the associated coefficient vectors are linearly independent. The decoding process is similar to solving a system of linear equations.

Note that the peer in the third layer can get k independent coded blocks very soon because the k relay peers in the second layer connected with the given peer just relay to it the independent coded blocks received from the source. Therefore, our system can accumulate a set of peers with the complete file very soon, which could promote the overall performance effectively.

III. A SECURE NETWORK CODING SIGNATURE SCHEME

Homomorphic hash functions are currently one of the best solutions to prevent pollution attacks for their effectiveness and high level security [4, 5]. However, existing schemes [4, 6] are not designed for group network

coding, and there is no mechanism to ensure that the true hashes of the file can be distributed to every peer effectively. In this section, we present a secure network coding signature scheme for group network coding authentication. And it can also authenticate all the hashes of every group.

In the following discussion, we shall use scalars and vectors defined over modular subgroups of Z. Firstly, the server is assumed to be trusted and it globally generates a set of security parameters (p, q, g), where p and q are two large random primes and $q \mid (p\text{-}1)$ (e.g. $|p|$=1024 bit and $|q|$=257 bit) . The parameter g is a $1\times r$ row-vector, denoted as (g_1, …, g_r). The elements of g are random elements of Z_p, all order q. More details about the parameters of the homomorphic hash can be seen in [6]. We assume that the source peer has a private key SK and a public key PK. Then the server publishes the security parameters (p, q, g, PK).

Given a file identifier id_f, a group identifier id_g and k original blocks b_i (i=1, …, k) of this group, the source peer computes the signatures for this group using Algorithm 1.

Algorithm 1

Step1: Compute the homomorphic hash for each block b_i = (b_{i1}, …, b_{ir}) as $\sigma_i = \prod_{j=1}^{r} g_j^{b_{ij}} \mod p$, for i = 1, …, k.

Step2: Compute the signature for the hashes as θ = Sign(SK, (id_f, id_g, σ_1, …, σ_k)), where Sign is a standard signature algorithm.

Step3: Generate the signature of the group σ = (σ_1, …, σ_k, θ).

When a peer first joins the system, it downloads the signatures of the file from its upstream peers. Given id_f, id_g, σ, and a coded block $B=\sum_{i=1}^{k} c_i \cdot b_i$ of this group, the receiving peer can check the validity of B by Algorithm 2.

Algorithm 2

Step1: Check the validity of the signature σ by standard signature verification algorithm using the given information (PK, (id_f, id_g, σ_1, …, σ_k), θ). If the checking result of σ is invalid, then algorithm2 aborts. The receiving peer must contact its upstream peers to regain σ.

Step2: Compute the homomorphic hash of B = (B_1, …, B_r) as $\sigma = \prod_{j=1}^{r} g_j^{B_j} \mod p$.

Step3: Compute the hash of B as $\sigma' = \prod_{i=1}^{k} \sigma_i^{c_i} \mod p$.

Step4: Judge whether σ is equal to σ'. If $\sigma = \sigma'$, the receiving peer accepts B; otherwise, it discards corrupted B.

Because the homomorphic hash function has the property that the hash of a linear combination of some input blocks can be constructed by a combination of the hashes of the input blocks, the correctness of the signature scheme is straightforward.

Because the computation of homomorphic hashes in our scheme may be expensive for some peers, to reduce the computation overhead, we make each peer check blocks probabilistically. Blocks of one group that have not been checked are kept in an insecure window. Different groups correspond to different insecure windows. Blocks are checked using batching method [6] and the batch window is equal to the insecure window when starting checking. Different insecure windows are checked independently. Whenever a peer verifies its insecure window and finds no corrupted blocks, the insecure window is reset. Because probabilistic check may let corrupted blocks propagate, to prevent corrupted blocks from propagating quickly and, more importantly, identify malicious peers, whenever a peer detects corrupted blocks using batching, it triggers an identity-based malicious peer identification scheme which will be described in section IV.

IV. AN IDENTITY-BASED MALICIOUS PEER IDENTIFICATION

In this section, we present an efficient identity-based malicious peer identification scheme based on the system topology and has two main approaches.

A. A Bottom-up Approach for Identification Scope Restriction

Firstly, every peer appends its generated encoded block with a time stamp that records the time when the block generated. And every peer maintains a table containing 1) the identities of its upstream neighbors that sent blocks insider its insecure windows, with the time stamps of these blocks; and 2) the identities of its downstream neighbors that received blocks encoded with insecure window blocks, also with the time stamps of the encoded blocks.

Figure 2 shows the process of identification scope restriction. When peer A detects corrupted blocks in its insecure window using batching, it sends the server a trace message with the identifier of the polluted group id_g, the identities of its upstream neighbors that sent it blocks of id_g inside its batch window, and the time stamps of these blocks. To prevent the corrupted blocks from propagating, A also sends the server an alert message with id_g, the identities of its downstream neighbors that received blocks encoded with its insecure window blocks of id_g, and the time stamps of these encoded blocks. If the same upstream neighbors as those with the trace message also sent A blocks of other groups (e.g. id_g') inside its insecure windows, and some downstream neighbors of A received blocks encoded with these blocks, then A would also report the identities of these downstream neighbors with the alert message, together with the corresponding group identifiers (e.g. id_g') and the time stamps of these encoded blocks (step1).

After receiving the messages from A, the server records them and sends trace messages to those upstream neighbors of A with received identities (step2). Meanwhile, the server sends alert messages to those downstream neighbors of A with received identities (step3). As in Figure 2, suppose peer P is one of those upstream neighbors of A. The trace message sent to P is associated with id_g, ID_A and the relevant time stamps of the blocks which were reported by A before. When P receives the trace message, it checks whether it sent A the blocks encoded with its insecure window blocks of id_g at the given time stamps. If P did, it continues to send a trace message and an alert message like A. If P didn't, it sends a halt message to the server (step4). Suppose C respectively received blocks encoded with

insecure window blocks of id_g and id_g' from A, then the alert message sent to C will be associated with id_g, id_g', ID_A and the time stamps of the encoded blocks reported by A. When C receives the alert message, it checks whether it has any insecure window blocks of id_g or id_g' with the given time stamps. If C has and it also has some downstream neighbors that received blocks encoded with these insecure window blocks, it will continue to send an alert message like A. Otherwise, the alert message stops at C (step5).

The peers in the same layer send trace messages concurrently and the server sends trace messages to peers layer by layer. Trace messages are propagated upward in this way until all the relevant peers send halt messages. At last, we restrict the identification scope as narrow as possible which is the area that the trace messages cover. Meanwhile, the alert messages are propagated downward layer by layer. To reduce the number of messages sent by the server, when there are multiple peers sending trace messages with the same upstream neighbor, the server just combines the trace messages before sending them to the upstream neighbor (step6). Similarly, the server combines the alert messages before sending them to the same downstream neighbor.

Peers in the identification scope start checking blocks concurrently right after they send trace messages, using a mechanism based on binary batching trees which works as follows: The peer first verifies blocks of id_g inside its insecure window in batches. If the batch verification does not find any corrupted blocks, all the batch window blocks are secure. Otherwise, the batch window is divided into two halves. Any corrupted parts are again subdivided into two halves until the individual corrupted blocks are identified. Meanwhile, the blocks of other groups from the same upstream neighbors as those batch window blocks of id_g are checked using the same method too. Any peer receiving the alert message checks the insecure window blocks specified in the alert message, also using the same method. Here all the blocks that need to be checked will not be used to encode new blocks until they are verified as secure.

Because the trace and alert messages are only of small size compared to the size of encoded blocks and require little processing at the server and peers, messages can propagate much faster than corrupted blocks and efficiently halt the propagation of corrupted blocks.

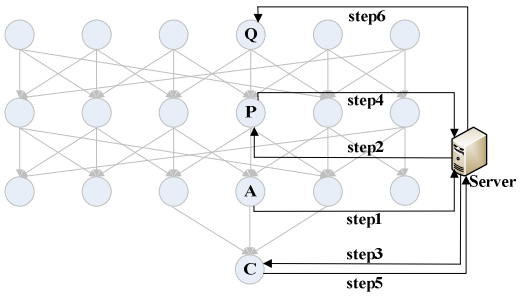

Figure 2. The identification scope restriction

B. A Top–down Approach for Malicious Peer Identification

After the identification scope restriction, the server starts identifying malicious peers in the identification scope from the lowest layer to the highest layer. Assume that layer n is the lowest layer. To check peer M of layer n, the server sends an identification message to all the downstream neighbors of M which sent trace messages with ID_M to the server before. Suppose peer N is one of these downstream neighbors. The identification message sent to N is associated with id_g, ID_M and the time stamps that were sent with the trace message and ID_M by N before.

After checking blocks using the proposed method, these downstream neighbors of M return the results to the server. For N, if any insecure window block of id_g from M with the given time stamp is checked as corrupted, then M is reported as malicious and will be removed from the system. If no corrupted blocks are found by any neighbors, then M is trusted. When M is identified as malicious, all the downstream neighbors of M will receive an alert message with ID_M from the server. Any peer that has insecure window blocks from M checks these blocks, and continues to send an alert message when necessary like C.

All the peers of layer n are checked concurrently and then the server goes on checking layer n+1. Any peer of layer n+1 need not be checked if any upstream neighbor was checked as malicious, because it could be innocent and received corrupted blocks from its upstream neighbors. If all the upstream neighbors in the identification scope need not be checked, the peer will receive a halt message from the server. In this way, peers in the identification scope are checked layer by layer until all the peers of some layer need not be checked or the identification finishes at the peer that first sent the trace message. Note that the feasibility of the identification relies on that peers must report the true checking results to the server. And the malicious peer cannot deny its behavior. To achieve these requirements, we make use of a non-repudiation transmission protocol proposed by Wang et al. [12].

For a peer I and a downstream peer J, the server generates a pool of secret keys $K_I = \{\gamma_i \mid \gamma_i = \mathrm{F}(skey_I, ID_J, i), i=1, \ldots, \lambda\}$, where $skey_I$ is the secret key of I registered at the server, and F is a secure hash function. After that, the server randomly selects a δ-element subset of keys K_I' from K_I based on $skey_J$ and ID_I; then, it sends K_I' to J.

When sending an encoded block B to J, I needs to compute an evidence $\Phi(B)$. Suppose B is a block of id_g and the time stamp is ts. $\Phi(B)$ should make sure that B is really a block of id_g with ts, and sent by I . $\Phi(B)$ consists of λ tags $\{\mu_1, \ldots, \mu_\lambda\}$. The tags are computed as $\mu_i = \mathrm{trunc}_\alpha(\mathrm{F}(B|id_g|ts, \gamma_i))$, where $i=1, \ldots, \lambda$, and trunc_α is a function that truncates the input into leftmost α bits. When J receives B from I, it computes $\mu_i' = \mathrm{trunc}_\alpha(\mathrm{F}(B|id_g|ts, \gamma_i'))$ where $i=1, \ldots, \delta$, and each γ_i' is a distinct element of K_I'. Then J checks whether the value of μ_i' can be found in $\Phi(B)$. If all μ_i' are in $\Phi(B)$, then J receives B that has a valid evidence. If J reports I malicious, it must provide the corrupted block B, id_g, ts, and $\Phi(B)$ to the server. The server computes μ_i'' =

$\text{trunc}_{\alpha}(F(B|id_g|ts, \gamma_i''))$, where i=1, …, λ-δ, and each γ_i'' is a distinct element of K_I/K_I'. If more than ω values of μ_i'' can be found in $\Phi(B)$, where ω is the threshold, the server accepts the report. Then the server further checks whether B is corrupted, if B is, the server confirms I is malicious.

C. Discussions

In P2P networks, collusion attacks possible exist. We consider multiple malicious peers collude to hide themselves. For example, consider a transmission path $X \rightarrow Y \rightarrow Z$, where X and Y are malicious, Z is infected by X. When Z triggers a malicious peer identification process, Y may not send trace message with ID_x. However, X will still be identified if it sends corrupted blocks to any innocent downstream neighbors. If X only sends corrupted blocks to Y, then X will not be discovered because of the shield of Y. However, Y will be identified as malicious instead. And X will also be discovered as long as it sends corrupted blocks to any innocent peers after Y removed from the system.

Malicious peers may prevent alert messages from propagating. For the transmission path $X \rightarrow Y \rightarrow Z \rightarrow S$, where X and Z are malicious, Y and S are infected by X. After Y sends an alert message to Z, Z will not send alert messages further. As a result, S has to detect corrupted blocks for probabilistic checking itself. In the following identification process, Z will be identified as malicious by S. Of course, Z may disparage Y with the corrupted block which was sent by Y but originated from X. However, Y have already sent alert message for this corrupted block to the server before. If Z sends trace message for this corrupted block, it will also be identified as malicious by the server.

In P2P networks, a malicious peer that may get more than one identity can enhance the strength of pollution attacks, which is called Sybil attack. The discussion of how to resist it is out of the scope of this paper. Of course, some prominent research results [13] can be integrated with our system to limit Sybil attack effectively.

V. PERFORMANCE EVALUATION

A. Throughput Evaluation

We start by evaluating the throughput through simulations, by comparing the content distribution scheme of ISNC with BitTorrent [3] in a well-connected mesh network. Throughput is defined as the service the system provides in one time units. Here we let two systems transmit the same file, so the throughput can be represented by the consumed time when the peers finish receiving the file, denoted by finish time. The shorter the finish time, the higher the throughput. The simulator is round-based, where in each round a peer can upload and download blocks. The simulation parameters are as follows: the network consists of 200 peers and the file size is set to 100 blocks. The group size of the file is 6 blocks. The upload capability of the source peer and relay peers in our system are 6 blocks per round. All the other peers can upload 3 blocks per round. The topology of our system is based on C_{12}^{6} extended uniform bipartite network. We also performed experiments with other parameters and observed similar results.

Figure 3 shows the finish times of our scheme and BitTorrent. We measure the finish time as the number of rounds required to complete the download. As Figure 3 shows, although the finish time of our scheme has a little larger variance because of the system topology, the average finish time of our scheme is about 15% shorter than that of BitTorrent. Thus our system can bring better throughput.

B. Corruption Evaluation

Figure 4 shows how the corruption varies during content distribution in ISNC through simulations. The relevant parameters are the same as those in Part A. The malicious peers are randomly chosen from all the peers with proportions of 15%, 25%, 35% respectively in three simulation circumstances. We also assume the probability of checking is 5% for each innocent peer.

As Figure 4 shows, as the proportion of malicious peers increases, the mean percentage of corrupted peers just increases a little. In the three circumstances, all the malicious peers can be identified by our scheme. We also implement the scheme proposed by Gkantsidis et al. [4]. The simulation parameters are the same and the results show that in their scheme, the mean percentages of corrupted peers are 20%, 24% and 26% where the proportions of malicious peers are 15%, 25% and 35%. Compared with their scheme, the percentage of corrupted peers is much smaller in our system. In addition, because their scheme can not identify malicious peers, the corruption is continual during the whole content distribution.

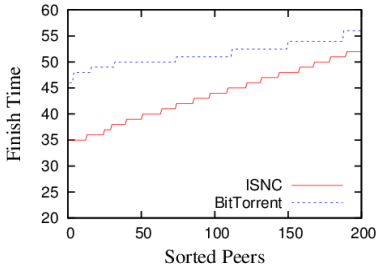

Figure 3. Finish times

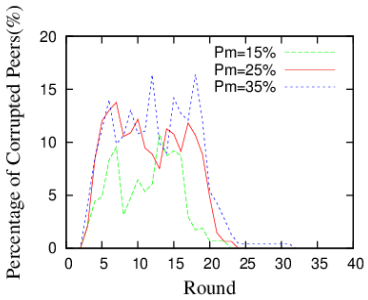

Figure 4. Corruption variation during content distribution in ISNC

C. System Overhead

In ISNC, the signatures of the file consist of elements in Z_p and the element is typically 128B, which is 1/2048 times the size of a block of size 256KB. Thus, the signatures are about only 0.05% of the total file size. Furthermore, every peer only needs to download signatures once and no signature is added to the transmitted blocks, whereas some other schemes [7, 8, 9, 12] require repeatedly distributing verification information or appending the signature to every block, which may bring a significant overhead.

VI. Related Work

Besides the work conducted by Gkantsidis et al. [4], some approaches utilizing homomorphic signatures are also proposed in [7, 8]. These approaches, however, require more expensive modular exponentiation computations, which results in high delays if intermediate peers are to compute and verify signatures. Thus, they are too expensive for P2P systems. Agrawal and Boneh [9] proposed a homomorphic MAC scheme. The scheme allows peers to verify every received block with little overhead. However, it is only collusion resistant up to a limited pre-determined bound. In P2P networks, collusion attacks are common. Thus, the scheme is not applicable to P2P systems. Recently, Kehdi and Li [5] proposed a light-weight scheme based on the null-space property of network coding. There are still some drawbacks of this scheme. First, the scheme is vulnerable to collusion attacks, where multiple malicious peers can collude to infer the null keys employed for the verification and let innocent peers accept corrupted blocks. Next, the null keys distribution phase may be attacked. Some schemes [10] aim at correcting errors at decoders. However, these schemes are applicable only when less than a threshold number of corrupted blocks are injected into the network or specified number of network links can be corrupted.

All the above schemes focus on the corruption detection or error correction. Researches on identifying attackers are much less. However, attacker identification is a more efficient approach to thwart pollution attacks in P2P systems. Recently, Wang et al. [12] introduced a malicious peer identification scheme which achieves high efficiency. However, the identification requires the server distribute multiple checksums of all the blocks to all the peers for every generation that experiences attacks. Also, all infected peers need to report their upstream peers, which sent corrupted blocks, to the server with the blocks. The scheme may incur significant communication overhead during the identification, especially in dynamic P2P networks. Next, because the success of the identification relies on peers receiving all the checksums, the identification is vulnerable to collusion attackers that may prevent certain peers from receiving some checksums. Moreover, although the corruption detection based on checksums is computationally efficient, its security is much lower than that of the schemes based on homomorphic hashes.

VII. Conclusion

In this paper, we have proposed a novel and integrated system solution for secure P2P content distribution based on network coding (ISNC) against pollution attacks. In ISNC, by having peers checking blocks probabilistically and making them trigger a malicious peer identification process when they detect corrupted blocks, we are able to efficiently reduce the computation overhead at each peer while preventing corrupted blocks from propagating, and also identify the malicious peers quickly. Compared with the existing related schemes, ISNC is especially applicable for P2P content distribution and can achieve both high security and overall efficiency.

Acknowledgment

This work is supported by National Natural Science Foundation of China under grants 60873225, 60773191, 70771043, National High Technology Research and Development Program of China under grant 2007AA01Z403, and Innovation Fund of Huazhong University of Science and Technology under Grants 2010MS068 and Q2009021.

References

[1] R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung, "Network information flow", IEEE Transactions on Information Theory, vol. 46(4), pp. 1204–1216, 2000.

[2] R. Koetter and M. Medard, "An algebraic approach to network coding", IEEE/ACM Transactions on Networking, vol. 11(5), pp. 782–795, 2003.

[3] C. Gkantsidis and P. Rodriguez, "Network coding for large scale content distribution", Proc. IEEE Infocom, 2005.

[4] C. Gkantsidis and P. Rodriguez, "Cooperative security for network coding file distribution", Proc. IEEE Infocom, 2006.

[5] E. Kehdi and B. Li, "Null Keys: limiting malicious attacks via null space properties of network coding", Proc. IEEE Infocom, 2009.

[6] M. Krohn, M. FreedMan and D. Mazieres, "On-the-fly verification of rateless erasure codes for efficient content distribution", Proc. IEEE symposium on Security and privacy, 2004.

[7] Z. Yu, T. Wei, B. Ramkumar, and Y. Guan, "An efficient signature-based scheme for securing network coding against pollution attacks", Proc. IEEE Infocom, 2008.

[8] D. Charles, K. Jain, and K. Lauter, "Signatures for network coding", Proc. of the 40th annual conference on information sciences and systems, 2006.

[9] S. Agrawal and D. Boneh, "Homomorphic MACs: MAC-based integrity for network coding", Proc. ACNS, 2009.

[10] R. Koetter and F. R. Kschischang, "Coding for errors and erasures in random network coding", IEEE Transactions on Information Theory, vol. 54(8), pp. 3579–3591, 2008.

[11] C. K. Ngai and R. W. Yeung, "Network coding gain of combination networks", Proc. IEEE information theory workshop, 2004.

[12] Q. Wang, L. Vu, K. Nahrstedt, and H. Khurana, "Identifying malicious nodes in network-coding-based peer-to-peer streaming networks", Proc. IEEE Infocom, 2010.

[13] H. Rowaihy, W. Enck, P. McDaniel, and T. La Porta, "Limiting sybil attacks in structured P2P networks", Proc. IEEE Infocom, 2007.