

# SAMcast - A Scalable, Secure and Authenticated Multicast Protocol for Large Scale P2P Networks

Waseem Ahmad, Ashfaq Khokhar  
Department of Electrical and Computer Engineering  
University Of Illinois at Chicago  
Chicago, IL 60607 Email: {wahmad,ashfaq}@ece.uic.edu

*(All rights are reserved. Distribution of the document is allowed only for non-commercial purposes.)*

## Abstract

Overlay networks have shown tremendous potential in solving large scale data dissemination problem by employing peer-to-peer communication protocols. These networks, however, have mostly been used for illegal dissemination of copyrighted material. This paper is aimed at investigating an incentive driven approach to encourage users to actively participate in overlay activities. The users are also discouraged from indulging in illegal distribution of copyrighted material by employing an efficient public key based broadcast encryption scheme along with a deterministic traitor tracing mechanism. We note that public key based broadcast encryption schemes require some mechanism by which a peer can verify the integrity of contents downloaded from other peers. SAMcast is the first protocol, to the best of our knowledge, which provides an efficient integrity verification mechanism along with public key based broadcast encryption. Our experiment results show that the proposed broadcast encryption scheme is highly scalable and the integrity verification is extremely efficient both in terms of computation and communication.

## I. INTRODUCTION

Overlay networks have proven very useful in solving the large scale data dissemination problem. These networks have also given rise to vibrant online social communities which have enormous business potential. To date, these communities have mostly been used for illegal file sharing networks and there has been little effort to make their use for legal distribution of the copyrighted data. We envision a scenario where an entity (say a movie studio) interested in distributing its media to a large number of subscribers, sets up an overlay network of the subscribers. To make effective use of the social networks, the studio gives incentive to all those legal subscribers who bring some of their peers to sign-up for the download service. There are two challenges involved in peer-to-peer distribution of this sort. The studio has to make sure that only those users who have subscribed to the service (we call them as privileged users) are able to access the broadcast contents. Secondly in a peer-to-peer communication model, every peer should be able to authenticate the content downloaded from its peers. In this paper, we present SAMcast which provides an efficient solution to both of the above mentioned problems. We employ Broadcast Encryption techniques[7] to make sure that only privileged users are able to decrypt and view the contents of the broadcast. There has been quite a work done in broadcast encryption both in symmetric [8,7,4] and asymmetric [3,5,2,6] cryptographic domains. Symmetric cryptography based solutions don't allow non-trusted<sup>1</sup> peers to engage in broadcast[2]. This essentially means that peer-to-peer communication models can't be applied to symmetric broadcast encryption schemes. Asymmetric schemes, however, don't suffer from this drawback. Since we are looking into employing peer-to-peer communication model, we will be using an asymmetric cryptography based broadcast encryption scheme. Apart from the enhanced security, another advantage of the asymmetric broadcast encryption is the simplified key management as there is only one public key for the entire network and each user has its own private key (details in SectionIII). One problem with allowing non-trusted peers to engage in broadcast is the possibility of a compromise to the data integrity as a node can change the contents before passing them onto its peers. This problem has been overlooked in most of the schemes proposed so far. SAMcast, however, solves this problem by allowing each peer to verify the contents that it has downloaded from other peers.

Before initiating the broadcast, the main server (movie studio in our example scenario) divides the broadcast content into blocks and builds a hash combined over all these blocks using an incremental hash function of [16] (details on the authentication scheme are provided in Section[IV]).

**Organization:** Rest of the paper is organized as follows. We outline the assumed network model and the basic components of the SAMcast protocol in SectionII. The broadcast encryption scheme employed by SAMcast is detailed in SectionIII. The content integrity verification and authentication framework is described in SectionIV. Incentive driven P2P dissemination approach is described in SectionV. Related work is described in SectionVI. Evaluation of the protocol is provided in SectionVII. Conclusions are drawn and future research direction is provided in SectionVIII.

## II. NETWORK MODEL

As is clear from the introductory discussion, we assume an overlay network setup by an organization such as a movie studio who is interested in selling its media online. Participants can subscribe to the feeds by registering themselves with

<sup>1</sup>Note that only main content server is considered as a trusted one. This server is setup directly by the organization interested in selling its broadcast.

the organization. The download management is provided through a web service or a stand alone application running on the subscriber's machine. This download management service is comprised of following components on the client peers' side.

- A middleware providing necessary P2P communication services. There are a number of nice open source solutions available for building basic P2P applications. Examples of such solutions include JXTA[] and FreePastry[] etc.
- A decryption module which decrypts the encrypted broadcast to its valid contents if the peer is not a traitor.
- A search and track utility which allows users to search for a particular content from a specific organization. This utility allows optimized downloading by leveraging the geographic proximity and available bandwidth of neighboring peers.
- An integrity verification module which verifies the integrity of the contents downloaded from the peers.

Following are the services provided at the trusted server's side apart from the basic P2P infrastructure related services.

- **Encryption Module:** A public key based encryption module which encrypts the content to be broadcasted using the public key of the network. (Note that there is only one public key for the entire network.) Asymmetric cryptographic scheme for broadcast encryption also allows to determine if a peer is a traitor or not.
- **Incremental Hashing Module :** The trusted server is also responsible for maintaining a hash for the entire content available for download. SAMcast employs an incremental hashing scheme which incurs much less verification overhead as compared to Hash Tree (Merkle Tree) based schemes.
- **Traitor Tracing Module :** The traitor tracing module is responsible for performing periodic checks of individual peers to determine if any one of them is involved in illegal activity. All the traitors are placed in a black list and their keys are revoked.

This paper focuses on providing authenticated secure multicast protocol, therefore we will only discuss asymmetric broadcast encryption scheme and incremental hashing based integrity verification scheme. Details of these components are provided in the next couple of sections.

### III. BROADCAST ENCRYPTION

Broadcast encryption ensures that only privileged users are able to access the content. There has been a number of useful contributions made to this area [7,4,8,2]. Some of these schemes rely on symmetric cryptography and others on asymmetric cryptography. The symmetric cryptographic schemes don't allow any node other than the main server to engage in the broadcast operation [2], thus rendering P2P communication models ineffective, therefore, we limit our focus to the asymmetric schemes. Moreover, symmetric schemes suffer from complex key management[2,6] and rekeying schemes to maintain backward and forward secrecy[7]. On the other hand, efficient asymmetric schemes have simplistic key management properties where there is only one public key and  $n$  private keys where  $n$  total number of subscribers. Most of the asymmetric schemes proposed so far are based on polynomial interpolation and employ discrete log problem. Asymmetric schemes employ public key based schemes to encrypt the session key which is used to encrypt and decrypt the actual broadcast content. We use a scheme which is similar to the one proposed in [9]. But we also provide details on how to perform selective encryption [13] of the underlying multimedia data based on the session keys. Moreover, the scheme in [9] does not consider the authentication problem while SAMcast provides an efficient solution to the authentication and integrity verification problem. One of the most focused areas in broadcast encryption research is "*traitor tracing*" problem. A traitor is a subscriber who gives away his key to some non-privileged users who can then access the broadcast contents illegally. It is much easier to trace a single traitor if his key has been used illegally. More challenging is the task to trace a group of traitors who collude to generate a new key which can be used to decrypt the broadcast contents. The basic asymmetric broadcast encryption scheme with traitor tracing capabilities is presented as follows.

**Assumptions :** Let  $k$  be the maximum number of colluded receivers (traitors) and  $z$  be the revocation threshold i.e. at most  $z$  private keys of traitors can be revoked. We set  $z \geq 2k$ .

**Asymmetric Key Generation:** Let  $G_p$  be a group of large prime order  $p$ . The main server selects a degree- $z$  polynomial  $f(x) = \sum_{t=0}^z a_t x^t \pmod p$  where coefficients  $a_i \in Z_p \forall i$ . The server's secret key is  $f(x)$  and its public key is

$$\langle g, g^{a_0}, g^{f(1)}, \dots, g^{f(z)} \rangle$$

**Subscription :** When a subscriber  $i$ ,  $i > z$  subscribes to the download service, the server gives the subscriber  $i$  with the share (private key for  $i$ ) which is  $(i, f(i))$ . The subscriber verifies his key by verifying

$$g^{a_0} = \prod_{t=0}^z g^{f(x_t)\lambda_t}$$

where  $x_0 = 1, x_1 = 2, \dots, x_{z-1} = z, x_z = i$ . If its turns out to be true then the subscriber  $i$  has a private key  $(i, f(i))$ . We call  $(j, f(j))$  an unused share if it has not been assigned to any subscriber.

**Public Key Encryption of the Session Key:** The main server randomly selects  $z$  unused shares

$$(j_1, f(j_1)), (j_2, f(j_2)), \dots, (j_z, f(j_z))$$

and also a random number  $r \in Z_p$  and a session key  $s$ . The receiver computes

$$T = \langle sg^{a_0}, g^r, (j_1, g^{rf(j_1)}), (j_2, g^{rf(j_2)}), \dots, (j_z, g^{rf(j_z)}) \rangle$$

and broadcasts  $\langle T, \text{selectiveEncrypt}(s, M) \rangle$  where *selectiveEncrypt* is a selective encryption function applied on  $M$  which is the content to be broadcasted.

**Selective Encryption :** We employ a selective encryption technique as proposed in [13] to efficiently encrypt the multimedia data. As mentioned above, the sender who wants to broadcast the content, generates a symmetric key  $s$  called as session key. This key is then used to perform selective encryption e.g. in the case of MPEG video, only DC coefficients and motion vectors are encrypted. It has already been proved [13] that selective encryption of this sort results in the same security level as attained by fully encrypting the entire content.

**Decryption :** Each subscriber, upon receiving  $\langle T, \text{selectiveEncrypt}(s, M) \rangle$ , computes  $s$  by

$$\begin{aligned} \frac{sg^{ra_0}}{[(g^r)^{f(i)\lambda_z} \cdot \prod_{t=0}^{z-1} (g^{rf(x_t)})^{\lambda_t}]} &= \frac{sg^{ra_0}}{g^{r \sum_{t=0}^{z-1} f(x_t)\lambda_t + f(i)\lambda_z}} \\ &\Rightarrow \frac{sg^{ra_0}}{g^{ra_0}} = s \end{aligned}$$

The subscriber subsequently uses  $s$  to decrypt the content.

**Traitor Tracing :** It was observed in [9], that if a pirate  $P$  gets a share by linear combination of keys of a subset of  $m$  traitors then the  $P$  and the traitors together cannot compute  $a_0$  or  $g^{ra_0}$ . Like in [9], we base our traitor tracing algorithm on this observation. If the main server suspects that users  $j_1, j_2, \dots, j_m, m \leq z$  are traitors, the server broadcasts cipher block *selectiveEncrypt*( $s, M$ ) and the block containing encrypted session key

$$\langle sg^{ra_0}, g^r, (x_{j_1}, g^{rf(x_{j_1})}), \dots, (x_{j_m}, g^{rf(x_{j_m})}), (l_1, g^{rf(l_1)}) \rangle$$

$, \dots, (l_{z-m}, g^{rf(l_{z-m})}) \rangle$

where  $l_1, l_2, \dots, l_{z-m}$  are arbitrarily chosen and different from  $x_{j_1}, x_{j_2}, \dots, x_{j_m}$ . A subscriber who is not a traitor can compute  $g^{ra_0}$  and thus  $s$ . We confirm that  $j_1, j_2, \dots, j_m$  are traitors if they together cannot decrypt the cipher block properly. This method of traitor tracing is formally termed as Black Box Tracing.

#### IV. INTEGRITY VERIFICATION OF THE CONTENT IN SAMCAST

One downside of having non-trusted nodes engaged in broadcast is that these nodes might compromise the integrity of the content. This is the direct result of having asymmetric cryptographic approach to the content security. Given the benefits of asymmetric schemes, it is important that this flaw be removed by employing efficient authentication and integrity verification mechanisms. While most of the previous work done in asymmetric secure multicast has ignored this issue, SAMcast attempts to build a highly efficient solution to this problem. There has been considerable amount of work done in integrity verification problem in various domains. Simple digital signatures based schemes fail to scale as they require considerable bandwidth and computational resources.  $k$ -time signature schemes[15] perform well compared to the one time signature schemes but still incur considerable bandwidth cost e.g. the  $k$ -time signature scheme proposed in [15] generates 300 bytes for each signature. Signature chains based solutions [10] are not tolerant to packet loss as loss of one packet would render verification for the entire stream ineffective and subsequent retransmission of the packets would yield high communication overhead. Merkle hash/signature trees have been used in various applications [14]. Each unit of data transmission (a packet) is treated as a leaf of the tree and each inner node is a combined hash of the hash values of its children. The root of the tree is stored in a trusted location. Integrity verification of each packet requires calculating the hash of the packet, combining it with the hash of its siblings and combining and verifying the hash values of relevant inner nodes all the way up to the root. If the value of the calculated hash matches with the stored hash, packet is deemed authenticated. The schemes employing Merkle Trees [11] have to transmit all relevant hashes along with the packet so as to allow receiver to verify its integrity. This results in tremendous bandwidth wastage.

Incremental multiset hashes based integrity verification was first employed in [16], to verify the integrity of memory operations in secure processors. A multiset is a finite unordered group of elements where an element can occur as a member more than once. [16] gave a complete construction of a family of collision-resistant incremental multiset hashes. For example, the multiset hash function based on addition operation is given as follows. Let  $B = \{0, 1\}^m$  represents the set of bit vectors of length  $m$  and let  $M$  be a multiset of elements of  $B$ . The number of times  $b \in B$  is in the multiset  $M$  is denoted by  $M_b$  and is called the multiplicity of  $b$  in  $M$ . Let  $H_K : \{0, 1\}^{m+1} \rightarrow Z_n^l$  be randomly selected from a pseudorandom family of hash functions. Let

$$L \approx n^l \approx 2^m, L \leq n^l, L \leq 2^m$$

$$\hat{h}_K(M) = \left[ H_K(0, r) + \sum_{b \in B} M_b H_K(1, b) \text{mod } n; \sum_{b \in B} M_b \text{mod } L; r \right]$$

where  $r \in B$  is a random nonce.

It was shown in [16] that incremental multiset hashes easily outperform Merkle trees if the integrity verification operation is performed over a sequence of operations on un-trusted memory as opposed to each single operation. This is true in our case where integrity verification operations are performed on block levels as compared to on each packet level. We can also employ a probabilistic verification scheme whereby a random block is picked up to verify its integrity. These sorts of non-deterministic techniques for integrity verification are used to defy denial of service attacks as well.

In SAMcast, the main server divides the entire content into  $n$  blocks. It creates an incremental hash over all these blocks.<sup>2</sup> Now a subscriber gets registered to download that content. The download management services hooks the subscriber up with some other peers in order to minimize the latency involved in downloading the content. Like in traditional P2P networks, download management services might download different blocks from different peers. In order to verify the integrity of these blocks, the client peer builds a hash of all the blocks when they are completely downloaded and match this hash with the server's hash. If the hashes match, then the integrity of the downloaded content is intact. If, however, the hashes don't match then the server and the client get engaged in a synchronized hash building process whereby they compare hash of each block with each other. Blocks which result in different hashes are marked as invalid and are downloaded from different set of peers this time. The providers of corrupt blocks are brought under investigation. If there happens to be more than a threshold number of complaints against a peer then that peer is removed from the overlay. The reason behind this removal is that it might have gone under some adversary's control or its link is so lossy that error correction schemes are unable to remove all the errors. Moreover, that peer is put into the list of those peers which are being investigated for their alleged roles as traitors.

## V. INCENTIVE DRIVEN APPROACH FOR SECURE BROADCAST IN SAMCAST

In the discussion so far, we have only considered a P2P communication approach where content is only *pulled* from other peers on demand. In our model, we encourage peers to get more subscribers to the network. Incentives are also provided to those peers who allow other peers to download content from their machines. These incentives can be in the form of cash credits or free downloads. This incentive driven scheme is aimed at encouraging peers to participate actively in the content dissemination process. Moreover, this scheme along with technical solutions to traitor tracing problem should discourage illegal dissemination of copyrighted data over such an overlay network. The information dissemination over the envisioned overlay network can be seen as consisting of pull operations where each peer contacts others for specific content.

## VI. RELATED WORK :

**Broadcast Encryption:** The area of Broadcast communication was first formally studied by Fiat and Naor[7]. As mentioned above, there are two categories of Broadcast encryption systems. Those which employ symmetric cryptography[3,7,4] and those which employ asymmetric cryptography [2,8,6,9]. Symmetric cryptographic schemes, though suitable in many applications, are inherently unscalable as they require a single trusted server to disseminate the content. Most of the symmetric cryptographic broadcast encryption schemes incur high storage costs at client nodes for storing a huge number of private keys. They also require complex rekeying schemes to maintain forward and backward secrecy [2]. On the other hand, the symmetric cryptographic scheme employed by SAMcast not only is scalable as it allows third parties to distribute content but also requires minimal key storage. There is only one public key for the entire network and each subscriber has a single private key. The public key based broadcast encryption schemes proposed in [9] are very similar to ours but they don't consider verification of the integrity of the content when un-trusted nodes are involved in the broadcast operations.

**Traitor Tracing :** Traitor Tracing schemes were introduced by Chor et al. [5], who employed a probabilistic design: each user possesses a different subset of a set of keys and tracing is achieved using the properties of the key assignment. The results were later implemented with concrete combinatorial designs in [17]. These techniques were extended by [7] who also considered the combination of traitor tracing schemes with efficient revocation methods. These schemes are not scalable, since (1) They do not support public key technology in an efficient fashion. (2) They employ combinatorial designs for the key-assignment that requires a tight guess of an a-priori bound on the number of users. (3) The cipher text size is an increasing function of the total number of revoked users in the system's life-time.

The schemes in [3], though, provided deterministic traceability but did not consider revocation of keys.

**Integrity Verification:** As discussed above, integrity verification is essential in public key based broadcast encryption schemes as non-trusted nodes are also engaged in broadcast operations. Gennaro and Rohatgi [10] introduced hash chains where hash of each packet is sent along with the next packet. This scheme does not tolerate packet losses and incurs high communication overhead. Perrig et al. [11] proposed TESLA and EMSS for efficient and secure multicast. These schemes require strict ordering of the packets which can't be guaranteed in the P2P distribution scenarios mentioned above. Moreover, if the supplying clients generate the keys and sign the digests as in TESLA, they may not be acceptable to other clients because clients are not assumed to be trustworthy. Habib et al. [14] employ Merkle Tree based efficient integrity verification. Their scheme reduces

<sup>2</sup>Note that incremental hashes are extremely useful when content is updated. In case of such updates, incremental hashes only need to generate the hash of the change and then add it to the stored hash. (We will have to make sure that all such updates are appended to the hash along with a time stamp so as to defy replay attacks[16]) In case of Merkle trees, we will need to update all inner nodes as well as the root of the tree

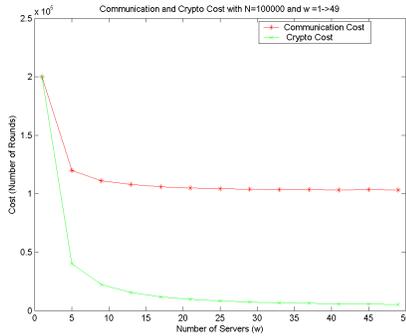


Fig. 1. Communication and Crypto cost for Total number of subscribers  $N = 100,000$  with varying number of peers to download from ( $1 \leq w \leq 49$ ).

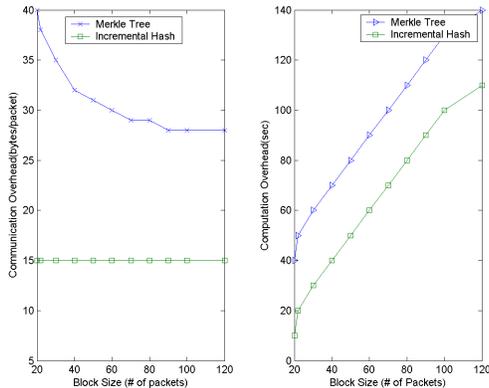


Fig. 2. Comparison of communication/computation overhead for Incremental hash and Merkle tree based integrity verification

the communication overhead by sending the digests of one subtree before sending any data. It was observed in [16] that incremental multiset hashes outperform Merkle Tree based integrity verification when integrity verifications are not very frequent. We make use of this observation in SAMcast to devise a memory integrity verification scheme based on incremental multiset hashes. Moreover, since incremental hashes require very small storage compared to other schemes and also because they can be computed very efficiently, we don't transmit signatures along with content blocks. This saves precious bandwidth for us. The integrity verification scheme is performed offline as outlined in the Section IV. This entire strategy makes our integrity verification scheme extremely efficient and lightweight.

## VII. EVALUATION :

The SAMcast protocol is still under development. We have, however, implemented the asymmetric broadcast encryption scheme along with the incremental hash based content integrity verification scheme. These schemes were simulated in a modified version of Peersim [1]. For ease of analysis, communication cost is measured as the average number of communication rounds required for each peer to download the content in its entirety. We have assumed a static network and we don't consider network churn. This keeps things simple and easy to analyze. Our first set of experiments is aimed at investigating the scalability of the proposed broadcast encryption framework. We performed this experiment by varying the number of peers that a specific peer can download the content from. As shown in the Figure.1, it is clear that by leveraging the bandwidth offered by neighboring peers, we can significantly improve the communication cost compared to the case where there is only one server responsible for content distribution. Our second set of experiments is aimed at checking the communication and computation overhead of the integrity verification scheme. As shown in the Figure.2, the communication overhead is almost negligible compared to Merkle Tree based schemes. This is because we don't transmit signatures with each block. This also saves a lot of computational effort which otherwise is required for computing the signatures for each transmitted block on the server side.

## VIII. CONCLUSIONS

We have proposed a secure and authenticated multicast scheme for peer-to-peer communication based large scale content distribution. This scheme is aimed at encouraging users to actively participate in the overlay activities and discouraging them

from indulging in illegal dissemination of copyrighted data. The scheme employs a public key based broadcast encryption mechanism with traitor tracing. It also allows non-trusted peers to engage in broadcast operations. To cater for the possible compromise of the integrity of the content, we provide an efficient incremental hash based content integrity verification scheme. This scheme outperforms Merkle Tree based schemes. The development of the protocol is an ongoing effort. We are looking forward to carry out extensive simulations of the protocol along with practical deployment of the protocol on planet-lab.

#### REFERENCES

- [1] Mrk Jelasity, Alberto Montresor, and Ozalp Babaoglu: *Gossip-based aggregation in large dynamic networks*. ACM Transactions on Computer Systems, 23(3):219252, August 2005
- [2] Y. Dodis, N. Fazio, A. Kiayias, and M. Yung: *Scalable public-key tracing and revoking*. In Proc. 22nd ACM Symposium on Principles of Distributed Computing, 2003. Full version at <http://eprint.iacr.org/2004/160>.
- [3] D. Boneh and M. Franklin. *An Efficient Public Key Traitor Tracing Scheme*. In Advances in Cryptology—Crypto '99, pages 338-353. Springer-Verlag, 1999. LNCS 1666. Full version available at [crypto.stanford.edu/dabo/pubs.html](http://crypto.stanford.edu/dabo/pubs.html).
- [4] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pinkas. *Multicast Security: A Taxonomy and some Efficient Constructions*. In Proceedings of IEEE INFOCOM '99, volume 2, pages 708-716, 1999.
- [5] B. Chor, A. Fiat, and N. Naor. *Tracing Traitors*. In Advances in Cryptology—Crypto '94, pages 257-270. Springer-Verlag, 1994. LNCS 839.
- [6] Y. Dodis and N. Fazio. *Public-Key Trace and Revoke Scheme Secure against Adaptive Chosen Ciphertext Attack*. In Public Key Cryptography—PKC '03, pages 100-115. Springer-Verlag, 2003. LNCS 2567.
- [7] A. Fiat and M. Naor. *Broadcast Encryption*. In Advances in Cryptology—Crypto '93, pages 480-491. Springer-Verlag, 1993. LNCS 773.
- [8] K. Kurosawa and Y. Desmedt. *Optimum Traitor Tracing and new Direction for Asymmetry*. In Advances in Cryptology—EuroCrypt '98, pages 145-157. Springer-Verlag, 1998. LNCS 1403.
- [9] W.G. Tzeng and Z.J. Tzeng. *A Public-Key Traitor Tracing Scheme with Revocation Using Dynamics Shares*. In Public Key Cryptography—PKC '01, pages 207-224. Springer-Verlag, 2001. LNCS 1992.
- [10] R. Gennaro and P. Rohatgi, *How to Sign Digital Streams*, technical report, IBM T.J. Watson Research Center, 1997.
- [11] A. Perrig, R. Canetti, D. Song, and D. Tygar, *Efficient and Secure Source Authentication for Multicast*, Proc. Network and Distributed System Security Symp., Feb. 2001.
- [12] A. Perrig, R. Canetti, J.D. Tygar, and D.X. Song, *Efficient Authentication and Signing of Multicast Streams over Lossy Channels*, Proc. IEEE Symp. Security and Privacy, pp. 56-73, Nov. 2000.
- [13] H. Yin, C. Lin, F. Qiu, J. Liu, G. Min, and B. Li, *CASM: A Content-Aware Protocol for Secure Video Multicast*, IEEE Transactions on Multimedia, Jan 2006.
- [14] Ahsan Habib, Dongyan Xu, Bharat Bhargava, Mike Atallah, John Chuang: *A Tree-based Forward Digest Protocol to Verify Data Integrity in Distributed Media Streaming*. IEEE Transactions on Knowledge and Data Engineering (TKDE), 17 (2005), pp. 1010-1014.
- [15] P. Rohatgi, *A Compact and Fast Hybrid Signature Scheme for Multicast Packet*, Proc. ACM Conf. Computer and Comm. Security, pp. 93-100, Nov. 1999
- [16] D. Clarke, S. Devadas, M. van Dijk, B. Gassend, and G. E. Suh. *Incremental multiset hash functions and their application to memory integrity checking*. In Asiacrypt 2003.
- [17] D. R. Stinson and R. Wei. *Combinatorial Properties and Constructions of Traceability Schemes and Frameproof Codes*. SIAM Journal on Discrete Mathematics, 11(1):41-53, 1998.