Message Dropping Attacks in Overlay Networks: Attack Detection and Attacker Identification

Liang Xie[1] Sencun Zhu[1,2]
[1]Department of Computer Science and Engineering, [2]College of Information Sciences and Technology
The Pennsylvania State University, University Park, PA 16802
{lxie, szhu}@ cse.psu.edu

***Abstract*— Overlay multicast networks are used by service providers to distribute contents such as web pages, streaming multimedia data, or security updates to a large number of users. However, such networks are extremely vulnerable to message dropping attacks by malicious or selfish nodes that intentionally drop packets they are required to forward. It is difficult to detect such attacks both efficiently and effectively, not mentioning to further identify the attackers, especially when members in the overlay switch between online/offline statuses frequently. We propose a random-sampling–based scheme to detect such attacks, and a path-resolving–based scheme to identify the attack nodes. Our schemes work for dynamic overlay networks and do not assume the global knowledge of the overlay hierarchy. Analysis and simulation results show that our schemes are bandwidth-efficient and they both have high detection/identification rates but low false positive rates.**

Index Terms: Overlay networks, multicast, message dropping attacks, network dynamics.

I. Introduction

Multicasting is an efficient mechanism for packet delivery in one-to-many data transfer applications. Although IP multicast has long been regarded as the right mechanism for large-scale group communication applications due to its efficiency, its full deployment has been very difficult and slow due to its dependency on routers and ISP's reluctance to turn on IP multicast. A new alternative way is to move the multicast functionality from routers to end-hosts, i.e. let end-hosts that form a multicast group to replicate and forward packets on behalf of the group. This is called overlay multicast, also called end system multicast or application-level multicast [8]. Based on standard unicast mechanisms, hosts participate in an application session share responsibility for forwarding information to other hosts. Thus, although additional overheads are put on end hosts, the mechanism is more applicable because of its easiness for deployment. Examples of applications that benefit from overlay multicast include the distributions of web pages, streaming multimedia data, real-time stock quotes, etc.

Although many research effort has been put on overlay multicast networks, most of them focus on network constructions, data deliveries, and routing. Security issues in overlay multicast networks have not been considered adequately. The existing work either investigates the impact of selfish cheating nodes on the performance of overlay multicast [11], key management[19], or improves the fault-tolerance or denial-of-service(DoS) resilience of overlay network by introducing redundancy [15], [17], [4], [19] in overlay multicast communications.

Contributions We study the *message dropping attack*, in which selfish or malicious nodes (referred to as compromised nodes hereafter) in an overlay hierarchy intentionally drop the messages they have received, causing denial of service to the nodes in its subtree. Unlike all the previous research[15], [17], [4], [19] which attempts to tolerate message dropping attacks, our work makes the first effort to detect this attack and further identify the compromised nodes.

More specifically, we first examine the severity of the message dropping attacks, then propose two efficient security schemes: a random sampling based scheme to detect potential message dropping attacks in the network and a path resolving based scheme to identify compromised nodes. Besides considering static overlay networks, we also study these schemes in dynamic overlay networks where global knowledge of the overlay hierarchy is not assumed. Theoretical analysis and simulation results show that both schemes are effective and bandwidth-efficient, and also have low false positive rates.

Organization The remainder of this paper is organized as follows: Section II discusses some related work on overlay multicast networks. Section III describes the system model and our design goal. In Section IV, we present our basic schemes of malicious node detection and identification in a static overlay. In Section V, we extend these two schemes for dynamic overlay networks. We evaluate our schemes through theoretical analysis and simulations in Section VI. Section VII draws the conclusion.

II. Related Work

Mathy et al [11] studied the impact of selfish nodes cheating about their distance measurements in application-level multicast overlay tree. Ngan et al [12] presented mechanisms that distinguish selfish nodes from their peers. The peers make their judgments strictly by observing the behavior of their upstream peers. Wright et al [16] presented k-redundant depender graphs for distributing public-key certificate revocation lists (CRLs), which provides every node in the graph with k disjoint paths to the root of the graph, thus guaranteeing delivery even when up to $k-1$ paths between them have failed. Song et al [15] improved the scalability of the above scheme by presenting expander graphs for constructing robust overlay networks that have constant degree. Yang et al [17] proposed to augment

1-4244-0423-1/06/$20.00 ©2006 IEEE

tree-like hierarchy with hierarchical overlay networks, which is actually also a type of graphs, to achieve DoS resilience. Drabkin et al [6] designed an overlay based Byzantine tolerant broadcast protocol, which overcomes Byzantine failure by combining digital signatures, gossiping of message signatures, and failure detectors.

Banerjee et al [4] introduced a probabilistic forwarding scheme for overlay multicast. In their scheme, every node forwards received packets to a randomly selected set of nodes. Zhu et al [19] described a k-RIP scheme in which the distribution server injects k copies of the same message into k randomly selected nodes in the network. These schemes provide probabilistic guarantee on message delivery ratio in the presence of message dropping attacks.

Jun et al [10] propose a technique, which is similar to TCP sliding window, for fault tolerance in overlay applications. In their scheme, each message distributed has a sequence number and periodically the root generates a signed certificate which contains the last sequence number. Nodes in the overlay either do not receive this (and hence know a parent is dropping packets) or receive it and realize that they have not receive the most recent multicast packets and hence again know that a parent is dropping packets. They then reconnect to the tree at a better point. Their method, however, assumes that the root maintains a topology of the entire overlay. Moreover, in a dynamic environment where packet losses caused by node joins/departures cannot be ignored, failing to receive a sequence message from the root does not necessarily mean a malicious parent to the node.

In addition to the greatly increased bandwidth overhead, many of these techniques require to maintain certain network connectivity or structure to provide security guarantee; moreover, they assume the global knowledge of the overlay hierarchy, as is usually impossible for dynamic environments. Unlike the above techniques, our work adopts a sampling-based approach to detect the existence of message dropping attacks in the first place, then tries to identify the attackers. As such, our techniques provide another security building block other than fault tolerance towards making overlays more secure.

III. Assumptions and Design Goal

The System Model There are potentially a large number of application scenarios of overlay multicast, which are characterized by different parameters, e.g., group size, membership dynamics, number of data sources. It seems unlikely that a single system model can describe all these scenarios. Therefore, we focus on a specific application scenario, which we believe is (or will be) very representative. We consider a commercial application of overlay multicast, in which a service provider distributes data (e.g., live content or streaming media) to a large number of subscribers (also called member nodes hereafter).

For simplicity, we assume that online nodes are self-organized into an overlay multicast delivery tree rooted at a distribution server of the service provider. The algorithms for constructing and maintaining overlay multicast trees [3], [8], [9] are out of the scope of our work. In this model, packets are propagated downwards in the hierarchy and reach every online member node if there is no any packet losses and attacks. We expect that the normal behavior of each node is to forward each packet it receives to its current child nodes in the overlay hierarchy. To focus our research on security issues and avoid complications from group member controls, here we assume a closed model where the population of the system remains stable.

We assume that each member node possesses a security credential such as a public key certificate, which is authorized by the service provider, so that it can join the overlay network. Every node only accepts authorized nodes as its parent node or child nodes. A key management scheme such as the one in [19] may be applied to revoke the nodes that are identified as compromised, as this work does not deal with the node revocation issue.

Attack Model Because an unauthorized node cannot join the overlay network without a valid credential, we only focus on authorized but misbehaving insider nodes. We assume that a compromised node launches the message dropping attack to cause denial of service to the other nodes in its subtree and multiple compromised nodes, controlled by the same attacker, may collaborate in launching this attack. Moreover, we assume an intelligent attacker who knows the defense strategies of the group controller (GC), the tree topology of the overlay, and can place the compromised nodes in the positions of his choice. However, due to the use of security credential for joining the network, a node cannot lie about identity.

Design Goal To detect this attack, the simplest solution is to query every online node some time after the server broadcasts a message. A negative acknowledgement indicates the existence of such an attack, and further investigation may be made to further identify the compromised nodes. Obviously, this scheme does not scale at all with the network size. Therefore, our goal is to perform attack detection and attacker identification in an *efficient*, *scalable* while still *effective* way.

IV. Sampling Schemes for Static Overlay Networks

This section studies the problem of attack detection and attacker identification in a static overlay, where member nodes remain online once they have joined the network. The nodes communicate with one another through a TCP-based protocol; therefore, we assume that packet losses are only caused by message dropping attacks. We make these assumptions, which may not be very realistic, to help understand our schemes for dynamic networks introduced in the next section.

We assume a static balanced tree-structured topology which is known to both the group controller and the attackers. Let the degree of the tree be *d* and the height *H*, the population of the overlay is $N = \frac{d^{H+1}-1}{d-1}$. We use *LCA* to denote the *lowest common ancestor* of a set of nodes. In Fig.1, node m and i share a LCA level c, denoted as $LCA(m, i) = c$.

A. Attacking Strategy

For an attacker, the optimal policy to place his compromised nodes in the overlay network is to keep as many legitimate nodes (victims) as possible inside the subtrees of the compromised nodes, so that broadcast messages cannot reach these victims. Meanwhile, the attacker does not want his compromised nodes to be easily detected by the GC's defense strategy. If we define an overall *attacking gain* $G_{adv} = (1 - P_{det}) \cdot \frac{\#victim}{\#total}$ for the attacker, where P_{det} denotes the possibility that an attack is detected, then the attacker's goal becomes to find an optimal tree level h^* for his compromised nodes and try to put them as sparsely as possible in the overlay, so that the attacking gain is maximized. When there are multiple attackers in the overlay, the solution of this optimization problem is that all compromised nodes are at the same optimal height h^* and there is no any compromised node placed within the subtree of the another. Otherwise, the number of victim nodes will be reduced without increasing the attacking gain. In the rest of the paper, we always refer to the optimal attack as the worst case when we discuss the GC's countermeasures. Such case strongly favors the attacker and guarantees a lower-bound for effectiveness of the GC's defense scheme.

B. Attack Detection

The problem of attack detection is trivial if the GC queries every node whether it has received a previously broadcast message. However, this clearly does not scale with group size. A more practical solution is to adopt a random sampling strategy, making a trade-off between detection rate and overhead. Below we introduce two sampling schemes.

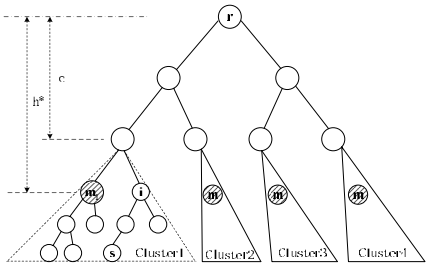


Fig. 1. The GC's sampling schemes: r-root node, m-malicious node, s-sampled node

1) Scheme I: An Intuitive Random Sampling Scheme: In this scheme, the GC samples every node with the same probability. It periodically selects a random subset of nodes and queries each of them regarding the receiving statuses of a sequence of previously broadcast messages. Every sampled node replies with an authenticated acknowledgment (ACK) in a bitmap format where each bit value represents either the node has received (1) or missed (0) the message. We name it a *positive* ACK when the bitmap is all-1, and a *negative* ACK otherwise. Since a compromised node tries to hide itself, it reports a positive one when receiving a query. The GC will detect the attack if it receives a negative ACK.

Security Analysis Suppose the optimal attacker places m malicious nodes as sparsely as possible at tree level h^*. We have the portion of nodes that are denied from receiving broadcast messages: $p_s = \frac{m \cdot N_c}{N} = \frac{md \cdot (d^{H-h^*}-1)}{d^{H+1}-1} \approx m \cdot (d^{-h^*} - d^{-H})$, where N_c is the number of victims in the subtree of a malicious node. When GC randomly samples a node, the probability that a victim node is sampled (hence an attack is flagged) is also p_s. Let S_n be the number of sampling messages the GC can send, which is a system parameter limited by the resources of GC, we compute the GC's detection rate for scheme I as follows:

$$\begin{aligned} P^I_{det} &= 1 - P^I_{miss} = 1 - (1 - p_s)^{S_n} \\ &= 1 - (1 - m \cdot d^{-h^*} + m \cdot d^{-H})^{S_n} \end{aligned} \quad (1)$$

As we mentioned, the optimal attacker's goal is to maximize $G_{adv} = P^I_{miss} \cdot \frac{\#victim}{\#total}$. We solve this optimization problem and get $h^* = H - \lfloor log_d[\frac{d^H}{m \cdot S_n} + 1] \rfloor$ for compromised nodes. Thus, we derive the worst cast of the GC's detection rate as follows:

$$P^I_{worst} = 1 - (1 - S_n^{-1} - 2m \cdot d^{-H})^{S_n} \quad (2)$$

Scheme I is simple but not optimal because it samples many intermediate nodes. To cause message dropping attacks to others, compromised nodes need to be intermediate nodes. If we sample an intermediate compromised node, we will not receive an honest ACK. Also, it is redundant if we sample multiple nodes on the same path because if the one closest to the leaf, if not compromised, reports a positive ACK, other nodes on the same path normally also report positive ACKs. These observations indicate the potential of a more effective sampling scheme instead of purely randomly sampling the nodes.

2) Scheme II: A Group-based Sampling Scheme: We present a group-based sampling scheme, in which the GC decides locally a subset of *leaf* nodes and sends its query message to each of these selected nodes. Note that the GC still accepts bitmap-format ACKs for a sequence of broadcast messages. The algorithmic detail is as follows:

1) Based on the limit of its sampling bandwidth, the GC first decides a LCA level $c = \lfloor log_d S_n \rfloor$, with which it starts to group nodes.
2) The GC then forms $k \leq d^c$ sampling clusters (subtrees), whose root nodes are at level c, as illustrated in Fig.1. If $k < d^c$, the GC may randomly choose *k* out of d^c members as parent nodes to form sampling clusters.
3) Finally, the GC randomly samples $w = \lceil S_n/k \rceil$ leaf candidates within each cluster. Queries and acknowledgments are direct interactions between the GC and sampled nodes.

In a special case when $c = 0$ and $k = 1$, the entire tree is a sampling cluster. This group-based technique ensures fairness on sampling leaf nodes. It keeps *k* sampling groups as sparse as possible, thus the GC may include more compromised nodes on its sampling paths. On the other hand, the GC needs sampling density (w) so that it may find potentially multiple attackers within a sampling group. The GC seeks an optimal balance between these two aspects.

Security Analysis In this case, the attacker's best strategy remains the same, i.e., to place his compromised nodes as sparsely as possible at an optimal tree level h^*. However, knowing the GC's defense policy, the attacker will not choose its $h^* \leq c$. The reason is that when the attacking level h^* is closer to the root than c, compromised nodes will always be included in the GC's sampling paths. More specifically, when there are $m \leq d^c$ malicious nodes, the attacker chooses to evenly distribute them into m-out-of-d^c clusters, each of them stays at optimal level h^*.

Consider an arbitrary leaf node s in Fig.1. Its probability of being randomly sampled is $p_s^{'} = \frac{k \cdot w}{d^H}$. For a malicious node at tree level h^*, its probability of being included in w random samplings within the cluster is:

$$P_{in} = \frac{k}{d^c} \cdot (1 - (1 - \frac{1}{d^{h^*-c}})^{w}) = \frac{k}{d^{h^*}} \quad if \quad k = d^c. \quad (3)$$

To simplify the problem, we always choose $k = d^c$ (i.e., w=1) in later discussions. We can easily see from the equation that $P_{in} = 1$ when $c = h^*$.

A false negative error might occur when the GC fails to include any of the compromised nodes on its sampling paths. We may compute the GC's probability of missing sampling all m compromised nodes as follows:

$$P_{miss}^{II} = (1 - P_{in})^m = (1 - \frac{k}{d^{h^*}})^{m}, \quad (4)$$

and the GC's overall detection rate as: $P_{det}^{II} = 1 - P_{miss}^{II}$. It can be observed that the false negative rate increases with the optimal attacking level h^* .

Since the attacker's goal is to maximize its G_{adv}, we solve this optimization problem for scheme II and get $h^* = \lceil log_d 2k \rceil$ for compromised nodes. Thus, the worst detection rate of scheme II is as follows:

$$P_{worst}^{II} = 1 - (1 - \frac{1}{w+1})^{m} \quad (5)$$

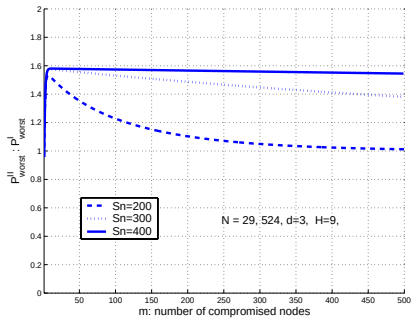


Fig. 2. Comparison of the worst-case detection rates in Scheme I and Scheme II.

3) Comparison of Scheme I and Scheme II: From Equ.2 and Equ.5, we derive that when $m > -k \cdot log_2[\frac{S_n - 1}{S_n}]$ (as always satisfies when $m > 1$), $P_{worst}^{II} > P_{worst}^{I}$, which indicates that scheme II has a higher worst-case detection rate than scheme I. Fig.2 makes the comparison based on the above analytical results. It is shown that the GC achieves a higher detection rate when adopting sampling scheme II. The figure also indicates that the difference between the two schemes becomes more distinct as the sampling size S_n increases.

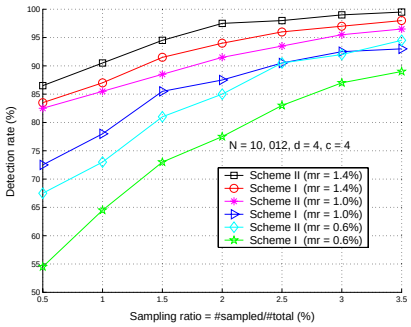


Fig. 3. Comparison of the average-case detection rates of the two schemes (f_m denotes the fraction of nodes that are malicious).

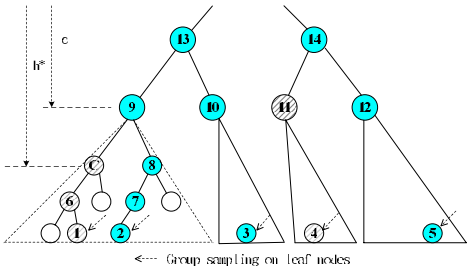
Based on simulation, Fig.3 compares the average-case detection rates of the two schemes as the function of sampling ratio and the fraction (f_m) of malicious nodes. The figure shows that, when using the same sampling size, scheme II outperforms scheme I in terms of detection rate. It also shows that more severe attacks take a higher risk of being detected by the GC, as can be derived from Equ.1 and Equ.4.

We notice that in both schemes, a larger sampling size leads to a higher detection rate at the cost of larger message overhead. Both the detection schemes introduce $2 * S_n$ additional messages (queries + ACKs) to the overlay. In the case when S_n/N is small and the bitmap form of broadcast messages is used, the message overhead approaches $O(1)$. Compared to a flooding-based approach, whose message complexity is $2|E| * 2 > 2 * S_n$, both our detection schemes are lightweight.

C. Attacker Identification

The previous attack detection schemes can only report with a certain probability if the message dropping attack exists or not; it cannot identify the compromised nodes. Next we propose a path-resolving (PR) scheme to identify the compromised nodes without increasing the number of sampled nodes. For the scheme to work, we assume that every leaf node knows all the nodes on its path to the root.

Specifically, as in our group-based sampling scheme, the GC periodically queries $S_n = k \cdot w$ leaf nodes, w from each cluster, regarding their message receiving statuses. Each sampled node j replies in its ACK a tuple of two data fields: a root path r_j including the ids of the enroute nodes from j to the root; a receiving status s_j, which consists of a bitmap indicating whether j has successfully received a sequence of previously broadcast message specified in the query. The GC uses a table T_s to store these query records, as illustrated in Fig.4(b). Also, it keeps a global list L_n for all member nodes in the overlay. Each entry i of L_n contains a suspicious level $p_f[i]$, reflecting the probability that node i has been compromised. This value is aggregated through multiple rounds of PR processes. When $p_f[i] = 1$, we say node i is a compromised node.



(a) The PR algorithm in a static overlay

Index j	Root Path r_j of a Sampled Leaf Node						ACK s_j (6-bits)
1	…	13	9	C	6	1	110000 → Negative
2	…	13	9	8	7	2	111111 → Positive
3	…	13	10	…	…	3	111111 → Positive
4	…	14	11	…w_4	…	4	111110 → Negative
5	…	14	12	…	…	5	111111 → Positive
j	…	…	n_j	…	…	$n_{j,H}$	s_j

hop 0 → hop H

(b) T_s: sampling responses acquired by The GC

Fig. 4. An illustration of the PR algorithm: in this example, node 1-5 are sampled leaves, among which 1, 4 reply Negative ACKs. C denotes a compromised node, and 11 is a selfish node, which also refuses to forward messages.

1) Scheme Overview: Our identification scheme is based on the following observation: in an overlay where there are potential message dropping attacks, when a (noncompromised) leaf node reports a *Positive*, all the nodes on the path to the root are good nodes. In contrast, if a (noncompromised) leaf node reports a *Negative*, there is at least one compromised node on the path. By analyzing the feedbacks from multiple sampled nodes through multiple rounds, we are able to narrow down the suspicious node set and identify each compromised node (In Section VI we will evaluate the case when sampled compromised nodes report Positive or Negative reversely).

Fig.4(a) illustrates such an example. The GC queries nodes 1-5 and receives an ACK from each of them. For node 1 that fails to receive the previous broadcast messages, the GC searches within its current cluster a nearest sampled node whose receiving status is Positive (in this case node 2). If such a node does not exist, the GC tries in the next adjacent cluster until it finds such a node (node 3). We may say that, to the GC's current knowledge, these two nodes share a longest successful sub-path. In this way, the range of node 1's suspicious sub-path is narrowed down.

2) Scheme Description: The PR scheme is also referred as GSPR, as it adopts the group-based sampling technique. The scheme consists of three basic steps. First, the GC reconstructs a simple spanning tree based on the root path information it receives from the sampled leaves. Second, the GC resolves this spanning tree according to the receiving statuses of the sampled leaves and derives the suspicious sub-paths. Third, the GC applies a statistical aggregation to compute a suspicious level for each node on the suspicious sub-paths. The GC also updates the global list L_n considering nodes' suspicious history.

a) Path Resolving on a Spanning Tree: The GC may adopt a similar method as the overlay construction to add nodes into a spanning tree, as all root paths reported share a common ancestor – the root node. The spanning tree provides the GC with partial knowledge of the overlay topology.

The path resolving process on the spanning tree is: for each leaf node $j \in T_s$ that fails to receive the queried broadcast messages, the GC searches for its nearest sampled leaf l that reports a Positive. More formally,

$$\arg\max_{l \in T_s} LCA(j,l), \quad \text{s.t.} \quad s_l = Positive. \tag{6}$$

Since these two nodes share a longest successful sub-path in the spanning tree, we obtain j's suspicious sub-path w_j by eliminating good nodes from r_j. Thus, the hop count of j's suspicious sub-path is: $|\omega_j| = H - LCA(j,l)$.

b) Statistical Aggregation: After obtaining a suspicious sub-path ω_j, the GC tries to compute for each node $i \in \omega_j$ a suspicious level $f(i,\omega_j)$. Considering that a node situates closer to the root is more of an attacker's interest for compromise so that his message dropping attack is more effective, we use the node's hop-count to the leaf as a weight to compute its suspicious level: $f(i,\omega_j) = \frac{2(H-h_i)}{|\omega_j|\cdot(|\omega_j|-1)}$.

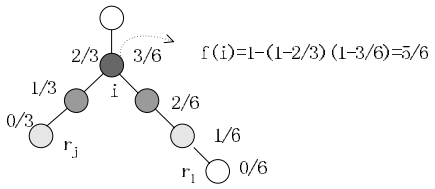


Fig. 5. An example when node n_i belongs to two suspicious paths.

A nodes is more likely to have been compromised when it appears in multiple suspicious sub-paths at the same time. We further aggregate the suspicious level of node i as follows:

$$f(i) = 1 - \prod_{j=1}^{\#sus} (1 - f(i,\omega_j)), \;\; f(i,\omega_j) = 0 \; if \; i \notin \omega_j, \tag{7}$$

where $\#sus$ denotes the total number of suspicious sub-paths in the spanning tree. Fig.5 illustrates a simple example of the statistical aggregation.

c) Multiple Rounds: In real applications, the GC may apply the GSPR algorithm over multiple rounds, each based on a sampled spanning tree. Suppose through round $1, 2, ...r-1$, node k has an aggregated suspicious level $p_f^{(r-1)}[i]$ in the global list L_n. During the rth round, the GC updates the global list entry as follows:

$$p_f^{(r)}[i] = 1 - (1 - p_f^{(r-1)}[i]) \cdot (1 - f(i)), \qquad p_f^{(0)}[i] = 0. \tag{8}$$

d) Security Analysis: Our GSPR algorithm is resilient to both consistent message dropping attacks and partial message dropping attacks where compromised nodes randomly drop messages with a probability $\alpha < 1$. For a sampled leaf child of an attack node, it will report a Positive with a probability of at most $(1-\alpha)^b$, where b is the bitmap size, reflecting the message interval between two consecutive samplings. This error rate is usually very small.

To evaluate the effectiveness of the algorithm, we define an *identification rate* $P_{I,iden}$, computed as an average suspicious level on actually compromised nodes after executing the algorithm for I rounds, i.e., $P_{I,iden} = \frac{1}{m} \cdot \sum_{n_i} p_f^{(I)}(n_i)$, where $\{n_1, ..., n_m\}$ are the compromised nodes in the overlay. We can also calculate the false positive rate for the algorithm, which is measured as the average suspicious level of legitimate nodes.

The GC's identification scheme has the same level of message complexity as its detection schemes, which is $I \cdot 2S_n \rightarrow O(1)$. The time complexity of the GSPR algorithm typically consists of three elements: spanning tree construction, path resolving and statistical aggregation. Thus, we may derive its running time as: $I \cdot 3H \cdot S_n \rightarrow O(logN)$. Since $S_n/N << 1$, our GSPR scheme is a bandwidth-efficient scheme.

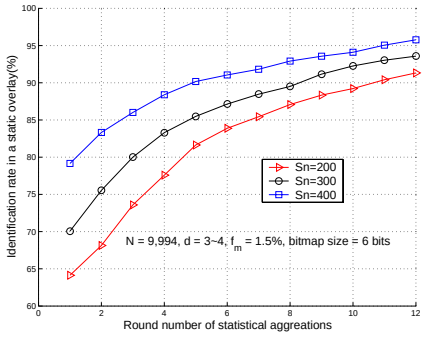


Fig. 6. Identification rate as the function of the number of rounds and the sample size.

Fig.6 shows our simulation result when the algorithm is executed for 12 rounds in a static overlay. We can observe the identification rate increases with the number of rounds and the sample size.

D. Further Attack and Security Analysis

Our previous discussions assumed leaf nodes are all benign nodes. However, malicious leaf nodes may exist in a set of sampled nodes. To avoid being detected, these nodes will report Positive ACKs once queried. Moreover, they may report false root path information to disrupt our detection/identification schemes.

1) Root Path Falsification Attacks: More specifically, we consider two types of root path falsification attacks: *truncation attack* and *modification attack* [7]. In the truncation attack a compromised node creates an attacker-favored route by shortening its actual root path. For instance, if it excludes the ids of the other compromised nodes on its root path, the GC may not be able to identify those compromised nodes. In the modification attack, a compromised node modifies its root path by altering intermediate node ids. This could mislead the GC to suspect legitimate nodes.

To prevent from accepting false root path, the GC needs to verify the authenticity of a root path consisting of multiple nodes and edges. In[14], the *Hash-based Strong Split Whisper* technique is proposed to provide path authenticity. However, it cannot be applied directly to our scenario because it involves proactive routing, whereas many of the routing protocols for overlay networks adopt reactive routing. Our approach is to first construct a spanning tree based on all the reported root paths, then randomly sample some nodes in the tree. Clearly, the more nodes to sample, the higher correctness guarantee, but the larger overhead. Therefore, a practical trade-off is required in this process. To tolerate false root path to some extent, we use the *principle of parsimony* (Ockham's Razor). This principle suggests adopting the simplest adequate explanation to observed data. In our case, we will accept a root path if it is consistent with the root paths from the majority of other sampled nodes.

2) Receiving Status Changing Attacks: Once being sampled, a compromised leaf may 1) reply a Positive ACK when it has not really received an attested broadcast message or 2) report a Negative ACK when it has actually received the message. These two cases have impacts on both attack detection and attacker identification, especially a high impact on the latter case because our path resolving algorithm relies heavily on the reported statuses. With the wrong information, the GC may not suspect a malicious node but suspect a legitimate node.

Fortunately, the chance of sampling a compromised leaf node is very small. First, compromised nodes only account for a fraction of the online nodes. Second, compromised nodes are less likely to be leaf nodes for an effective attack. Nevertheless, we cannot avoid sampling compromised nodes. Therefore, it is important to tolerate this receiving status changing attack to some degree as we may not know if these nodes actually have been compromised. In Section VI-C.4 we will study through simulations the detection rate under this false receiving status changing attack as well as the false positive rate caused by it.

V. Sampling Schemes for Dynamic Overlays

Now we study the problems of attack detection and attacker identification in a dynamic environment – a typical case in overlay applications. We first introduce some background knowledge on the dynamic behaviors of overlay networks, and then adapt our previous schemes to meet the new challenges due to network dynamics.

A. The System Model

In addition to the general system model introduced in Section III, in a dynamic overlay, a member node is in either of the two statuses: *presence* (online) or *absence* (offline), and it may switch between these two statuses as long as it keeps its membership in the network. We use the term "presence duration" and "absence duration" to denote a continuous time period a node stays online and offline, respectively. Previous study based on multiple sessions showed that presence duration in a multicast session follows either an exponential distribution or a Zipf distribution [1]. For simplicity, we use the former pattern in our dynamic case. We assume that presence durations of nodes follow an exponential distribution with mean R/μ, and absence durations have a mean of $1/\mu$. Thus, R is the ratio between two duration means.

Let N_{on} and N_{off} be the populations of online and offline members when the time interval begins, respectively, then $N_{on} + N_{off} = N$. According to the queuing theory, when the system is in its steady status, within a certain time interval, the number of members that come online is equal to the number of members go offline, i.e., $N_{on} \cdot \mu/R \cdot T = N_{off} \cdot \mu \cdot T$. Therefore, the average population of online nodes in steady status is $N_{on} = \frac{N \cdot R}{R+1}$.

The details of the protocols for constructing and maintaining dynamic overlay tree structures are presented in [18], [3], [8], [9]. Basically, a node coming online first contacts a well-known *rendezvous point* (RP) or a dedicated server to locate the root of the tree, which will assign this node to an appropriate child node based on such criteria as topological closeness or available bandwidth. Based on the criteria, this child node decides whether to accept this joining node as its own child node or introduce this node to one of its own child node. This process is repeated until this node is settled down in the tree hierarchy. Each joined node keeps its children list and root path (i.e., the node ids on its path to the root) up to date by exchanging REFRESH and PATH messages with its neighbors. In addition, we change these protocols so that a node sends its initial root path to the RP after it is settled down; this provides us the minimal and rough knowledge of the network topology. Note that for scalability, we do not require nodes to report updated root paths caused by network dynamics.

In our schemes we assume reliable communications in the overlay, and that packet losses are caused by system dynamics and message dropping attacks. The group controller(GC) knows from the RP which nodes have recently come online and their root paths. Note that according to the joining protocol, nodes that recently come online are most likely to be leaf nodes in the tree.

B. Attack Detection in Dynamic Overlays

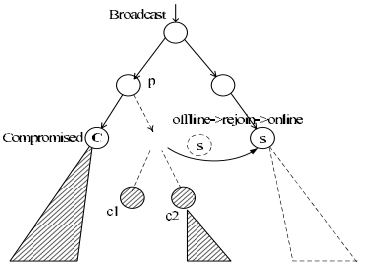


Fig. 7. Packet loss caused by node arrivals/departures in a dynamic overlay. The *repair time* starts as node s leaves and ends when node $c1, c2$ and their children reconnect to the overlay.

1) New Challenges from System Dynamics: System dynamics in the overlay raise several new challenges. First, because a node may change its status frequently, the GC does not have the precise knowledge of the tree topology. Moreover, a node to be sampled may have already gone offline as time elapses. Second, it is very hard to tell if packet losses are due to system dynamics or message dropping attacks. In [18], a node periodically exchanges control messages (REFRESH and PATH) with its neighbors and discovers a neighbor's absence after the time period expires. Fig.7 shows a case when node s has gone offline, its parent p and children c_1, c_2 are informed promptly. Although it usually only takes a short time (e.g., one period) for the temporarily isolated child nodes to recover the connectivity by joining to p directly, broadcast messages may get lost if they are delivered during this very moment. This appears to be more complex than in the static case where packet losses are solely attributed to malicious attacks. Moreover, as nodes join and leave the overlay very frequently, such packet losses may vary with time. Clearly, no perfect deterministic solution exists due to the limited topology information and node dynamics. As such, we will provide a statistical solution in Section V-B.3.

2) The Dynamic Group-based Sampling algorithm: In a dynamic overlay, the first problem is how to select nodes to sample. An intuitive solution is one in which the GC uniformly samples member nodes. The disadvantage is that it may involve a large number of onlineness tests because some sampled nodes are offline. Hence, we adopt an *exponentially weighted sampling* strategy in which the GC samples the most recently joined nodes with high probabilities. Our motivation is that the recently joined nodes are more likely to be leaf nodes. Thus, from the receiving statuses of leaf nodes the GC can know more intermediate nodes. In addition, leaf nodes are less likely to be compromised.

For member nodes with presence mean $1/\theta$, the probability $p_j(t)$ that a member node j remains online at time *t* after its last coming online is computed as $p_j(t) = e^{-\theta \cdot t}$. This simply means that nodes more recently seen nodes are more likely to remain online than those seen earlier. Sampling such nodes reduces the number of onlineness tests.

a) Algorithmic Details: Similar as in IV-B.2, the GC considers its bandwidth limit and decides a sampling size S_n, it then acquires $(R+1) * S_n/R$ most recently joined nodes from the RP and group these nodes based on their initial root paths according to the same LCA level $c = \lfloor log_d S_n \rfloor$. As a result, the GP gets k clusters of recent nodes and it queries $\lceil S_n/k \rceil$ such nodes from each cluster.

However, due to presence dynamics of the nodes, some of them might have gone offline as time elapses. Thus, the server may not be able to successfully sample all $\lceil S_n/k \rceil$ recent nodes from each cluster. We propose two solutions to address this problem: either the GC ignores these unavailable nodes (hence the total number of samplings may be below S_n) or the GC makes a parent substitution, that is, if the node to be sampled is currently not available, its parent node (learned from the initial root path) will be queried instead. We adopt the latter strategy since it enables the GC to maintain a relatively stable sampling size; moreover, the parent node for replacement is also likely to be a leaf node.

In summary, this algorithm utilizes the rough knowledge of the network topology from the RP and groups most probable leaf nodes to achieve sampling breadth and density for a dynamic overlay.

3) Statistical Detection of Attacks: Here we propose an attack detection scheme which adopts the dynamic sampling

algorithm and a statistical technique to detect potential message dropping attacks in the overlay. We define *living rate* $r_a = \frac{N_{on}}{N} = \frac{R}{R+1}$, as the average fraction of online nodes. Also, we define *receiving rate* $r_v = \frac{\#Positive}{S_n}$, as the fraction of sampled online nodes that reply Positive ACKs. Thus, we may use $r_l = 1 - r_v$ to estimate the *packet-loss rate* of the dynamic overlay.

Our dynamic detection scheme consists of two stages. The first stage starts after the overlay has been constructed and the network enters a stable status. At this point, no compromised nodes or few of them have started to launch attacks in the overlay, as it would take some time for an attacker to take over many nodes. Therefore, packet losses are mainly attributed to overlay dynamics. In each round, the GC samples S_n online nodes to learn a packet-loss rate $r_l = 1 - \frac{\#Positive}{S_n}$. This process is repeated for $I (I > 30)$ rounds and eventually the GC computes an average loss rate $\bar{r_l}$. Alternatively, the GC may refer to dynamic packet loss rates measured from live streaming overlays, such as the CoopNet in [13], or resilient overlay networks (RONs) [2].

To keep the packet loss rate r_l more steady over time in a dynamic overlay, so that we may apply the statistical detection scheme, we adopt an active approach in which each node monitors the packet loss rate it is experiencing (learned through the GC's query bitmaps). When its packet loss reaches an unacceptable level, the node contacts the server and executes a fresh relocation in the overlay. In [10] and [13], the similar approach is used and it has been shown that the packet loss rate as a function of time can be smoothed by active relocations in the overlay.

In the second stage, when message dropping attacks may exist in the overlay, both the attacks and system dynamics contribute to packet losses. The GC samples the recently joined nodes and measures the current packet-loss rate $r_l^{'} = 1 - \frac{\#Positive^{'}}{S_n}$. This rate is also averaged over I rounds of measurements and the GC obtains a sampled mean $\bar{r_l}^{'} = \frac{1}{I}\sum_{i=1}^{r} r_l^{'}(i)$.

Now the GC is able to apply a statistical technique which helps determine if the current packet loss rate significantly deviates from the one learned from the first stage. If the deviation is significant, it concludes that there are message dropping attacks in the dynamic overlay. More specifically, the GC performs the following hypothesis test to make the decision:

$$H_0 : \ \bar{r_l}^{'} = \bar{r_l} \quad vs. \quad H_1 : \ \bar{r_l}^{'} \neq \bar{r_l}$$

Reject H_0 with significance level α if:

$$|\bar{r_l}^{'} - \bar{r_l}| > \frac{s}{\sqrt{r}} t_{\alpha/2}(R-1),$$

where s is the standard deviation of the packet-loss rate $r_l^{'}$.

a) Security Analysis: Clearly, for dynamic networks we cannot provide as precise guarantee on the detection rate as for static networks due to the lack of topology information and node dynamics. We have to rely on statistical methods to detect such attacks. This is because of the necessary attacking scale for the attack to cause nontrivial and increasing damages. That is, there should be some number of compromised nodes launching this attack frequently. Otherwise, if only few nodes occasionally launch this attack, we may not care about this attack because normal users may go offline as well, causing the dropping of broadcast messages occasionally. We will show in the next section the detection capability of our scheme with respect to the number of compromised nodes and the network dynamics.

C. Attacker Identification in Dynamic Overlays

We adapt the previous GSPR algorithm for attacker identification in dynamic overlays. Similarly, the GC will first sample the nodes and receive some ACKs, then reconstruct the spanning tree and conduct path resolving, finally it uses statistical aggregations to identify those compromised nodes. During these processes, we consider the influences from system dynamics.

Selection of sampling intervals is an important issue in a dynamic overlay. If the GC uses a large interval (i.e., it samples the leaf nodes after broadcasting a large sequence of messages), system dynamics may influence the sampling responses; if the GC samples after each broadcast, high message overhead will be incurred. We propose the GC determines its sampling interval according to R, the dynamic ratio of the overlay. When the overlay becomes more dynamic, it reduces its sampling interval to ensure the correctness of sampling responses.

In a dynamic overlay, nodes may have different life times. As a node stays longer in the overlay, its suspicious level will accumulate due to the nondecreasing function in Eqn.8. This may cause false positives. We introduce an aging factor $0 < \beta \leq 1$ to address this problem:

$$p_f^{(r)}[i] = \beta^{(r)} \cdot (1 - (1 - p_f^{(r-1)}[i]) \cdot (1 - f(i))), \quad (9)$$

where $\beta^{(0)} = 1$ and $\beta^{(r)} = 0.5\beta^{(r-1)}$. This equation guarantees that a legitimate node which is occasionally suspected will not be eventually identified as an illegal one. We show the effectiveness of our dynamic attacker identification scheme through simulations in Section VI-C.4.

VI. Performance Evaluation

This section reports through simulations the effectiveness of our attack detection and attacker identification schemes in dynamic overlay networks.

A. Performance Metrics

We use the following performance metrics in the study:

- *Detection Rate*: The percentage of tests that correctly reports the message dropping attacks in a dynamic overlay. It reflects the power of our detection scheme.
- *Identification Rate*: The GC's ability to identify the compromised nodes in the overlay, measured by the average suspicious level of compromised node.

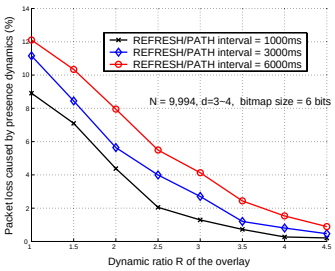


Fig. 8. Normal packet loss rate as a function of node dynamics and control message parameters.

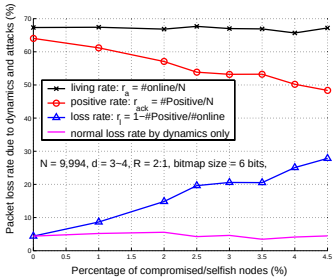


Fig. 9. The overall packet loss rate as a function of node dynamics and message dropping attacks.

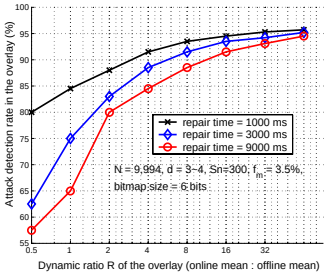


Fig. 10. The detection rate as a function of the sampling rate and the fraction of malicious nodes.

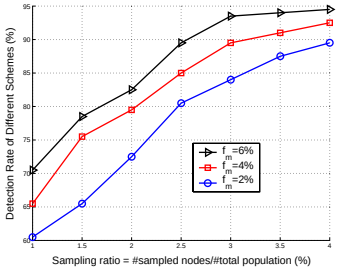


Fig. 11. Effectiveness of the attack detection scheme in dynamic overlays

- *Sampling Rate*: The percentage of sampled nodes over the entire population, i.e., $r_s = \frac{\#sampled}{\#total}$. It reflects the scalability of a sampling algorithm.

B. Simulation Settings

We first generate a random graph of 9,994 nodes and then construct a tree out of the graph based on the joining algorithm that is also used in [18], [3], [9]. More specifically, every joining member searches from the root downwards along the tree for the (possible) nearest node as its parent, thus, geometrically adjacent nodes become neighbors in the tree. The link delay between any two nodes is randomly selected from a uniform distribution between 10 and 200 ms, and the out-degree d of a node is chosen between 3 and 5. We set the presence dynamics of members according to an exponential distribution, and adjust the dynamic ratio R (the online duration mean over the offline duration mean) to simulate various dynamic environments. Our simulation programs were written using the csim simulation library [5]. We use the method of independent replications for our simulations and all our results have $\alpha = 95\%$ confidence intervals that are within 5% of the reported values.

C. Simulation Results

1) Impact of Network Dynamics on Packet Loss Rates: In the first stage of our statistical detection scheme, we evaluate the impact of node joins/departures on normal packet losses in various dynamic environments. More specifically, we apply the dynamic sampling algorithm in different overlays (determined by the ratio R) and adjust control message parameters (REFRESH/PATH message interval) to see their influences on packet losses. Fig.8 shows the average packet loss rate r_l as a function of dynamic ratio R and the message interval. The simulation result indicates that when an overlay becomes more static, it will have less packet losses. Also, a longer control message interval usually results in more packet losses to the overlay. These results are consistent with our discussion in V-B.1. When nodes join and leave more frequently, the overlay will have more isolated nodes who are temporarily unaware of their parents' departures until the next refresh time arrives.

From the network point of view, the processing of node joins/departures should be as prompt as possible, so that a high receiving rate and a minimal interruption can be guaranteed. Our schemes lower down the control message interval to reduce the repair time, and hence the normal packet losses. Also, we adopt the technique described in V-B.3 to stabilize the change of packet loss rate over time. These two steps significantly improve the detection rate of our statistical method.

2) Impact of Attacks on Packet Loss Rates: In the second stage of our statistical detection scheme, both the attacks and system dynamics contribute to packet losses in the overlay. We examine the feasibility of using the statistical method to detect different severity levels of attacks in a dynamic environment. Fig.9 shows the simulation result of impacts from both system dynamics and message dropping attacks to the packet loss rate, when applying the dynamic sampling algorithm in an overlay whose dynamic ratio $R = 2:1$. The relationship between the changes of loss rate and severity levels of attacks is clearly shown in the figure. When there is no attack nodes, or only few nodes occasionally launch attacks in the overlay, presence dynamics alone contribute about $r_l \approx 4.42\%$ message losses to the network. However, as the number of compromised nodes increases, their portion of packet losses in the overlay becomes more dominant, i.e., the overall loss rate deviates much from the normal ratio (stable over time). This result is consistent with our discussion in V-B.3.

3) Results on Attack Detection: Now we may adopt the statistical detection scheme to check the existence of message dropping attacks in a dynamic overlay. Fig.10 shows the effectiveness of our detection scheme in various dynamic environments, with different dynamic ratio Rs and different repair times. In most cases, the GC achieves reasonably high detection rates. As the overlay becomes more static, i.e, ratio R increases, the detection rate gets improved accordingly, because packet losses are largely attributed to attacks at this point. The figure also shows that, when the normal packet losses decrease due to a reduced repair time in a dynamic environment, the statistical scheme achieves a higher detection rate. These result are consistent with our previous discussions.

Fig.11 shows the simulation result when we use different sampling rate r_s for our statistical scheme to detect different severity levels of attacks in a dynamic overlay ($R = 2:1$). We can see from the figure that, in most cases, the sampling-based scheme detects message dropping attacks very effectively. Similar as in the static case, when the GC chooses a larger sampling size or there are more compromised nodes in the

overlay, a higher detection rate can be achieved.

4) Results on Attacker Identification: Fig.12 shows the simulation result when the GC applies the GSPR algorithm in various dynamic environments. In our test, we set the range of the dynamic ratio R from $2:1$ to $8:1$ and uniformly choose 3.5% of the total population as attack nodes. The figure clearly indicates that, given a reasonable sampling size (limited by the GC's resource), the GSPR algorithm is able to identify most of the compromised nodes even in a highly dynamic overlay ($R = 2:1$), after a limited number (less than 10) of rounds of statistical aggregations. As the overlay becomes less dynamic (R increases), the GC achieves a higher identification rate. The figure also indicates that a lager sampling size results in a more accurate identification, because more malicious nodes are likely to be included in the GC's sampling paths and be identified as suspicious nodes.

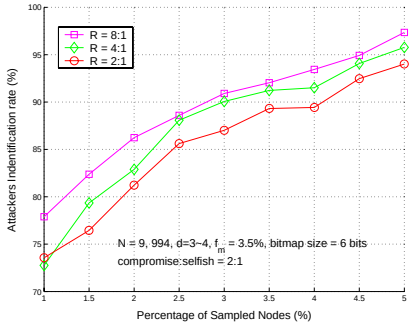


Fig. 12. Identification rate as a function of the sampling ratio and the dynamic ratio

In addition, we also measure the false positive rate of our identification scheme based on the same set of parameters as in Fig.12. We define the false positive rate as the percentage of legitimate nodes which are mistakenly identified as compromised nodes. Our test result indicates that under different levels of system dynamics, the GSPR scheme keeps its false positive rate below a reasonable value of 2%.

VII. Conclusions and Future Work

We proposed a random-sampling–based scheme to detect the message dropping attacks, and a path-resolving–based scheme to identify the compromised/selfish nodes. Our schemes work for dynamic overlay networks as well, because they do not assume the global knowledge of the overlay hierarchy as previous work did. Analysis and simulation results show that our light-weight schemes have high detection/identification rates but low false positive rates.

We note that attack detection and attacker identification in dynamic networks are very challenging issues. To address the problem, we have made several assumptions, for example, on the distribution of node membership durations and an estimated ratio between compromised nodes and selfish nodes. We will consider relaxing these assumptions in our future research. Also, we will investigate techniques to counter richer attack models (e.g. attacker collaborations) and allow other trusted nodes than the GC to help in the process of detection/identification.

Acknowledgments

This work was supported in part by Army Research Office (W911NF-05-1-0270) and the National Science Foundation (CNS-0524156).

References

[1] K. Almeroth and M. Ammar. Multicast group behavior in the internet's multicast bckbone (mbone). IEEE Communications, June 1997.

[2] D. Anderson, H. Balakrishnan, F. Kaashoek, and R. Moris. Resillient overlay networks. In *Proc. 18th ACM Symposium on Operating Systems Principles (SOSP)*, 2001.

[3] S. Banerjee, B. Bhattacharjee, and C. Kommareddy. Scalable application layer multicast. In *Proc. of ACM Sigcomm*, 2002.

[4] S. Banerjee, S. Lee, B. Bhattacharjee, and A. Srinivasan. Resilient multicast using overlays. *SIGMETRICS Perform. Eval. Rev.*, 31(1):102–113, June 2003.

[5] CSIM. Website at http://www.mesquite.com.

[6] V. Drabkin, D. Wallach, and P. Druschel. Incentives-compatible peer-to-peer multicast. In *International Conference on Dependable Systems and Networks (DSN'05)*, 2005.

[7] Y.-C. Hu, A. Perrig, and M. Sirbu. Spv: secure path vector routing for securing bgp. In *SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 179–192, 2004.

[8] Y. hua Chu, S. G. Rao, and H. Zhang. A case for end system multicast (keynote address). In *SIGMETRICS '00: Proceedings of the 2000 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 1–12, New York, NY, USA, 2000.

[9] J. Jannotti, D. K. Gifford, K. L. Johnson, M. F. Kaashoek, and J. W. O'Toole, Jr. Overcast: Reliable multicasting with an overlay network. In *Proc. of 4th USENIX OSDI Symposium*, pages 197–212, San Diego, CA, USA, October 2000.

[10] S. Jun, M. Ahamad, and J. Xu. Robust information dissemination in uncooperative environments. In *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS'05)*, June 2005.

[11] L. Mathy, N. Blundell, V. Roca, and A. El-Sayed. Impact of simple cheating in application-level multicast. In *Proc. of IEEE INFOCOM*, volume 2, pages 1318–1328, Hong Kong, China, March 2004.

[12] T. Ngan, S. Wallach, and P. Druschel. Incentives-compatible peer-to-peer multicast. In *2nd Workshop on Economics of Peer-to-Peer Systems*, Cambridge, Massachusetts, USA, June 2004.

[13] V. Padmanabhan, H. Wang, P. Chou, and K. Sripanidkulchai. Distributing streaming media content using cooperative networking. In *ACM NOSSDAV*, May 2002.

[14] L. Sabramanian, V. Roth, I. Stoica, S. Sehnker, and R. Katz. Listen and whisper: Security mechanisms for bgps. In *Proceedings of the First Symposium on Networked Systems Design and Implementation (NSDI 2004)*, 2004.

[15] D. Song, D. Zuckerman, and J. Tygar. Expander graphs for digital stream authentication and robust overlay networks. In *In Proc. of IEEE Symp. on Security and Privacy*, pages 258– 270, Berkeley, CA, USA, 2002.

[16] R. Wright, P. Lincoln, and J. Miller. Efficient fault-tolerant certificate revocation. In *Proc. of ACM CCS 2000*, 2000.

[17] H. Yang, H. Luo, Y. Yang, S. Lu, and L. Zhang. Hours: Achieving dos resilience in an open service hierarchy. In *DSN '04: Proceedings of the 2004 International Conference on Dependable Systems and Networks (DSN'04)*, page 83, Washington, DC, USA, 2004.

[18] B. Zhang, S. Jamin, and L. Zhang. Host multicast: A framework for delivering multicast to end users. In *Proceedings of IEEE Infocom*, pages 1366–1375, June 2002.

[19] S. Zhu, C. Yao, D. Liu, S. Setia, and S. Jajodia. Efficient security mechanisms for overlay multicast-based content distribution. In *ACNS*, pages 40–55, New York, USA, June 2005.