Available online at www.sciencedirect.com

ScienceDirect

Computer Communications 30 (2007) 793–806

computer communications

www.elsevier.com/locate/comcom

Efficient security mechanisms for overlay multicast based content delivery [☆]

Sencun Zhu [a,*], Chao Yao [b], Donggang Liu [c], Sanjeev Setia [b], Sushil Jajodia [b]

[a] *Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA 16802, USA*
[b] *Center for Secure information Systems, George Mason University, Virginia 22030, USA*
[c] *Department of Computer Science and Engineering, University of Texas at Arlington, Texas 76019, USA*

Received 12 May 2006; received in revised form 29 September 2006; accepted 3 October 2006
Available online 3 November 2006

Abstract

This paper studies the security issues that arise in an overlay multicast architecture where service providers distribute content such as web pages, static and streaming multimedia data, realtime stock quotes, or security updates to a large number of users. In particular, two major security problems of overlay multicast, *network access control* and *group key management*, are addressed. We first present a bandwidth-efficient scheme, called CRBR, that seamlessly integrates network access control and group key management. Performance analysis and simulation results show that our scheme incurs much smaller communication overhead than two other well-known schemes when they are directly applied in overlay multicast. Next we propose a DoS-resilient key distribution scheme, called k-RIP, that delivers updated keys to a large fraction of nodes with high probability even if an attacker can *selectively* compromise nodes in the multicast data delivery hierarchy and command these compromised nodes to drop keying packets. The proposed schemes do not rely on knowledge of overlay topology and can scale up to very large overlay networks.

Keywords: Overlay multicast; Network access control; Group key management; DoS-resilient

1. Introduction

IP Multicast [10] was proposed as an efficient way of delivering data from one or multiple data sources to a group of interested users simultaneously. The main idea of IP multicast is to construct a delivery tree that connects all group members with the direct support of Internet routers that can understand the Internet Group Management Protocol (IGMPv3) [15]. During the last decade, researches on multicast have been very extensive. The topics of interest included multicast routing, reliable multicast, secure multicast, and others. In practice, IP multicast has evolved from being a pure research topic to being experimentally deployed in the MBONE [12] to being supported by major router manufacturers such as Cisco. The class-D IP addresses, which range from 224.0.0.0 to 239.255.255.255, are specifically allocated for IP multicast. Thus, the Internet is becoming increasingly multicast capable. However, IP multicast has seen very slow commercial deployment, despite its efficiency and potential applications. Today many ISPs and carriers are still reluctant to provide multicast services due to both technical reasons and marketing reasons [11].

Recently, with the prosperity of peer-to-peer (P2P) networking, researchers have proposed alternative solutions to bypass the limitations in IP multicast. The solution is overlay multicast, also called end system multicast or application-level multicast [8], which shifts multicast support from core routers to end systems. In end multicast, group members communicate via an overlay structure built on top of unicast paths between various pairs of hosts;

[☆] A preliminary version of this paper appeared in Proceeding of Applied Cryptography and Network Security, 2005.
[*] Corresponding author. Tel.: +1 814 865 0995; fax: +1 814 865 3176. *E-mail address:* szhu@cse.psu.edu (S. Zhu).

therefore, the underlying physical topology is completely hidden from hosts and no direct router support is needed. Although end multicast usually incurs much higher communication costs compared to IP multicast when distributing the same amount of data, its independence of network routers makes it a very promising technique for multicast communications.

Several studies [8,7,16,29] have investigated research problems in overlay multicast such as algorithms for tree or mesh construction, routing, reliability, and resource allocation. However, security issues in overlay multicast have received relatively little attention so far. Previous work on overlay network security either investigates the impact of selfish cheating nodes on the performance of overlay multicast trees [19], or investigates schemes that improve the fault-tolerance or denial-of-service (DoS) resilience of overlay networks by introducing path redundancy [3,24,26,27].

1.1. Contributions

We consider the security issues that arise in an overlay multicast architecture where service providers distribute content such as web pages, static and streaming multimedia data, realtime stock quotes, or security updates (e.g., new virus signatures, certificate revocation lists) [18].

We concentrate on two major security problems of overlay multicast: *network access control* and *group key management*. In IP multicast, network access control and group key management were considered as two *independent* issues and they were studied *separately*, one in the network layer [14] and the other in the application layer [5,20,22,25]. In this paper, we propose a bandwidth-efficient scheme called CRBR that seamlessly integrates network access control with group key management. CRBR exploits the special property of overlay multicast that a node is both a group member and a router. We show through analysis and simulation that CRBR greatly outperforms other two representative group rekeying schemes: LKH [25] and SDR [20] when they are applied in overlay multicast. Moreover, using a queueing model, we show the impact of node presence dynamics (i.e., coming online/going offline) on the performance of group rekeying protocols.

We also propose a simple but effective DoS-resilient key distribution scheme, called k-RIP (stands for k-Random Injection Points), that delivers updated keys to a large fraction of nodes via an overlay network. Specifically, in addition to propagating one copy of updated keys using a multicast tree rooted at the source node, our scheme injects k additional copies of updated keys into the multicast tree through k randomly selected nodes in the network. These selected nodes propagate the message to both their child nodes (if any) and parent nodes, thereby spreading the message over the multicast tree. Our simulation and analysis results show that k-RIP can greatly increase the probability that nodes receive messages even if an attacker can *selectively* compromise nodes in the multicast tree, compared to a scheme in which only one copy of the message is injected via the root node. Unlike previously proposed schemes [3,24,26,27], k-RIP does not rely on the knowledge of the overlay topology. Thus, it is scalable to very large overlay networks.

1.2. Organization

The remainder of this paper is organized as follows. Section 2 discusses some related work on group key management, network attacks and countermeasures. Section 3 describes the system model and our design goal. In Section 4, we present our scheme CRBR for providing both network access control and group key management, followed by its security and performance analysis. In Section 5 we describe our k-RIP key distribution scheme. Finally, Section 6 concludes this paper.

2. Related work

We introduce the related work in four categories: *group key management*, *network attacks and countermeasures*, and *resilient overlay multicast*.

2.1. Group key management

Group key management has been extensively studied in the context of secure multicast in IP multicast. The previous group rekeying schemes can be categorized into stateful and stateless protocols. The stateful class of protocols includes several protocols based upon the use of logical key trees, e.g., LKH [25], OFT [5], ELK [21]. In these protocols, the key server uses key encryption keys that were transmitted to members during previous rekeying operations to encrypt the keys that are transmitted in the current rekeying operation. Thus, a member must have received all the key encryption keys of interest in all the previous rekey operations to decipher the current group key. Adding redundancy in key distribution [23,28] does not fully address the issue in the case of burst packet loss or nodes going offline frequently. *Stateless* group rekeying protocols [17,20] form the second class of rekey protocols. In these protocols, a legitimate user only needs to receive the keys of interest in the current rekey operation to decode the current group key. The stateless feature makes these protocol very attractive for applications in which members go offline very frequently. However, these protocols usually have much higher communication overhead than the stateful protocols. Our scheme also provides the stateless property, but it incurs significantly smaller communication overhead than the other schemes in the context of overlay multicast. Moreover, it also provides network access control.

2.2. Network attacks and countermeasures

Mathy et al. [19] studied the impact of selfish nodes cheating about their distance measurements in application-level multicast overlay tree. Badishi et al. [4] proposed a gossip-based multicast protocol called Drum, which combines multiple techniques such as push, pull, random port selections, and resource bounds, for mitigating DoS attacks in secure gossip-based multicast. Wright et al. [26] 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. [24] improved the scalability of the above scheme by presenting expander graphs for constructing robust overlay networks that have constant degree. Yang et al. [27] proposed to augment tree-like hierarchy with hierarchical overlay networks, which is actually also a type of graphs, to achieve DoS resilience.

All these schemes provide stronger fault-tolerance or DoS resilience at the cost of higher (re)construction complexity to maintain their security property, especially when nodes join or leave the tree frequently. Moreover, these schemes are subject to selective attacks in which an attacker can prevent a large number of nodes from receiving messages by compromising (or becoming) the nodes close to the root. Our random injection points scheme directly works with the existing overlay multicast schemes without changing trees into graphs, and it is especially suitable for distributing small-size but critical messages. Our scheme is robust to selective attacks; therefore, we believe that the combination of our scheme with the other DoS-resilient schemes will make a distribution system more robust to DoS attacks.

2.3. Resilient overlay multicast

Banerjee et al. [3] introduced a probabilistic forwarding scheme for overlay multicast. In their scheme, every node forwards received packets to a randomly selected set of nodes, assuming that every node has global knowledge of overlay topology or it can discover other nodes on the fly. Our k-RIP scheme also uses randomness, but the randomness is used by the key server to inject packets into the overlay network, not used by the regular nodes to forward packets. The main reason we do not employ their scheme directly is because of scalability consideration. For large-scale and dynamic overlay networks, the overhead for discovering other nodes on the fly or maintaining global topological knowledge would be very large. In k-RIP neither the key server nor the nodes need knowledge on overlay topology. This allows k-RIP to scale up to arbitrarily large overlay networks.

3. System model and design goal

This section describes our system model and design goal. For ease of presentation, we use the terms "join" and "leave" to denote the actions of a subscriber coming online and going offline, respectively, whereas use "add" and "revoke" to denote the actions of becoming a member and cancelling the membership status of a subscriber, respectively.

3.1. System model

There are potentially a large number of application scenarios of overlay multicast, which are characterized by different parameters. For example, *group size* could vary from tens to millions, *number of data sources* from one to many, *membership dynamics* from static to frequent subscribing/unsubscribing, *node presence* from always online to frequently going offline, and *types of traffic* from realtime information to delay-insensitive bulk data. 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. Security solutions for this scenario can be applied to many other scenarios as well.

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) simultaneously. For simplicity, we assume that online nodes are self-organized into an overlay multicast delivery tree rooted at the distribution server of a service provider, although our security schemes work for various distribution infrastructures, such as trees, meshes, or other types of graphs. The algorithms for constructing and maintaining overlay multicast trees [2,8,16] are out of the scope of our work.

The population of the system could be up to hundreds of thousands or even millions of nodes. We assume that a node may join or leave a multicast group very frequently and at any time. For example, a user may subscribe to multiple service providers for different programs. She may switch between multiple channels to find an interesting program to join. The user may also leave a channel immediately after she has received the data of interest to her.

In this model, the service provider has three types of servers: *a key server*, *a data server*, and *a distribution server*. These servers play different roles and can communicate securely. A key server (or many key servers for scalability) provides subscription services to users. Before an end host is able to join the group for the first time, it needs to subscribe to the key server (e.g., through a website). After successful subscription based on certain policies or rules (e.g., agreeing to pay service fee), a host is provided with a service credential that allows it to join the multicast delivery tree later. A host must also contact the key server to cancel its membership later when needed. The key server also manages the update of data encryption keys (DEKs). When it changes its DEK, it sends a new DEK to the data server, which encrypts the future messages with the new key. The key server also sends to the distribution server its (updated) network access control policy or access control list indicating which nodes are currently authorized to join the group. The data server is mainly engaged in processing the data to be distributed, e.g., computing encryptions and digital signatures. It transmits the prepared data to the distribution server for distribution.

3.2. Design goal

Security requirements of overlay multicast are similar to those of other networks. Some of the general security properties are *authentication*, *confidentiality*, *network access control*, *availability*, *anonymity*, and *fairness*.

In this paper, however, we focus on two of these security issues in the context of overlay multicast. First, we want to provide data confidentiality and network access control. Data confidentiality ensures that only authorized nodes can understand the multicast data. It must be provided because an unauthorized user may attempt to receive multicast data by eavesdropping on the communication links of authorized nodes or even of Internet routers. Network access control is also critical because it ensures that only authorized nodes can join the overlay multicast tree; otherwise, the resources of a legitimate node are consumed for forwarding data to unauthorized nodes. Second, we want to provide a DoS-resilient key distribution scheme that delivers keys to existing member nodes with high probability even if some selectively compromised nodes *drop* the keys they are supposed to forward. Note that we do not address another type of DoS attack in which an attacker floods the network with junk data. Instead, we assume the existence of an appropriate multicast source authentication scheme by which member nodes can immediately verify the data from the data server and the key server and drop false data.

4. A certificate revocation based group rekeying scheme (CRBR)

An important and challenging issue for providing multicast data confidentiality is group key management. To enforce both backward confidentiality (i.e., a new user should not be able to decipher the data distributed before its subscription) and forward confidentiality (i.e., a revoked user should not be able to decipher the future data), it is required to distribute a new group data encryption key (DEK) to all authorized members in a *secure*, *reliable*, and *timely* fashion when group membership changes. This is referred to as *group rekeying*.

Unicast-based group rekeying, in which the key server sends a new DEK to every individual node, has the communication complexity of $O(N)$ keys. Recently proposed group rekeying schemes [5,20,25] use logical key trees to reduce the complexity of a group rekeying operation from $O(N)$ to $O(\log N)$. Further, it has been proposed that groups be rekeyed periodically instead of on the basis of every membership change [22,28]. Periodic or batched rekeying can reduce both the processing and communication overhead at the key server, and improve the scalability and performance of key management protocols. Note that in all these schemes, the key server includes keys for *all* the member nodes when distributing its rekeying message, and every member receives the entire message although it is only interested in a small fraction of the content.

Network access control, which is another critical security service, was studied *independently* with group key management in IP multicast. It is usually enforced by Internet edge routers [14]. Specifically, each router maintains inclusion or exclusion access control lists (ACLs) for all supported multicast groups, and every member presents its authorization certificate or token to its edge router to join a group. Network access control is hard to implement in IP multicast because routers are required to authenticate packets, to establish trust relationship with individual group controllers, and to keep their ACLs up-to-date.

4.1. Scheme overview

We exploit the property of an overlay network that nodes are both hosts and routers when designing group rekeying and network access control schemes. In IP Multicast, all group members are end hosts, and they have no responsibility for forwarding keying materials to other group members. In contrast, for group communication in an overlay network, the nodes in the delivery tree also act as routers. As such, the key server only has to deliver a new DEK securely to a small number of nodes, which are its immediate children, and these children then forward the new DEK securely to their own child nodes. In this way, a group key is propagated to all the online member nodes in a hop-by-hop fashion. The amortized transmission cost per node is one key, independent of the group size.

For the above scheme to work, a basic requirement is the existence of a secure channel between every pair of neighboring nodes. We employ conventional public key techniques for establishing pairwise keys between two nodes. The use of public key techniques can additionally provide network access control because public key techniques such as digital signatures support strong source authentication. In overlay multicast, because nodes are both routers and hosts, network access control will be achieved as long as every node authenticates every other node that contacts it for joining the network.

In our system model, it is very natural that data access control (through encryption) and network access control (through authentication) be integrated. A node should have both privileges if it is authorized, and it should not have either of them if it is revoked. This motivates us to update group keys and invalidate the public keys (or certificates) of revoked nodes simultaneously. Moreover, since periodic group rekeying is much more scalable than individual rekeying [22] and certificate revocation information is also distributed periodically [6], group rekeying and the distribution of certificate revocation information can be performed with the same time interval. An appropriate rekeying interval is application dependent and requires a trade-off between security and performance. The selection of rekeying interval is out of the scope of this paper.

4.2. Scheme specifications

This subsection describes the details in CRBR.

- *Node registration:* The key server issues every member a public-key certificate when a member subscribes to the group.
- *Security update generation:* The key server generates a new certificate revocation list (CRL) and a new DEK K_g for every group rekeying. It further computes a digital signature over the CRL, K_g, and a timestamp. Denote SU as a *security update* that includes the CRL, the timestamp, and the above digital signature (note that K_g is not included in SU). The key server sends SU and K_g to the distribution server securely.
- *Security update distribution:* For the ease of presentation, here we use a traditional, non-DOS-resilient scheme for the distribution of security updates. This is referred to as base scheme (a DOS-resilient scheme will be presented in Section 5). In the base scheme, the distribution server forwards SU and K_g to each of its child nodes. SU is sent in cleartext, whereas K_g is encrypted with a pairwise key shared between two nodes in every link. A node establishes a pairwise key with another node and then propagates K_g to it only if the CRL indicates that node is still a legitimate member. Also note that every node can verify the authenticity of the received group key K_g by verifying the signature. After successfully verifying the message, these child nodes forward the message to their own child nodes. Recursively, the security update and K_g are propagated to all online nodes in a hop-by-hop fashion.
- *Local recovery:* A node that has missed one or several security update information because it was offline can authenticate itself to any one of the online nodes to obtain the up-to-date security update and the group key when it joins the network, because that node knows if the joining node is legitimate or not based on the CRL it possesses.

4.2.1. Certificate management

The key server issues every node a unique public-key certificate if the node is authorized. The node is also given the public key of the key server, which allows the node to verify the certificates (hence their public keys) presented by other nodes.

The key issue in using digital certificate is certificate management. In general, there are mainly two challenges. The first challenge is for a node to verify a received certificate in an efficient and timely fashion in the presence of complex CA hierarchy. In our applications, fortunately, there is no CA hierarchy since every user receives a certificate from the same key server. The second challenge is certificate revocation. On one hand, it is more efficient and economic for a CA to issue certificates with long validity periods, because there is considerable computational and communication overhead (e.g., computing a digital signature) involved in issuing a certificate. On the other hand, it is more likely that a certificate with a long validity period needs to be revoked explicitly before it is expired, because in some applications users may unsubscribe from the program at any time. To address this problem, a conventional approach is for the key server to issue a certificate revocation list (CRL) that contains the information of all revoked certificates. The issue is: how can the key server make the revocation information available to other nodes in an efficient and timely manner. There are mainly two techniques: pull or push. In a pull-based scheme, a node contacts a dedicated directory to verify a received certificate from another node. Since every node needs to authenticate itself to multiple nodes, including its parent, children, and multiple other nodes it has to contact during its join process, the pull-based scheme makes the directory a performance bottleneck.

We adopt a push-based approach in which the key server distributes its CRLs periodically. To minimize the communication overhead, we employ the following techniques.

- The key server assigns a unique integer to every node as the identifier of the certificate of the node. The integer starts from ‘0’ and is incremented by one for a new node. Note that the key server does not assign the ids of revoked nodes to any new nodes. Also, the certificate of a user does not contain any personal information about the user. Instead, the key server has a database that records the personal information of a user and its certificate.
- The CRL is a bit string of size Z, where Z is the number of nodes that have so far subscribed to the system. Every certificate is mapped to a bit in the bit string and the id of the certificate is the index of the bit in the bit string. A bit value of ‘0’ indicates that a corresponding certificate is invalid and ‘1’ indicates valid. When a node is revoked from the system, its corresponding bit in the bit string is set ‘0’.

4.2.2. Node join/leave

When a node u joins the multicast delivery tree after its subscription process, it follows the existing overlay multicast routing protocol, except that it authenticates to all the nodes it contacts with and also verifies any messages from those nodes. For example, nodes u and its parent node v can authenticate to each other and establish a pairwise key K_{uv} based on their certificates and their public/private keys. Node v also checks if the bit indexed by node u’s id in its CRL is ‘1’. Then node v sends node u the current K_g encrypted by K_{uv} and SU. Node u can verify the authenticity of K_g based on SU. Note that a pairwise key is not merely for delivering K_g. In all the overlay multicast routing protocols [2,8,7,16,29], two neighboring nodes in the multicast tree exchange KEEPALIVE messages periodically. They can use their pairwise key for authenticating these KEEPALIVE messages to each other.

When a node leaves the delivery tree, its membership is not changed. Hence, no group rekeying is caused by a node leave. Also, the node may rejoin the tree later. In this case, the same process as the join process is executed.

4.3. Security analysis

In our scheme, no unauthorized nodes can get the group key K_g because a member node only forwards K_g to other member nodes. Nor can a compromised node inject a false group key into the network because the group key is signed by the key server. In addition, an unauthorized node cannot join the multicast tree. Our scheme also provides weak anonymity in the sense that the certificate of a user only has an integer field to uniquely identify the user. It is hard for a node to figure out the identities of other users in the system; however, it may know the IP addresses of other nodes it is communicating with.

4.4. Performance evaluation

This subsection compares the performance of our scheme CRBR with two well-known group rekeying schemes: LKH [25] and SDR [20], assuming that they are employed in overlay multicast and perform batched rekeyings. Note that although the original LKH algorithm was described in the context a single node join/leave, based on the same rekeying principle it is not hard to extend it for multiple node joins/leaves, for example as in [22,28]. The purpose of this comparison is to show that it is more desirable to design a specific group rekeying scheme like CRBR than to directly apply other schemes that were not designed for overlay multicast.

The metric of interest is key server bandwidth overhead, that is, the bandwidth the key server has to allocate to transmit its keys. Because security updates are propagated in the entire multicast tree, the more the key server distributes, the more network and node resources are consumed. Hence, key server bandwidth overhead is also an indication of the total network bandwidth overhead and an individual user's bandwidth overhead, which we may care the most in practice.

Two scenarios are studied. The first scenario considers the bandwidth overhead of the key server for multicasting keys to online nodes. The second scenario considers the bandwidth overhead of the key server for unicasting the current keys to individual nodes that have missed one or several previous group rekeying operations because they were offline. To study the performance of these schemes quantitatively, below we first present an analytical model for node presence dynamics.

4.4.1. The analysis of node presence dynamics

A member node can be in either of two states: presence (online) or absence (offline), and it can switch its state between these two states until its membership duration is expired and is then revoked from the group. We use the term "presence duration" and "absence duration" to denote a continuous time period a node stays in a group and stays outside a group, respectively. A previous study [1] based on multiple sessions in MBone showed that presence durations in a multicast session follow either an exponential distribution or a Zipf distribution. For simplicity, in this study we assume that the durations of node membership follow an exponential distribution with mean $1/\theta$. We also assume that presence durations of nodes follow another exponential distribution with mean $1/\mu$, and further assume that the absence durations of nodes are exponentially distributed with mean R/μ. Thus R is the ratio of the average time for which a node is absent to the average time for which it is present Fig. 1.

Fig. 2 depicts our analytic model. Let the group rekeying interval be T. When the system is in its steady state, during T the number of new subscribers J is equal to the number of revoked subscribers L. Based on queueing theory, the revocation rate of the system is $N \cdot \theta$ where N is the population of the system. Thus, $L = N \cdot \theta \cdot T$.

In this model, the rate of an online user going offline is μ and the rate for an offline user coming online is μ/R. Let N_i and N_o be the populations of online and offline users just after a rekeying operation, respectively, then $N_i + N_o = N$. Denote S as the number of nodes switching from offline state to online state in T. In the steady state, S is also the number of nodes switching from online to offline in T. For periodic batched rekeying, both node additions and revocations are processed at the end of a group rekeying. Thus, $S = N_i \cdot \mu \cdot T = N_o \cdot \mu/R \cdot T$. We have $N_i = \frac{N}{R+1}$ and

$$S = \frac{N \mu T}{R+1}. \qquad (1)$$

We will use S in our evaluation study shortly.

4.4.2. The impact of presence dynamics on group rekeying

4.4.2.1. Scenario I: Multicast cost. For LKH, we adopt the analytic result in [28] to compute the communication cost of the LKH-based group rekeying scheme during one batched rekeying. This analytic result, based on a logical key tree, shows the average number of keys that will be distributed as the function of group size N and number of joining/leaving nodes S to be processed as a batch. For SDR, the upper bound of group rekeying cost is $2r$, where r is the accumulated number of nodes that have been revoked from the system so far. We use simulations to show its average cost in different rekeying periods.

The setting for the comparison is as follows. We assume that in the steady state, $N = 65536$. Let the average membership duration be $1/\theta = 30$ days and the group rekeying interval be $T = 1$ day. Let the size of a key be 20 bytes (128-bit AES encryption, with a key version field and encoding

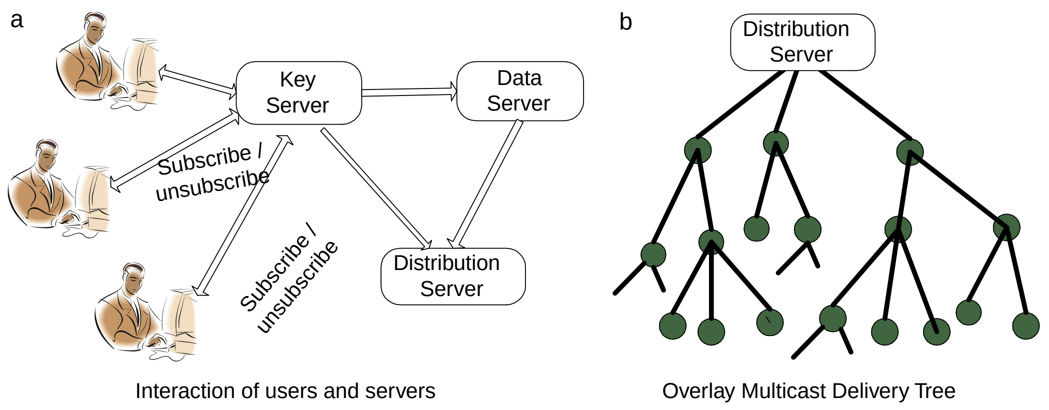


Fig. 1. The system model.

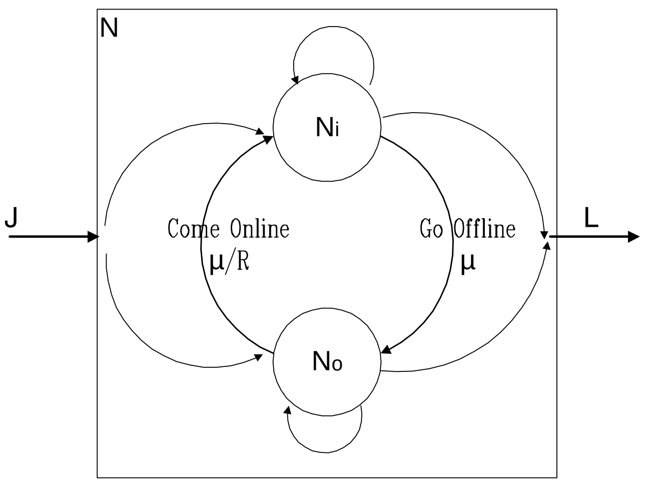


Fig. 2. An analytical model.

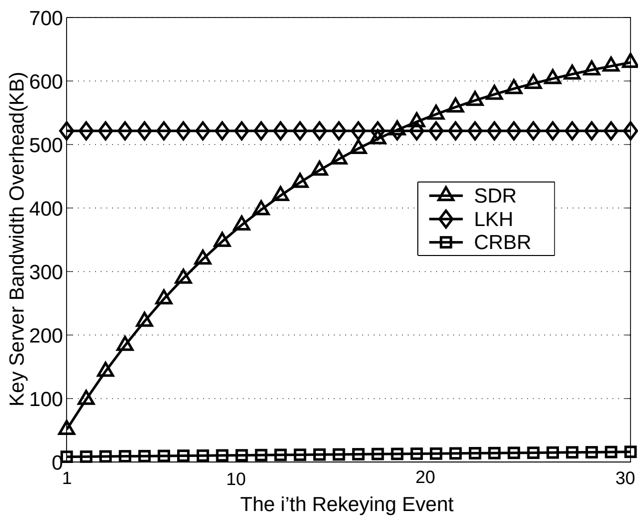


Fig. 3. Rekeying cost of LKH, SDR, and CRBR in different rekeying times.

overhead). Fig. 3 depicts the bandwidth overhead of LKH, SDR, and CRBR in the first thirty rekeying events. We can observe that LKH has the same bandwidth overhead during different rekeying operations, whereas in SDR the bandwidth overhead starts at a small value and eventually exceeds that in LKH. The bandwidth overhead in CRBR increases slightly with time, but it is still far smaller than that in the other two schemes.

Note that although in this simulation setting the bandwidth overhead seems not a big concern even for individual users in any one of these three schemes, the bandwidth saving of our scheme over the other two schemes is very meaningful for a very large-group. For example, when N reaches one million, under the same setting every rekeying cost in LKH is 8.3 MB, in SDR it becomes several megabytes when r reaches hundreds of thousands, whereas in CRBR it is upper bounded by 128 KB. In addition, small message size could also reduce the rekeying latency. Finally, because of its small-size, the rekeying message in CRBR can be distributed using our DoS-resilient k-RIP scheme (introduced in Section 5), where the key server distributes many copies of the rekeying message to prevent DoS attacks.

4.4.2.2. Scenario II: Unicast cost. When the key server distributes its security update, none of the N_o offline nodes receive it. As we showed earlier, S of these nodes come back online during T. When LKH is employed, these S nodes would need to ask the key server for retransmission of the current group key to decrypt the current data because of the statefulness property of LKH, whereas in SDR and CRBR, these nodes can get retransmission from other nodes.

Next we show the overall retransmission cost in these schemes. In SDR or CRBR, only the current group key (no KEKs) is retransmitted to a requesting node. Thus the overall bandwidth overhead is S keys if we do not count other packet overhead. In LKH, a node needs to receive the current group key and some of its KEKs that have been updated. We assume that in LKH, a node that comes online needs to receive on average $h/2$ keys, where h is the height of the key tree maintained by the key server. Therefore, during T, the key server needs to retransmit $\frac{Sh}{2}$ keys.

Fig. 4 plots the bandwidth overhead for unicast-based key updating in LKH and SDR/CRBR with the same

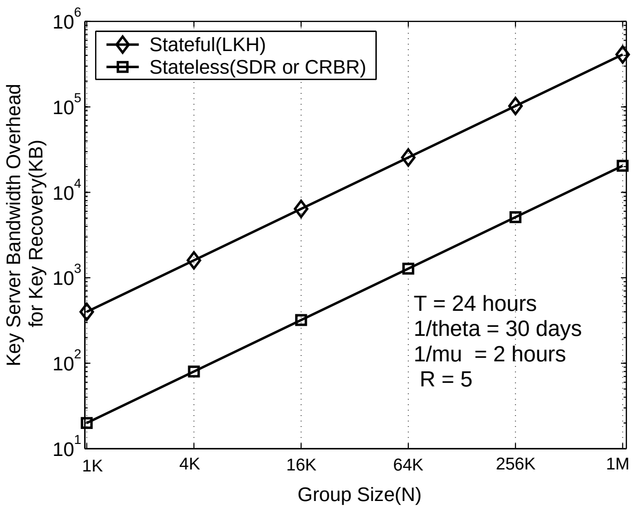


Fig. 4. The bandwidth overhead of key server for unicasting keys to nodes.

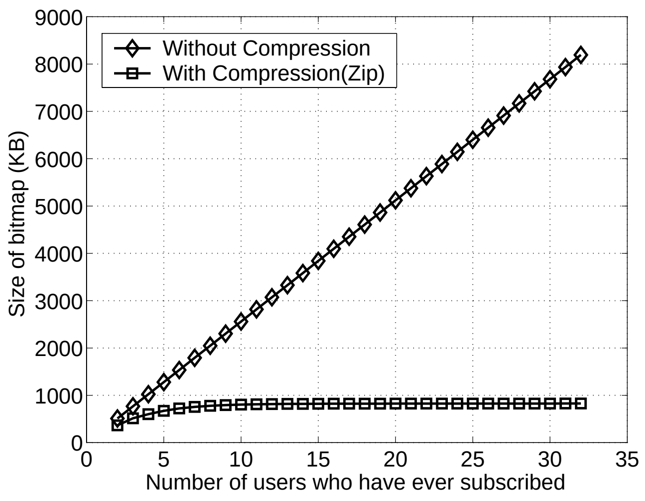


Fig. 5. Effectiveness of compressing a CRL.

network characteristics as in the previous comparison. Here the *y*-axis is in logarithmic scale. We can observe that the key server bandwidth overhead is non-trivial and it increases linearly with group size. For example, for a group of size of 65,536, in LKH the key server has to transmit 25.6 MB to help nodes update keys. In SDR or CRBR, the cost is 1.28 MB. When *N* reaches one million, the cost in LKH becomes greater than 400 MB.

4.4.2.3. Reducing the CRL size by compression. Next we investigate the possibility of further reducing the CRL size. Recall in our CRL management scheme, additional '1's are appended to the bitmap when new users subscribe to the group and the existing '1's are set to '0's for the nodes unsubscribing from the group. Because of this asymmetry, we expect that the CRL size can be further reduced by using a standard compression algorithm. The key server can sign the compressed CRL instead.

To confirm our conjecture, we run the following simulations. We set the initial population two millions; hence, the initial bitmap has 2 Mbits. During the next rekeying interval, one million new users are added to the group (may also include some returning nodes) and 1 million existing users are *randomly* selected to unsubscribe from the group. This process is repeated until the number of users who have ever subscribed to the group reaches 32 M. Clearly, the population of valid users is always 2 M. A standard zip algorithm is used to compress the bitmap. Fig. 5 shows the effectiveness of compression compared to the original CRL size. We can observe that without compression the CRL size increases linearly with the number of subscribers; in case of 32 M subscribers, the CRL size is over 8 MB. Using a zip algorithm, however, the CRL size increases very slowly; in case of 32 M subscribers, it is only about 828 KB. This indicates that compression helps reduce CRL size greatly.

Overall, the above analysis and simulation shows that CRBR outperforms LKH and SDR in terms of bandwidth for the applications under consideration. Especially, as shown next, the small CRL size greatly helps mitigate DoS attacks and node failures.

5. A DoS-Resilient key distribution scheme (*k*-RIP)

This section describes our DoS-Resilient key distribution scheme called *k*-RIP. The scheme can also be used for distribution of other small-size but critical information (e.g., new virus and worm signatures, CRLs) in overlay multicast group.

In overlay multicast, messages are normally injected into the network from the distribution server (i.e., the root node), and are then forwarded hop-by-hop to all the other nodes in the tree. If a malicious node in the tree intentionally discards the message it receives from its parent node, its downstream nodes will not receive the message. This attack is specially severe when the malicious node is very close to the root. We note that this attack is also effective to non-tree based delivery infrastructure. Schemes [24,26,27] based on more complex graphs are more resilient to the attack in general, but they are still subject to selective attacks in which an attacker selectively compromises several nodes close to the injection point.

Note that we cannot solely rely on detection and retransmission mechanisms to address this attack. If every node that detects message losses asks the key server for retransmission, the key server will become the performance bottleneck. Therefore, it is very important that the majority of the member nodes could receive messages even in the presence of DoS attacks.

5.1. Scheme overview

To address the above attacks, we propose that in addition to propagating its message through the root node, the key server also randomly picks *k* nodes (not including the root) in the tree and sends its message to these *k* nodes.

All these k nodes propagate the message towards their children (if any) as well as their parents if their children or parents have not received the message yet. Thus, if a small number of nodes do not forward the message, other nodes might still be able to get it from their children or parents with high probability.

In this scheme, we can simply use sequence numbers to suppress duplicated messages, thus every node only receives one copy of the message and forwards the message to another node at most once. Moreover, this scheme has the additional benefit of reducing the overall latency for all online nodes to receive the message. On the other hand, this scheme incurs the bandwidth overhead for the key server to transmit k additional copies of the message. However, for small-size messages (e.g., tens or hundreds of kilobytes) and a small k (e.g., ≤ 20), in practice this transmission overhead should not be a big concern for the key server.

5.2. Node selection

The very first question is which k nodes to select? To answer this question, we need to consider two factors: latency and DoS-resilience. Ideally, we should select k nodes such that the overall latency is minimized and the number of nodes that can receive messages is maximized. In practice, it is hard to achieve the above goal because the key server might not have the precise knowledge of the tree topology due to the presence dynamics of the member nodes. The key server might know which nodes have joined the tree from a rendezvous point (RP) in many routing algorithms [2,29] because a joining node contacts a RP for information assisting the node to find a position in the tree. However, for scalability the RP does not keep track of the position of a specific node in the tree or if a node is online or offline. Thus, for our scheme to work, a practical issue is to determine which nodes are online.

5.2.1. A Heuristic Selection Algorithm

A simple solution works as follows. The key server randomly selects its member nodes to connect to. If a member node is unreachable, it picks another one. The key server repeats this process until it discovers k online nodes. One problem with this scheme is that the key server might not know the IP addresses of its member nodes because nodes might have dynamic IP addresses. This problem can be addressed by letting the RP record the IP addresses of the nodes that have recently contacted it. Because nodes normally do not change their IP addresses during a session, the key server can use these IP addresses directly[1].

Using the same group characteristics as that used in Section 4.4, we know that for a system that has the registered population of N and the average network size of $N_a = \frac{N}{R+1}$, the key server needs to try an average number of $\frac{kN}{N_a} = k(R+1)$ times to find k online nodes. This shows that the efficiency of this algorithm relies on the node presence dynamics. For a small R, this selection algorithm should work fine. When R is large, we may exploit the following heuristics to increase the *hit ratio*. The idea is that the key server could make a good guess of online members based on the joining times of the members. Again, we assume that presence durations in a multicast session approximately follow an exponential distribution [1]. Assume that the mean of presence durations is $1/\eta$, which can be calculated if every member node records its every presence period and reports its mean presence time to the RP when joining the tree.

The probability $p_i(t)$ that a member node i is still online t time after it joins (also referred to as hit ratio) is $p_i(t) = e^{-\eta \cdot t}$. $p_i(t)$ decreases with t, indicating that the nodes joining more recently are more likely to be online than those joining earlier. Thus, the RP simply tells the key server the ids of m distinct nodes that joined the tree most recently, such that $\sum_{i=1}^{m} p_i(t_i) \geq \upsilon k$, where t_i ($1 \leq i \leq m$) is the time difference between the current time and the joining time of that node. Here $\upsilon \geq 1$ is a parameter reflecting the probability that the key server finds k online nodes from m candidates, and it is variable and should be determined by the presence dynamics of an actual application.

We note that there is a potential attack against this selection algorithm if the message (e.g., the CRL in CRBR) is distributed periodically, because multiple malicious nodes may join the tree just before the distribution time point. Based on our selection algorithm, the RP will likely report their ids to the key server. Thus, these malicious nodes are selected as injection points, reducing the effectiveness of our scheme. To mitigate this attack, it is important that the key server randomly picks nodes from the m candidates for presence test, not preferring the nodes that joined more recently.

5.3. Evaluation of effectiveness

This subsection reports the effectiveness of our k-RIP scheme in increasing hit ratio, reducing propagation latency, and increasing DoS-resilience.

5.3.1. Simulation setting

We first generate a random graph of 10,000 nodes and then construct a tree out of the graph based on the joining algorithm that is also used in [16,29]. Specifically, every joining node searching 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 outdegree of a node is a random number between 1 and 5. Our simulation programs were written using the Csim simulation library [9]. We use the method

[1] Note that for nodes behind network address translators (NATs), if they can join the overlay network [13], they should also be reachable to the server.

of independent replications for our simulations and all our results have 95% confidence intervals that are within 5% of the reported values.

5.3.2. *Increasing hit ratio*

Fig. 6 shows that the hit ratio decreases as m increases. When m is small, say less than 100, the hit ratio is close to 1. However, if m reaches the group size, the hit ratio is about 20% ($R = 4$). This shows that our heuristic algorithm does increase the hit ratio greatly.

5.3.3. *Reducing latency*

We first evaluate the effectiveness of k-RIP in reducing the propagation delay when no nodes are compromised. Fig. 7 plots the latency histogram when k nodes are randomly selected from 10,000 nodes. In the base scheme, there is only one injection point, which is the root. Note

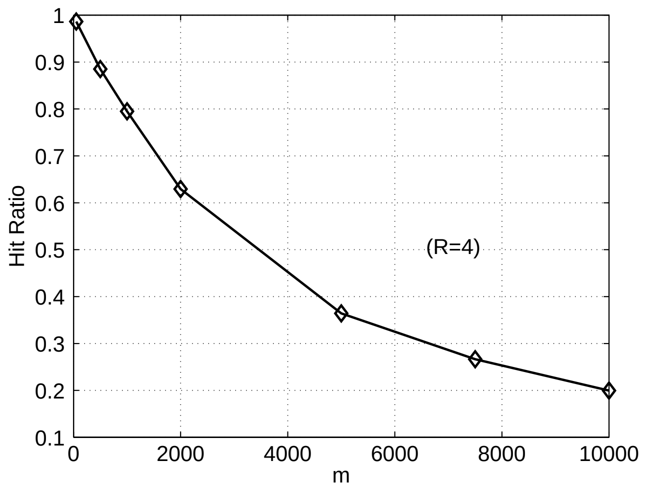


Fig. 6. Hit ratio as a function of m, the size of the most recently joined node list ($k = 10$).

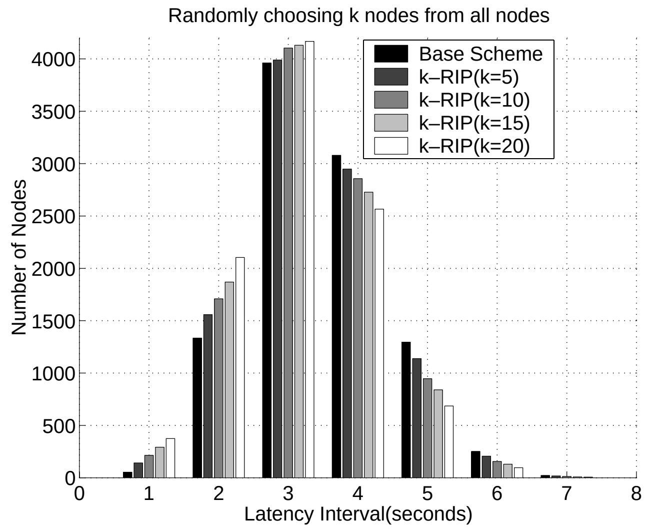


Fig. 7. A histogram showing the number of nodes receiving the distributed message in each time interval.

that for simplicity we calculate the message propagation latency in each link as triple of the link delay to mimic TCP-like unicast schemes; hence, the results should only be interpreted as relative performance. We can see that with the increase of k, more and more nodes can receive a distributed message in shorter time intervals. Note that here the number of nodes receiving the message after three seconds decrease. This is mainly because of the irregularity of the topology tree used in this simulation. The number of nodes which are farther away from their closest injection points becomes less and less.

We have also calculated the average latency in each case (not shown in this figure). When $k = 20$, the average latency is about 15% less than that in the base scheme. Note that the reduced latency is not very significant over the base scheme because the nodes are randomly selected. Due to the tree structure, most of the nodes are selected from leaves or close to leaves.

We investigate the effectiveness of the heuristic algorithm by randomly choosing k ($k \leq 20$) injection points from $m = 50$ most recently joined nodes, which shows that it is not as effective as choosing k nodes from all N_a nodes. When $k = 20$, the average latency is only about 11% less than in the base scheme. This is because the recently joined nodes are mostly leaf nodes in the tree.

5.3.4. *Increasing DoS-resilience*

We first show the analytical model, followed by simulations.

5.3.4.1. *Analytical model.*

Fig. 8 depicts an example multicast tree that has degree of $d = 2$ and group size of N_{a} nodes (excluding the root node that is the distribution server). The solid nodes are good nodes and the empty one is a compromised node that drops messages going through it. Let h be the height (in hops) of the compromised node from the root and s be the number of nodes in the subtree rooted at the compromised node. Then we have $s = \frac{N_{\mathrm{a}}}{d^h} - \frac{1-(\frac{1}{d})^{h-1}}{d-1}$.

Let z be the number of good nodes that can receive messages. Let base scheme be the one in which only a single copy of a message is injected via the root node. It is easy to see that in base scheme $z = N_{\mathrm{a}}–s$. In our k-RIP scheme, besides the root node, k randomly picked nodes also inject the message into the tree simultaneously. If a good node in the subtree rooted at a child node of the compromised node is selected, all the nodes in this subtree will receive the message. If one good node is selected from each subtree rooted at each child node of the compromised nodes, all the good nodes in the network will receive the message. More generally, denote $p(i)$ as the probability that at least one node is selected from each of $i(0 \leq i \leq d)$ subtrees rooted at i child nodes (but no nodes are selected from the other $(d - i)$ subtrees) of the compromised node. Further denote $x = (s - 1)/d$ and $y = N_{\mathrm{a}} - s + 1$, as shown in Fig. 8. Basic

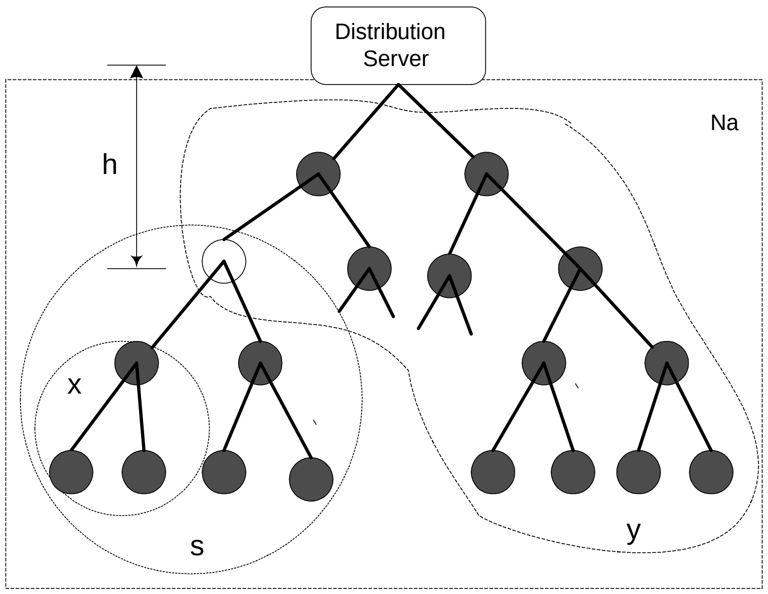


Fig. 8. An example tree of size N_a excluding the root. The empty node is a compromised node whose tree size is s.

probability and combinatorics arguments can be used to derive $p(i)$:

$$p(i) = \begin{cases} \dfrac{\binom{y}{k}}{\binom{N_a}{k}} & (i = 0) \\ \dfrac{\binom{y+ix}{k}}{\binom{N_a}{k}} - \sum_{j=0}^{i-1} \binom{i}{j} p(j) & (0 < i \leq d) \end{cases}$$

The expected value of z is:

$$E[z] = \sum_{i=0}^{d} \binom{d}{i} p(i)(y - 1 + ix). \quad (2)$$

This analytic result has been validated by simulations.

5.3.4.2. A single compromised node. Fig. 9 illustrates the effectiveness of our k-RIP scheme compared to the base

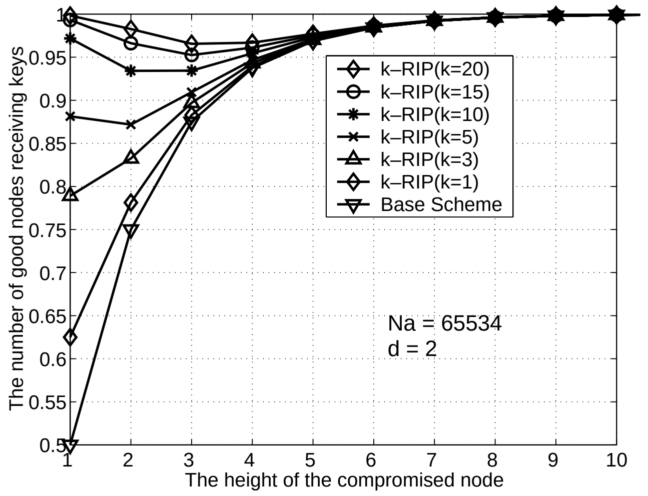


Fig. 9. The fraction of nodes that can receive messages as a function of the location of one single compromised node in the multicast tree and k.

scheme, based on Eq. (2). We observe that in the base scheme, a compromised node with the height $h = 1$ could prevent half of existing nodes from receiving the message, whereas our scheme allows a much larger fraction of nodes to receive the message. For example, when $k = 3$, about 80% nodes can receive the message. More nodes get the message when k increases. For example, when $k = 10$, about 97% nodes could receive it. Recall that in a group of size one million, a rekeying message size is 128 KB. When $k = 10$, the overall bandwidth cost is only about $1.28M$, making this k-RIP scheme very efficient. The figure also indicates that in the base scheme, to cause denial of service to more nodes, an attacker should manage to become as close to the root as possible. However, when our scheme is deployed, that is not necessarily the best strategy for the attacker. For example, when $k = 10$, becoming a node with $h = 2$ or $h = 3$ gives the attacker a little more advantage than becoming a node with $h = 1$.

Next we study the effectiveness of our heuristic selection algorithm that selects k nodes from mostly recently joined m nodes (denoted as k-RIP-h, dashed lines in Fig. 10), compared with the base scheme in which we randomly select k nodes from N_a nodes (denoted as k-RIP, solid lines). We set the outdegree of a node $d = 3$ and the size of the candidate set $m = 50$. Fig. 10 indicates that except when $h = 1$, the effectiveness of k-RIP is only slightly affected when choosing k ($k \leq 20$) nodes from 50 most recently joined nodes instead of choosing from all $N_a = 10{,}000$ nodes.

5.3.4.3. Multiple compromised nodes. Fig. 11 shows the effectiveness of our scheme when all the d^h nodes in height h are compromised, instead of a single compromised node in the previous case. This simulates the worse case for the base scheme. When $h = 1,2,3$, in the base scheme almost no good nodes receive keys. The effectiveness of our

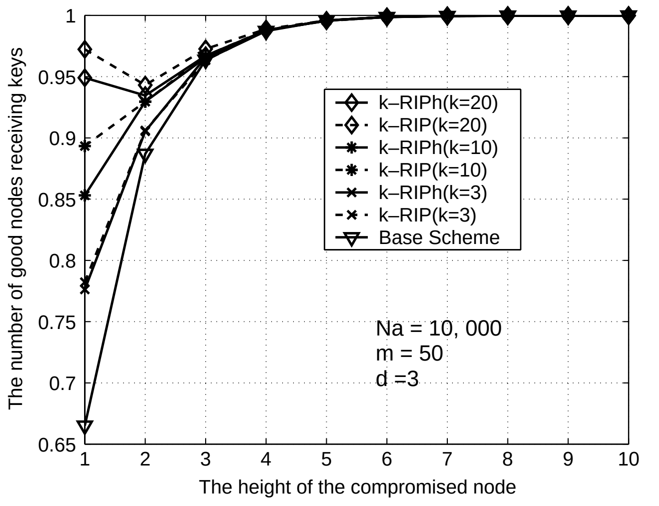


Fig. 10. The effectiveness of our heuristic selection algorithm (through simulation, 95% CI is within 5% interval of a reported value).

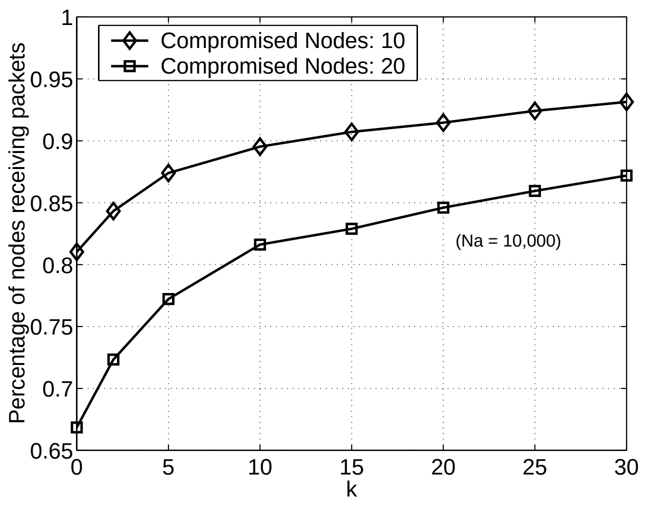


Fig. 12. The effectiveness of k-RIP in increasing message acceptance ratio when nodes are randomly compromised ($N_a = 10{,}000$).

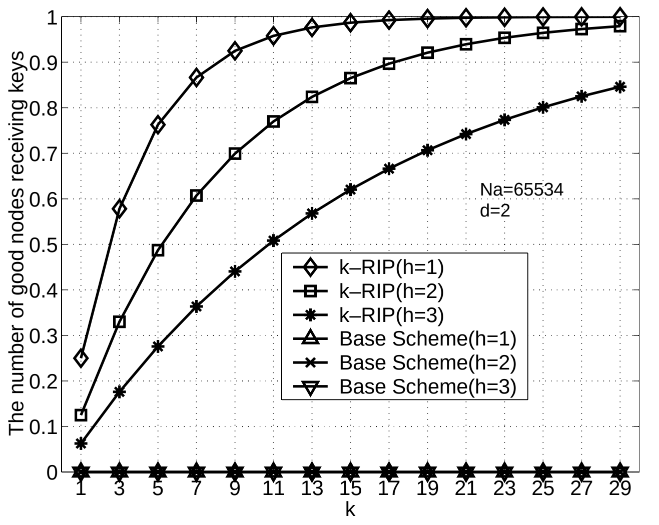


Fig. 11. The fraction of nodes that can receive messages when all the nodes in the height h are compromised.

scheme decreases with h. For example, in the case of $h = 3$ (i.e., all the *8* nodes in level 3 are compromised), when $k = 20$ only about 72% nodes receive keys. Note that if the key server has the global knowledge of the multicast tree, higher robustness can be achieved with smaller communication cost. For example, in the case of $h = 3$, there are totally 16 subtrees of these 8 compromised nodes. By selecting one node from each of the 16 subtrees, all good nodes can receive keys.

We also investigate the impact of randomized node compromises through simulations. If we set every node in the tree to be compromised with the same probability, most of the compromised nodes will be the leaf nodes, which cannot cause DoS attacks to other nodes. Therefore, we instead let the compromised nodes to be randomly chosen from the first 500 nodes that joined the tree (they are closer to the root node than other nodes). We can observe from Fig. 12 that when the number of injections increases, the acceptance ration also increases. When there are 20 compromised nodes, if there are no additional injections, about 67% nodes receive the message (this is much better than the worst case); if there are $k = 30$ injections, this ratio can be increased to 87%. This means that for a group of 10,000 nodes, with 30 proactive injections, 20% (or $10{,}000 \times 20\% = 2000$) nodes can receive the message in the first round. Therefore, we consider our technique is very effective, in addition to being also very efficient due to the small-size of the rekey message output from CRBR.

6. Conclusions and future work

We have presented a bandwidth efficient scheme that integrates network access control and group key management. Performance analysis and simulation study show that our scheme incurs much smaller communication overhead than two other well-known schemes. We also proposed a DoS-resilient information distribution scheme that delivers small-size but critical messages (e.g., keys) to a large fraction of nodes with high probability even if an attacker can selectively compromise nodes in the multicast data delivery hierarchy. The scheme has only considered to distribute one message and used a fixed k. In the future, we will consider selecting k dynamically based on the receiving status in the past.

Acknowledgements

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, M. Ammar, Multicast group behavior in the Internet's Multicast Backbone (Mbone), IEEE Communications (1997).

[2] S. Banerjee, B. Bhattacharjee, C. Kommareddy, Scalable application layer multicast, in: Proceedings of ACM SIGCOMM, 2002.

[3] S. Banerjee, S. Lee, B. Bhattacharjee, A. Srinivasan, Resilient multicast using overlays, in: Proceedings of ACM Sigmetrics, 2003.

[4] G. Badishi, I. Keidar, A. Sasson, Exposing and eliminating vulnerabilities to denial of service attacks in secure gossip-based multicast, in: Proceedings of Dependable Systems and Networks (DSN), 2004.

[5] D. Balenson, D. McGrew, A. Sherman, Key management for large dynamic groups: one-way function trees and amortized initialization, IETF Internet Draft (in preparation), August 2000.

[6] CCITT Recommendation X.509: The Directory-Authentication Framework, 1988.

[7] Y. Chu, S. Rao, S. Seshan, H. Zhang, Enabling conferencing applications on the internet using an overlay multicast architecture, in: Proceedings of ACM SIGCOMM, 2001.

[8] Y. Chu, S. Rao, H. Zhang, A case for endsystem multicast, in: Proceedings of ACM Sigmetrics'00, 2000.

[9] <http://www.mesquite.com/>.

[10] S. Deering, Multicast routing in internetworks and extended LANs, Computer Communication Review 18(4), 1988. ACM SIGCOMM '88 Symposium: Communications Architectures and Protocols.

[11] C. Diot, B. Levine, B. Lyles, H. Kassem, D. Balensiefe, Deployment issues for the IP multicast service and architecture, IEEE Network 14 (2000) 88–98.

[12] H. Eriksson, Mbone: the multicast backbone, Communications of the ACM (1994) 54–60.

[13] B. Ford, P. Srisuresh, D. Kegel, Peer-to-peer communication across network address translators, in: Proceedings of USENIX Annual Technical Conference, 2005.

[14] H. He, T. Hardjono, B. Cain, Simple Multicast Receiver Access Control, draft-irtf-gsec-smrac-00.txt, November 2001.

[15] Internet Group Management Protocol, Version 3. <http://www.networksorcery.com/enp/rfc/rfc3376.txt>.

[16] J. Jannotti, D. Gifford, K. Johnson, M. Kaashoek, J.O'Toole, Overcast: reliable, multicasting with an overlay network, in: Proceedings of Fourth USENIX OSDI Symposium, 2000.

[17] D. Liu, P. Ning, K. Sun, Efficient self-healing group key distribution with revocation capability, in: Proceedings of the 10th ACM CCS, 2003.

[18] J. Li, P. Reiher, G. Popek, Resilient self-organizing overlay networks for security update delivery, IEEE Journal on Selected Areas in Communications 22 (2004).

[19] L. Mathy, N. Blundell, V. Roca, A. Elsayed, Impact of simple cheating in application-level multicast, in: Proceedings of IEEE Infocom, 2004.

[20] D. Naor, M. Naor, J. Lotspiech, Revocation and tracing schemes for stateless receivers, in: Advances in Cryptology-CRYPTO 2001. Springer-Verlag Inc., LNCS 2139, 2001, pp. 41–62.

[21] A. Perrig, D. Song, D. Tygar, ELK, a new protocol for efficient large-group key distribution, in: Proceedings of the IEEE Symposium on Security and Privacy 2001, Oakland, CA, May 2001.

[22] S. Setia, S. Koussih, S. Jajodia, E. Harder, Kronos: a scalable group rekeying approach for secure multicast, in: Proceedings of the IEEE Symposium on Security and Privacy, Oakland, CA, May 2000.

[23] S. Setia, S. Zhu, S. Jajodia, A comparative performance analysis of reliable group rekey transport protocols for secure multicast, Performance Evaluation 49(1/4) (2002) 21–41 (special issue), in: Proceedings of Performance 2002, Rome, Italy, September.

[24] D. Song, D. Zuckerman, J. Tygar, Expander graphs for digital stream authentication and robust overlay networks, in: Proceedings of IEEE Symposium on Security and Privacy, 2002.

[25] C. Wong, M. Gouda, S. Lam, Secure group communication using key graphs, in: Proceedings of SIGCOMM, 1998, Vancouver, British, Columbia, pp. 68–79.

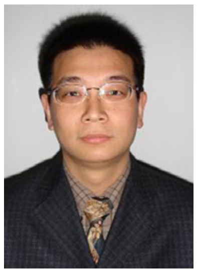
[26] R. Wright, P. Lincoln, J. Millen, Efficient fault-tolerant certificate revocation, in: Proceedings of ACM CCS, 2000.

[27] H. Yang, H. Luo, Y. Yang, S. Lu, L. Zhang, HOURS: achieving DoS resilence in an open service hierarchy, in: Proceedings of Dependable Systems and Networks (DSN), 2004.

[28] Y. Yang, X. Li, X. Zhang, S. Lam, Reliable group rekeying: design and performance analysis, in: Proceedings of ACM SIGCOMM, 2001.

[29] B. Zhang, S. Jamin, L. Zhang, Host Multicast: a framework for delivering multicast to end users, in: IEEE Infocom, 2002.

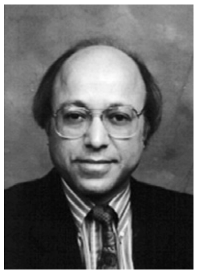
Sencun Zhu is an assistant professor in Department of Computer Science and Engineering and College of Information Sciences and Technology, the Pennsylvania State University. He received the Ph.D. degree in Information Technology from George Mason University in 2004. His research interests include network and systems security, ad hoc and sensor networks, and peer-to-peer computing.



Chao Yao received the BS and MS degree from the Department of Computer Science at Huazhong University of Science and Technology, China, in 1997 and in 2000, respectively. Currently, he is a Ph.D. candidate in George Mason University. His research direction is information security.

Donggang Liu received the BS degree from the Department of Computer Science at Beijing Institute of Technology, China, in 1998, the MS degree from Institute of Computing Technology, Chinese Academy of Science, China, in 2001, and the Ph.D. degree from the Department of Computer Science at North Carolina State University, in 2005. Currently. He works as an assistant professor in the Department of Computer Science and Engineering, UT Arlington.

Sanjeev Setia received the M.Sc. (Tech.) degree from BITS, Pilani, India in 1987, and the M.S. and Ph.D. degrees in computer science from the University of Maryland, College Park in 1990 and 1993 respectively. He is currently an associate professor in the Department of Computer Science at George Mason University. His current research interests are in multicast security, ad hoc networks, peer to peer computing, and performance evaluation of computer systems.



Sushil Jajodia is BDM International Professor of Information Technology and the director of Center for Secure Information Systems at the George Mason University, Fairfax, Virginia. He served as the chair of the Department of Information and Software Engineering during 1998-2002. He joined GMU after serving as the director of the Database and Expert Systems Program at the National Science Foundation. Before that he was the head of the Database and Distributed Systems Section at the Naval Research Laboratory, Washington. Dr. Jajodia received his Ph.D. from the University of Oregon, Eugene. He has authored five books, edited twenty two books, and published more than 250 technical papers in the refereed journals and conference proceedings. He received the 1996 Kristian Beckman award from IFIP TC 11 for his contributions to the discipline of Information Security, and the 2000 Outstanding Research Faculty Award from GMU's School of Information Technology and Engineering. Dr. Jajodia has served in different capacities for various journals and conferences. He is the founding editor-in-chief of the Journal of Computer Security and on the editorial boards of ACM Transactions on Information and Systems Security, IEE Proceedings on Information Security, International Journal of Cooperative Information Systems, and International Journal of Information and Computer Security. He is the consulting editor of the Springer International Series on Advances in Information Security. He also serves as the chair of the IFIP WG 11.5 on Systems Integrity and Control. He has been named a Golden Core member for his service to the IEEE Computer Society, and received International Federation for Information Processing (IFIP) Silver Core Award "in recognition of outstanding services to IFIP" in 2001. He is a past chairman of the ACM Special Interest Group on Security, Audit, and Control (SIGSAC) and the IEEE Computer Society Technical Committee on Data Engineering. He is a senior member of the IEEE and a member of IEEE Computer Society and Association for Computing Machinery. The URL for his web page is http://csis.gmu.edu/faculty/jajodia.html.