Adding Confidentiality to Application-Level Multicast by Leveraging the Multicast Overlay*

Cristina Abad
Fac. de Ing. en Electricidad y Computación
Escuela Superior Politécnica del Litoral
cabad@fiec.espol.edu.ec

Indranil Gupta
Department of Computer Science
University of Illinois at Urbana-Champaign
indy@cs.uiuc.edu

William Yurcik
National Center for Supercomputing Applications
University of Illinois at Urbana-Champaign
byurcik@ncsa.uiuc.edu

Abstract

While scalability, routing and performance are core issues for Application-Level Multicast (ALM) protocols, an important but less studied problem is security. In particular, confidentiality (i.e. data secrecy, achieved through data encryption) in ALM protocols is needed. Key management schemes must be simple, scalable, and must not degrade the performance of the ALM protocol. We explore three key management schemes that leverage the underlying overlay to distribute the key(s) and secure ALM. We evaluate their impact on three well-known ALM protocols: ESM, ALMI and NICE. Through analysis and simulations, we show that utilizing the ALM overlay to distribute key(s) is feasible. For a given ALM protocol, choice of the best key management scheme depends on the application needs: minimizing rekeying latency or minimizing data multicasting latency.

1. Introduction

Several application-level multicast protocols (ALM) have been proposed over the last few years [1]. Only recently have papers dealing with security issues in ALM emerged [12, 13]. Security can be defined in terms of confidentiality, integrity and availability. In the ALM context *confidentiality* means that only group members should be able to read the multicasted messages. For *integrity*, hosts should be able to verify the source and that messages have not changed in transit. In ALM, hosts are trusted to follow the protocol. A host mis-forwarding or delaying messages can easily disrupt *availability*.

In the peer-to-peer (P2P) and ALM areas, some problems that affect availability have been studied [5, 8]. Key distribution in ALM–necessary for confidentiality–has not been studied yet. Key distribution schemes for IP multicast [11, 17] have been proposed but none for ALM. While some of the existing key distribution schemes could potentially be adapted to ALM protocols, doing so may require imposing significant additional structure to the overlay or having the key distribution node have full knowledge of the group members and/or distribution tree. Furthermore, tree structure and neighboring relationships may change constantly adding more complexity than in usual IP multicast settings. There is a need for key distribution schemes to use with ALM protocols without imposing significant restrictions. These schemes should be straight-forward to implement and should not impose additional structure to the overlay (e.g. should not impose additional overlay neighboring requirements).

We explore three key distribution/management schemes that exploit ALM overlays to add confidentiality to application-layer multicast. We study the case of three existing ALM schemes: ESM [6], ALMI [14] and NICE [4]. Our analysis and simulation results show that it is feasible to implement key management schemes leveraging the ALM overlay. The schemes add low overhead but choice of an appropriate scheme depends on the ALM protocol and characteristics of the group.

*Funded in part by a grant from the Office of Naval Research within the National Center for Advanced Secure Systems Research

2. Background and Related Work

Application-Level Multicast Protocols. ESM, ALMI and NICE have been selected for evaluation purposes. End System Multicast (ESM) [6], uses a two-step process to build and refine the multicast tree. First it builds a mesh with some desirable performance properties. Then, it constructs a spanning tree of the mesh rooted at the source–using the reverse shortest path between each recipient and source. ALMI's [14] tree-building algorithm is centralized: the rendezvous point (RP) calculates the minimum spanning tree. A new node receives from the RP an ID and the address of its new parent. NICE [4] nodes self-organize into a layered topology. Each layer is organized organized into clusters (each with k members). Only the leader of a cluster is part of the next layer. Layer 0 contains all nodes, and the last layer contains only one node. Data delivery uses source-specific trees implicitly defined by a forwarding rule. Each cluster leader periodically checks the size of the cluster to keep it within the desired bounds ($[k, 3k-1]$).

Security Issues. Key distribution for IP multicast has been extensively studied. Some solutions [11, 16, 17] allow rekeying of the multicast group on membership changes to ensure secrecy. Rodeh et al. [15] analyze three group communication systems rekeying schemes that depend only on unicast messaging and do not impose a multicast tree or hierarchy.

In P2P/overlay security, research has concentrated on anonymity [9], availability [5, 8] and authentication [10]. Recently, ALM security has been studied: Mathy et al. [12] analyzed the malicious effect of peers mis-reporting distance measurements, and Nicolosi et al. [13] studied how to send secure acknowledgements.

3. ALM Key Management

We believe that the multicast overlay can be efficiently used to distribute key(s) to secure ALM protocols that can provide forward and/or backward secrecy by changing the key on group membership changes. This leads to the following three schemes (see Fig. 1):

- One shared key for the group – Cryptographic costs of multicasting are low. To ensure secrecy, group needs to be rekeyed on membership changes.
- Shared keys between neighbors – Each pair of peers that are neighbors in the multicast distribution tree/path share a key. Membership changes are easily processed, but multicast message transmission overhead increases significantly (messages are re-encrypted at each hop[1]).

[1] Key Encryption Keys (KEK) can be used. Sender generates KEK and encrypts message with it. Only the KEK needs to be re-encrypted.

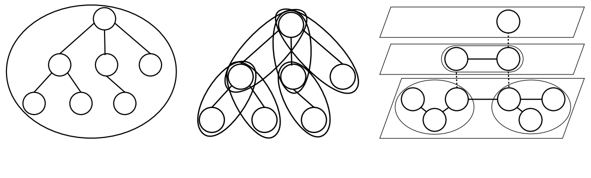


Figure 1. Key management approaches

Table 1. Nomenclature

X_{pubk}	X's public key
X_{privk}	X's private key
p,g,n	Diffie-Hellman parameters
$X\text{-}Y_{sck}$	Secret channel shared between X and Y
G_k	Group-wide shared secret key
M_{id}	Group member ID
K	Message encryption key
m	Message

- Subgroup shared key – (When applicable) subgroups or layers can each share a key among its members. Its costs are in-between the two above.

For the case of a single group key, a Rendezvous Point (RP) would be in charge of rekeying the group, act as a certification authority and perform access control. The access control problem is orthogonal to key distribution. A certification authority is needed to sign peer certificates and validate credentials.

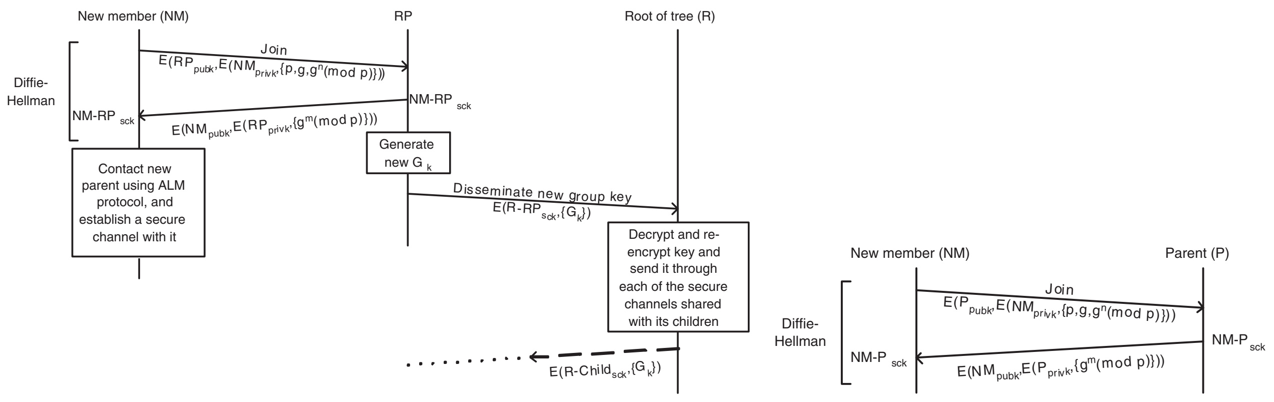
Next, we describe in a generalized way each of the key management schemes and any specific issues involving their implementation in each of the selected ALM protocols. Section 4 presents an analysis of the costs involved. Table 1 lists the nomenclature used in this section.

3.1. Group-wide Shared Secret Key

A group-wide secret key is shared among members. Rekeying is necessary on membership changes.

Join (Fig. 2(a)). The node and RP establish a secure channel (e.g. using Diffie-Hellman). The RP generates a new group key and sends it to the root of the tree, which starts its dissemination through the secure channels of the tree by sending it to its children using the secure channels they share. The children forward the key, re-encrypting it at every hop. The peer joins the group and establishes a private shared key (secure channel) with its parent. Every pair of neighbors can securely communicate using these secure channels.

In ESM the RP needs to maintain a secure channel only with the root. In ALMI the RP maintains secure channels with each group member. To disseminate the new key, it sends it to any member. NICE peers maintain secure channels with their cluster(s) neighbors.


New member (NM)
RP
Root of tree (R)
Diffie-Hellman
Join
E(RP_pubk, E(NM_privk, {p,g,g^n(mod p)}))
NM-RP sck
E(NM_pubk, E(RP_privk, {g^m(mod p)}))
NM-RP sck
Contact new parent using ALM protocol, and establish a secure channel with it
Generate new G k
Disseminate new group key
E(R-RP_sck, {G_k})
Decrypt and re-encrypt key and send it through each of the secure channels shared with its children
E(R-Child_sck, {G_k})
New member (NM)
Parent (P)
Diffie-Hellman
Join
E(P_pubk, E(NM_privk, {p,g,g^n(mod p)}))
NM-P sck
E(NM_pubk, E(P_privk, {g^m(mod p)}))
NM-P sck

(a) Group-wide scheme (b) Neighbors scheme

Figure 2. Joining

Leave (Fig. 3(a)). When a member leaves, the RP is notified, generates a new group key and distributes it as described above, after the tree has been rebuilt. For extra safety, the RP can notify the group members that the leaving peer should no longer be included in the tree (e.g. by sending the peer's ID together with the key).

In ALMI it is straightforward to know when the tree has been rebuilt, since the RP centrally organizes it, it can wait for acknowledgements from the members before sending the new group key. In NICE, the cluster leaders from Layer 0 to which the leaving member belonged can send an ACK to the RP notifying that the clusters have been re-organized. In ESM, the children of the leaving node can notify the RP when they have joined new parents.

Multicasting. To securely multicast a message, the root/sender first encrypts it with the shared private key (E(G_k,{m}))[2]. Messages do not need to be sent through secure channels. Each qualified receiver has the group key and uses it to decrypt the messages. No extra operations are required for ESM, ALMI or NICE.

3.2. Shared Keys between Pairs of Neighbors

The RP is not involved in the security operations of this scheme. Re-encryption of the message key is needed at each hop. Extending ALM protocols to use this scheme is straightforward and no extra operations – other than key agreement and encryption– are required.

Join (Fig. 2(b)). Peer establishes key with parent. No extra operations needed for backward/forward secrecy.

Leave. No extra operations are required.

[2]For sender authentication, the message should be signed first.

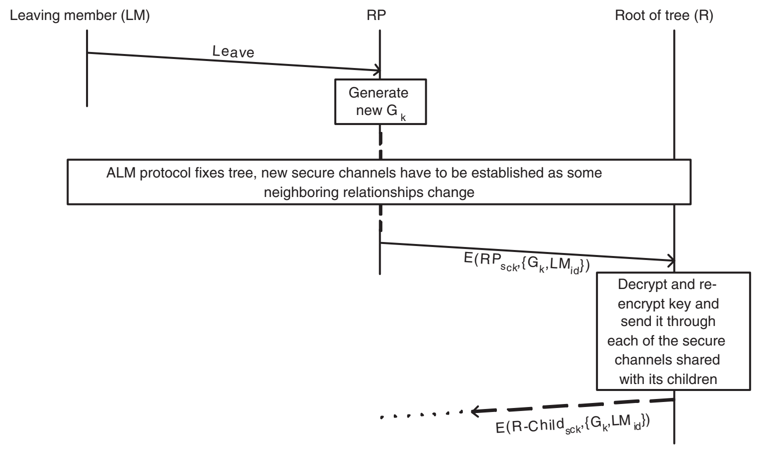
Multicasting (Fig. 3(b)). Sender generates a key and encrypts the message with it. It sends the encrypted message to each of its children, together with the key (sent through the secure channel shared with each). The encryption key is re-encrypted along the way.

3.3. Shared Secret Keys for Each Subgroup

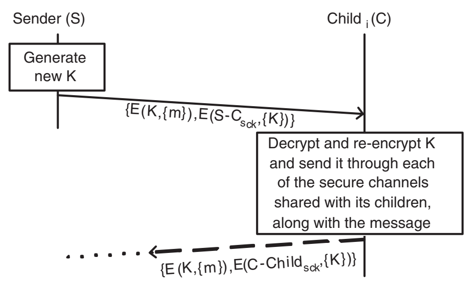
This scheme is a generalized variant of the two above. Each subgroup shares a secret key. The overlay should have a hierarchy or some kind of subgrouping for this scheme to be adopted. Of the ALM schemes analyzed in this paper, only NICE has a layered approach that leads naturally to the use of hierarchical keys. Actually, its layered architecture can be directly extended to use the spatial clustering approach described in [3]. Each layer shares a key, each cluster shares a key, and each cluster member shares a secret key with its cluster leader. The operations described next are specific to NICE.

Join. A joining node is assigned to a layer zero (L_0) cluster. To ensure backward secrecy, the cluster leader generates a new cluster key and distributes it among its peers. Then, it contacts the key server or RP to request a rekey of L_0. The key server generates a new shared layer key and sends it to all members in the next layer (L_1), encrypted with the current L_1 secret key. The members of L_1 decrypt the key and the cluster leaders of L_0 distribute it among their peers. If the new member joins any layer, other than L_0, each layer needs to be rekeyed.

Leave. Each of the leaders of the clusters the departing member belongs to requests a layer rekey to the key server. Layers are rekeyed as described above. If departing member is a cluster leader, a remaining member of the cluster is selected as the new leader and joins all


Leaving member (LM)
RP
Root of tree (R)
Leave
Generate new G k
ALM protocol fixes tree, new secure channels have to be established as some neighboring relationships change
E(RPsck,{Gk,LMid})
Decrypt and re-encrypt key and send it through each of the secure channels shared with its children
E(R-Childsck,{Gk,LMid})

(a) Join, group-wide scheme


Sender (S)
Child i (C)
Generate new K
{E(K,{m}),E(S-Csck,{K})}
Decrypt and re-encrypt K and send it through each of the secure channels shared with its children, along with the message
{E(K,{m}),E(C-Childsck,{K})}

(b) Multicasting, neighbors scheme

Figure 3. Leaving and multicasting

the layers to which the leaving member belonged. The affected clusters are rekeyed. New keys for the affected layers are requested as described. If a cluster shrinks to less than three members, it merges with a cluster in the same layer. The leader of the smallest cluster is demoted and removed from the next upper layer.

Multicasting. When multicasting a message (using NICE's implicit approach) the message has to be re-encrypted along the way, as it traverses different layers.

4. Analytical Comparison

We present an analysis of the costs with respect to point-to-point messages and cryptographic operations involved when members join and leave the multicast group, for each of the described solutions. The analysis presented in this section is summarized in Table 2.

Member joins–if no backward secrecy is desired–add only a small overhead: creating the secret channels with its neighbors and/or RP. In this paper, only the case of ESM with no backward secrecy is analyzed. For the other cases it is assumed that backward secrecy is desired, as it makes the analysis more interesting.

Secure Channels. To securely disseminate group key changes through the multicast tree ESM and ALMI need $n-1$ (the number of edges in the tree) secure channels. In ESM only the source/root needs to maintain a secure channel with the RP. ALMI builds a shared tree in which any member can send a multicast message and the RP maintains information about all group members; thus, it makes sense for the RP to maintain secure channels with each of the members (n). In NICE the RP generates a new layer key when needed, and sends it through a secure channel to a layer member which then distributes it to the rest of the layer. Since there are $\log_k n$ layers, the RP needs to maintain $\log_k n+1$ secure channels. To securely distribute layer and cluster keys within group members, every member must maintain a secure channel with its cluster leader. Since every member belongs to only one cluster for which it is not its leader, and since the leader of the higher layer does not have a leader, the number of secure channels inside the tree is $n-1$.

Key Agreement. In Table 2, for the key agreement column, a value of 2 (two) means that a new member needs to agree on two keys: one with parent and one with RP. For ESM with shared keys between neighbors, a secure channel with the RP is not needed. For ALMI, when a member leaves, its children ($O(m)$) need to rejoin the group. Since m is a small number independent of the size of the group, $O(m)$ is equivalent to $O(1)$. For NICE, when a member joins the group, it needs to establish secure channels with each of the members of the cluster it is joining ($O(k)$). A member leaving the group may lead to the merging of two clusters, in which case new secure channels within the new cluster need to be established ($O(k)$). Since k is a small number independent of the group size, $O(k) \equiv O(1)$.

Point-to-Point Messages. The number of point-to-point messages added to each protocol depends on the size of the group/subgroup being rekeyed. For solutions involving a shared group key, the whole group needs to be rekeyed on membership changes (if forward/backward secrecy is desired), and thus the $O(n)$

Table 2. Some properties of the different secure application-level multicast solutions

Scheme		Pnt2pnt messages	Cryptographic operations: Key agreement	Cryptographic operations: Symmetric key	Secure channels: In tree	Secure channels: With RP
ESM	Shared group key				$n-1$	1
	Join	$O(n)$	2	$2(n-1)$		
	Join (no backward secrecy)	$O(1)$	2	0		
	Leave	$O(n)$	0	$2(n-1)+2$		
	Shared keys (neighbors)				$n-1$	0
	Join	$O(1)$	1	0		
	Leave	$O(1)$	0	0		
ALMI	Shared group key				$n-1$	n
	Join	$O(n)$	2	$2n$		
	Leave	$O(n)$	$O(1)$	$2n$		
	Shared keys (neighbors)				$n-1$	0
	Join	$O(1)$	2	0		
	Leave	$O(1)$	$O(1)$	0		
NICE	Shared group key				$n-1$	1
	Join	$O(n)$	$\Theta(1)$	$O(n)$		
	Leave	$O(n)$	$O(1)$	$O(n)$		
	Shared keys (neighbors)				$n-1$	0
	Join	$\Theta(1)$	$\Theta(1)$	$O(1)$		
	Leave	$O(1)$	$O(1)$	$O(1)$		
	Spatial clustering				$n-1$	$\log_k n + 1$
	Join	$O(n)$	$\Theta(1)$	$O(n)$		
	Leave	$O(n)$	$O(1)$	$O(n)$		

value. NICE with spatial clustering has $O(n)$ in this field because if L_0 needs to be rekeyed, every member needs to receive the new key. For solutions involving no shared group key but rather shared keys between neighbors, $O(1)$ point-to-point messages are used, since the joining member needs to establish secure channels only with its new parent (or new neighbors for NICE).

Symmetric Key. Symmetric key operations are encryption/decryption operations needed to forward a message along the multicast tree through the secure channels each pair of neighbors shares. For the solutions that involve rekeying (group, layer, etc.), the number of symmetric key operations is in the same order as the point-to-point messages, since each point-to-point message sent goes through a different secure channel, and needs to be re-encrypted along the way.

5. Experimental Comparison

We present simulation results comparing the discussed protocol/key management combinations. We believe that the simulations results and analysis are valuable. Even when real-life results may vary, differences will be due to network delay but the cryptographic operations overhead should not differ much from our simulated values given that these are performed at each host and are not affected by external network conditions.

Simulations were performed with *myns* [2]. Key management, rekeying and cryptographic operations, and the ALMI protocol were added to *myns*, which already had versions of ESM and NICE. The simulation parameters are listed in Table 3. The network topology was generated using the GT-ITM transit stub topology generator [18]. The generated topology has 9312 nodes (routers); the average node degree is 4.19. Each end host is directly connected to a router at the edge of the topology. Simulated links are lossless and have the same delay. The RP is always available. Peers stay connected and participate in the protocol from the time they join until the end of the simulations. Figure 5 shows the node link distribution of the nodes in the generated topology.

The estimated time for performing cryptographic operations was obtained from the Crypto++ v5.1 [7] benchmark (on a Pentium 4, 2 GHz, Windows XP PC). The symmetric cryptographic operations were done with DES (1024 bit keys); encrypted messages had 100 Kb. Diffie-Hellman was used for key agreement.

Table 3. Parameters used in simulations

Nodes	9312
Source sending rate	1.5 pkts/sec
Process time of a key agreement op.	0.0037 sec
Process time of a symmetric crypto-op.	0.005 sec
Simulation runs per plotted point	10
Link delay	1.0 msec

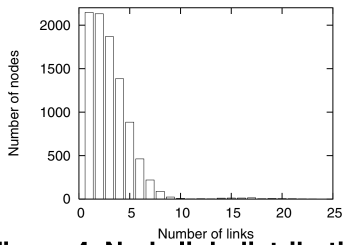
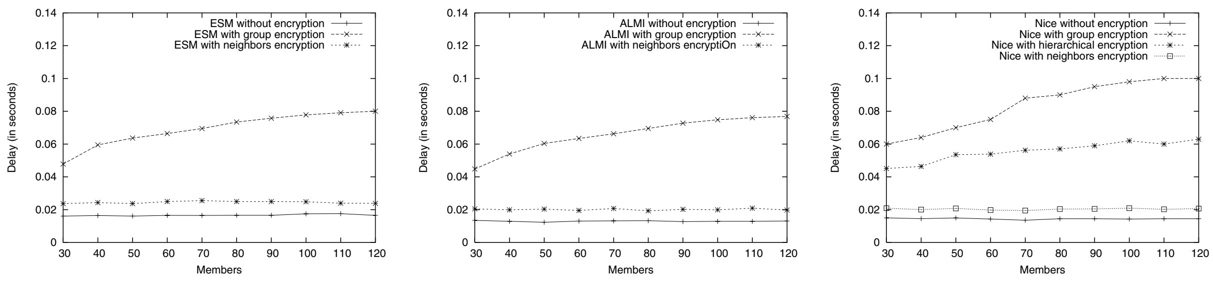


Figure 4. Node link distribution



(a) ESM (b) ALMI (c) NICE

Figure 5. Average join latency, for all group members, across all runs

Join. The join latency average for different group sizes is shown in Fig. 5. For ESM and ALMI, joining with the group-wide scheme is significantly slower than with the neighbors scheme. The join latency when a group-wide shared key is used grows with the group size, because the group is rekeyed on membership changes. NICE follows the same pattern as ESM and ALMI: joining with the subgroups scheme is faster than with the group-wide scheme and slower than with the neighbors scheme. Thus, the group-wide shared key scheme is most appropriate for groups in which group membership changes are not too frequent. For dynamic applications such as peer-to-peer overlays with high churn, the neighbors or subgroup scheme is a better fit.

Multicasting. The data delivery latency for different group sizes is shown in Fig. 6. DVMRP is plotted as a baseline for comparisons. The results are opposite to those obtained for the join latency average. The best results are obtained when a group-wide key is used (0.019-0.026 sec). The worst results come from using the neighbors scheme (0.04-0.045 sec). The subgroup scheme (in NICE) is in between the other two schemes. Thus, the group-wide shared key scheme is more appropriate for applications with high data send rates.

Data Losses. Fig. 7 shows the average percentage of undelivered/lost packets. ALM reliability is not affected by the choice of key management scheme.

To summarize, joining with the group-wide scheme is significantly slower, and its delay increases as the group size grows due to the rekeyings after membership changes. Multicasting with the group-wide scheme is faster than with the neighbors scheme due to the hop-by-hop re-encryption. The percentage of lost packets is independent of the key distribution scheme used.

6. Conclusions

We described and analyzed three key management schemes that leverage the ALM overlay and enable encryption of multicasted data to achieve confidentiality in application-level multicast. The impact of these schemes was analyzed and simulated for three protocols: ELM, ALMI and NICE. Our results show that implementing key management schemes that make use of the ALM overlay for key distribution is feasible. The added overhead is reasonable, but choosing the appropriate scheme depends on the ALM protocol being used and the characteristics (join/multicasting rate) of the group. The group-wide key scheme has the worst impact to join latency, but the best results with respect to data transmission latency. The subgroup scheme lies in between the other two schemes. Data delivery reliability is not affected by the choice of key management scheme.

References

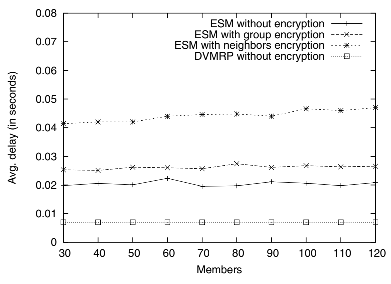
[1] C. Abad, W. Yurcik, and R. H. Campbell. A survey and comparison of end-system overlay multicast solutions suitable for network centric warfare. In *Proc. of SPIE*, vol. 5441, pp. 215–226, Orlando, FL, July 2004.

[2] S. Banerjee. *myns* simulator, Sept. 2002. <`http://www.cs.umd.edu/~suman/research/myns/`>. (Feb. 2005).

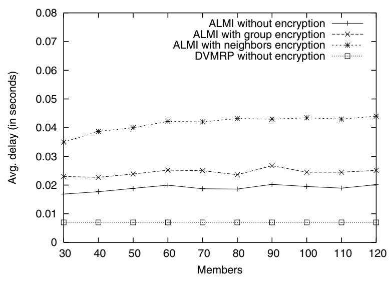
[3] S. Banerjee and B. Bhattacharjee. Scalable secure group communication over IP multicast. *IEEE J-SAC*, 20(8):1511–1527, Oct. 2002.

[4] S. Banerjee, B. Bhattacharjee, and C. Kommareddy. Scalable application layer multicast. In *Proc. of ACM SIGCOMM*, pp. 205–217, Pittsburgh, PA, Aug. 2002.

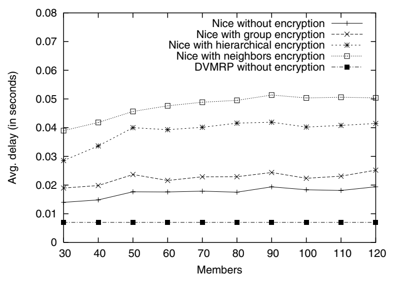
[5] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. Wallach. Secure routing for structured peer-to-peer overlay networks. In *Proc. of Usenix Symp. on Operating Sys. Design and Impl. (OSDI)*, pp. 299–314, Boston, MA, Dec. 2002.



(a) ESM

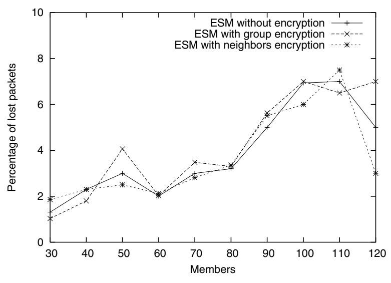


(b) ALMI

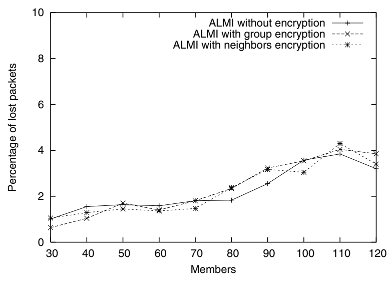


(c) NICE

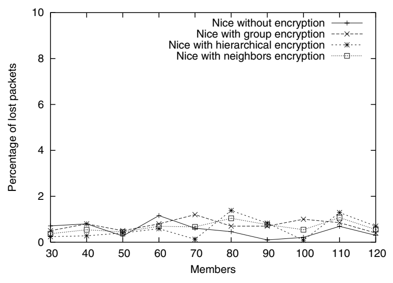
Figure 6. Multicast data delivery average latency, for all group members, across all runs



(a) ESM



(b) ALMI



(c) NICE

Figure 7. Percentage of lost packets (0% loss if all multicasted data reaches all members)

[6] Y.-H. Chu, S. G. Rao, S. Seshan, and H. Zhang. A case for end system multicast. *IEEE J-SAC*, 20(8):1456–1471, Oct. 2002.

[7] W. Dai. Crypto++ v5.1, Aug. 2003. <`http:\\www.cryptopp.com`>. (Feb. 2005).

[8] J. R. Douceur. The Sybil attack. In *Proc. of Intl. Workshop on Peer-to-Peer Sys. (IPTPS)*, pp. 251–260, Cambridge, MA, Mar. 2002. Springer-Verlag, LNCS 2429.

[9] M. J. Freedman and R. Morris. Tarzan: A peer-to-peer anonymizing network layer. In *Proc. of ACM Conf. on Comp. and Comm. Seccurity (CCS)*, pp. 193–206, Washington, DC, Nov. 2002.

[10] W. Josephson, E. Sirer, and F. Schneider. Peer-to-peer authentication with a distributed single sign-on service. In *Proc. of Intl. Workshop on Peer-to-Peer Sys. (IPTPS)*, pp. 250–258, San Diego, CA, Feb. 2004. Springer-Verlag, LNCS 3279.

[11] X. Li, Y. Yang, M. Gouda, and S. Lam. Batch rekeying for secure group communications. In *Proc. of Intl. World Wide Web Conf. (WWW)*, pp. 521–534, Hong Kong, May 2001.

[12] L. Mathy, N. Blundell, V. Roca, and A. El-Sayed. Impact of simple cheating in application-level multicast. In *Proc. of IEEE INFOCOM*, vol. 2, pp. 1318–1328, Hong Kong, Mar. 2004.

[13] A. Nicolosi and D. Mazières. Secure acknowledgment of multicast messages in open peer-to-peer networks. In *Proc. of Intl. Workshop on Peer-to-Peer Sys. (IPTPS)*, pp. 233–248, San Diego, CA, Feb. 2004. Springer-Verlag, LNCS 3279.

[14] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel. ALMI: An application level multicast infrastructure. In *Proc. of Usenix Symp. on Internet Technologies and Sys. (USITS)*, pp. 49–60, San Francisco, CA, Mar. 2001.

[15] O. Rodeh, K. P. Birman, and D. Dolev. A study of group rekeying. Tech. Rep. TR2000-1791, Dept. of C.S., Cornell University, Ithaca, NY, Mar. 2000.

[16] S. Setia, S. Koussih, S. Jajodia, and E. Harder. Kronos: A scalable group re-keying approach for secure multicast. In *Proc. of IEEE Symp. on Security and Privacy (S&P)*, pp. 215–229, Oakland, CA, May 2000.

[17] N. Weiler. Semsomm – A scalable multiple encryption scheme for one-to-many multicast. In *Proc. of IEEE Intl. Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, pp. 231–236, Cambridge, MA, June 2001.

[18] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee. How to model an internetwork. In *Proc. of IEEE INFOCOM*, vol. 2, pp. 594–602, Kobe, Japan, Mar. 1996.