A taxonomy of multicast security issues (updated version)

<draft-irtf-smug-taxonomy-01.txt>

Status of this Memo

1. Abstract

With the growth and commercialization of the Internet, the need for secure IP multicast is growing. In this draft we present a taxonomy of multicast security issues. We first sketch some multicast group parameters that are relevant to security, and outline the basic security issues concerning multicast in general, with emphasis on IP multicast. Next we suggest two `benchmark' scenarios for secure multicast solutions. Lastly we review some previous works.

Table of Contents:

1. Abstract i
2. Introduction 1
3. A Taxonomy of multicast security issues..... 2
3.1 Multicast group characteristics..... 2
3.2 Security requirements and trust issues..... 3
3.3 Performance parameters..... 5
4. Benchmark Scenarios..... 5
4.1 Single source broadcast..... 6
4.2 Virtual Conferences..... 7
5. A mini-survey of related work..... 7
5.1 Works on group key management..... 8
5.2 Works on individual authentication.....10
5.3 Works on membership revocation.....11
5.4 Working prototypes.....13
5.5 Architectures.....13
5.6 Fighting piracy.....14
Acknowledgments.....15
References.....16
Authors address.....18

2. Introduction

In addition to traditional unicast communication, the Internet Protocol supports a multicast mode where a packet is addressed to a group of recipients. The main motivation behind this mode is efficiency, both in sender resources (one transmission serves all recipients) and in network resources (far less traffic). The main challenge in efficient multicast transmission is routing: how to get a packet to its intended recipients with minimal latency and bandwidth consumption. See work done at the MBONED and IDMR working groups. Reliable multicast is being studied in the IRTF Reliable Multicast working group.

The growth and commercialization of the Internet offers a large variety of scenarios where multicast transmission will greatly save in bandwidth and sender resources. Immediate examples include news feeds and stock quotes, video transmissions, teleconferencing, software updates, and more. (See [Quinn] for a more complete survey on multicast applications.) Yet, multicast transmission introduces security concerns that are far more complex than those of simple unicast. Even dealing with the `standard' issues of message and source authentication and secrecy becomes much more complex; in addition other concerns arise, such as access control, trust in group centers, trust in routers, dynamic group membership, and others.

Security solutions should mesh well with current multicast routing protocols, and should have as small overhead as possible. In particular, a realistic solution must maintain the current way by which {\em data packets} are being routed; yet additional control messages may be introduced, for key exchange and access control. These messages need not necessarily be sent via multicast.

As a first step towards a workable solution, we present a taxonomy of multicast security concerns and scenarios, with a strong emphasis on IP multicast. First we list multicast group characteristics that are relevant to security. Next we list security concerns and some trust issues. We also discuss important performance parameters.

It soon becomes clear that the scenarios are so diverse that there is little hope for a single security solution that accommodates all scenarios. Thus we suggest two `benchmark' scenarios for multicast security solutions. One scenario involves a single sender (say, an on-line stock-quotes distributor) and a large number of passive recipients (say, hundreds of thousands). The second scenario depicts relatively small interactive groups of up to few thousands of participants.

Lastly we present a brief survey of existing work on multicast security. (The authors apologize in advance for any misinterpretations and omissions. Please write and complain. They will be happy to update and correct the draft.) Two main issues emerge, where the performance of current solutions leaves much to be desired:

- Source authentication: How to make sure that information is arriving unmodified from a particular group member (as opposed to information coming from "one of the group members").

- Membership revocation: How to prevent a leaving member from future access to the group resources.

3. A Taxonomy of multicast security issues

3.1 Multicast group characteristics

We list salient parameters of multicast groups. These parameters crucially affect the security architecture that should be used.

Group size: Can vary from several tens of participants in small discussion groups, through thousands in virtual conferences and classes, and up to several millions in large broadcasts.

Member characteristics: These include computing power (do all members have similar computing power or can some members be loaded more than others?) and attention (are members on-line at all times?).

Membership dynamics: Is the group membership static and known in advance? Otherwise, do members only join, or do members also leave? how frequently does membership change and how fast should changes be updated? Are membership changes bursty?

Expected life time: Is the group expected to last several minutes? days? an unbounded amount of time?

Number and type of senders: Is there a single party that sends data? several such parties? all parties? Do few senders generate most of the traffic? Is the identity of the senders known in advance? Are non-members expected to send data to the group?

Volume and type of traffic: Is there heavy volume of communication? Must the communication arrive in real-time? what is the allowed latency? For instance, is it data communication (less stringent real-time requirements, low volume), audio (must be real-time, low volume), or video (real-time, high volume)?

3.2 Security requirements and trust issues

We list several security requirements and trust-related concerns. Not all issues are relevant to all multicast applications; yet they should be kept in mind when designing a system.

Group management and access control: Making sure that only registered and legitimate parties have access to the communication addressed to the group. (Sometimes it may be necessary also to allow only group members to send data to the group.) Sometimes this is enforced by having a group key that is known only to group members; other, more hierarchical solutions exist as well. Here several security concerns are involved:

* How to authenticate potential group members
* How to securely distribute the group key(s).
* How to revoke membership of leaving members
* How to prevent joining members from access to past group communication.
* How to periodically refresh the group key(s).
* How to log information and allow for external auditing.

Ephemeral secrecy: Preventing non group-members from having easy access to the transmitted data. Here a mechanism that only delays access, or prevents access only to crucial parts of the data may be sufficient. (For instance, to maintain ephemeral secrecy when transmitting a video it is sufficient to encrypt only the low-order Fourier coefficients in an MPEG encoding.) Ephemeral secrecy is often sufficient to protect multicasted contents, in cases where the content itself is not confidential.

Long-term secrecy: Making sure that the data remains secret to non-group members, for a substantial amount of time after transmission. This may often not be a requirement for multicast traffic. In particular, the larger the multicast group the weaker the secrecy assurance is (even if the cryptography is perfect).

Forward Secrecy: Making sure that encrypted data remains secret even if the key is compromised (either by cryptanalysis or by break-in) at a later date. This requirement is needed only for applications that require long-term secrecy. Thus in many multicast applications it is not necessary.

Sender and data authenticity: Making sure that the received data originates with the claimed sender and was not modified on the way. Authenticity takes two flavors: Group authenticity means that a group member can recognize whether a message was sent by a group member. Source authenticity means that it is possible to identify the particular sender within the group. It may also be desirable to verify the origin of messages even if the originator is not a group member.

Anonymity: Several flavors are possible. One is keeping the identity of group members secret from outsiders or from other group members. Another is keeping the identity of the sender of a message secret. A related concern is protection from traffic analysis.

Non-repudiability: This refers to the ability of receivers of data to prove to third parties that the data has been transmitted, together with the source. Non-repudiability is somewhat contradictory to anonymity, and it is not clear whether it should be implemented in an IP-layer protocol.

Service availability: Maintaining service availability against malicious attack is ever more challenging in a multicast setting, since clogging attacks are easier to mount and are much more harmful.

3.3 Performance parameters

We list relevant performance parameters. Relative importance of these parameters may vary from application to application. These parameters should always be measured against the degree of security achieved.

Latency, bandwidth and work overhead per data packets. These are the most immediate costs and should typically be minimized at the highest priority. Here distinction should be made between the load on strong server machines and on weak end-users. An additional important parameter here is the amount of buffering needed at the sending side and at the receiving side, both in terms of required space and in terms of packet delay.

Latency, bandwidth and work overhead per control packets. These are typically less frequent, thus efficiency here is somewhat less crucial. The control messages usually deal with key distribution and refreshment.

Group initialization, and member addition and deletion overheads. Group initialization occurs once. In groups with highly dynamic membership, efficient addition (and especially deletion) of members may be an important concern.

Sender initialization, the overhead of a sender when it starts transmitting to the group.

Congestion control, especially around centralized control services at peak sign-on and sign-off times. (A quintessential scenario is a real-time broadcast where many people join right before the broadcast begins and leave right after it ends.)

Resume overhead: The work incurred when a group member becomes active after being dormant (say, off-line) for a while.

4. Benchmark Scenarios

As seen above, it takes many parameters to characterize a multicast security scenario, and a large number of potential scenarios exist. Different scenarios call for different solutions; it seems unlikely that a single solution will accommodate all scenarios.

We present two very different scenarios for secure multicast, and sketch possible solutions and challenges. These scenarios seem to be the ones that require most urgent solutions; in addition, they span a large fraction of the concerns described above, and solutions here may well be useful in other scenarios as well. Thus we suggest these scenarios as benchmarks for evaluating security solutions.

4.1 Single source broadcast

Here a single source wishes to continuously broadcast data to a large number of passive recipients. The source can be a news agency that broadcasts stock-quotes and news-feeds to paying customers, or possibly a Pay-TV station. We list a number of characteristics:

The number of recipients can be up to hundreds of thousands and more. The source is typically a top-end machine with ample resources. It can also be parallelized or split to several sources in different locations. The recipients are typically lower-end machines with limited resources. Consequently, the security solution must optimize for efficiency at the recipient side.

The life-time of the group is usually long. Yet, the group membership is dynamic: members join and leave at a relatively high rate. In addition, at peak times (say, before and after important broadcasts) a high volume of sign-on/sign-off requests are expected. It can be assumed that members have a long-term relationship with the group; this may facilitate processing of sign-on/sign-off requests.

The volume of transmitted data may vary considerably: if only text is being transmitted then the volume is relatively low (and the latency requirements are quite relaxed); if audio/video is transmitted then the volume can be very high and very little latency is allowed.

Authenticity of the transmitted data is a crucial concern and should be strictly maintained: a client must never accept a forged stock-quote as authentic. In particular, it should not accept a stock quote that originated with any other group member than the specified sender. Another important concern is preventing non-members from using the service. This can be achieved by encrypting the data; yet the encryption may be relatively weak/ephemeral since there is no real secrecy requirement - only prevention from easy unauthorized use.

The required latency of the communication varies from application to application. Member revocation would be performed within minutes or seconds from the time it is requested (but it is typically not required to remove members within fractions of a second)

There is typically a natural group owner that manages access-control as well as key management. However, the sender of data may be a different entity (say, Yahoo! broadcasting Reuters stock-quotes via its home-page).

4.2 Virtual Conferences

Typical virtual conference scenarios may include on-line meetings of corporate executives or committees, town-hall type meetings, interactive lectures and classes, or multiparty video games. A virtual conference involves several tens to hundreds of peers, often with roughly similar computational resources. Usually most, or all, group members may a-priori wish to transmit data (although often there is a small set of members that generate most of the bandwidth).

The group is often formed per event and is relatively short-lived (say, few minutes or hours). Membership is often static: members join at start-up, and remain signed on throughout. Furthermore, even if a member leaves it is often not crucial to cryptographically revoke their group membership. Bandwidth and latency requirements vary from application to application, similarly to the case of single source broadcast. However, latency (and especially sender initialization) should typically be very small in order to facilitate the simultaneity and interactivity of virtual conferences.

Authenticity of data {\em and sender} may be the most crucial security concern. In some scenarios maintaining secrecy of data and anonymity of members may be important as well; in other scenarios secrecy of data is not a concern at all. There is often a natural group owner that may serve as a trusted center. Yet, it is always beneficial to distribute trust as much as possible.

5. A mini-survey of known related work

Following is a short survey of multicast security related work. The authors apologize in advance for any misinterpretations and omissions. Please write and complain. They will be happy to update and correct the draft.
The first three sections of the survey describe work on three main issues described above: group key management, individual authentication, and membership revocation. The last section describes work on prototypes which implement various elements of multicast security.

5.1 Works on group key management

The works below concentrate on establishing and managing a common key among all group members. This key can be used for encryption and group authentication, but is insufficient for individual authentication. Group management is closely related to user revocation methods (see Section 5.3) since it should prevent a leaving group member from further decrypting the group communication.

The GKMP protocol [GKMPA,GKMPS] generates and maintains symmetric keys for the members of a multicast group. In this protocol a multicast group has a dedicated Group Controller (GC) which is responsible for managing the group keys. The GC generates the group keys in a joint operation with a selected group member. Afterwards it contacts each group member validates its permissions, and sends it the group keys (encrypted using a key which is mutually shared between the GC and that member). This approach may have scalability problems since a single entity, the GC, is responsible for sending the keys to all group members.

The Scalable Multicast Key Distribution scheme (SMKD) [Ballardie] is based on the Core-Based Tree (CBT) routing protocol and provides secure join to a CBT group tree in a scalable approach. It utilizes the hard-state approach of CBT in which routers on the delivery tree know the identities of their tree-neighbors. When a CBT group is initiated in this scheme the core of the tree operates as the group controller and generates the group session keys and key distribution keys. As routers join the delivery tree they are delegated the ability to authenticate joining members and provide them with the group key. This approach is highly scalable. However, it is tied to a specific routing protocol, and does not provide a separation between the routing and the security mechanism. (In particular, it puts high trust in the routers, since each router in the delivery tree obtains the same keys as the group controller.)

The MKMP [MKMP] key management protocol enables the initial Group Key Manager to delegate the key distribution authority to other parties in a dynamic way. It first generates the group key. Then it delegates the key distribution ability to selected parties by sending a message to the multicast group soliciting these parties. This message contains keys and access lists which can only be decrypted by the solicited parties. After they obtain this material they can operate as Group Key Managers. This dynamic approach has the advantage that the group topology can be adapted on-line. MKMP uses a single key for the entire group and thus does not require hop-by-hop decryption/re-encryption of the payload.

The Iolus scheme [Mittra] handles the scalability problem by introducing a "secure distribution tree". The multicast group is divided into subgroups which are arranged hierarchically. There is a Group Security Controller (GSC) managing the top-level group, and Group Security Intermediaries (GSIs) for managing the different subgroups. Each subgroup has its own sub-key which is chosen by its manager. A GSI knows the keys of its subgroup and of a higher level subgroup, so it can "translate" messages to/from higher levels. A disadvantage of this approach is the latency incurred by GSIs decrypting and re-encrypting each data packet (although the use of encryption indirection enables this latency to be constant and independent of the packets length). The removal of an untrusted GSI is also complex.

The work of Poovendran et al [PACB] identifies two major drawbacks of the GKMP protocol which result from the use of a single group controller: the group controller is a single point of failure, and its heavy load might affect scalability. It is suggested to use a panel of three controllers, where every two panel members can operate as a group controller. It is furthermore proposed to improve scalability by segmenting the group into clusters, which are each managed by a sub-controller panel.

The Internet draft of Hardjono et al [HCD] suggests a hierarchical framework for group key management, similar to that of Iolus. The network is divided into regions: many "leaf regions" and a single "trunk region" which which is used to connect leaf regions and does not contain any member hosts. Each region can have a different key management protocol. In particular, different leaf regions can have different intra-region group key management protocols, and the trunk region operates an inter-region group key management protocol.

In [HCM] an intra-region group key management protocol is presented in detail. To enable scalability a domain is further divided into areas, where each host belongs to a single area. There is a single Domain-Key-Distributor and many Area-Key-Distributors which are responsible for each area. A host only communicates with the AKD of its area. A key for the members of a multicast group in a domain is generated by the DKD and is propagated to the hosts through the AKD's. This scheme presents an interesting new concept: the group key is common to members in the entire domain, while the control messages for key updates are transferred via the Area-Key-Distributors, using two levels of keys. This method enjoys "the best of two worlds": First, the data packets need not be re-encrypted en-route and can be routed using any multicast routing protocol. Second, the group (or domain) controller need not keep track of all group members; instead, it can keep track only of the AKDs. This facilitates scalability while maintaining independence from the data routing mechanism. Note that this protocol is for managing a multicast group inside a domain (a "leaf" in the terms of [HCD]) whereas a different protocol can be used for inter-domain ("trunk") key management of the group.

Kruus [Kruus] focuses on identifying and surveying security related issues for multicast group key management. The paper describes several approaches for group key management and for user revocation.

Banerjee and Bhattacharjee [BB] suggest using spatial clustering of group members in order to scale rekeying and other multicast based applications. The group members are divided to clusters, which each have a cluster leader. In a higher level, the cluster leaders are divided to clusters, which have their own leaders, and so on. An analysis of the rekeying algorithm shows that the amortized cost of rekeying is constant.

The previous schemes have a single group controller (GC), which is a single point of failure, or otherwise use several GCs in a way which compromises the security of the whole group, or part of the group, if any one of the GCs is broken into. Alternatively, one could use a distributed pseudo-random system [NPR] which uses several servers to produce keys. The system has n servers, and a host must contact k servers in order to obtain a key. The system ensures that two or more hosts, that need to obtain the same key, learn the same key value even if they contact different, non-intersecting, sets of k servers. The system provides better security in the sense that an adversary must break into at least k servers in order to learn keys.

Rodeh et al. [Rodeh] introduced algorithms for management of group-keys in group-communication systems. Unlike prior work, based on centralized key-servers, this solution is completely distributed

and fault-tolerant and its performance is comparable to the centralized solution.

5.2 Works on individual authentication

In order to authenticate that a message was sent by one of the group members it is sufficient to use Message Authentication Codes (MACs, see e.g. [HMAC]) with a single shared key known to all group members. However, this method does not suffice to enable individual authentication, i.e. it cannot be used to authenticate a message as originating from a specific party.

Source authentication can be achieved if the sender of the message signs it using a digital signature scheme. However, the computational complexity of computing and verifying digital signatures, as well as the length of the signature, is significant. RSA signatures might be an appealing choice of a signature scheme since it is possible to use them in a mode which considerably reduces the running time of the verification algorithm. (Furthermore, the use of Batch RSA [Fiat] enables the source to sign many messages in parallel, with a computational overhead which is not considerably larger than signing a single message. The sender should, however, know all the messages that should be signed before it can sign the first message.)

It is possible to use signature schemes based on elliptic curves, which are very efficient both in processing time and in bandwidth. Another interesting approach is to use on-line/off-line signature schemes. These enable the signer to perform most of its computation off-line, even before it learns the message that it should sign. When this message becomes known the signer only has to perform a very efficient computation in order to complete the signature.

The schemes of Gennaro and Rohatgi [GR] enable to efficiently sign streams of data. Basically, the idea is to partition data packets into chains. Each data packet includes a hash of the next packets in the chain, and then only the first packet in the chain needs to be signed. There are two types of schemes, for data streams which are available off-line (and therefore the whole stream can be examined by the source before it sends the first packet), and for real-time data. A major drawback of the suggested schemes is that they do not deal well with unreliable communication channels and might therefore not be suitable for large scale multicast groups. Furthermore, the scheme for real-time data introduces a considerable communication overhead per packet.

Wong and Lam [WL] address the problem of source authentication in the presence of unreliable communication. Their scheme allows a receiver to individually authenticate each packet. The idea is to let the

sender buffer a number of packets; once enough packets are buffered the sender computes a hash-tree of the packets (a la Merkle [Merkle]), and signs the root. This signature is attached to each packet, together with the appropriate hashes that allow verification of the packet independently of other packets. It is also suggested that signatures will be performed using a variant of Fiat-Shamir signatures, which (using some heuristics) are more efficient than other common signature schemes.

Rohatgi [Rohatgi] describes a scheme that gets rid of buffering altogether, even at the sender side. The idea is to prepare and sign the hash tree in the [WL] scheme ahead of time, where the leaves correspond to public keys of one-time signatures a la Lamport. This way, each packet can be signed (using the one-time signature scheme) and sent without delay. Similarly, each packet can be verified upon arrival independently of other packets. The main drawback of that scheme is the size of the signature, which is up to 270 byes per packet.

Another approach to making the [GR] scheme resilient to packet loss was taken by Perrig et. al. [PCTS]. (A similar idea appears in [G].) First, they let the hash of each packet appear in the next packet (rather than in the previous one). Then, they include the hash of each packet in several additional packets "down the stream" from the hashed packet. This provides resilience to packet loss, provided that the choice of packets that contain the hash of each packet is a good one. They also suggest other optimizations that reduce the bandwidth overhead of their scheme.

Building on previous work [FN1,DFFT], Canetti et al [CGIMNP] suggest individual authentication schemes which are based on efficient MACs rather than on public key signatures. These schemes are designed to be secure against coalitions of up to k of group members, where k is a parameter which affects the overhead. To explain the approach, let us present a simplified example: The idea is to use some number, n, of MAC keys. The pre-designated sender has all keys, where each one of the receivers has n/2 keys, chosen at random from the n keys. Now, each message is MACed with each one of the n keys, and a recipient verifies the MACs whose keys it knows. A coalition of bad parties can make some `victim' accept forged messages only it the coalition knows all the MAC keys that the victim knows. The parameters are set so that the probability that such a bad event occurs is small.

Yet another approach to providing source authentication uses only symmetric cryptography, more specifically on message authentication codes (MACs), and is based on delayed disclosure of keys by the sender. The idea, common to all above schemes, is to have the sender attach to each packet a MAC computed using a key k known only to itself. The receiver buffers the received packet without being able to authenticate it. If the packet is received after some ``deadline'', it is discarded. A short while later, the sender discloses k and the receiver is able to authenticate the packet. (See more details within.) Consequently, a single MAC per packet suffices to provide source authentication, provided that the sender has synchronized its clock with the sender ahead of time.

This technique was first used by Cheung [Cheung] in the context of authenticating communication among routers. It was then used in the Guy Fawks protocol [Fawks] for interactive unicast communication. In the context of multicast streamed data it was proposed by several authors [BCC,Briscoe,PCTS]. An Internet Draft based on the TESLA scheme of [PCTS] was recently written [PCBST].

5.3 Works on membership revocation

In order to prevent new group members (respectively, leaving members) from accessing data sent before they joined (respectively, after they leave), the group controller needs to change a multicast group key whenever membership in the group changes. While it is rather straightforward to efficiently update the group key when a new member joins the group, this is not the case when a member is removed from the group since this member already knows the group key.

The approach taken in many group key management protocols [GKMPA,SMKD,MKMP] to remove untrusted members is to generate a new group key and send it independently to each of the remaining group members (using secret keys which are shared between each of the members and the group controller), thus essentially creating a new multicast group without the untrusted member. This approach is non-scalable.

As discussed in Section 5.1, an alternative approach is to divide the multicast group to subgroups with independent subgroup keys. When a member is removed it is only required to send individually encrypted messages to members of the subgroup of the removed member. This approach, taken in [Mittra,PACB,HCD,HCM], is more scalable. It requires that each subgroup contains a trusted controller (e.g. the Group Security Intermediary in the Iolus system of [Mittra]). If this party becomes untrusted then a more complex revocation procedure (which is not described in these drafts) should be run. (Note that the scheme of [HCM] suggests that subgroups (domains) are further partitioned to smaller subgroups (areas) with their own controllers, to provide better scalability).

Broadcast encryption [FN] is a scheme to encrypt messages from a single source to a dynamically changing group of recipients. When a member is leaving the scheme can be used to send the new group key to the remaining members. The scheme uses a parameter k which is the maximum tolerable size of a corrupt coalition of former group members that might try to learn a key they should not get. The overhead of the scheme depends on the maximum number of potential members in the group, and the maximum size of the corrupt coalition, but not on the number of members which are removed. Therefore overhead is better than in other user revocation methods if the number of leaving/joining members is large. The scheme is based on using a set of keys and applying a clever method of assigning subsets of these keys to group members. This assignment makes sure that for every corrupt coalition of k users it is possible to encrypt a message such that the keys known to its coalition members do not suffice for decryption, whereas the keys of any other member do.

Wallner et al [WHA] (and, independently, [WGL]) introduce a scalable, tree based user revocation scheme. For a group of n members there is a total of 2n keys but each member is only required to store log(n) keys. When a group member is removed the group controller sends a single message of size 2log(n) to all members, and each member performs log(n) (rather efficient) computations in order to generate the new group key. The removed member cannot compute the new group key even if it receives this message. The basic idea of the scheme is to imagine the users as the leaves of a binary tree, assign a key to each node, give each user the keys in the path from its leaf to the root and use the root key as the group key. When a user is removed all the keys it holds are replaced. Two drawbacks of the scheme are that it requires the center to keep track of 2n keys, and requires each member to receive and process each member-revocation message in order to learn the current group key.

The scheme of [WHA] was generalized by [WGL] for trees of arbitrary degree. It is possible to reduce the length of the member revocation message that the controller broadcasts to only log(n) encryptions (instead of 2log(n) in [WHA]). Different schemes that achieve this property are presented in [CGIMNP] and in [MS]. The scheme of [MS] affects the broadcast overhead of the member join operation, it increases it from O(1) in the scheme of [WHA] to log(n). The scheme of [CGIMNP] does not increase the overhead of member join. The security of the scheme of [CGIMNP] can be rigorously proven based on the security (i.e. pseudorandomness) of the cryptographic function that is used.

A member revocation scheme of a different flavor is suggested in [NP1]. It enables to revoke the keys of up to k members, and is secure against a coalition of all the revoked members. Th personal key of each member is of constant size, and the revocation message is of size O(k). The main idea is to give each member a personal key which is a share of a (k+1)-out-of-n secret sharing scheme. When k members have to be removed their keys are broadcast and the new group key is set to be the secret. Each other member has k+1 shares and can reveal the secret, whereas the revoked members have nothing but their k shares. The scheme was generalized to multiple revocations and to interleaving with traitor tracing.

5.4 Working prototypes

A prototype of the Iolus system has been implemented [Mittra]. It uses a client application which interfaces between applications and the Iolus GSC/GSIs. It is claimed there that the basic prototype is rather a simple to implement and to use. There is only small penalty for the decryption/encryption process of a GSI, and this penalty does not depend on the size of the payload. Note however that the Iolus system does not provide any individual authentication mechanism.

A toolkit for secure internet multicast is described in [CEKPS]. It emphasizes a separation between control and data functions. This enables applications to have fine grain control over the data path, while keeping the control plain transparent to the applications. The toolkit can operate without end-to-end support for multicast, using data reflectors connected via unicast tunnels. It is written in Java. Similar to Iolus, a multicast group is divided to subgroups (domains), however the toolkit offers better flexibility, supports individual authentication (by using digital signatures), and operates over non-multicast enabled backbones.

5.5 Architectures

Several works address the architectural issues involved in the key management aspect of secure multicast [HCD,GMKPA.GMKPS,WHA,WGL]. These works have been described above.

In [CCPRRS] a host architecture for secure IP multicast protocols is suggested. The architecture is based on the IPSEC architecture [Ken98] and tries to use IPSEC components (IKE, AH, ESP) as much as possible. As in IPSEC, the architecture calls for separation of the key management (to take place in the application layer) from the data transformations (to take place mostly in the IP layer). Yet, an additional data-processing module is added in the application/UDP layer. This module is responsible for source authentication, which is not taken care of by the IPSEC transformations. The suggested

architecture is flexible and can adopt many of the key management protocols described above.

5.6 Fighting piracy

Secure multicast is used to ensure the secrecy of the communication inside the multicast group. However nothing prevents one of the legitimate members of the group from helping other, illegitimate parties to receive the messages that are sent to the group. This problem is well known in the context of television broadcasting, and is commonly referred to as "piracy". There has been a considerable amount of work on this subject, motivated by the pay-TV industry. It is mostly relevant to the one-to-many multicast scenario.

The fight against piracy consists of two stages: (1) tracing the pirates, and (2) taking action to prevent them from further operation. We describe below some methods which can be used for tracing. Once the source of piracy is detected it is possible to prevent it from further decryption using the user revocation schemes we discussed above. It might be also desirable to take legal actions against the pirates.

There are two methods by which pirates can operate. They can either

(1) distribute decryption keys that enable to decrypt the group communication, i.e. perform illegal "key distribution", or
(2) decrypt the communication themselves and distribute the decrypted content, i.e. perform illegal "content distribution".

The content distribution attack is harder for large scale implementation and easier to detect, since the pirates should essentially operate their own broadcast station. The key distribution attack is therefore preferable to the pirates.

In order to prevent illegal key distribution pay-TV providers provide decryption keys in smartcards which are supposed to be "tamper proof". That is, it should be hard to extract from a key from a smartcard. However, most smartcards have weaknesses which enable to extract the keys they contain (see e.g. [C]).

It is easier to trace the source of a key distribution attack than that of a content distribution attack. The reason being that it is possible to give each user a somewhat different key (and then, given a pirate decoder, recognize which keys were used to construct it). It is a much harder task to distribute to each user a different copy of the data such that each copy would seem to have the same content, but a pirate copy would identify its source (even if the content was passed through different transformations, e.g. lossy compression, and was generated from several legal copies). This task is known as watermarking, and is discussed below.

Fighting illegal key distribution:

A obvious method to trace the source of keys is to give each user a personal key. The content should be encrypted with a random key, which is itself encrypted with each of the personal keys. The drawback of this scheme is that it sends an additional encryption per user, and this overhead is only reasonable for small groups. There are more advanced methods whose overhead is much smaller [CFN,NP,CFNP,BF]. These methods are secure as long as the pirates use less than k keys, where k is a parameter.

Fighting illegal content distribution:

In order to trace the source of the content it is essential to insert in it "water marks" which the pirates cannot remove. See [CKLS,PAK] for a discussion of marking methods. An additional issue is the distribution of marks into the copies that are delivered to users, i.e the problem of efficiently generating unique fingerprints. An efficient fingerprinting method is suggested in [BS] (however, fingerprinting schemes are less efficient than schemes for tracing illegal key distribution).

Self enforcement is an interesting concept: it intends to prevent piracy (rather than trace its source) by discouraging legitimate users from giving their keys to others. In order to achieve this property each user's key contains some information which is private to the user (for example, his credit card number). It is hoped for that users would be discouraged from providing keys which contain this information to others. Efficient tracing schemes are described in [DLN].

Acknowledgments

Much of this text is reproduced from [CGIMNP], written with Juan Garay, Gene Itkis, Daniele Micciancio and Moni Naor. The authors are grateful to the people with whom they interacted on this topic, including all the above and in addition Naganand Doraswamy, Rosario Gennaro, Dan Harkins, Shai Halevi, Dimitris Pendarakis, Tal Rabin, Pankaj Rohargi and Debanjan Saha.

References

[Fawks] R. Anderson, F. Bergadano, B. Crispo, J. Lee, C. Manifavas, R. Needham, "A new family of authentication protocols", Operating Systems Review, 32(4):9--20, 1994.

[Ballardie] Ballardie A., "Scalable Multicast Key Distribution", RFC 1949, May 1996.

[BB] S. Banerjee. S. Bhattacharjee, "Spatial Clustering for IP Multicast: Algorithms and an Application", Technical Report, CS-TR 4177, Department of Computer Science, University of Maryland, College Park, August 2000.

[BCC] F. Bergadano, D. Cavalino, B. Crispo, "Individual Single Source Authentication on the Mbone", ICME 2000. A talk containing this work was given at IBM Watson, August 1998.

[BF] Boneh D.,Franklin M.. "An Efficient Public Key Traitor Tracing Scheme", CRYPTO 1999: 338-353

[BS] Boneh D., Shaw D., "Collusion-Secure Fingerprinting for Digital date", IEEE Tran. on Information Theory, Vol 44, No. 5, pp. 1897-1905, 1998.

[Briscoe] B. Briscoe, "MARKS: Zero side-effect multicast key management using arbitrarily revealed key", First International Workshop on Networked Group Communication, 1999. On-line version at http://www.labs.bt.com/people/briscorj/ papers.html#MARKS.

[C] Dan Boneh, Matthew K. Franklin: An Efficient Public Key Traitor Tracing Scheme. CRYPTO 1999: 338-353 http://www.cryptography.com/dpa/qa/index.html

[CCPRRS] Canetti R., Cheng P. C., Pendarakis D., Rao, J., Rohatgi P., Saha D., "An architecture for secure IP multicast", manuscript.

[CGIMNP] Canetti R., J. Garay, G. Itkis, D. Micciancio, M. Naor, B. Pinkas, "Multicast Security: A Taxonomy and Efficient Authentication", to be presented at INFOCOM '99.

[CEKPS] Chang I., R. Engel, D. Kandlur, D. Pendarakis, D. Saha, "A Toolkit for Secure Internet Multicast", manuscript, 1998.

[Cheung] S. Cheung,"An efficient message authentication scheme for link state routing", in 13th Annual computer Security Aplications Conference, 1997.

[CFN] Chor B., Fiat A., Naor M., "Tracing Traitors", Proc. Advances in Cryptology - Crypto '94, Springr-Verlag LNCS 839 (1994), 257--270.

[CFNP] Chor B., Fiat A., Naor M., Pinkas B., "Tracing Traitors", IEEE Transactions on Information Theory, Vol. 46, No. 3, May 2000.

[CKLS] Cox I., Kilian J., Leighton T., Shamoon T., "A Secure, Robust Watermark for Multimedia", Information Hiding Workshop, Cambridge, UK, Springer-Verlag LNCS 1174, (1996), 185--206.

[DLN] Dwork C., Lotspiech J., Naor M., "Digital Signets: Self-Enforcing Protection of Digital Information", 28th Symposium on the Theory of Computation (1996), 489--498.

[DFFT] Dyer M., T. Fenner, A. Frieze, A. Thomason, "On Key Storage in Secure Networks", J. of Cryptology, Vol. 8, 1995, 189-200.

[EGM] Even S., O. Goldreich, S. Micali, "On-line/off-line digital signatures", Advances in Cryptology - Crypto '89, Springer-Verlag LNCS 435, pp. 263-277, 1990.

[Fiat] Fiat A., J. of Cryptology 10:75-88 (1997).

[FN] Fiat A., M. Naor, "Broadcast Encryption", Adv. in Cryptology - Crypto '92, Springer-Verlag LNCS 839, pp. 257-270, 1994.

[FN1] Fiat A., M. Naor, unpublished work. Appears in Alon N., "Probabilistic Methods in Extremal Finite Set Theory", in "Extremal Problems for Finite Sets", 1991, 39-57.

[GR] Gennaro R., P. Rohatgi, "How to Sign Digital Streams", Advances in Cryptology - Crypto '97, Springer-Verlag LNCS 1294, pp. 180-197, 1997.

[G] P. Golle, "Authenticating Streamed Data in the Presence of Random Packet Loss", manuscript, Jan. 2000.

[HCD] Hardjono T., Cain B., Doraswamy N., "A Framework for Group Key Management for Multicast Security", internet draft draft-ietf-ipsec-gkmframework-00.txt, July 1998.

[HCM] Hardjono T., Cain B., Monga I., "Intra-Domain Group Key Management Protocol", internet draft, draft-ietf-ipsec-intragkm-00.txt, November 1998.

[MKMP] Harkins D., N. Doraswamy, "A Secure, Scalable Multicast Key Management Protocol (MKMP)".

[GMKPA] Harney, H., C. Muckenhirn, "Group Key Management Protocol (GKMP) Architecture", RFC 2094, July 1997.

[GKMPS] Harney, H., C. Muckenhirn, "Group Key Management Protocol (GKMP) Specification". RFC 2093, July 1997.

[Ken98] Stephen Kent, Randall Atkinson, "Security Architecture for the Internet Protocol", Internet Draft draft-ietf-ipsec-arch-sec-07.txt, July 1998.

[HMAC] Krawczyk H., M. Bellare, R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, April 1997.

[Kruus] Kruus P., "A Survey of Multicast Security Issues and Architectures," 21st National Information Systems Security Conf., Arlington, VA, October 5-8, 1998.

[MS] McGrew D. A., and Sherman A. T., "Key Establishment in Large Dynamic Groups using One-way Function Trees", submitted to IEEE Trans. on Software Engineering. 1998.

[Merkle] Merkle R. C., "A Digital Signature based on a Conventional Encryption Function", Crypto '87.

[Mittra] Mittra S., "Iolus: A Framework for Scalable Secure Multicast". In Proceedings of ACM SIGCOMM '97, Cannes, France, September 1997.

[NP] Naor M., Pinkas B., "Threshold Traitor Tracing", Crypto '98. http://www.wisdom.weizmann.ac.il/~bennyp/PAPERS/ttt.ps

[NP1] Naor M., Pinkas B., "Efficient Trace and Revoke Schemes", Proceedings of Financial Crypto '2000, Anguilla, February 2000.

[NPR] Naor M., Pinkas B., Reingold O., "Distributed Pseudo-Random Functions and KDCs", Advances in Cryptology - Eurocrypt '99 Proceedings, LNCS 1592, Springer-Verlag, pp. 327-346, April 1999.

[PCBST] A. Perrig, R. Canetti, B Briscoe, D. Tygar, D. Song, "TESLA: A source authentication mechanism", draft-irtf-smug-tesla-00.txt, Aug. 2000.

[PCTS] A. Perrig, R. Canetti, D. Tygar, D. Song, "Efficient Authentication and Signature of Multicast Streams over Lossy Channels", 2000 IEEE Symposium on Security and Privacy, Oakland, CA, May 2000.

[PAK] Petitcolas F., Anderson R., Kuhn M., "Information hiding - a survey", Proceedings of the IEEE, 87(7):1062--1078, July 1999.

[PACB] Poovendran R., Ahmed S., Corson S., Baras J., "A Scalable Extension of Group Key Management Protocol", Proceedings of the 2nd ATIRP Conference, University of Maryland, College Park, MD, Feb. 2-6, 1998.

[Quinn] Bob Quinn, "IP Multicast Applications: Challenges and Solutions", draft-quinn-multicast-apps-00.txt, Nov 1998.

[Rodeh] O. Rodeh, K. Birman, D. Dolev, "Optimized Group Rekey for Group Communication systems", Network and Distributed System Security 2000, Febuary, San Diego, California.

[Rohatgi] P. Rohatgi, "A Compact and Fast Signature Scheme for Multicast Packet Authenticatio". In 6th ACM Computer and Communications Security Conference (CCS) , Nov 1999.

[WHA] Wallner D. M., E. G. Harder, R. C. Agee, "Key Management for Multicast: Issues and Architecture", internet draft draft-wallner-key-arch-01.txt, September 1998.

[WGL] Wong C. K., Gouda M., Lam S. S., "Secure Group Communication Using Key Graphs", SIGCOMM '98. Also, University of Texas at Austin, Computer Science Technical report TR 97-23.

[WL] Wong C. K., Lam S. S., "Digital Signatures for Flows and Multicasts", IEEE ICNP '98. Also, University of Texas at Austin, Computer Science Technical report TR 98-15.

Authors' Addresses:
==========

Ran Canetti
IBM TJ Watson Research Center
POB. 704, Yorktown Heights,
Tel. 1-914-784-7076
canetti@watson.ibm.com

Benny Pinkas
STAR Lab
InterTrust Technologies
4750 Patrick Henry Drive
Santa Clara, CA 95054-1851
bpinkas@intertrust.com