Key Management for Secure Internet Multicast using Boolean Function Minimization Techniques

Class Presentation:
Dingbang Xu

Outline

- Introduction
- Related Work Overview
- Key Management Scheme
- Properties and Performance Analysis
- Conclusion and Future Work

Why do we need secure multicast?

- The internet today supports a basic form of multicast service. The multicast group is a open group.
- Support for privacy and authentication in multicast distribution can be useful (pay-per-view of digital media, pay-per-use games and multicast distribution of stock market information).

How to add secure services to multicast?

- Secure Multicast Group
  - Group Controllers (trusted servers).
  - Group members.
- Join to a group

(request, credentials' check, provide keys, etc.)

- De-registration (removal of a member)

Problems related to member removal

- A complex scalability problem
- An example:
  - n members, sharing a session key.
  - One member has to be removed
  - Session key be changed and sent to n-1 members.
  - Communicating new session key in scalable and secure fashion (It is NOT a trivial task).

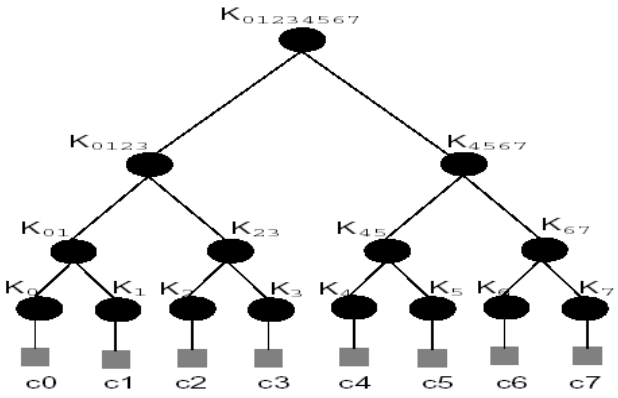
A Simple Solution

- Use separate secure unicast connection from the controller to EACH remaining member.
- Requirement: Each client share a unique key with the controller.
- Poor scalability:
  - n-1 secure unicast connections
  - n secret keys

Related Work

- GKMP
- SMKD with CBT
- Mittra et al. : Dividing a large group into multiple subgroups organized in a multi-level hierarchy.
- Wong et al. : uses a hierarchy of keys.
  - a key update requires O(logN) messages
  - Each client maintains a key ring of O(logN) keys
  - The controller manages a tree of O(N) keys.

Key hierarchy (by Wong et al.)



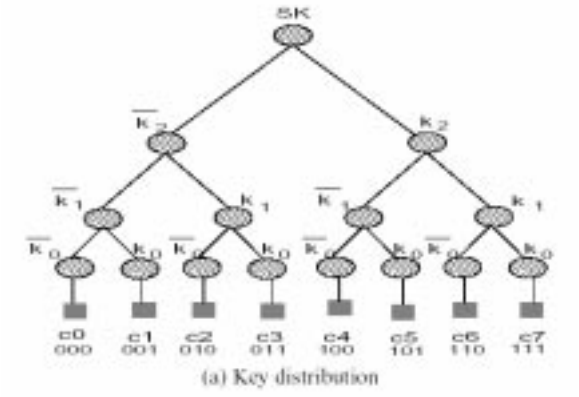
The approach in this paper

- Dynamically generate the most suitable key hierarchy by composing different keys.
- How to deal with member departure:
  - Individual departure
  - Cumulative member removal: many members departure (This paper's focus)
  - The controller only maintain O(logN) keys

Key Management Scheme

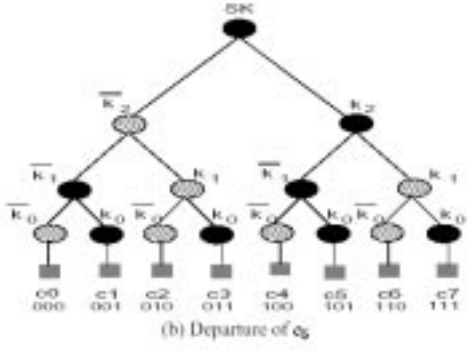
- Each member: a unique user ID (UID)
  - UID=$X_{n-1}X_{n-2}\ldots X_0$ (total number of members may be $N=2^n$)
  - X_i be $1(x_i)$ or $0([x_i])$ ($[x_i]$ means the complement of x_i : x_i bar)
- Common Session Key: SK
- r: current round number
- One member has n auxiliary keys: $K_{n-1},K_{n-2}\ldots,K_0$
  - K_i be $k_i(X_i=1)$ or $[k_i](X_i=0)$ (k_i and $[k_i]$ are not complements of each other), e.g., member c_5 (UID 101) possesses k_2, $[k_1]$ and k_0.
- Controller manage all auxiliary keys (k_{n-1}, $[k_{n-1}]$,k_{n-2}, $[k_{n-2}]$, …, k_0, $[k_0]$)

Individual member removal



(a) Key distribution

Individual member removal (Cont'd)



(b) Departure of c_5

Individual member removal (Cont'd)

- c_5 (UID 101) possesses k_2, $[k_1]$ and k_0
- The new session key is encrypted and sent in 3 different messages:
  - $SK(r+1)_{[k0]}$, $SK(r+1)_{k1}$, $SK(r+1)_{[k2]}$
- Analysis:
  - Controller maintains keys: O(logN)
  - # messages (Update session key): O(logN)

Multiple members removal

- A simple solution is applying "individual member removal" m times to remove m members.
- This simple solution is NOT efficient.
- Efficient approach: Aggregate the removal of several members.

Multiple members removal (Cont'd)

- Example:
  - The members (c0, c4) with UID 000 and 100 need to be removed
  - Remaining members Sr={c1, c2, c3, c5, c6, c7}
  - Two messages containing new session key be multicast.
  - $SK(r+1)_{k0}$ (can be decrypted by c1, c3, c5, c7)
  - $SK(r+1)_{k1}$ (can be decrypted by c2, c3, c6, c7)

Multiple members removal (Cont'd)

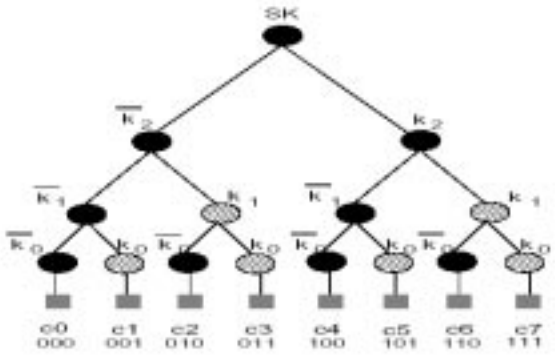


Fig. 2. Example of multiple members departing in the same round.

Multiple members removal (Cont'd)

- The problem of cumulative group member removal becomes grouping the remaining members that share common bits in their UIDs (and hence common keys) which are different from those of the removed members, in an efficient and systematic way.
- This problem is equivalent to the minimization of the Boolean membership function m().(If $m(X_0, X_1,\ldots, X_{n-1})=1$, then UID $X_{n-1}X_{n-2}\ldots X_0$ is in the group)

Multiple members removal (Cont'd)

An example: remove c0 and c4.

$M(X_2, X_1, X_0)=[X_2][X_1]X_0+[X_2]X_1[X_0] +[X_2]X_1X_0 +X_2[X_1]X_0 +X_2X_1[X_0] +X_2X_1X_0$

$=X_1 + X_0$

Multiple members removal (Cont'd)

- Simplification of Boolean function is the form known as the sum of products expression (SOPE)
- A minimal expression:
  - No other equivalent expression involving fewer products
  - No other expression involving the same number of products but a smaller number of literals (variables).

How to derive a minimal expression?

- Karnaugh map
- Quine-McCluskey algorithm

Here we will give an example on Karnaugh map method

How to derive a minimal expression? (Cont'd)

Input ($X_2X_1X_0$)	Output
000	0
001	1
010	1
011	1
100	0
101	X
110	1
111	1

TABLE I
BOOLEAN MEMBERSHIP FUNCTION.

How to derive a minimal expression? (Cont'd)

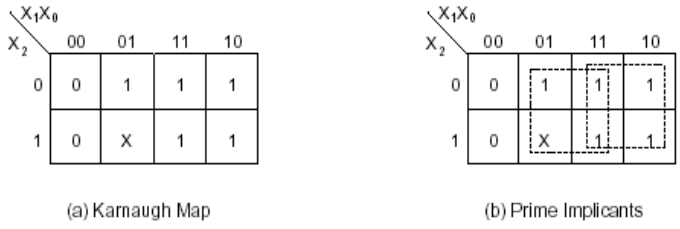


Fig. 3. Karnaugh map minimization of membership function.

Properties and Performance Analysis

- Cumulative Removal of 2 members (Worst case performance)
  - Lemma 1: Excluding C1 and C2, every member of the secure multicast group can decrypt at least one of the n messages of the following expression.
  - $\{SK(r+1)\}_{f(kn-1, [kn-2])}$, $\{SK(r+1)\}_{f(kn-2, [kn-3])}$, …, $\{SK(r+1)\}_{f(k0, [kn-1])}$)
  - Theorem 1: Re-keying a secure multicast group of size 2^n when two group members are to be removed require at most n messages.

Properties and Performance Analysis (Cont'd)

- Aggregate Removal of 2^{n-1} Members (worst case performance)
  - The number of messages required will be at most N/2

X_3X_2 \ X_1X_0	00	01	11	10
00	1	0	1	0
01	0	1	0	1
11	1	0	1	0
10	0	1	0	1

Fig. 4. Worst case for aggregate removal of $2^{(n-1)}$ members.

Properties and Performance Analysis (Cont'd)

- Average Case Performance
  - Please refer to the paper

Conclusion and Future Work

- An efficient key management and distribution scheme for secure multicast.
- It scales very well.
- For N group members:
  - # keys maintained by the controller: O(logN)
  - Message complexity (message number) for a single member departure is O(logN)
  - Message complexity for multiple members departure outperforms other schemes the authors have known

Conclusion and Future Work (Cont'd)

- How to deal with collusion attacks?
- How to update auxiliary keys?
- Analyze the average case overhead using simulations
- Build the proposed scheme into a toolkit for secure Internet multicast services
- Benchmark and optimize the components of the toolkit.
- Use the toolkit to enhance secure multicast applications.

Thank you!

- Thank you!

Questions and comments?

- Any Questions or comments?