Combinatorial Optimization of Group Key Management

M. Eltoweissy, James Madison U.,
M. H. Heydari, James Madison U.,
Linda Morales, Texas A&M – Commerce, &
Hal Sudborough, U. Texas – Dallas,

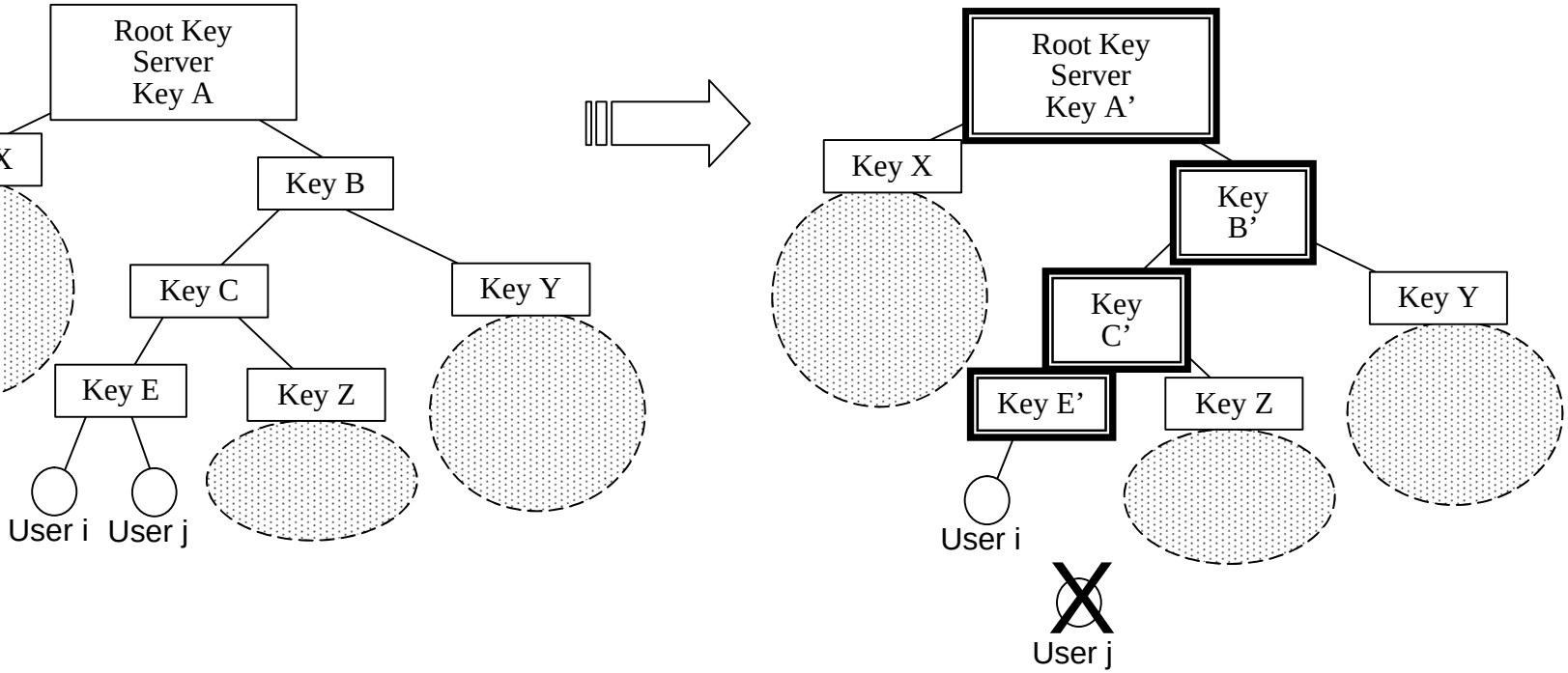
Why is key maintenance needed?

- Forward security is required when a user leaves the multicast group.
  - All keys known to the evicted user must be changed
  - New keys must be communicated to appropriate members
- Backward security
  - Keys changed when a user joins the group.

Basic System Problem

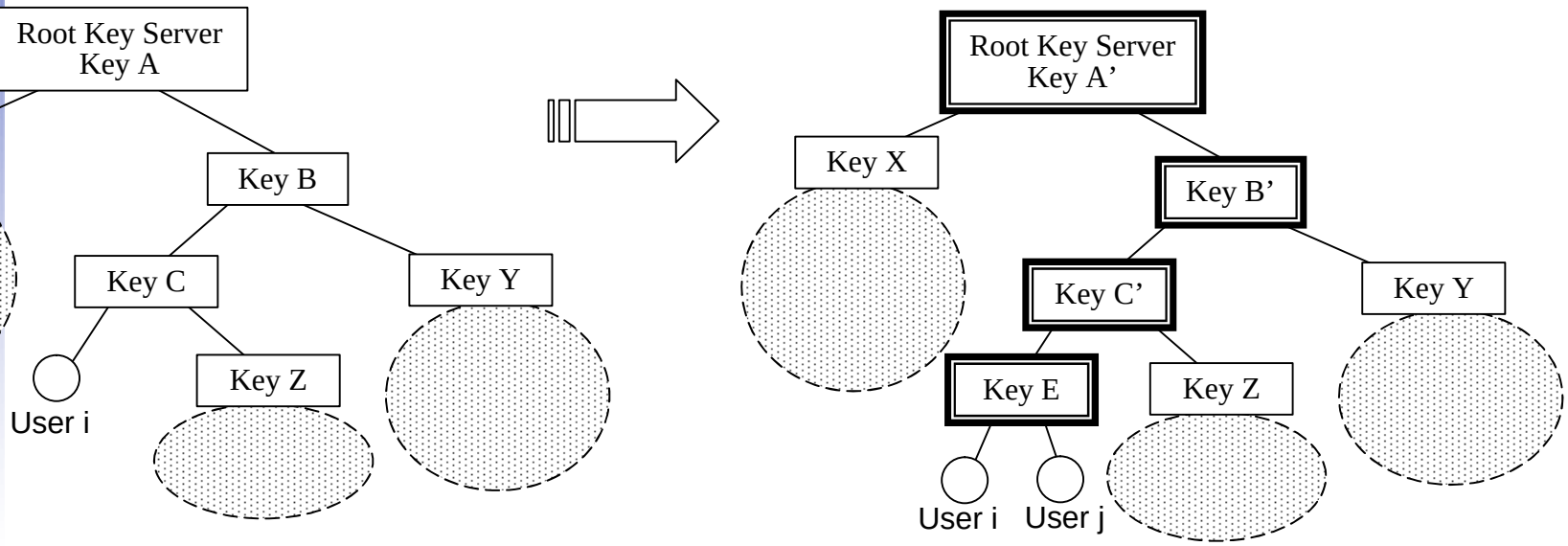
- Ensure message security for a dynamic multicast group by encrypting messages
- When users join or depart encryption keys must be changed.
- Minimize: (a) the number of keys users must know, and (b) the number of re-key messages

Binary Tree – Deleting a user



- Delete User j from key management tree
- Update key information
- $\log_2 n$ keys per user
- $\log_2 n$ re-key messages

Binary Tree – Adding a user



- Insert new user into key management tree
- Update key information
- $\log_2 n$ keys per user
- $\log_2 n$ re-key messages

Exclusion Problem

Let U = {1, …, n}. Given n,k,m, find a collection C of subsets of U, if possible, such that:

1. For every $i \in U$, there are m subsets in C such that the union of the m subsets equals U – {i}.
2. No element $i \in U$ is in more than k subsets of C.

Exclusion Basis System (EBS(n,k,m)) Problem: For given n,k,m, is there an EBS(n,k,m)?

EBS(n,k,m) Systems

- Each subset represents a key
- The elements of a subset represent users who know that key
- m is the number of re-key messages needed to distribute new keys *to all but one user*
- k is the number of keys each user knows.

Example (EBS(8,3,2))

- Consider the collection C = { $\alpha = \{5,6,7,8\}$, $\beta = \{2,3,4,8\}$, $\gamma = \{1,3,4,6,7\}$, $\delta = \{1,2,4,5,7\}$, $\varepsilon = \{1,2,3,5,6,8\}$ }
- The following unions are sufficient to isolate any element:

$$\alpha \cup \beta = \neg 1 \qquad \beta \cup \gamma = \neg 5$$
$$\alpha \cup \gamma = \neg 2 \qquad \beta \cup \delta = \neg 6$$
$$\alpha \cup \delta = \neg 3 \qquad \beta \cup \varepsilon = \neg 7$$
$$\alpha \cup \varepsilon = \neg 4 \qquad \gamma \cup \delta = \neg 8$$

- Note: no element is in more than 3 subsets

This can be described by a table:

	#1	#2	#3	#4	#5	#6	#7	#8
α	0	0	0	0	1	1	1	1
β	0	1	1	1	0	0	0	1
γ	1	0	1	1	0	1	1	0
δ	1	1	0	1	1	0	1	0
ε	1	1	1	0	1	1	0	1

Theorem.

There is an EBS(n,k,m) whenever $C(k+m,m) \geq n$.

Binary trees vs. EBS(n,k,m):

Note: $C(2n,n) = \Omega(4^n/n^{3/2})$.

Thus, $C(\log_2 n, (\log_2 n)/2) = \Omega(n/\log^{3/2} n)$.

So, EBS($c \bullet n/\log^{3/2} n$, $(\log_2 n)/2$, $(\log_2 n)/2$) is possible, for some constant $c>0$.

- $\therefore$ The parameters k and m (above) in the best EBS are half as large as in a standard binary tree system.

Example: For 10^6 users

A binary tree system requires each user to know 20 keys and needs 20 multicast re-key messages.

As C(23,11) = 1,352,078, the exclusion basis system EBS(10^6,12,11) exists and requires 12 keys/user and 11 multicast re-key messages.

Scalability

We need to consider how easy it is for a system to maintain an EBS(n,k,m) when a user is added or a user is deleted. For example, consider EBS(10,3,2). What does the system do when user 11 arrives?

Add 11th user & create EBS(20,3,3)

	1	2	3	4	5	6	7	8	9	10	11
A_1	0	0	0	0	1	1	1	1	1	1	0
A_2	0	1	1	1	0	0	0	1	1	1	0
A_3	1	0	1	1	0	1	1	0	0	1	0
A_4	1	1	0	1	1	0	1	0	1	0	1
A_5	1	1	1	0	1	1	0	1	0	0	1
A_6	0	0	0	0	0	0	0	0	0	0	1

Add 11th user & create EBS(20,4,2)

	1	2	3	4	5	6	7	8	9	10	11
A_1	0	0	0	0	1	1	1	1	1	1	0
A_2	0	1	1	1	0	0	0	1	1	1	1
A_3	1	0	1	1	0	1	1	0	0	1	1
A_4	1	1	0	1	1	0	1	0	1	0	1
A_5	1	1	1	0	1	1	0	1	0	0	1
A_6	1	1	1	1	1	1	1	1	1	1	0

Multi-level Security

	#1	#2	#3	#4	#5	#6	#7	#8
A1	0	0	0	0	1	1	1	1
A2	0	1	1	1	0	0	0	1
A3	1	0	1	1	0	1	1	0
A4	1	1	0	1	1	0	1	0
A5	1	1	1	0	1	1	0	1

Top Secret: 1, 5, & 8

Secret: 3 & 6

Unclassified: 2, 4, & 7

Multi-level Security

	#1	#2	#3	#4	#5	#6	#7	#8
A1	0	0	0	0	1	1	1	1
A2	0	1	1	1	0	0	0	1
A3	1	0	1	1	0	1	1	0
A4	1	1	0	1	1	0	1	0
A5	1	1	1	0	1	1	0	1
T	1	0	0	0	1	0	0	1
S	1	0	1	0	1	1	0	1
U	1	1	1	1	1	1	1	1

If user #1 left, then Re-key using:

A1(A3(A3'), A4(A4'), A5(A5'), T(T'), S(S'), U(U'))

A2(...)

Without modification, EBS(n,k,m) suffers collusion

If users share information, they could know all of the keys.

One can design techniques to make collusion attacks difficult.

For example,

- hide the keys from the users, *i.e.* give the users encrypted programs containing the keys,

rather than the keys themselves.

or

- combine the advantages of an EBS with the collusion avoidance of tree based systems.

Conclusions

We have introduced the concept of an Exclusion Basis System EBS(n,k,m) and proved, for which values of n,k,m, it exists.

We have given a constructive lower bound for any exclusion system and hence given optimum solutions.

We have shown that an optimum EBS is scalable.