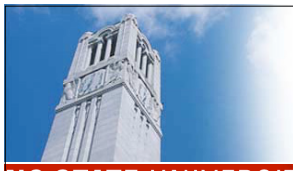


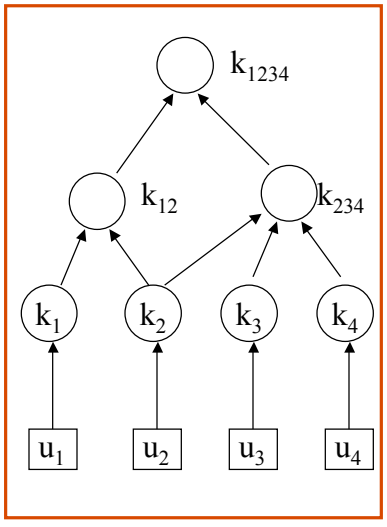
NC STATE UNIVERSITY Computer Science

CSC 774 Network Security

Topic 7.2 Group Key Distribution
-- Logical Key Hierarchy

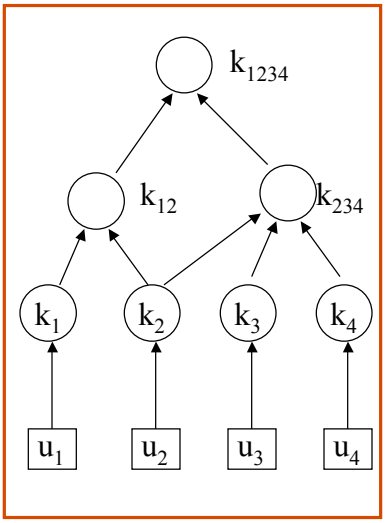
Key Graph

- A key graph is a DAG with two types of nodes
  - U-nodes: represent users
    - Each u-node has one or more outgoing edges, but no incoming edges
  - K-nodes: represent keys
    - Each k-node has one ore more incoming edges
    - Root: the k-node without outgoing edge



Key Graph (Cont'd)

- A key graph G specifies a secure group
  - The key corresponding to the root k-node is used as the group key.
- Join
- Leave

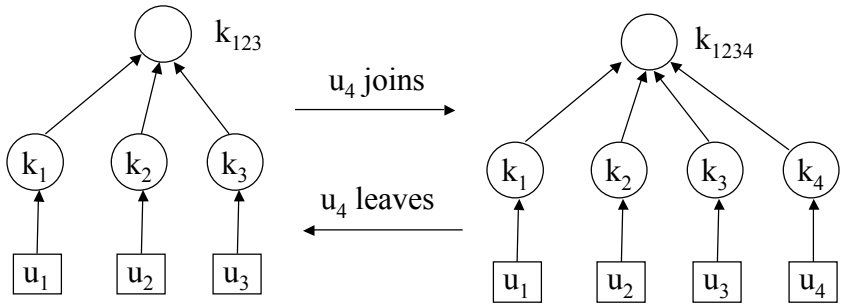


Key Graph (Cont'd)

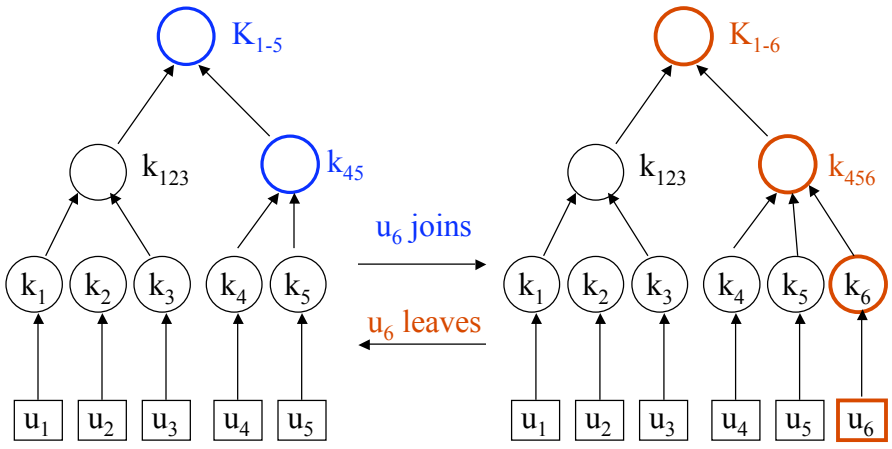
- Special cases
  - Key star
  - Key tree
    - Also known as Logical Key Hierarchy (LKH)

Key Star

- Naïve group key distribution
  - Join
  - Leave



Key Tree

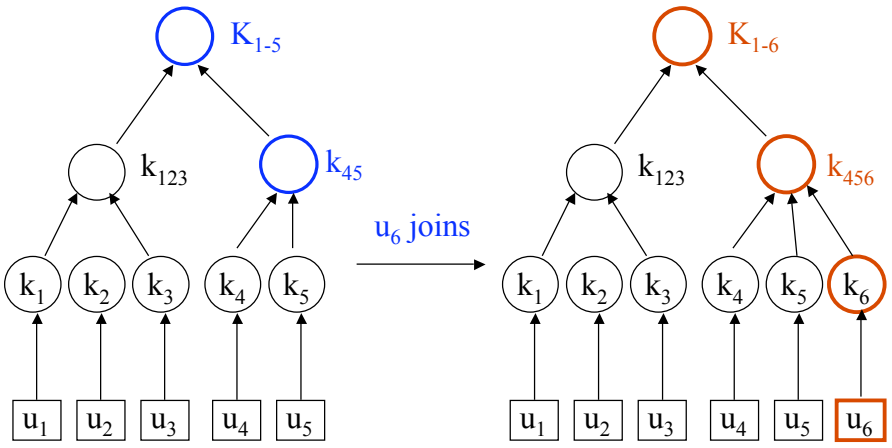


Joining A Key Tree

- User-oriented rekey
  - Idea: send each user precisely the new keys needed by the user
  - Approach
    - For each k-node with updated key (k → k'), construct a message by encrypting the new key and all its ancestors by the old key (k)
    - Send this message to the subset of users that precisely needs these keys
    - Send one rekey message to the joining user with all the new keys encrypted by the joining user's individual key

Joining A Key Tree (Cont'd)

- What are the rekey messages?

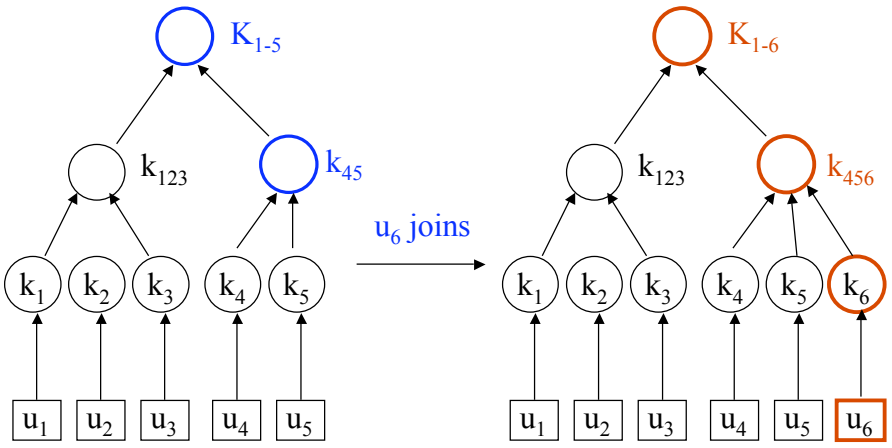


Joining A Key Tree (Cont'd)

- Key-oriented rekey
  - Idea: each key is encrypted individually
  - Approach:
    - For updated k-node (k → k'), construct two messages
      - New key (k') encrypted with the old key (k)
        - » Send to the users that share k
      - New key encrypted with the joining user's individual key
        - » Send to the joining member
    - Combine the rekey messages

Joining A Key Tree (Cont'd)

- What are the rekey messages?

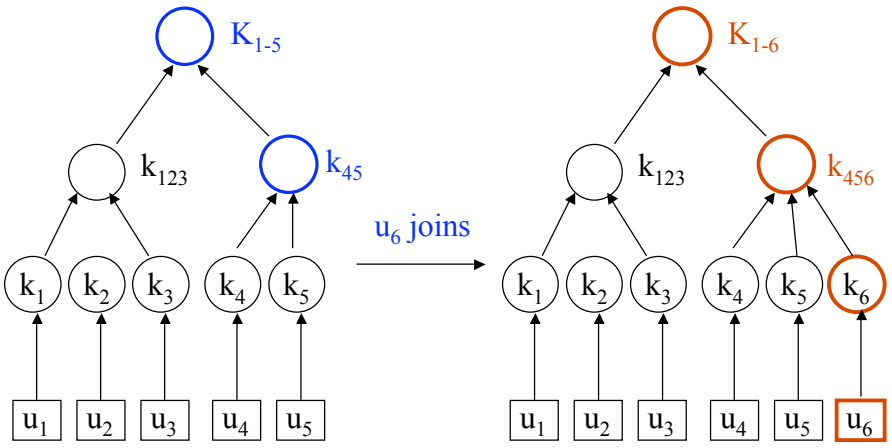


Joining A Key Tree (Cont'd)

- Group-oriented rekey
  - Idea: prepare rekey message for the entire group
  - Approach:
    - Similar to the key-oriented rekey, but merge all the messages for old members into one message
    - Reduce communication overhead if multicast (or broadcast) is available
    - Not all the contents are necessary for each (old) user

Joining A Key Tree (Cont'd)

- What are the rekey messages?

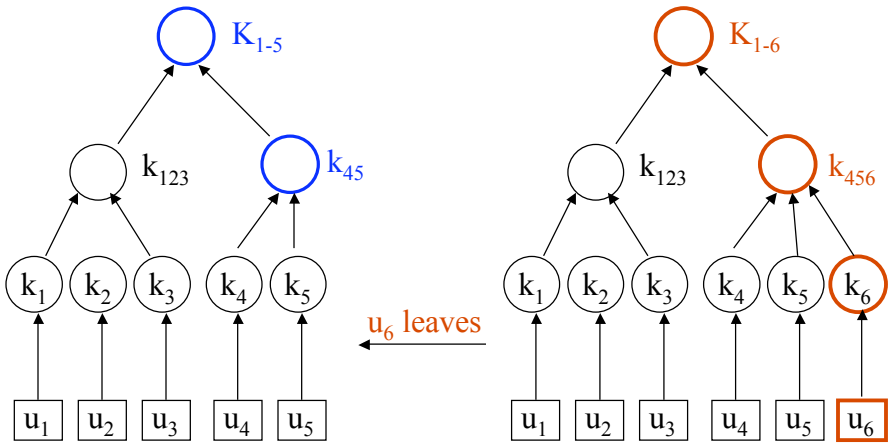


Leaving A Key Tree

- User-oriented rekey
  - Idea: each user gets a rekey message with all keys it needs encrypted with a key it holds
  - Approach:
    - Update the keys in the path from the removed user to the root
    - For each updated k-node x ($k \rightarrow k'$), and for each unchanged child y, construct a rekey message by encrypting the new keys of x and all its ancestors's new keys by the key K of y
    - Multicast this message to the users that have K.

Leaving A Key Tree (Cont'd)

- What are the rekey messages?

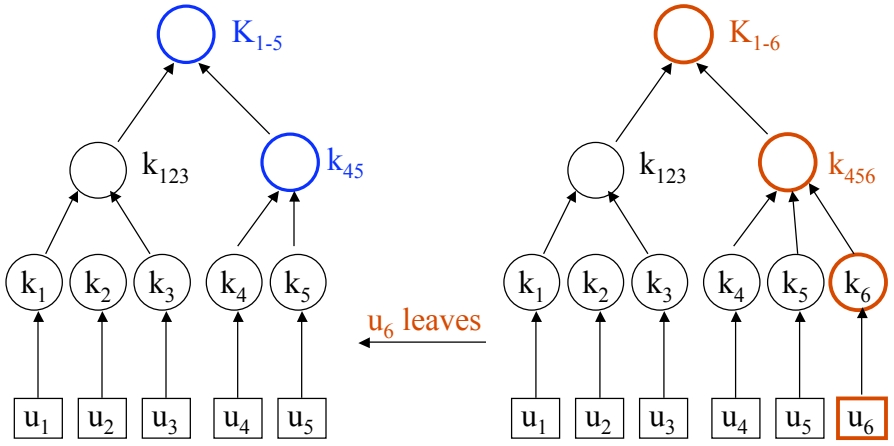


Leaving A Key Tree (Cont'd)

- Key-oriented rekey
  - Idea: encrypt each new key individually
  - Approach:
    - Update the keys in the path from the removed user to the root
    - For each updated k-node x ($k \rightarrow k'$), for each non-updated child y of x, use y's key K to encrypt k', use k' to encrypt x's ancestor's key, …
    - Send this message to the users having K

Leaving A Key Tree (Cont'd)

- What are the rekey messages?



Leaving A Key Tree (Cont'd)

- Group-oriented rekey
  - Idea: construct a single rekey message for the entire group.
  - Approach:
    - Similar to key-oriented rekey, but combine all rekey messages into one

Leaving A Key Tree (Cont'd)

- What are the rekey messages?