Key Grids: A Protocol Family for Assigning Symmetric Keys

Amitanand S. Aiyer, Lorenzo Alvisi, and Mohamed G. Gouda
ICNP2006
Presented by Jinsong Han
Motivation

• What is the smallest number of symmetric keys that need to be assigned to each process in a network of \( n \) processes so that each process can communicate securely with each other process in the network?
Naïve Solution
Basic Solution

• Each process holds a secret key.
• Two processes exchange the ID
• Knowing the ID of the process on the other side, compute a session key

\[
\begin{align*}
\text{X, } K_x & \quad \text{Y, } K_y \\
\downarrow & \quad \downarrow \\
K_{x, y} & \quad K_{x, y} \\
\text{X} & \quad \{M\}K_{x, y} \\
\text{Y} & \quad \text{Y}
\end{align*}
\]
Previous work: Key Grid

• Each process holds $O(\sqrt{n})$ keys
• A key assignment mechanism
• A key selecting algorithm
• Two types of key
  – Grid key
  – Direct key
• Utilize a Grid mapping to assign key and select key
Identities Generation

- Identities
  - n processes, marked by log(n) bits
  - For example n = 32, log (n) = 5

\[ b_0, b_1, b_2, b_3, b_4 \]

\[ \text{umax: length of A Bits} \quad \text{vmax: length of B Bits} \]

As much as possible, each part has the same number of bits

\[ \text{umax} \sim \text{vmax} \]
Note each process will be *Assigned* to a Grid node, but not now. 😊
Key Assignment to Grid Nodes

- Each \((u, v)\) has a grid key \(g(u, v)\)
Key Assignment to Grid Nodes

- Each pair of grid elements \((u, v)\) and \((u', v')\), where \(u = u'\), or \(v = v'\)
- Specify a direct key \(d(u, v)(u', v')\)
• Each process \((u, v)\) has:
• All \(g(u', v)\), where \(u = u'\); or \(g(u, v')\) where \(v = v'\)
• All \(d(u, v)(u', v)\), where \(u = u'\); or \(d(u, v)(u, v')\), where \(v = v'\)
Number of Keys For Each

• Each process holds a number of grid and direct keys
• $g(u', v)$ and $g(u, v')$, approximate $\text{umax} + \text{vmax}$
• $d(u', v)(u, v)$ and $d(u, v')(u, v)$, approximate $\text{umax} + \text{vmax}$
• Total number of keys is $O(4 \sqrt{n})$
Session Key (SK) Generation

SK: \{g(u', v), g(u, v')\}, Shared key = g(u', v) \text{ xor } g(u, v')
Usage of Key Cont.

SK: \{d(u, v) (u’, v)\}, Shared key = d(u, v)(u’, v)
Usage of Key Cont.

SK: \{d(u, v) (u, v')\}, Shared key = d(u, v)(u, v')
Contribution of This Work

• Extend key grid from 2-dimension to 3-dimension
• Then to k-dimension, where $k = \log(n)$
Key assignment in 3D

• Each p(u, v, w) has grid keys of the forms:
  – AB-g(u, v’), AB-g(u’, v)
  – AC-g(u, w’), AC-g(u’, w)
  – BC-g(v, w’), BC-g(v’, w)

• Each p(u, v, w) has direct keys of the forms:
  – AB-d(u, v)(u, v’), AB-d(u, v)(u’, v)
  – AC-d(u, v)(u, w’), AC-d(u, v)(u’, w)
  – BC-d(u, v)(v, w’), BC-d(u, v)(v’, w)
Number of Keys For Each in 3D

• Each process holds a number of grid and direct keys
• Grid keys: approximate $2(\text{umax} + \text{vmax} + \text{wmax})$
• Direct keys: approximate $2(\text{umax} + \text{vmax} + \text{wmax})$
• Total number of keys is $O(\sqrt[3]{12n})$
A General case, k-D

- Construct a k-D grid
- Number of Grids: \( \binom{\log n}{2} \)
- Assign each process
- \((k-1)(\text{umax}_0 + \text{umax}_1 + \ldots + \text{umax}_{k-1})\) grid keys
- \((k-1)(\text{umax}_0 + \text{umax}_1 + \ldots + \text{umax}_{k-1})\) direct keys
- Total \(O(2\log n(\log n - 1)^{\log n / \sqrt{n}}) = O(4\log^2 n)\) keys
Proof of the lower bound

• What is the smallest number of keys that need to be assigned to each process in order that each process can communicate securely with each other process?
• The lower bound
  – $O(\log n)$ keys.
Issue: how can reach $O(logn)$?