

Efficient Techniques for Realizing Geo-Spatial Access Control

Marina Blanton

**Department of Computer Science
Purdue University**

Joint work with Mikhail Atallah and Keith Frikken

ASIACCS'07

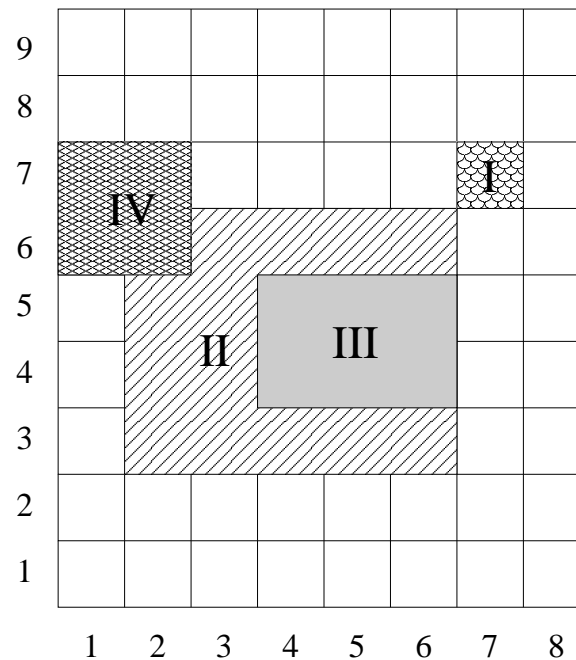
March 2007

Introduction

- In **geo-spatial access control systems** we are considering:
 - there is a **large area** with controlled access
 - the space is partitioned into **small cells**
 - a **user is granted access to a specific sub-area**
 - a **user is able to access resources at the cells of her sub-area**
- **Access to a cell means either:**
 - **physical access to a facility**
 - **access to information about a cell (weather, traffic conditions, etc.)**

Introduction (cont.)

- The entire area is modeled as an $m \times n$ grid
- User sub-areas are considered to consist of rectangles



Key Management in Access Control Systems

- **Key management** in access control systems refers to **assigning cryptographic keys** to resources and users for policy enforcement purposes
 - **proper enforcement** of access control policies is essential
 - **minimization of key material and computation** is important
- The problem of key management is well studied in other domains:
 - systems with a **hierarchy of user classes**
 - **time-based access control** policies
- We are unaware of other optimization work on key management in **geo-spatial context**

Key Management in Geo-Spatial Systems

- **The model** is such that:
 - access to each cell is secured with a **cryptographic key**
 - a user is given a small number of keys that permit her to access her specific sub-area of the grid
 - given her secret keys, a user is able to compute (or derive) the key for each cell within her area
- It is important that:
 - each user receives only a small number of keys
 - derivation of a cell's key is efficient

Our Solution

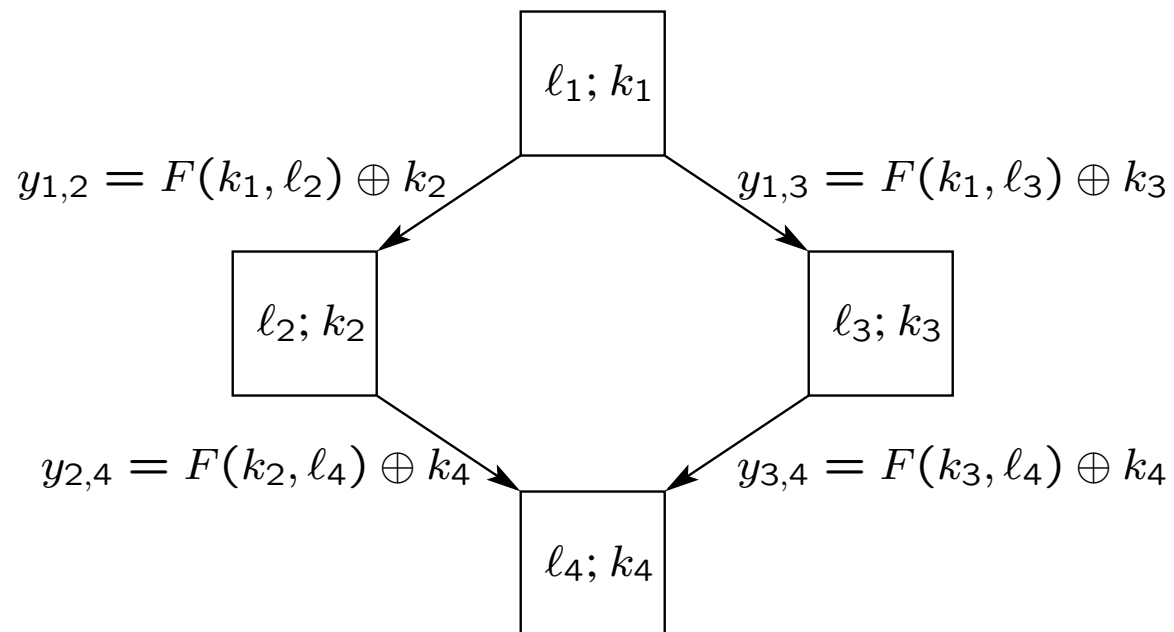
- We use existing key derivation techniques and **build a novel data structure**
 - such data structure allows a user with access to an **arbitrary rectangular area** to store a **small constant number of keys**
 - **derivation of a cell's key** within user's area consists of **a few efficient cryptographic operations**
 - the server is asked to maintain **public data** not significantly larger than the number of cells on the grid

The Key Derivation Mechanism

- The **key derivation mechanism** we utilize works for an **arbitrary directed graph**:
 - **each node** v is assigned a **secret key** k_v and a **public label** ℓ_v
 - **each edge** (v, w) has **public label** $y_{v,w} = F_{k_v}(\ell_w) \oplus k_w$
 - **anyone with access to a node** v 's **key** k_v is **able to compute the key of any of its descendants**
 - **key derivation is performed by following one edge at a time**
 - **given a parent's key** k_v , **computation of a child's key is done as**
$$k_w = F_{k_v}(\ell_w) \oplus y_{v,w}$$

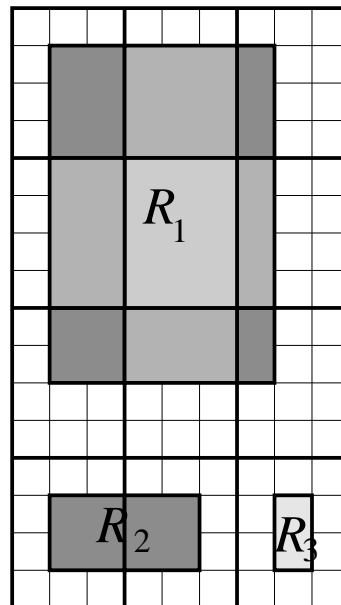
The Key Derivation Mechanism (cont.)

- **Example:**



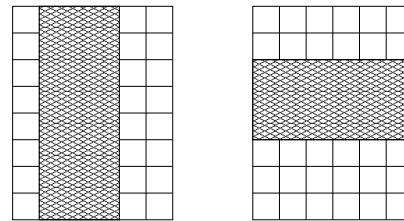
Our Approach

- The main idea is to **partition the grid into sub-grids of equal size**
- If user access area spans across such multiple sub-grids, we can manipulate keys at this level of granularity
- If user access area lies within a sub-grid, further recursively partition each sub-grid



Our Approach (cont.)

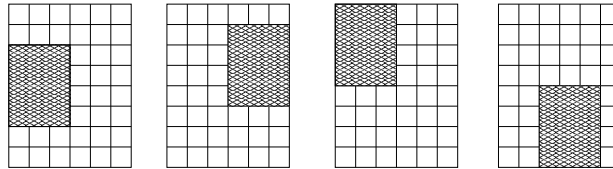
- First consider **special cases**:
 - **rectangles that span the grid**



- **each row** in horizontal spanning and **each column** in vertical spanning can be treated as a **single super-cell**
- this reduces the problem to **one-dimensional interval**
- now we are able to utilize **techniques from time-based access control to be able to handle varying widths**
- we obtain solutions with **3 keys per arbitrary span, constant key derivation time, and public storage slightly bigger than $O(m)$.**

Our Approach (cont.)

- Another special case is **rectangles that share a grid boundary**



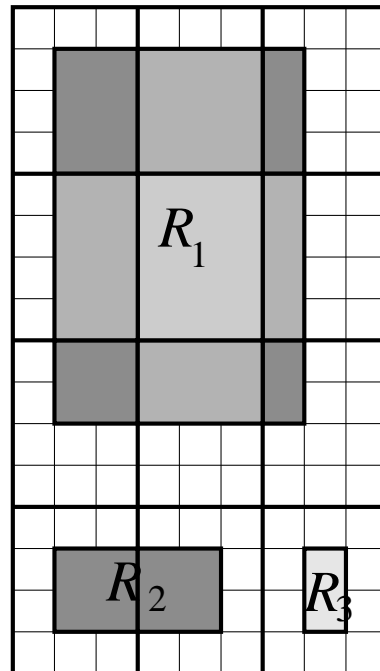
- assume that we have a rectangle that touches the right boundary of the grid
- then we can **give a user keys for the left-most cells of the rectangle** and permit derivation of all cells to its right
- efficient techniques for such key derivation already exist
 - for example, from a fixed position it is possible to reach each cell on the right in 3 hops by slightly increasing server's public storage
- having addressed horizontal key derivation, we **apply techniques from time-based access control to support rectangles of varying height**

Our Approach (cont.)

- **The overall solution:**
 - **we partition the grid into sub-grids of equal size and treat each sub-grid as a giant cell**
 - **we can apply solutions that follow from prior literature to be able to represent each rectangle composed of sub-grids**
 - **efficiency is achieved from the fact that the number of such sub-grids is rather small**
 - **this allows us to represent user access rights that cover a whole number of sub-grids**
 - **remaining parts (sides and corners) are addressed by the special cases**

Our Approach (cont.)

- **The overall solution (cont.):**
 - user access rights that entirely lie within a single sub-grid are addressed through **recursive partitioning of each sub-grid**



Our Results

- **For $m \times n$ grid** (assuming $m \geq n$):
 - a user with access to an arbitrary rectangular sub-area stores a **constant number of keys**
 - **key derivation** within the authorized region involves a **constant number of operations**
 - the server maintains $O(mn(\log \log m)^2 \log^* m)$ **public information**
 - all **cryptographic operations are very efficient**
- The solution generalizes to **higher dimensions** (e.g., space and time)