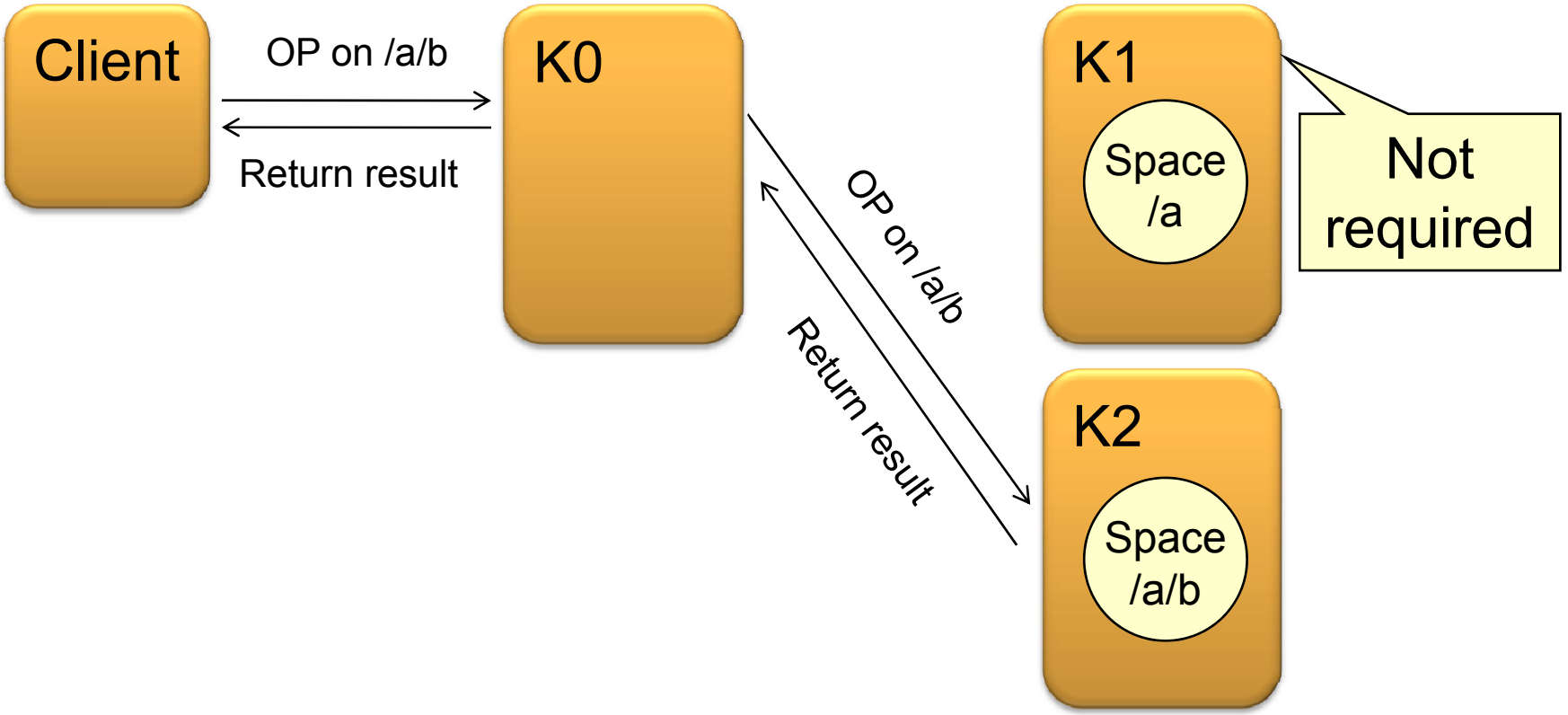
Distributed Subspaces and Security

Michael Lafite

WP5 Session
Milano, 8-9 January 2009

Without security

- Performing an operation on a distributed subspace.
  - Managing kernels of ancestor spaces not required
    → can't become bottleneck.



© TripCom

Basic idea

- For each space a kernel manages it must have all the security data required to authorize/deny operations on that space.
  - Policy of the space itself.
  - Policies of all the ancestor spaces of the space.
- → Copy security policies of ancestor spaces to remote kernels.

Copying policies

- Transfer policies during creation of space:
  - `SecurityData createRemote(SpaceURI spaceToBeCreated, KernelAddress managingKernel);`
- Forward policy update:
  - Use existing `setPolicy(…)` API method.
  - Change entry flow so that DM can forward the operation.
  - Recursive update algorithm similar to RD.

New security policies

- Which are the new security threats? Which are the crucial checks?

- Is client C allowed to create a distributed subspace of space S?
  - → new action CREATE_REMOTE
- Is client C allowed to create a space on the kernel?
  - New kernel policy
  - Only checked if there is no space policy
  - (also needed for root spaces)

Implementation

- Implementation until end of January:
  - New action CREATE_REMOTE
  - New kernel policy
  - Transfer of policies
  - Authorization of operations on remote subspaces.

- Some features will/may not be implemented:
  - (Optimized) policy update.
  - Secure forwarding of policy updates.

End of Document