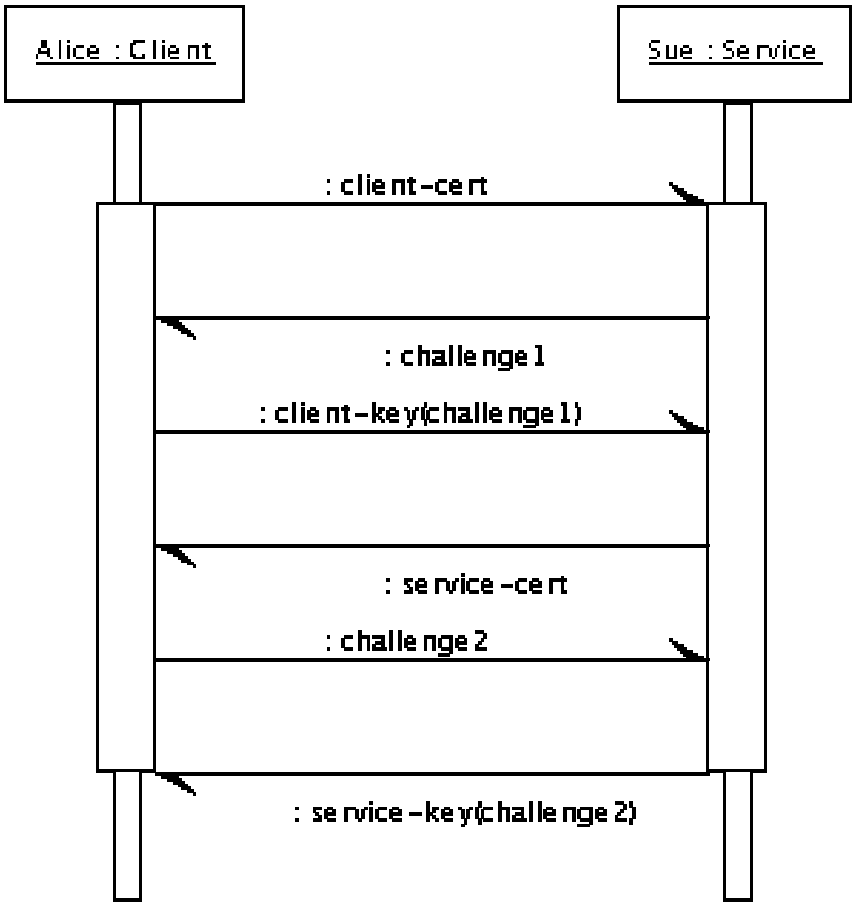
Security Group

D7.6 Design Ideas

E-mail: Akos.Frohner@cern.ch

Mutual Authentication

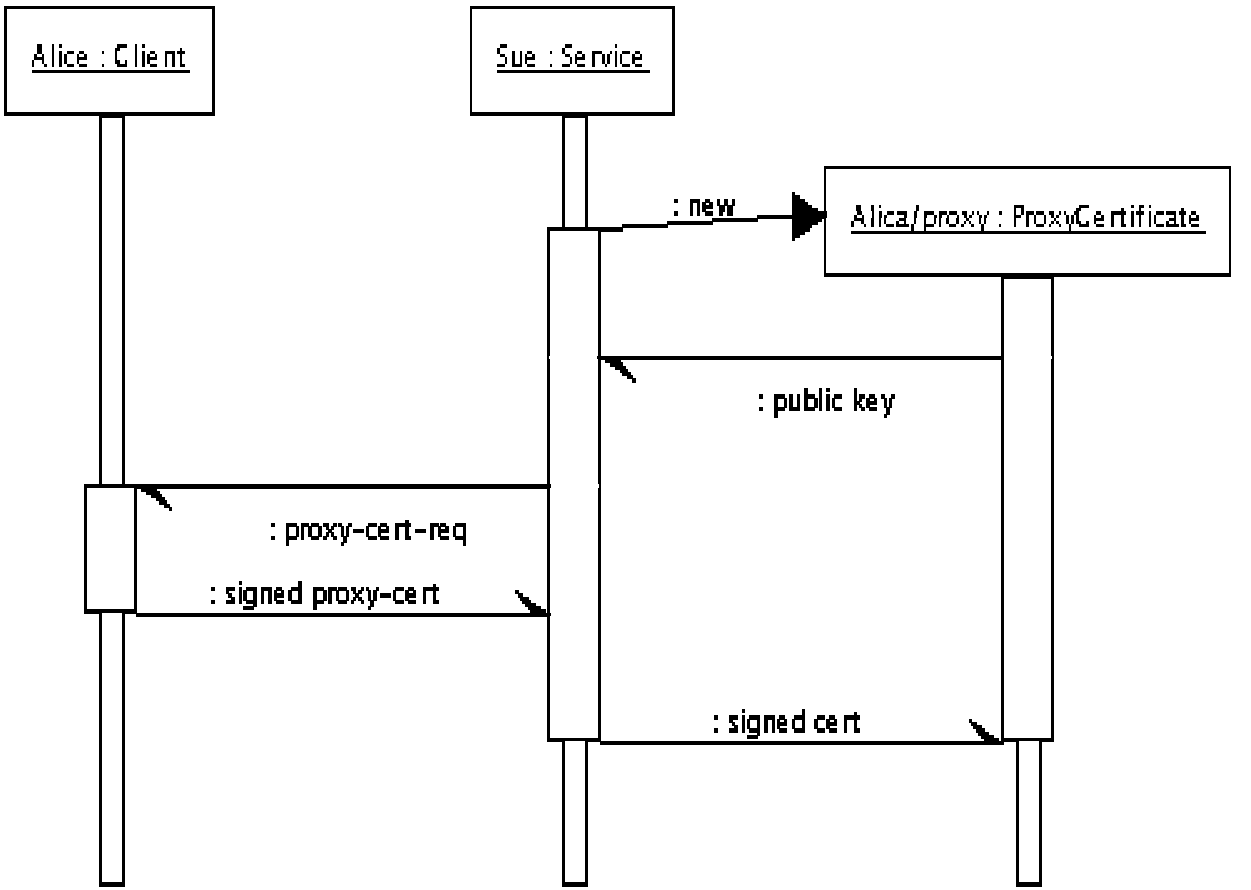


GSI – certificate based authentication

- challenge = random data
- key(data) = encoding with key
- validation: decode(public key, encode(private key, data)) = data

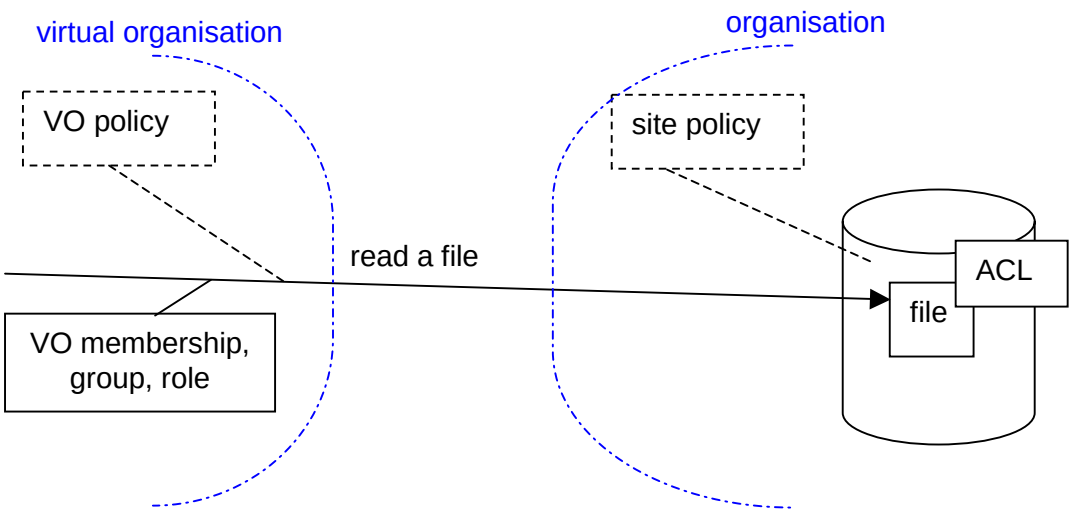
Short-time certificates! -> no CRL

Delegation



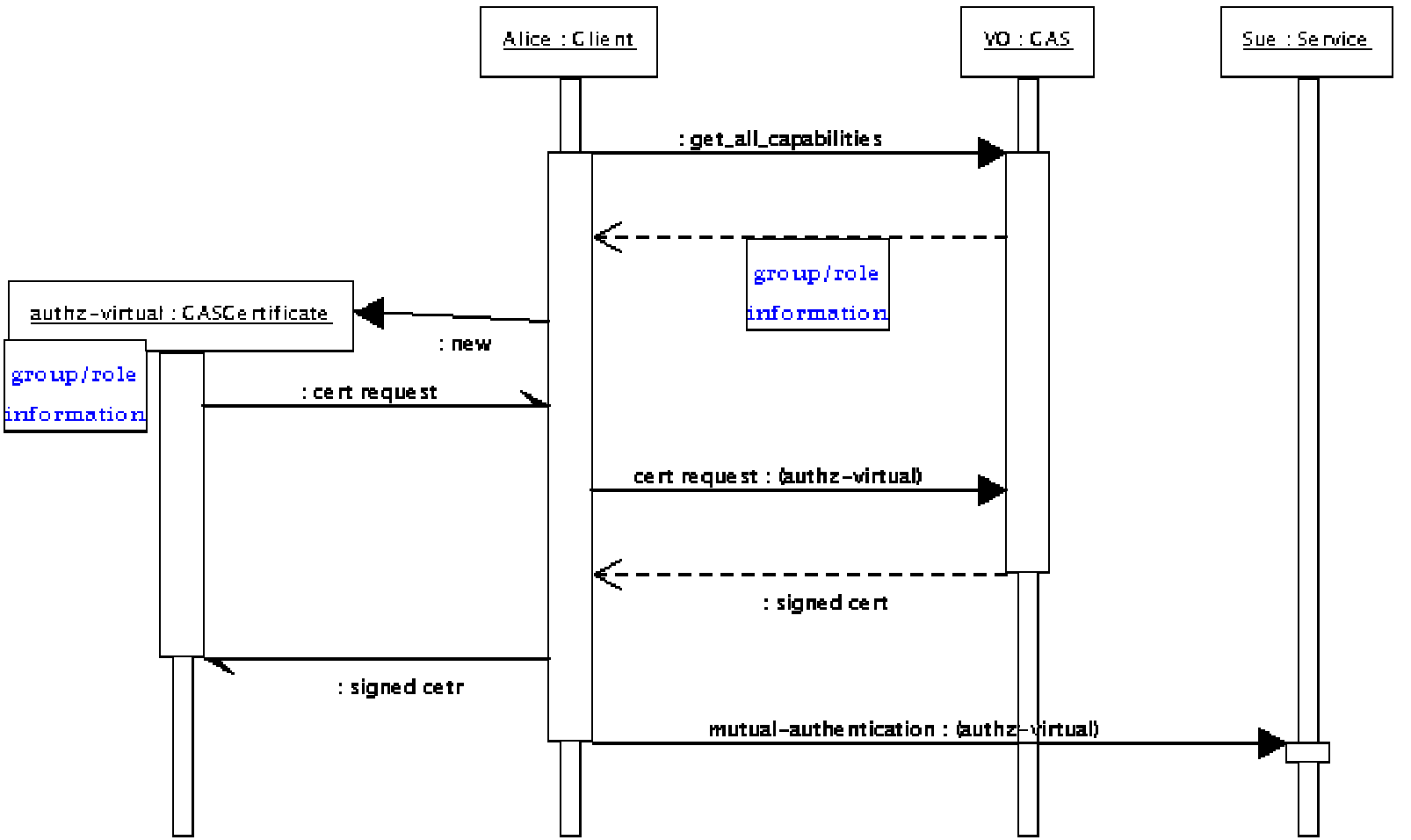
- proxy certificate is generated on the server side
- private key not crosses the net
- rights of the proxy are subset of the original rights

Membership (dataflow)

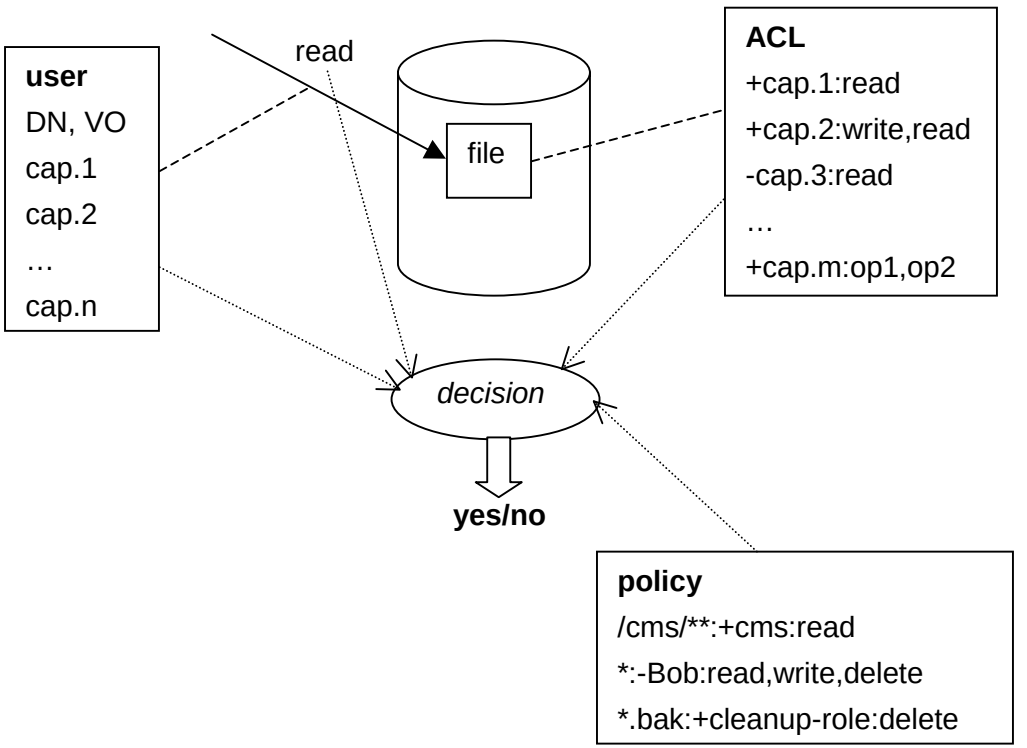


- Authenticate a user at a service
- Gather additional information associated to the user or the actual session (e.g. group membership, role, time)
- Gather additional information associated to the protected service or object (e.g. file permissions)
- Get local policy applicable to the situation (e.g. temporarily disabled user)
- Make an authorization information based on the identity and the additional information

Membership (sequence)



Access Control List



- user – list of capabilities
- operation
- protected object – access control list
- (policy: pattern + ACL)

-> yes/no decision

capability:

- DN
- VO DN
- group/role/...

New File or Directory in an SE

- the original owner (creator) is marked for accounting
  not user for authorization!
- creator have admin (getacl, setacl) permissions
- additional permissions from the enclosing object (default ACL), site and VO policy
- delete is a file attribute
- mark group/VO for accounting?

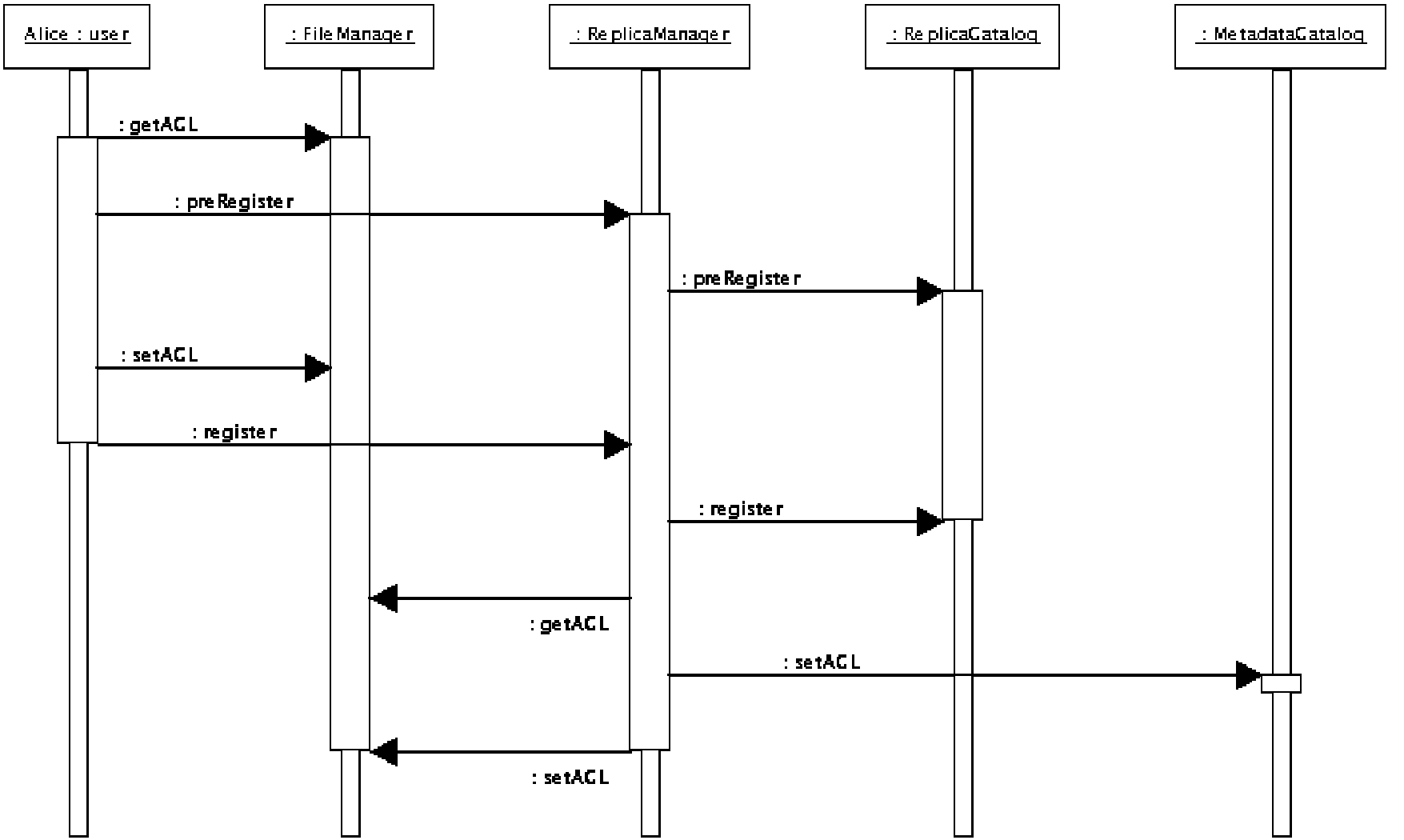
File

- creator: Alice
- ACL
  - +Alice:getacl,setacl, read,write,delete

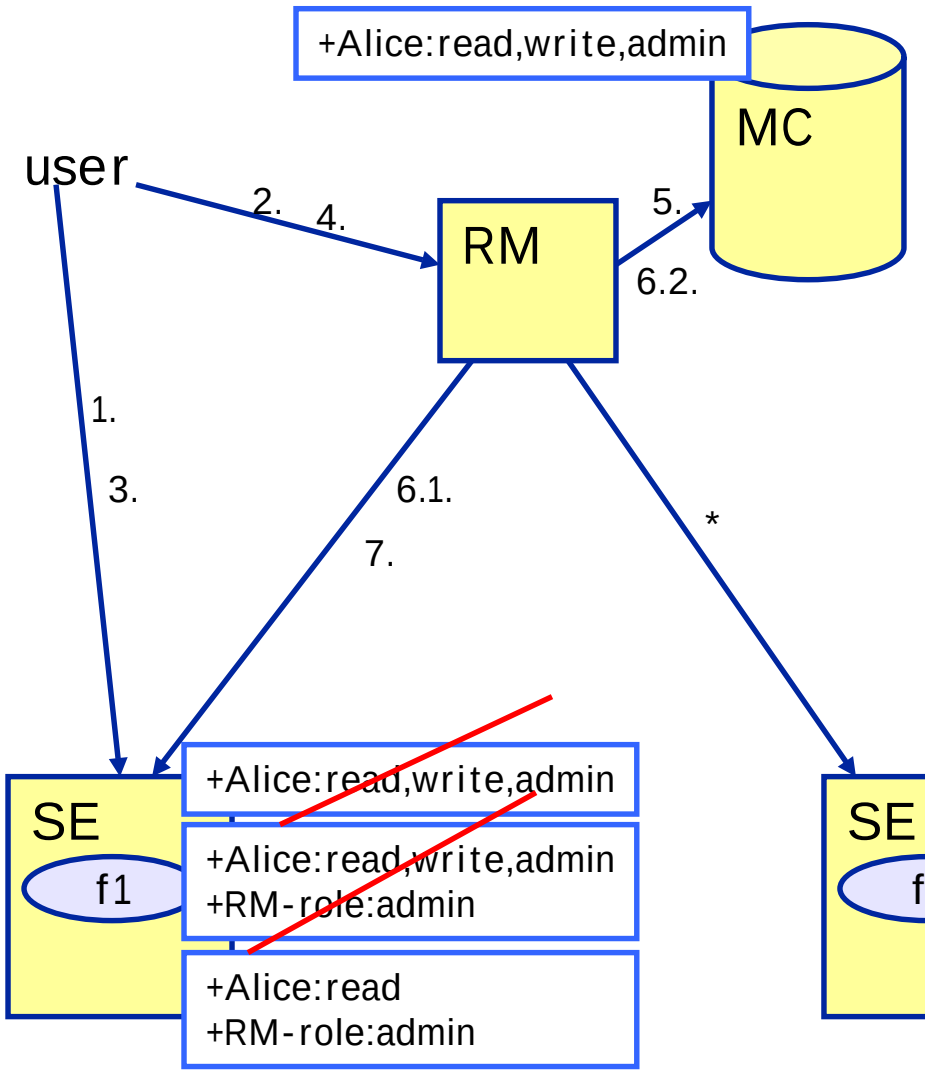
Directory

- creator:Alice
- ACL
  - +Alice:getacl,setacl,create,list,delete
- default ACL
  - dir:+Alice:getacl,setacl,create,list,delete
  - file:+Alice:getacl,setacl,read,write,delete

File Replication (sequence)

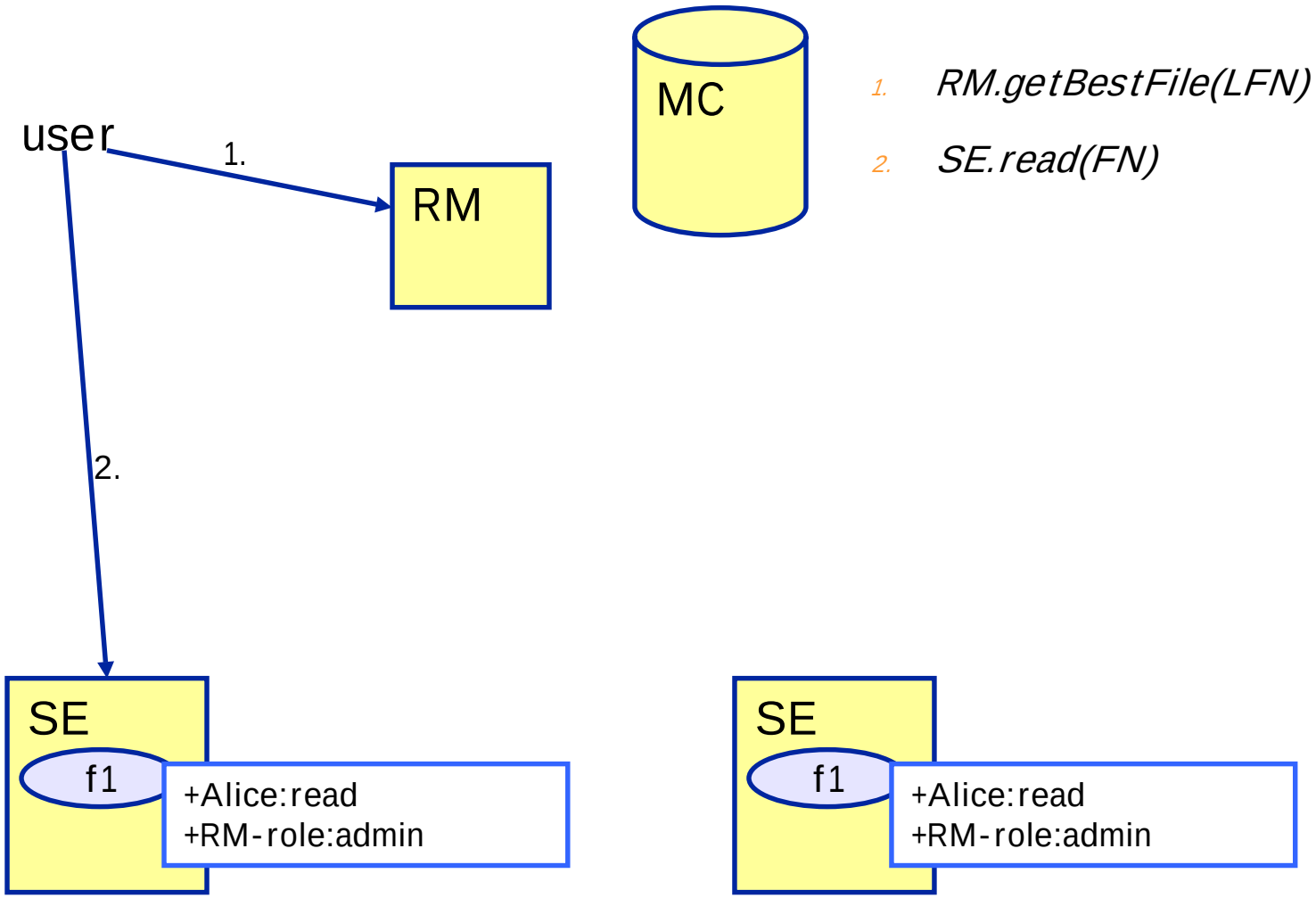


File Replication



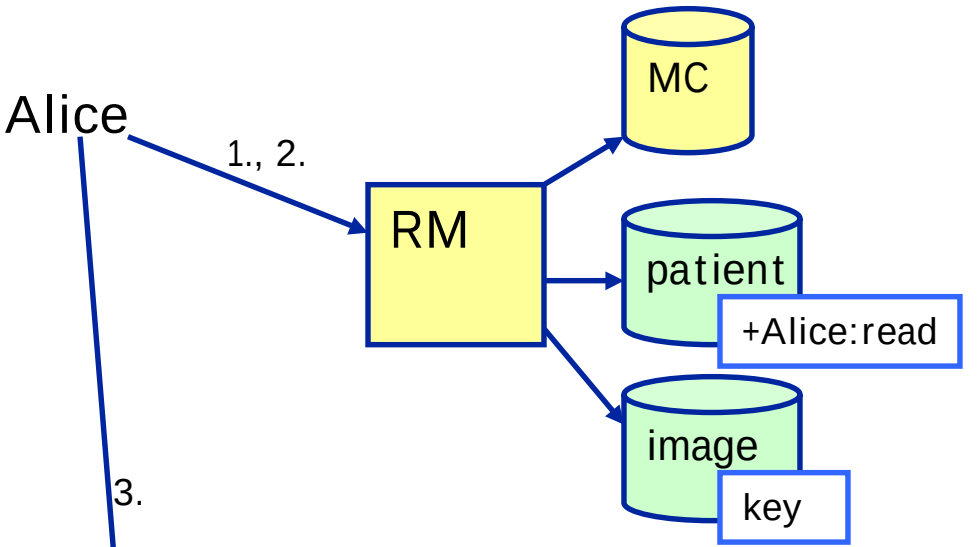
1. *SE.getACL*(+Alice:read,write,admin)
2. *RM.preRegister -> RM-role*
3. *SE.setACL*(+Alice:read,write,admin; RM-role:admin)
4. *Alice: RM.register*
5. *RM: MC.register*
6. *SE.getACL, MC.setACL* (+Alice:read,write,admin; RM-role:admin)
7. *SE.setACL*(+Alice:read; RM-role:admin)

Normal File Access

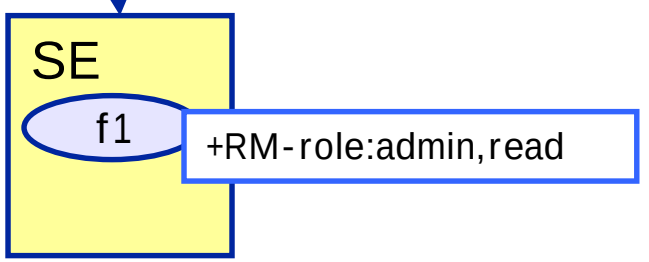


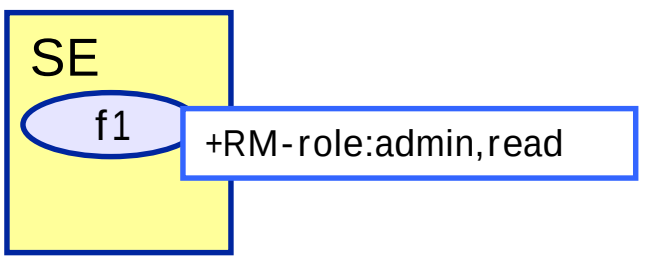
1. *RM.getBestFile(LFN) -> SE, FN*
2. *SE.read(FN)*

Medical Image Access

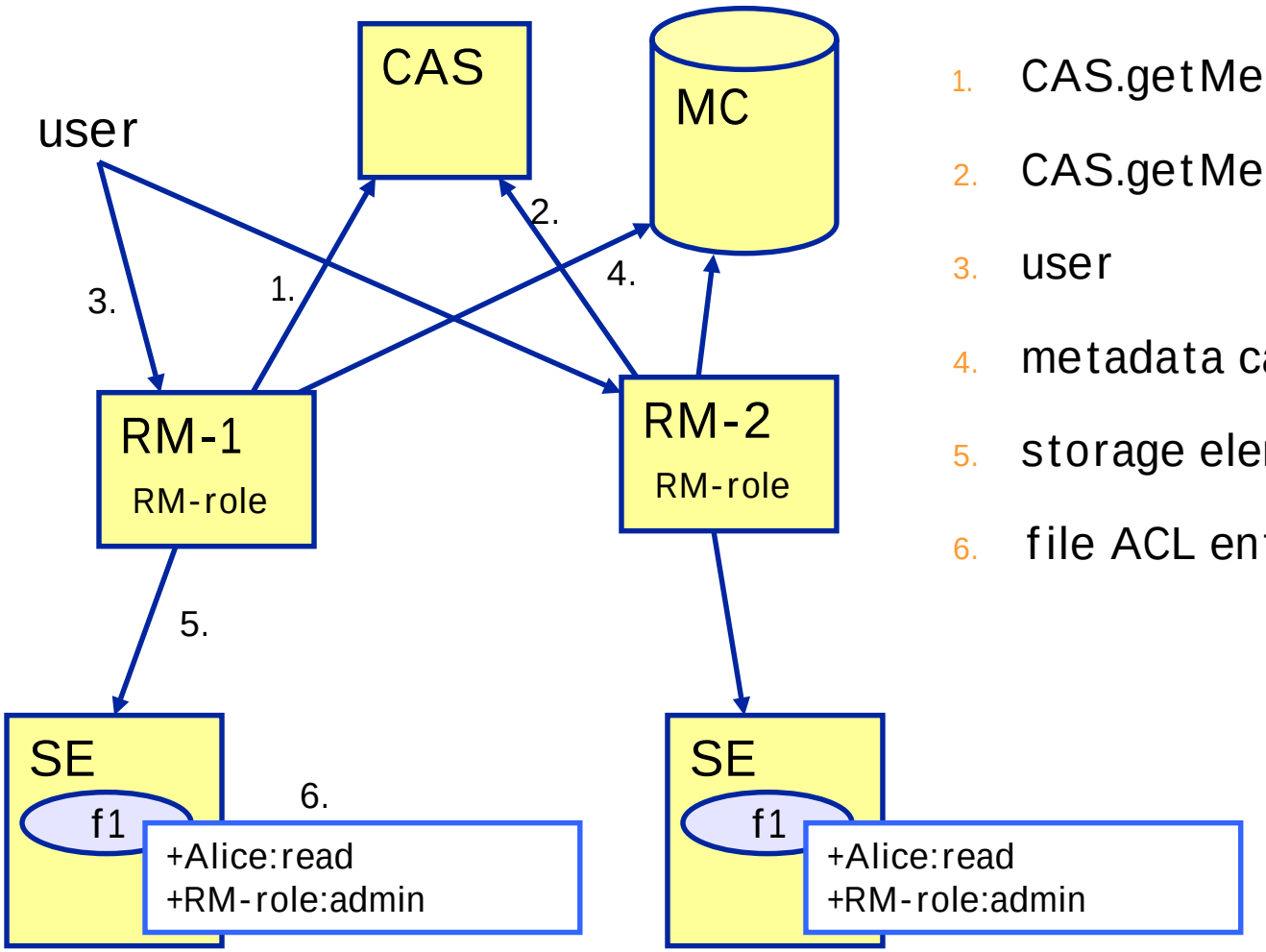


1. *RM.getBestFile(LFN) -> SE, FN*
2. *RM.getAppMetaData*
   -> restricted-cert, key
3. *SE.read(FN, restricted-cert)*
4. *decode(key, FN)*



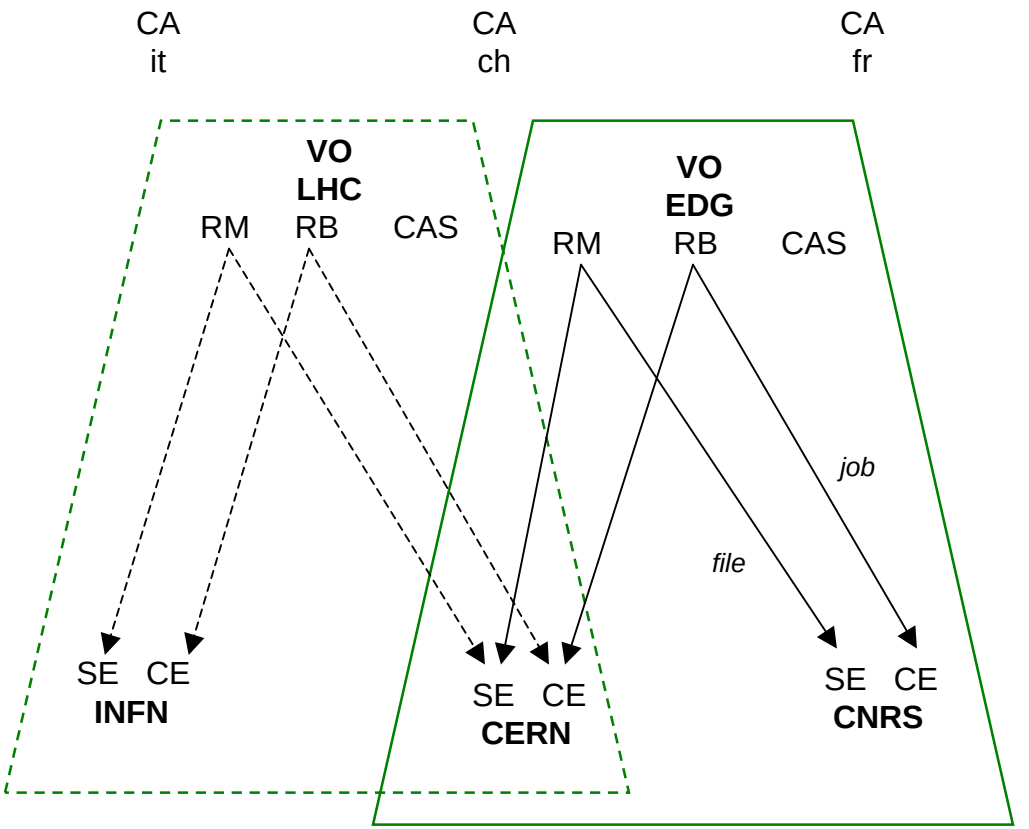


RM - role



1. CAS.getMembership -> RM-role
2. CAS.getMembership -> RM-role
3. user
4. metadata catalog
5. storage element
6. file ACL entry

Administrator Roles



Certificate Authorities

Virtual Organisation administrators

- CAS admin
- RM admin
- RB admin

Site administrators

- SE admin
- CE admin

Other issues

- initial credential: userid/password (PAM), kx509, ...
- renewable, forwardable certificates
- CAS: does more, then necessary
- encoding of capabilities (structure vs. DN)
- mapping CAS: composition of (Virtual) Organisations
- mutual authorization: use only VO-role playing service
- ACLs for jobs: monitor, stop, resume, kill
- using multiple vs. single VO (multiple vs. one cas-certificate)

...