Access Control

Model: access matrix
rows represent principals
columns represent objects

Implementation:

1. using access control lists (ACLs)
   keep non-null entries of column j with object j

2. using capabilities
   keep non-null entries of row i with principal i

ACL entry = principal name + access rights

capability = object name + access rights

Assume managers for the various types of object

On an access request, the manager must check that the requesting principal has the appropriate right to access the object

- check ACL list entry

- check capability passed by principal with request

ACLs versus capabilities

ACLs

subtle expression of policy
- entries may be per individual principal

slow to check (so don’t scale)
- if expressive e.g. allow negatives and exceptions
- if there are many principals and groups

awkward to delegate rights e.g. to a printer

easy to revoke
- but change to ACL entries may not have instant effect

capabilities

quick to check (so scale well)

problems/issues - see below

Capability-based access control - issues

1. **protection** of capabilities

   must prevent construction, tampering, theft

2. **propagation**

   can principals pass on copies?
   must they ask the object manager?
   if so - how is this enforced?

3. **delegation**

   is an example of propagation
   often with restricted rights for a limited time

4. (selective) **revocation**

   if the access control policy changes and
   certain principals should lose their rights
   their capabilities should be revoked without affecting others.

Capabilities in centralised systems

Several capability architectures were designed and built
e.g. Plessey PP250, CAP

Capabilities can be protected by hardware and/or the OS

* held in protected OS space per process

* held in typed capability segments in user space
  - operations insert, delete, use as argument.

Capabilities in distributed systems

Can’t be protected by hardware/OS

Have to be transferred across networks

Need to generalise cf. security ... capability = certificate

Capabilities in distributed systems

Must be protected by encryption

The object manager keeps a SECRET (random number) with each object.

A capability is constructed using:

check digits = f(SECRET, protected fields)

protected fields	check digits

When a capability is presented with an operation invocation,
the manager checks that:

f(SECRET, protected fields)= check digits

If not, the invocation is rejected.

Role-based Access Control (RBAC)

Services may classify their clients into named roles, e.g.

login service: logged-in-user (user-id, machine-id)

patient monitoring service: surgeon, doctor, nurse, patient
each with appropriate parameters

online exam service: candidate, examiner, chief examiner

digital library service: reader, librarian, administrator

Access rights are assigned to roles for:

* use of the service (method invocation)
* access to individual objects or broad categories of object managed by the service

Roles may be parametrised by principal-ID for fine-grained access

e.g. File-service: reader (file-ID, principal-ID)
reader (some-file-category)

NOTE GENERALITY AND SCALABILITY

Design of capabilities

Traditional OS capabilities in centralised systems:

object-id	rights

Role based access control (RBAC)

role	params

RBAC in a distributed system:

check digits = f(SECRET, protected fields)

protected fields	check digits

role	params	check digits

we shall expand on this .....

Capabilities/Certificates

1. protection

protect against tampering - adding rights
NOT against theft - eavesdropping on the network

2. propagation

still no control over transfer to other principals

3. delegation

object manager must be asked to generate a capability with reduced rights to pass to another principal for delegation

works indefinitely - duration not controlled

4. revocation

- create a hit list of principals adds to overhead on all accesses

- change the SECRET
  not selective - a new SECRET can be used to generate new capabilities
  - old ones will no longer work
  principals with old capabilities will have to ask for new ones to be constructed
  principals rejected by a new policy are refused access.

Principal-specific capabilities (Li Gong, Cambridge 1989)

Include the name of the principal as a field in the capability
- for generation and checking

1. protection from tampering - YES - as before
   protection from theft - YES

2. propagation
   a capability for the new principal
   must be created by the object manager

3. delegation
   a capability for the delegated principal
   must be created by the object manager

4. revocation - can be selective
   e.g. keep a hot list of principals to be rejected by new policy

Principal-specific capabilities - design

Li Gong (1989), Opera group MSSA: Sai Lai Lo, Richard Hayton (1993):

Include the name of the principal (e.g. persistent user-ID)
as a protected field in the capability

OASIS, Richard Hayton (1996)

Include the name of the transient principal (e.g. process-ID)
in the encryption function but

NOT as a protected field in the capability

check digits = f(SECRET, process-ID, protected fields)

better security - transient, session-length capabilities

how to capture - role-based access control - see OASIS
- persistent credentials - see OASIS

anonymity if required
- role authorises access (RBAC)
- transient principal need not indicate identity

OASIS: Open Architecture for Secure Interworking Services

* Oasis services name their clients in terms of **roles**

* Oasis services, and Oasis-aware services,
  specify policy in terms of roles
  - for service use (method invocation)
  - for object access *cf. ACL*

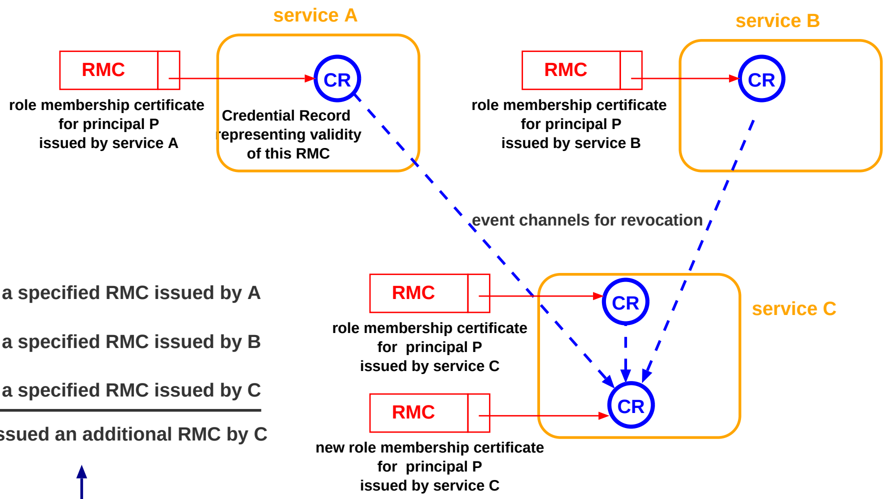
entering a role:

* Oasis services specify policy for role entry
  - in RDL (Role Definition Language)
    * Clients present **credentials** for entering a named role in order to use a service
      - RMCs of this and other services,
        possibly with conditions on the role parameters
      - persistent auxiliary credentials (e.g. registered student, registered doctor)

* Credentials checked against policy specification

* If successful , client is issued a role membership certificate (RMC)
  cf. principal-specific capability

OASIS: entering a role



P has a specified RMC issued by A

P has a specified RMC issued by B

P has a specified RMC issued by C

P is issued an additional RMC by C

This is the policy specification by service C
of the conditions for principal P to enter some role

It is expressed in a formal language, RDL (Role Definition Language)

OASIS - using RMCs

* client presents RMC with each service use
  - method invocation
  - access to service-managed objects

* services may be OASIS-aware and require, but not issue, RMCs

* service at which RMC is presented checks back with issuing service which checks RMC.
  used service caches RMC to avoid repeated check-back

OASIS philosophy:

independently developed services

independent design of RMCs

policy expression is per service / per domain

service (domain) level agreements on use of others' RMCs

anonymity if and when required

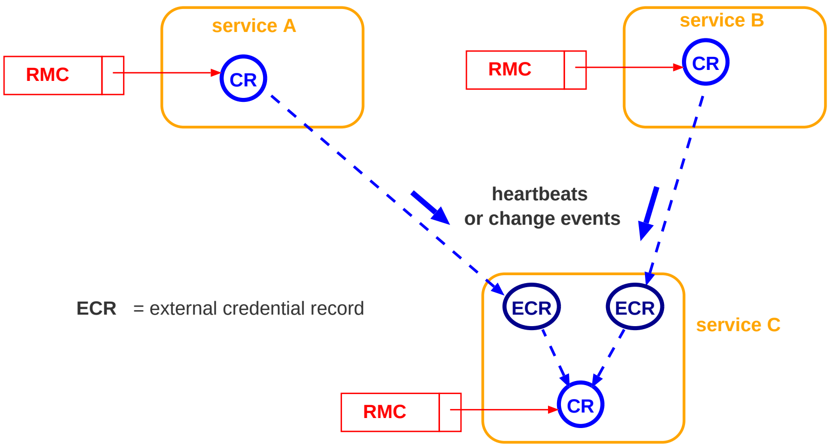
OASIS - revocation

* RMCs are transient - session-length

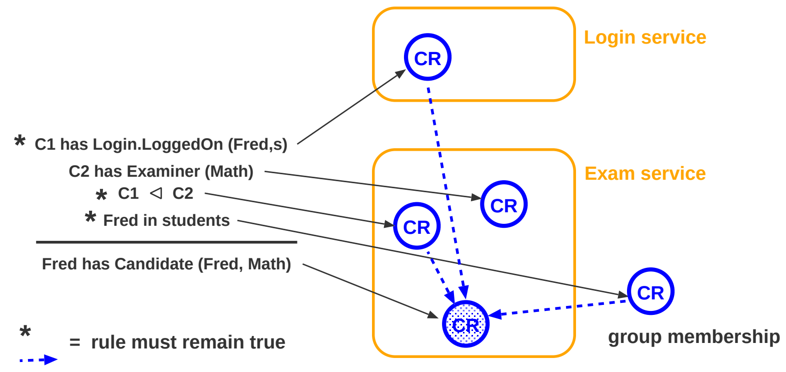
* when a role is entered the role-issuing service creates a corresponding credential record (CR), see A-12

* the services' CR data structures are linked by event channels

* if a role entry condition becomes false, this is signalled as an event so that all dependent RMCs can be revoked



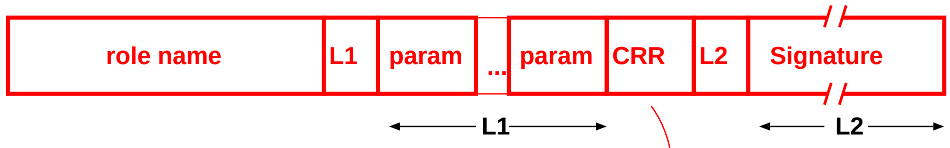
RDL specifies which role entry conditions must remain true



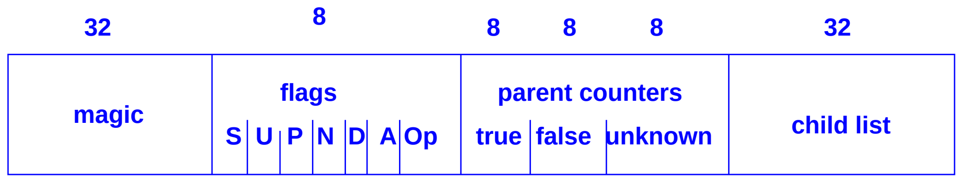
* C1 ◁ C2 *means that an examiner has issued Fred with a persistent certificate for the role exam-candidate at some time in the past*

* Fred in students *this is shown as a registration database lookup it might also be managed by issuing a persistent student-registration certificate*

A possible role membership certificate



A possible credential record



Distributed Systems Engineering issues

services and links may fail
services may be heavily loaded
in a large-scale implementation service replication may be needed

the non-arrival of a condition violation event may mean

- all is well (condition still holds)
- failure of link or service (condition unknown)

OASIS uses an inter-service heartbeat at a negotiated rate

- if a heartbeat fails to arrive any CR dependent on that service is marked UNKNOWN

- the service may suspend the corresponding RMCs according to policy

Policy issues

OASIS: policy for role entry
policy for service use
policy on absence of heartbeats

Policy specification - e.g. Govt. documents

"patients can specify who may access their medical record"

"doctors may access medical records" ....

Work in progress:
near natural language specification,
automatic translation to mathematical form

stored in policy stores - OASIS secured

checked on role entry, service use, etc.

audit of accesses - audit services - OASIS secured
"patients may see who has accessed their medical records"

policy change - another event

policy version - tag audit record with policy version that authorised access