Dynamically Authorized Role-Based Access Control for Secure Distributed Computation

CORBA CSIv2 in Action

C. Joncheng Kuo & Polar Humenn

Center for Systems Assurance

Syracuse University

March 20, 2002

Terminology

- **RBAC**: Role-Based Access Control
- **CSIv2**: Common Secure Interoperability Version 2
- **ATLAS**: Authorization Token Layer Acquisition Service

Outline

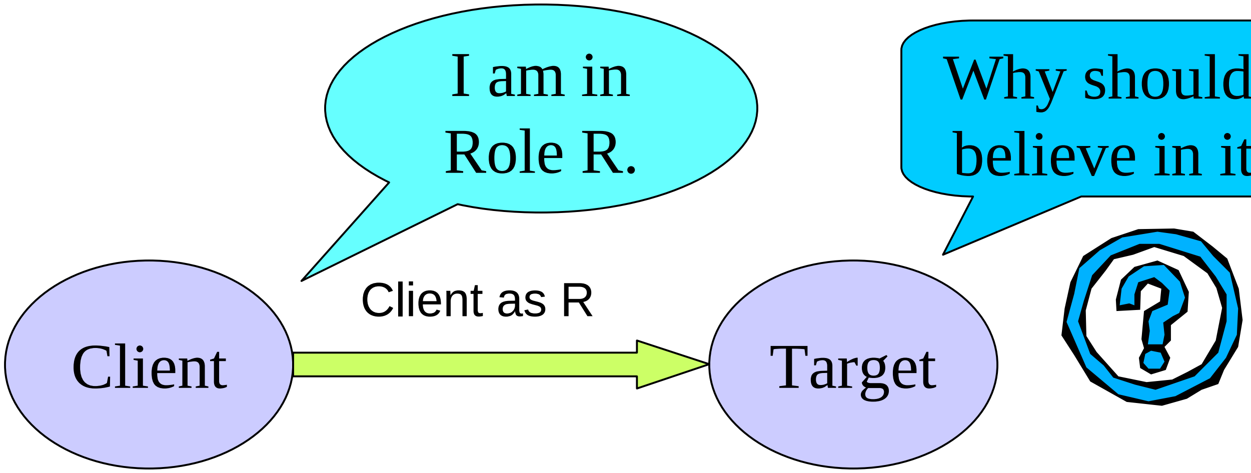
- CSIv2 and RBAC
- Role authorization
- Role administration
- Example: secure computation system
- Conclusions

What is in CSIv2?

- CSIv2 defines the Security Attribute Service protocol, which provides:
  - Identity Assertion:
    - Allows a client to claim to make a CORBA request on behalf of an identity other than its authenticated principal.
  - Authorization Service:
    - Transfers a client’s authorization data to a target.

Our Approach: Use CSIv2 to Do Role-Based Access Control

- A client uses a CSIv2 Identity Token to claim to be in a role.

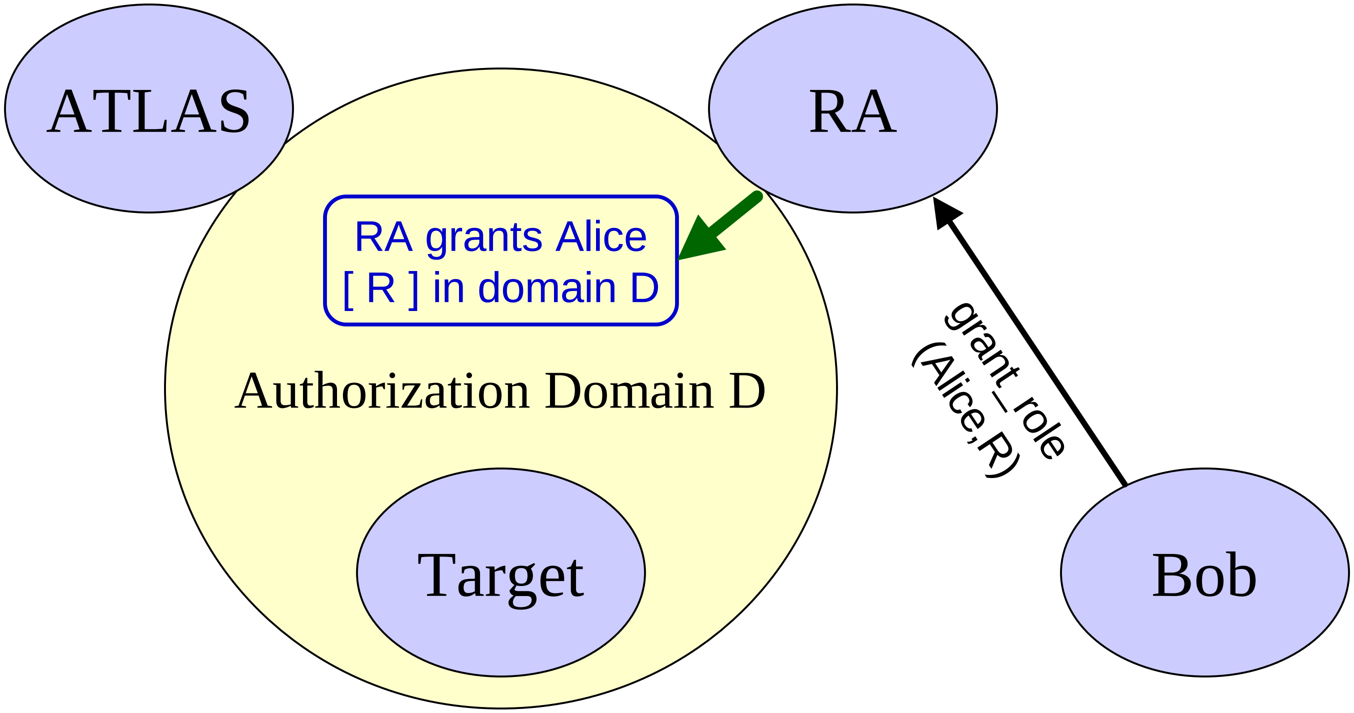


Role Authorization

- We use a **role certificate** to grant a principal the right to act in particular roles.
- Elements of a role certificate:
  - **Subject**: to whom roles are granted.
  - **Issuer**: the issuer of a certificate.
  - **Validity period**: the valid time of a certificate.
  - **Roles**: a list of roles granted to the subject.
  - **Authorization domain**: in which a certificate is accepted.

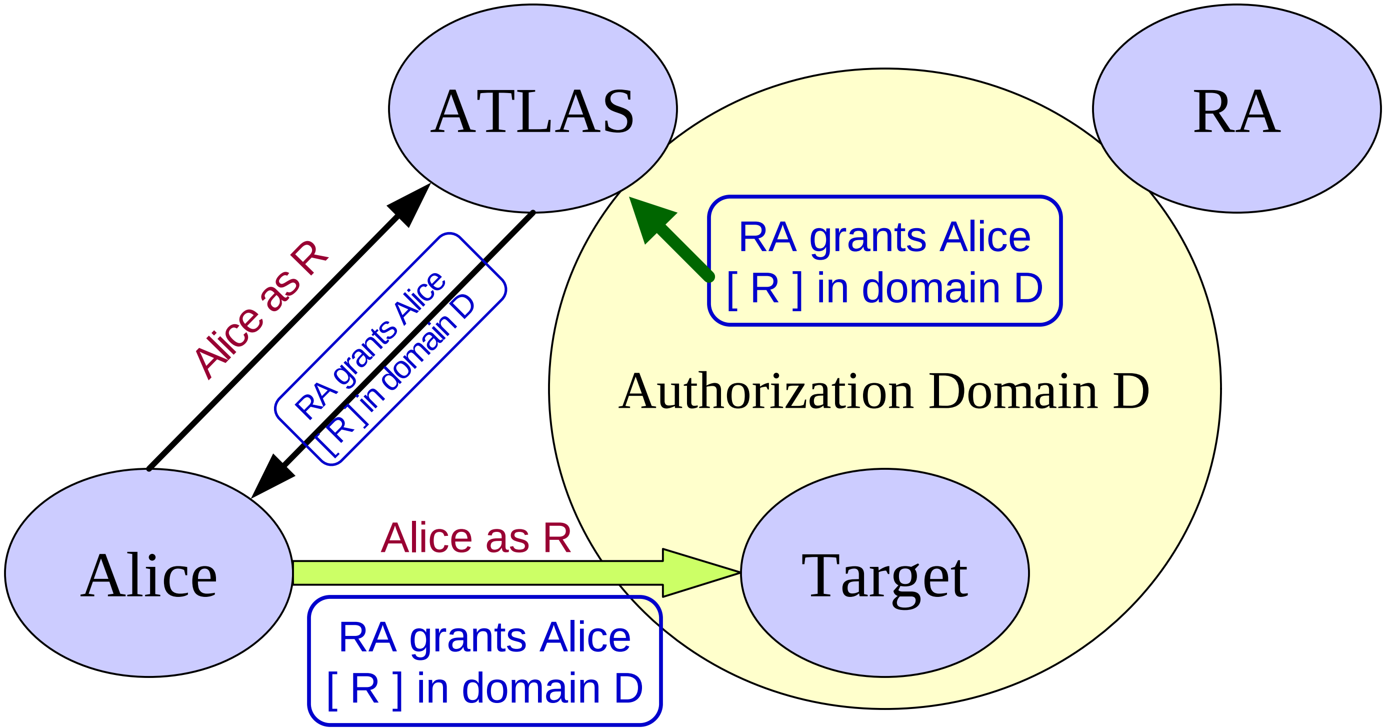
Role Administration

- We use a Role Authority (RA) to issue role certificates.



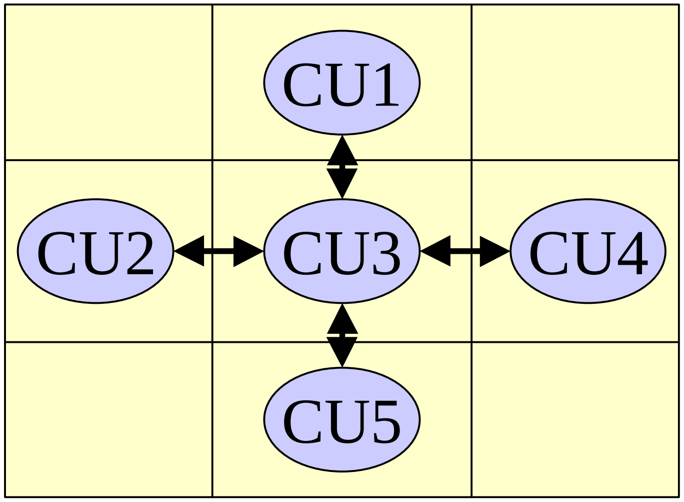
The Use of Role Certificates

- A client uses an ATLAS object to retrieve role certificates.



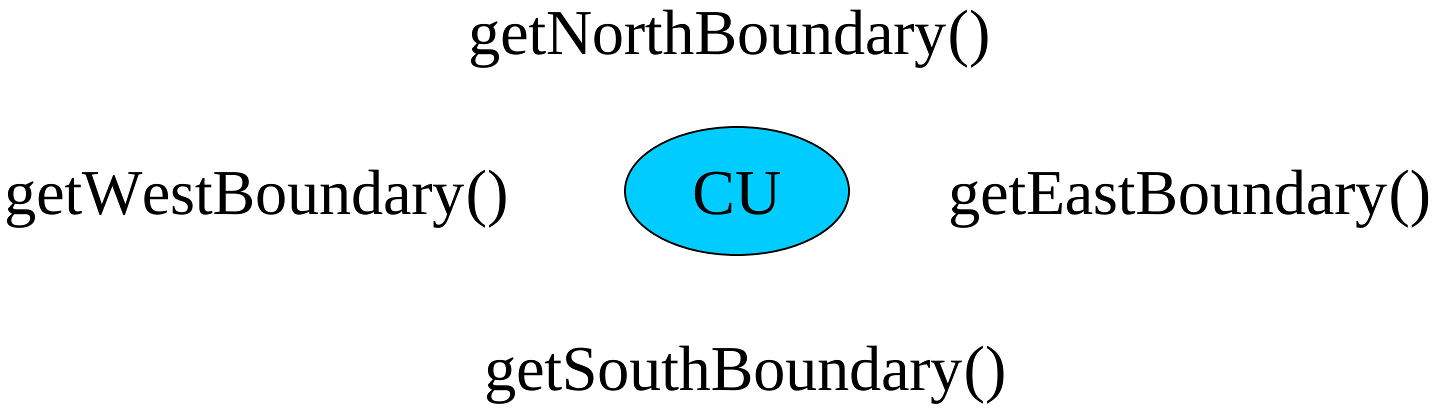
A Secure Computation System Business Logic

- A large simulation is partitioned into distributed objects, called Computational Units (CU).
- Each CU has four neighbors: east, west, south, north.
- CUs exchange specific boundaries with specific neighbors.



Interface of a CU

- Each CU provides four operations on its interface for its neighbors to retrieve data.



getNorthBoundary()

getWestBoundary() CU getEastBoundary()

getSouthBoundary()

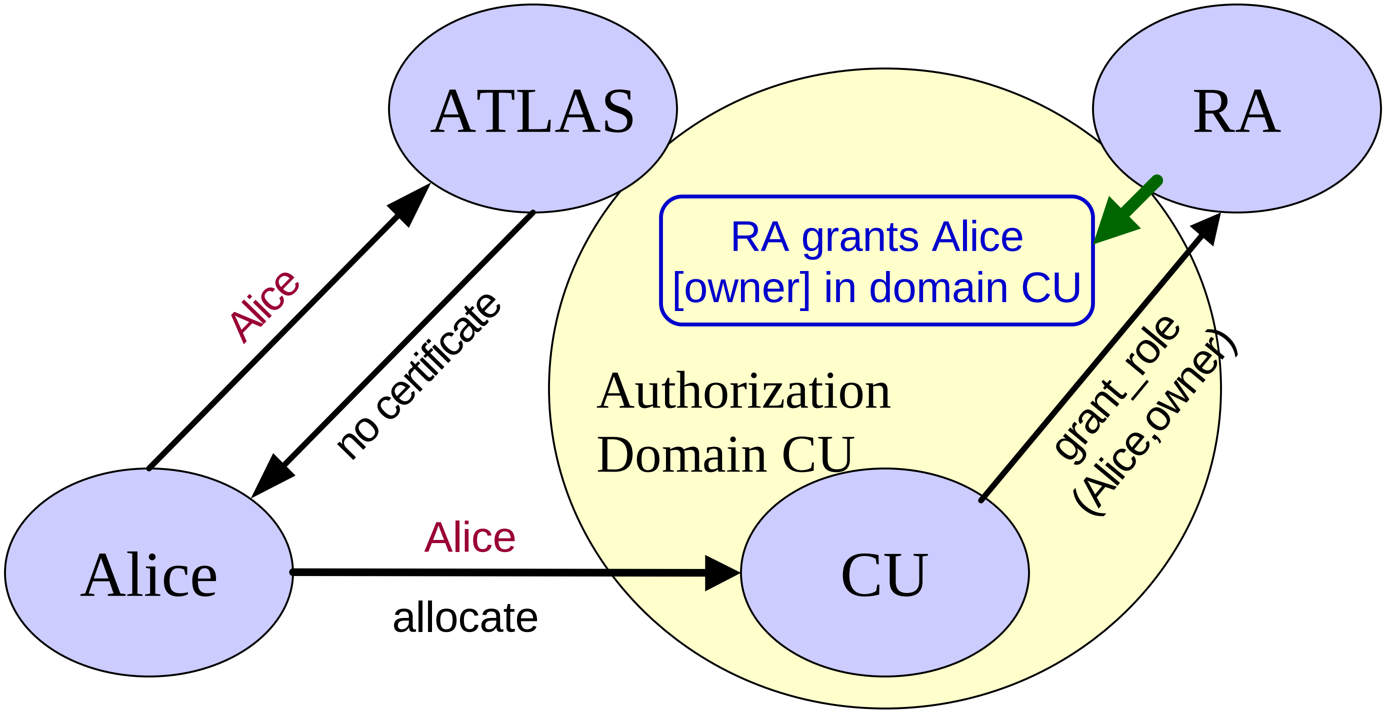
Access Control Policy of a CU

- Each CU models its neighbors as roles and defines its access control policy according to the business logic of the application:

east_neighbor	**cando**	[getEastBoundary]
west_neighbor	**cando**	[getWestBoundary]
south_neighbor	**cando**	[getSouthBoundary]
north_neighbor	**cando**	[getNorthBoundary]
anyone	**cando**	[allocate]
owner	**cando**	[release, setNeighbor, calculate, ...]

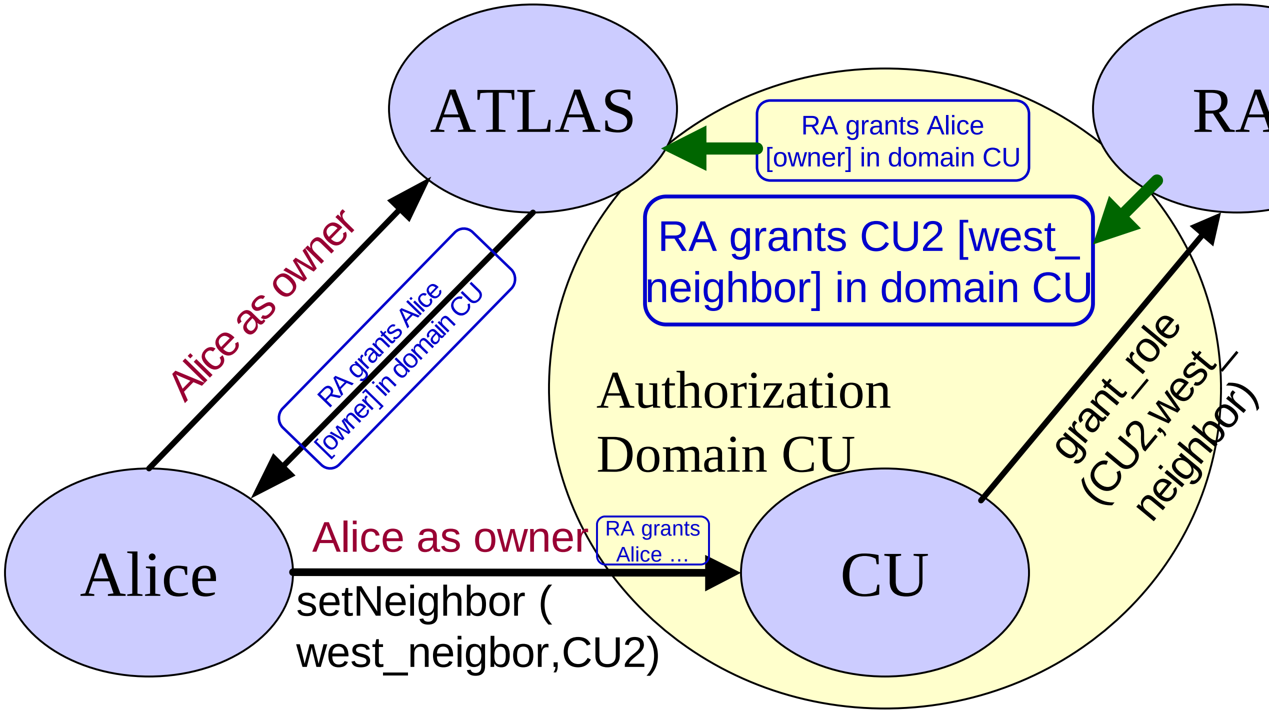
Allocating a CU

- CU administers its own authorization domain.



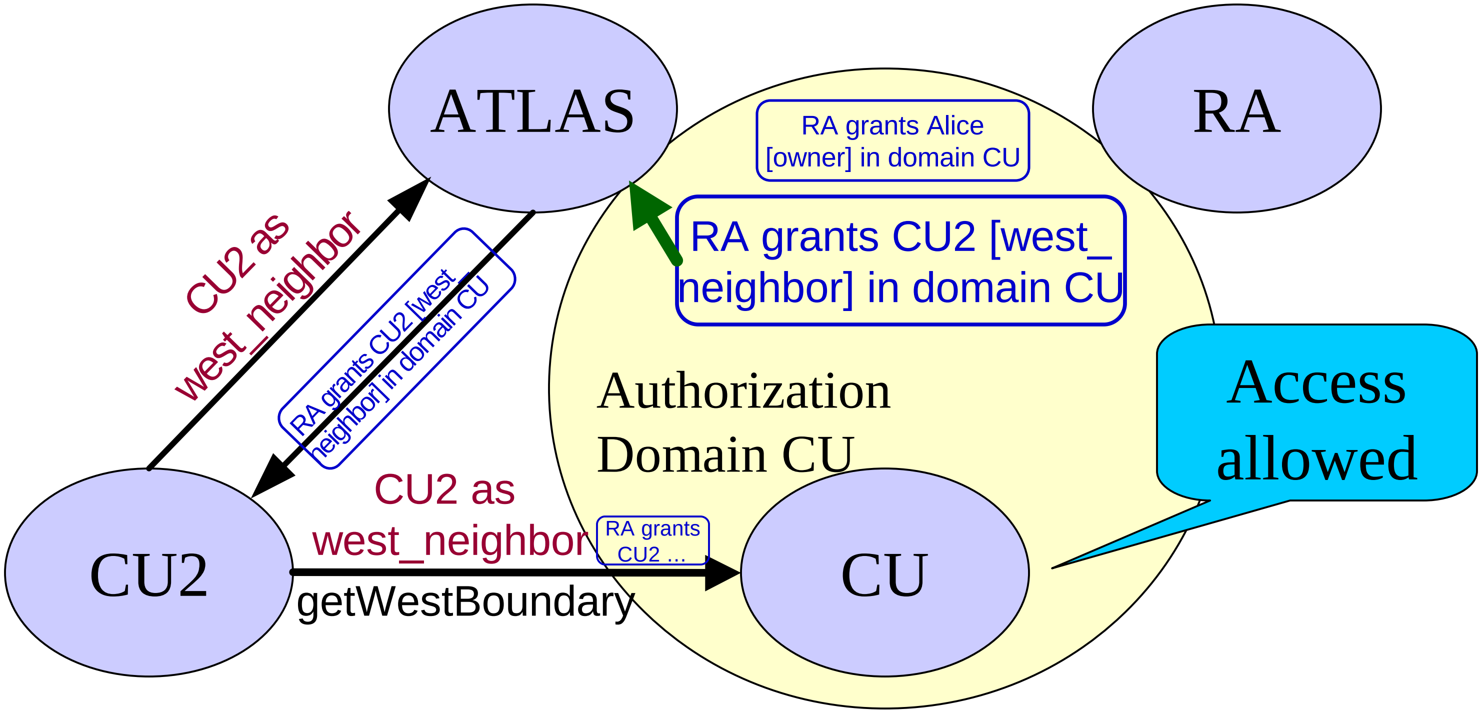
Configuring a CU

- The “owner” of a CU configures the business of the CU.


ATLAS
RA
RA grants Alice [owner] in domain CU
RA grants CU2 [west_ neighbor] in domain CU
Alice as owner
RA grants Alice [owner] in domain CU
Authorization Domain CU
grant_role (CU2,west_ neighbor)
Alice
Alice as owner
RA grants Alice …
CU
setNeighbor (west_neigbor,CU2)

Business of a CU

- The west neighbor access the west boundary.


ATLAS
RA
RA grants Alice [owner] in domain CU
RA grants CU2 [west_ neighbor] in domain CU
CU2 as west_neighbor
RA grants CU2 [west_ neighbor] in domain CU
Authorization Domain CU
Access allowed
CU2
CU2 as west_neighbor
RA grants CU2 ...
getWestBoundary
CU

Highlights

- Access control policy specifies permissions (as allowed operations) with respect to roles, not individual principals.
- Access control policy closely follows the static business logic of a CU.

Experimental Implementation

- Implemented at Center for Systems Assurance at Syracuse University
- Using:
  - ORBAsec SL3 from Adiron, LLC
  - ORBacus from IONA
  - JCSI from Wedgetail for Kerberos
  - iSaSiLk from IAIK for SSL
  - J2SDK from Javasoft

Potential Problems

- Revocation of Role Certificates
  - CUs need to revoke role certificates when released.
  - Can be done by means of various cost
    - Certificate Revocation Lists (expensive)
    - Unique Authorization Domain Names that are coordinated with the CU. (less expensive)
  - Not yet thoroughly investigated.

Conclusion

- The CORBA standards CSIv2 and ATLAS are effective in implementing RBAC in a dynamic fashion.