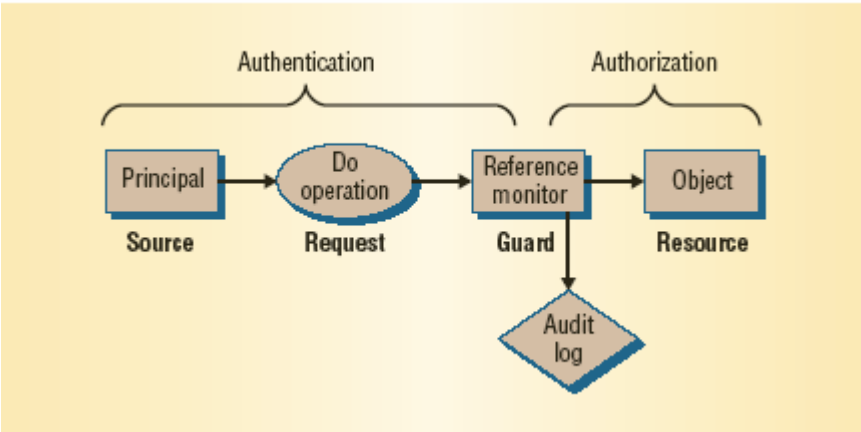
Protection and Security

Issues:

authentication: verifying a claim of identity
authorization: verifying a claim of permission
audit: verifying the (non)occurrence of previous actions



Reference Monitor Model

- **Au**thentication
- **Au**thorization
- **Au**dit

(**Au** = gold)

aka: AAA

From: “Computer Security in the Real World”, Lampson, 2004.

Security Goals and Principles

Goals:

- integrity - modification only by authorized parties
- confidentiality - access only by authorized parties
- non-repudiation - inability to disclaim authorship
- authenticity - verifiability of source
- availability - continuous access by authorized parties

Principles:

- least privilege - minimization of rights
- separation of duties (by task, by person)
- economy of mechanism - simplest means of enforcement
- acceptability – adoptable/usable by user community
- complete mediation - universal enforcement of control
- open design - secrecy of enforcement mechanisms is not important

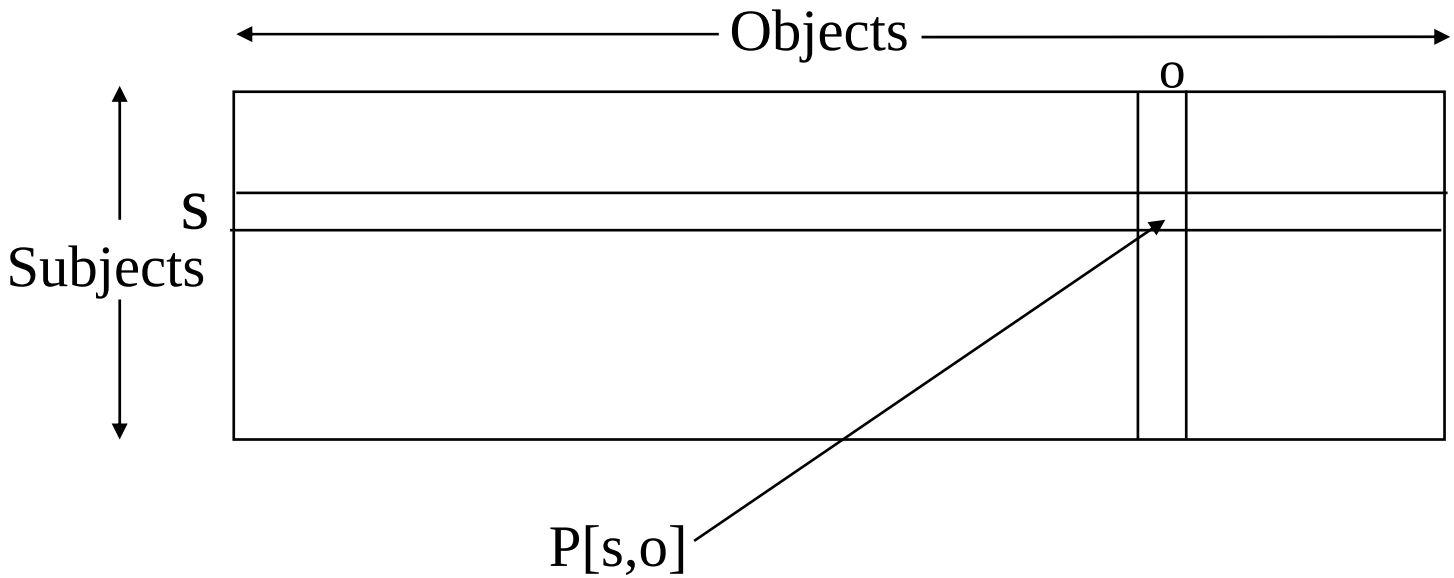
Elements of a Secure System

- Specification/Policy
  - secrecy
  - integrity
  - availability
  - accountability
- Implementation/Mechanism
  - isolation (impractical)
  - exclusion (code signing, firewalls)
  - restriction (sandboxing)
  - recovery
  - punishment
- Correctness/Assurance
  - trusted computing base
  - defense in depth
  - usability
  - theory

From: “Computer Security in the Real World”, Lampson, 2004

Access Matrix

Access Matrix Model



Access Matrix

objects

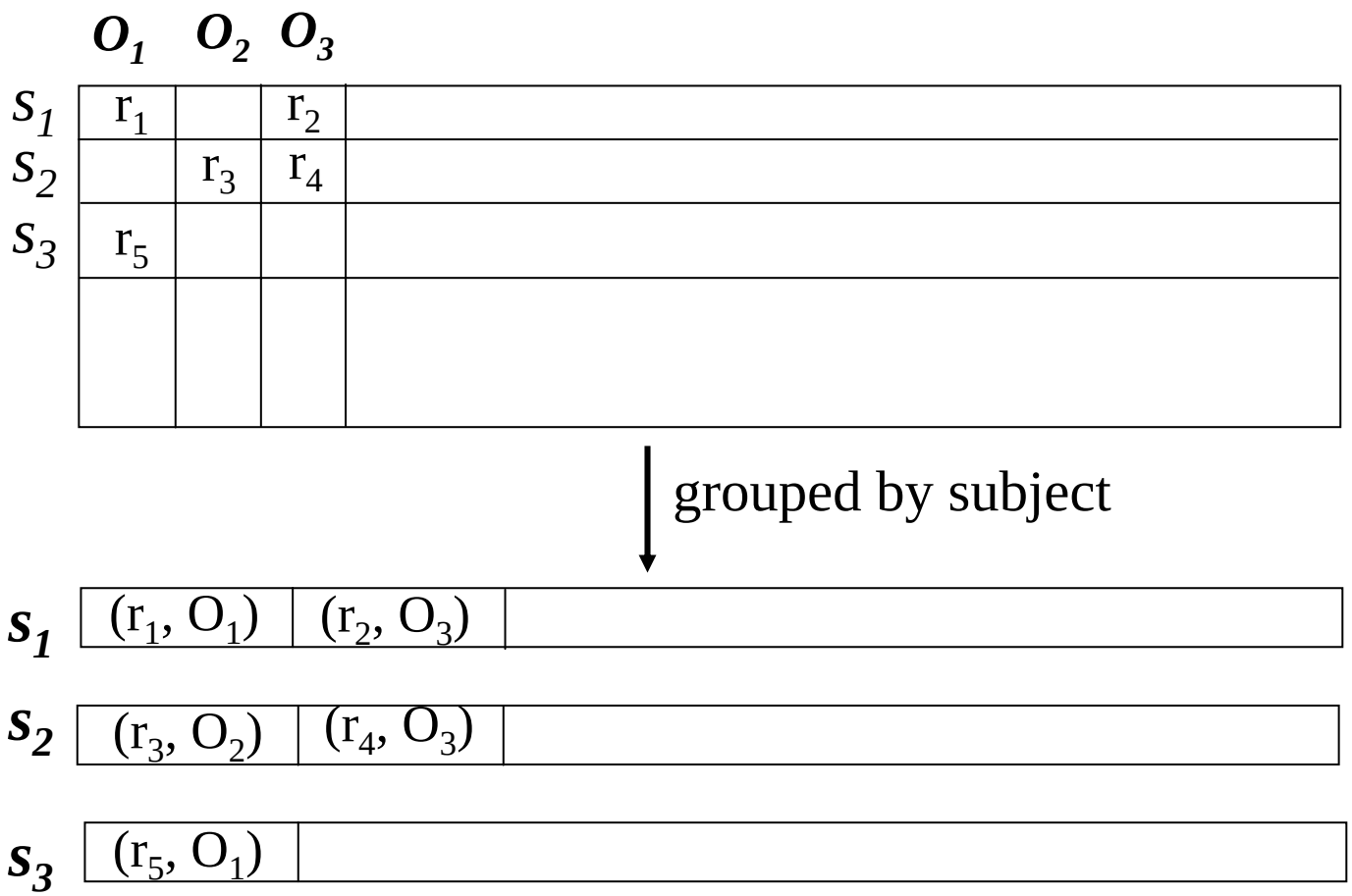
subjects

	S_1	S_2	S_3	F_1	F_2	D_1	D_2
S_1	control	owner block unblock	owner control	read* write*	read write	seek	owner
S_2	block unblock	control		owner	update	owner	seek*
S_3			control	delete	owner execute		

Manipulating the Access Matrix

Rule	Command (by S_0)	Conditions	Operation
R_1	transfer {a/a*} to S,X	a* in A[S_0,X]	store {a/a*} in A[S,X]
R_2	grant {a/a*} to S,X	*owner* in A[S_0,X]	store {a/a*} in A[S,X]
R_3	delete a from S,X	*control* in A[S_0,S] or *owner* in A[S_0,X]	delete a from A[S,X]
R_4	w = read S,X	*control* in A[S_0,S] or *owner* in A[S_0,X]	copy A[S,X] into w
R_5	create object X		add column for X to A; place *owner* in A[S,X]
R_6	destroy object X	*owner* in A[S_0,X]	delete column for X from A
R_7	create subject S		add a row for S to A; place *owner* in A[S0,S]; place *control* in A[S,S]
R_8	destroy subject S	*owner* in A[S_0,X]	delete row for S from A;

Capability Lists



Capability Lists

Access Control Lists

	O_1	O_2	O_3	
S_1	r_1		r_2	
S_2		r_3	r_4	
S_3	r_5			

Grouped by object

O_1	O_2	O_3
(s_1, r_1)	(s_2, r_3)	(s_1, r_2)
(s_3, r_5)		(s_2, r_4)

Access Control Lists

Role-Based Access Control (RBAC)

	O_1	O_2	O_3	
S_1	r_1	r_2		
S_2	r_1	r_2		
S_3		r_3	r_4	
S_4		r_3	r_4	
S_5		r_3	r_4	

grouped by multiple subjects

Role assignment

S_1	***Role*$_1$**
S_2	***Role*$_1$**
S_3	***Role*$_2$**
S_4	***Role*$_2$**
S_5	***Role*$_2$**

***Role*$_1$**	(r_1,O_1)	(r_2,O_2)	
***Role*$_2$**	(r_3,O_2)	(r_4,O_3)	

Privilege assignment

Role-Based Access Control (RBAC)

- Roles of model particular jobs or duties in an organization
- Single user may play multiple roles at the same or different times
- Multiple users may play the same role at the same or different times
- The user-role assignment may be made separately from the role-permission assignment

Classes, Levels, Domains

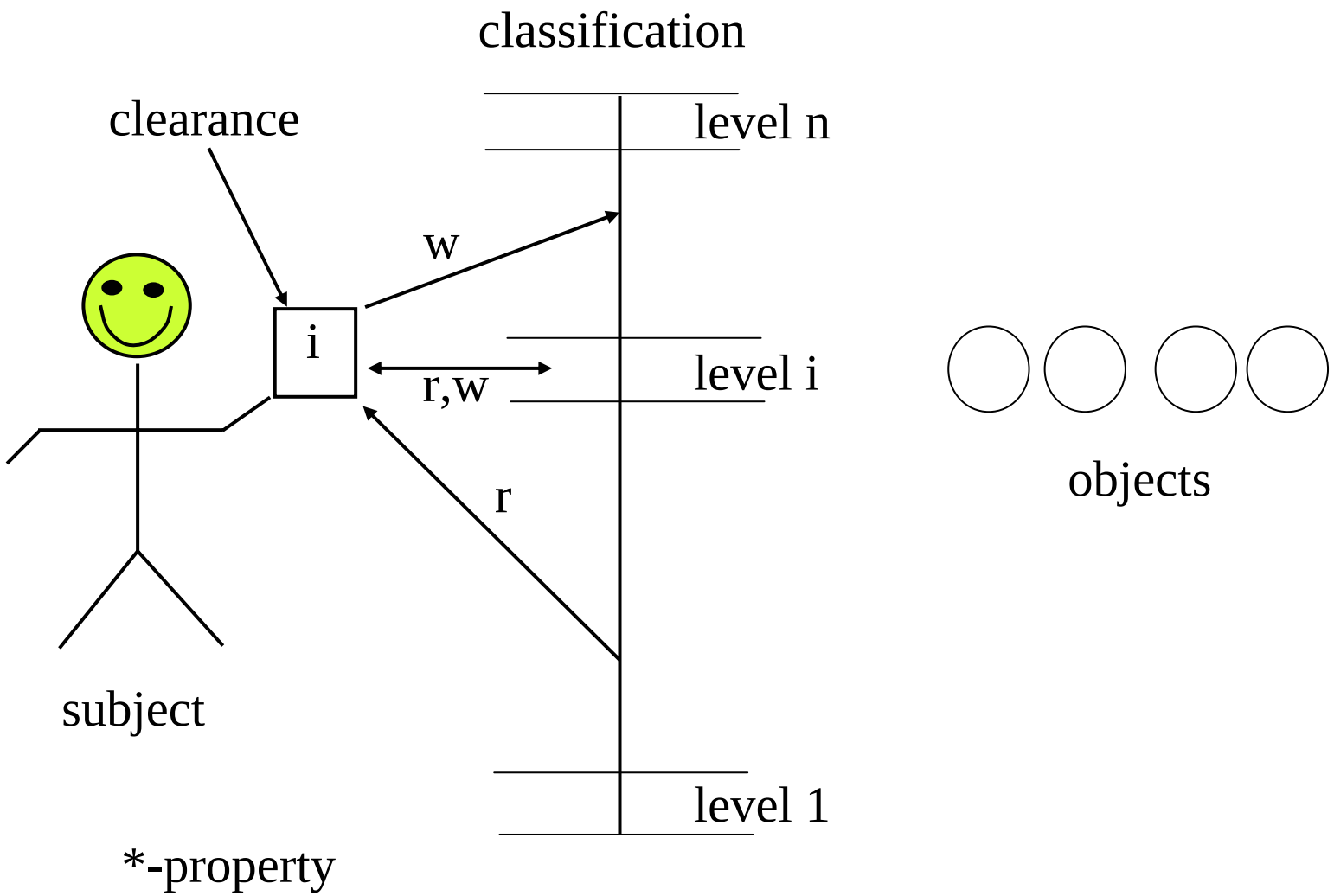
	O_1	O_2	O_3	O_4	O_5	
S_1	r_1	r_1		r_1		
S_2			r_1	r_3	r_1	
S_3	r_2	r_2	r_3		r_3	

Grouped by multiple objects

O_1 & O_2	O_3 & O_5	O_4
(s_1, r_1)	(s_2, r_1)	(s_1, r_1)
(s_3, r_2)	(s_3, r_3)	(s_2, r_3)

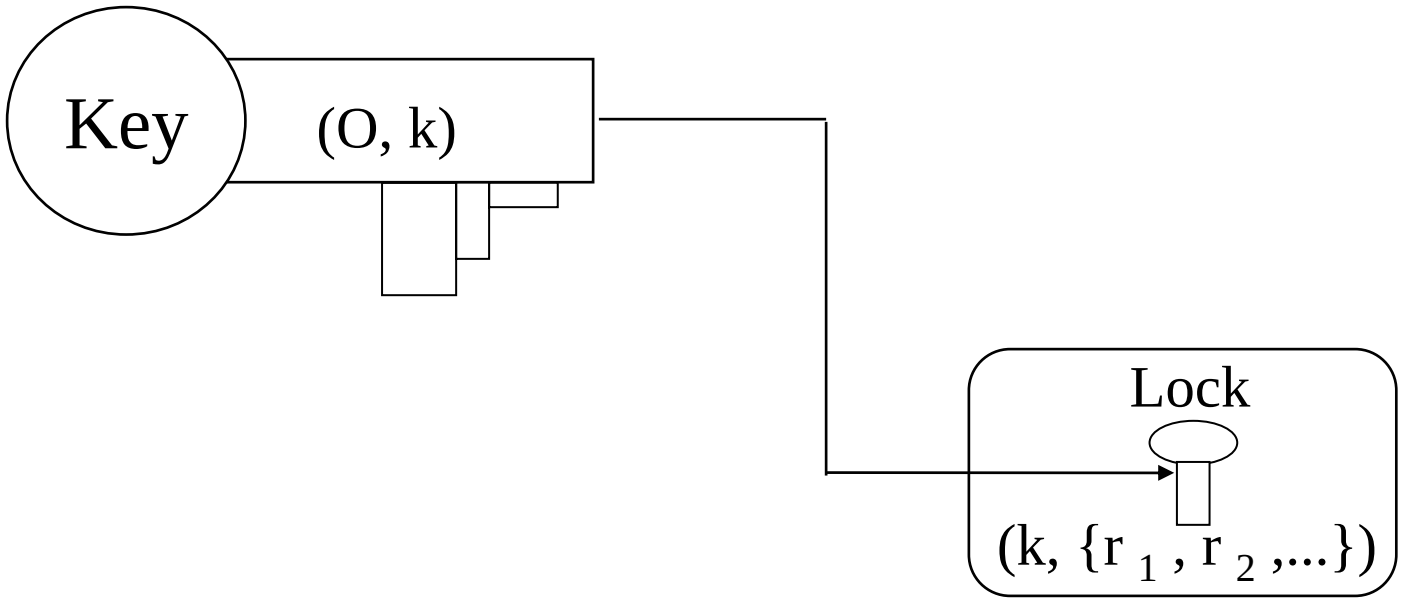
classes, levels, domains

Bell-LaPadula Model



Lock and Key Method

subjects possess a set of keys:



objects are associated with a set of locks:

Comparison of methods

	Capability list	Access Control links	Locks & Keys
propagation	 1	 3	 1
review			 4
revocation			 4
reclamation	 2		

1. need copy bit/count for control
2. need reference count
3. need user/hierarchical control
4. need to know subject-key mapping