Cours

16 mars Politiques de sécurité et contrôle d’accès

23 mars Politiques de sécurité et contrôle d’accès (suite)

30 mars Vers le contrôle des flots d’information

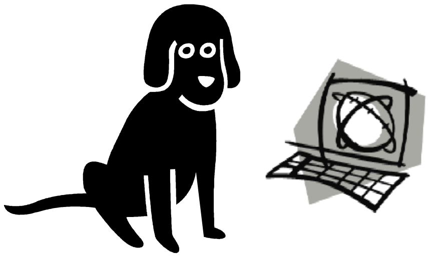
6 avril La fiabilité du logiciel

27 avril La cryptographie

4 mai « Sur Internet, personne ne sait que vous êtes un chien », vingt ans après

11 mai Les protocoles

18 mai Assurance et modèles formels



Security policies and access control

Chaire Informatique et sciences numériques
Collège de France, cours du 16 mars 2011

Security policies and mechanisms

Specification and implementation

For any system:

- Specification: *What is it supposed to do?*
- Implementation: *How does it do it?*
- Correctness: *Does it really work?*

In security:

- Specification: *Policy*
- Implementation: *Mechanism*
- Correctness: *Assurance*

Caveats

But:

- Some mechanisms are presented as policies.
- Mechanisms sometimes come before policies.
- Assurance can guide policies and mechanisms.
- Assurance is sometimes replaced with "security by obscurity".
- Attacks can exploit gaps at any level.

Security properties

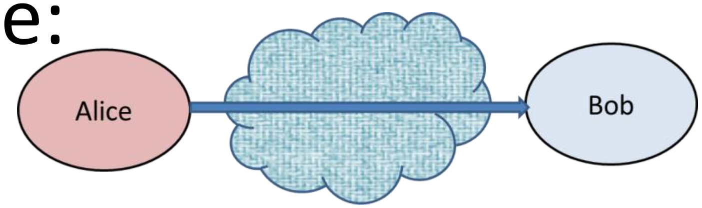
The main security properties are:

- ***Integrity properties***
  (no improper modification of information)
- ***Secrecy properties***
  (no improper disclosure of information)
- ***Availability properties***
  (no improper denial of service)

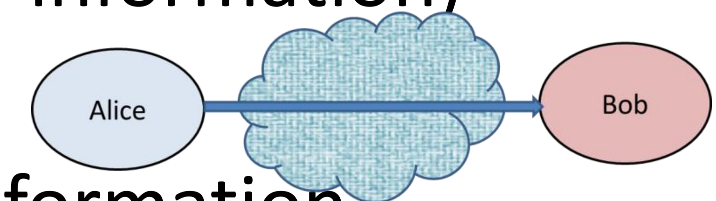
Security properties

The main security properties are:

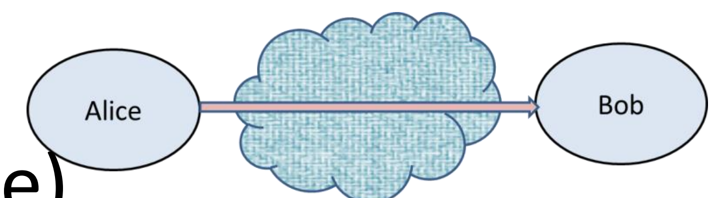
- ***Integrity properties***
  (no improper modification of information)



- ***Secrecy properties***
  (no improper disclosure of information)



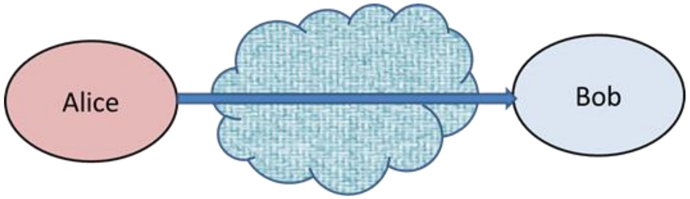
- ***Availability properties***
  (no improper denial of service)



Variations on integrity

Authenticity is often the same as integrity,

- with a difference only in emphasis,
- or with a requirement of freshness.



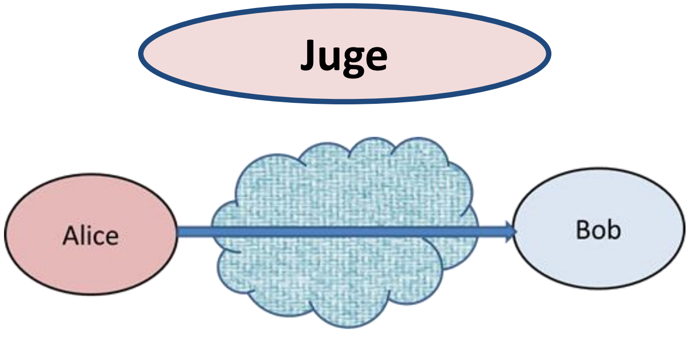
Variations on integrity

Authenticity is often the same as integrity,

- with a difference only in emphasis,
- or with a requirement of freshness.

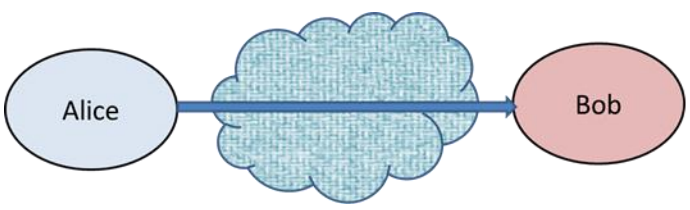
Other concepts are closely related to integrity:

- ***non-repudiation***,
- ***accountability***.



Variations on secrecy

- Similarly, ***confidentiality*** is basically secrecy.
- So is ***privacy***, often, in the context of personal information. (More on this later.)
- ***Anonymity*** is basically an instance of secrecy.
- ***Pseudonymity*** is anonymity plus linkability.
- ***Plausible deniability*** is the contrary of non-repudiation and might be viewed as a weak form of secrecy.



Security policies

Security properties are combined into security policies. For example, a bank may want:

- authenticity of clients at ATMs, on the Web,
- non-repudiation of transactions,
- integrity of the books,
- integrity of the messaging systems,
- secrecy for client data and for internal data,
- availability of the alarm system.

Security policies (cont.)

Policies may include less standard properties:

- exclusivity of duties (re. conflicts of interest),
- dual control for sensitive transactions.

Security properties are often in conflict

- because of the conflicting goals of each party (e.g., integrity vs. secrecy),
- because each party has its own goals (e.g., anonymity vs. non-repudiation).

HOME / BUSINESS / TECHNOLOGY

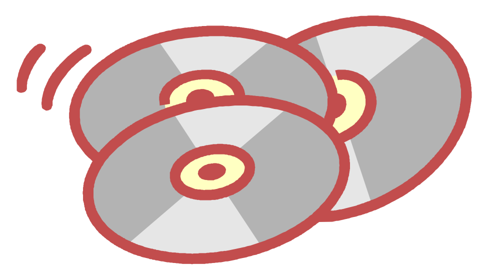
The Boston Globe

Security firm: Sony CDs secretly install spyware

Company denies it, saying program aims to foil music piracy

By Hiawatha Bray

Globe Staff / November 8, 2005



Basics of access control

Access control

Access control is prominent at many levels:

- memory-management hardware,
- operating systems, file systems, and the like,
- middleware,
- applications,
- firewalls,

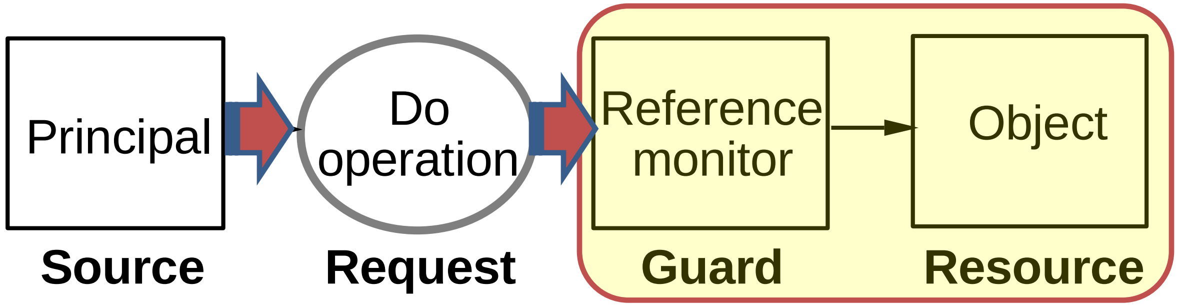
and also in physical protection.

Access control (cont.)

- Access control is a **mechanism**.
  - It aims to guarantee secrecy, integrity, and availability properties, and more.
- Access control can also be seen as a **model**, as specification for lower-level mechanisms.
  - (Higher-level policies are often not explicit.)

The access control model

- Elements:
  - Objects or resources
  - Requests
  - Sources for requests, called principals (or subjects)
  - A reference monitor to decide on requests



An access control matrix

[Lampson, 1971]

objects / principals	file1	file2	file3	file4
user1	rwx	rw	r	x
user2	r	r		x
user3	r	r		x

Implementing access control

Authentication

Access control depends on **authentication**:

- Access control (authorization):
  - Is principal *A* trusted on statement *s*?
  - If *A* requests *s*, is *s* granted?
- Authentication:
  - Who says *s*?

Other machinery

- Auditing
- Recovery

  …

The reference monitor and mediation

The principle of complete mediation

[Saltzer and Schroeder, 1975]

Every access to every object must be checked for authority.

This principle can be enforced in several ways:

- The OS intercepts some of the requests.
  The hardware catches others.
- A software wrapper / interpreter intercepts some of the requests. (E.g., as in VMs.)

Strategies for representing an access control matrix

In practice, a matrix is typically represented in terms of ACLs and capabilities.

- ACL: a column of an access control matrix, attached to an object.
- Capability: (basically) a pair of an object and an operation, for a given principal.
  It means that the principal may perform the operation on the object.

More on ACLs

principals \ objects	file1	file2	file3	file4
user1	rwx	rw	r	x
user2	r	r		x
user3	r	r		x

- An ACL says which principals can access a particular object.
  - It is a column of an access control matrix,
  - typically maintained "near" the object that it protects.
- ACLs can be compact and easy to review.
- Revoking a principal can be painful.

More on capabilities

objects / principals	file1	file2	file3	file4
user1	rwx	rw	r	x
user2	r	r		x
user3	r	r		x

- An alternative is to associate capabilities with each principal.
  - A capability means that the principal can perform an operation on an object.
- These capabilities form a row of an access control matrix for the principal
- Capabilities are often easy to pass around (so they enable delegation).
- They can be hard to review and to confine.

Implementing capabilities

$\Rightarrow$ *Principals should not be allowed to forge capabilities.*

This leads to implementations of capabilities

- stored in a protected address space, or
- with special tags with hardware support, or
- as references in a typed language, or
- with a secret, or
- with cryptography, e.g., certificates.

ACLs vs. capabilities

- ACLs and capabilities are dual.
- Both can yield practical implementations of access matrices.
- In actual systems, they are often combined.

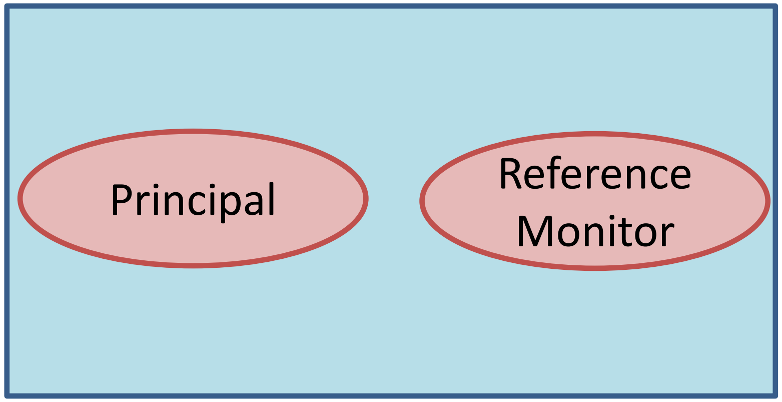
push vs. pull

- The reference monitor relies on proofs of identity, the access policy, and other evidence.
- It can gather this evidence by two methods:

push vs. pull

- The reference monitor relies on proofs of identity, the access policy, and other evidence.
- It can gather this evidence by two methods:

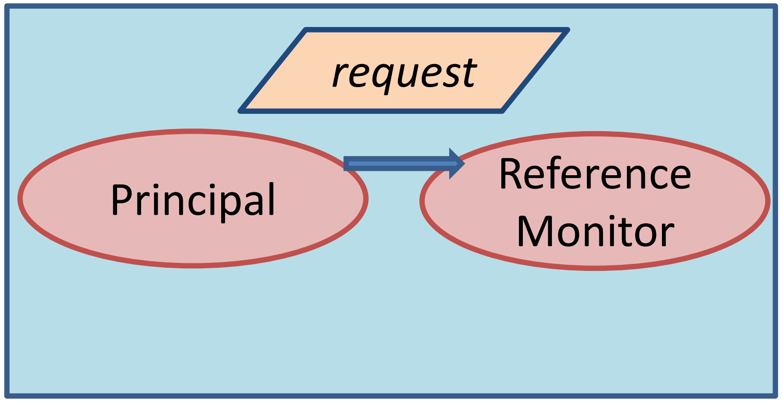
push: Principals present evidence with requests



push vs. pull

- The reference monitor relies on proofs of identity, the access policy, and other evidence.
- It can gather this evidence by two methods:

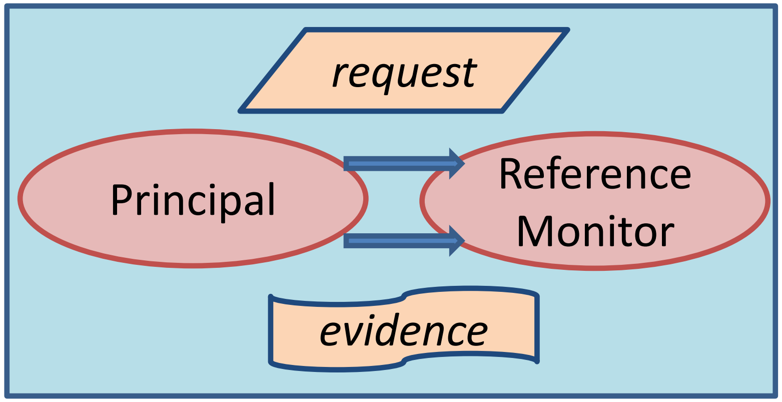
push: Principals present evidence with requests



push vs. pull

- The reference monitor relies on proofs of identity, the access policy, and other evidence.
- It can gather this evidence by two methods:

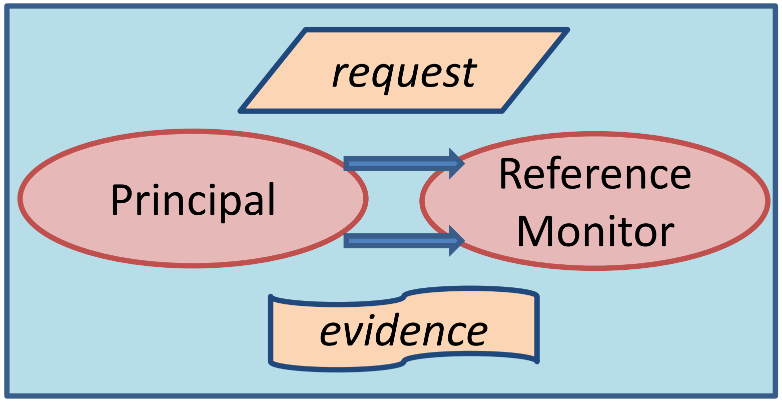
push: Principals present evidence with requests



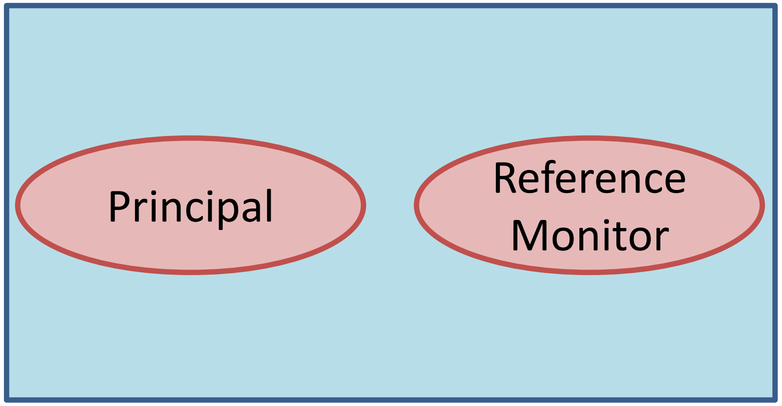
push vs. pull

- The reference monitor relies on proofs of identity, the access policy, and other evidence.
- It can gather this evidence by two methods:

push: Principals present evidence with requests



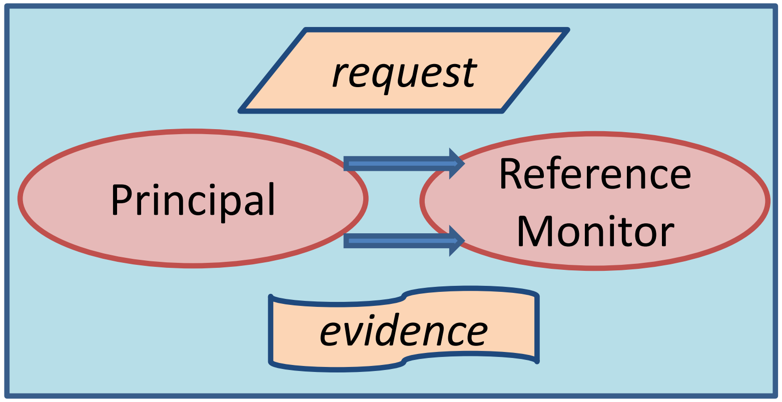
pull: Reference monitors gather evidence, with help from others.



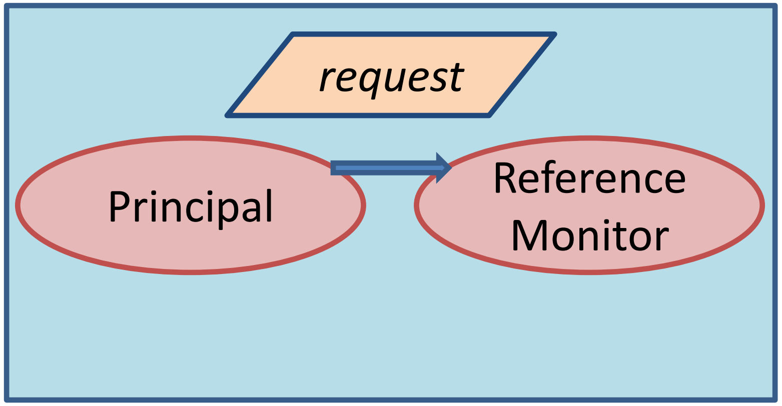
push vs. pull

- The reference monitor relies on proofs of identity, the access policy, and other evidence.
- It can gather this evidence by two methods:

push: Principals present evidence with requests



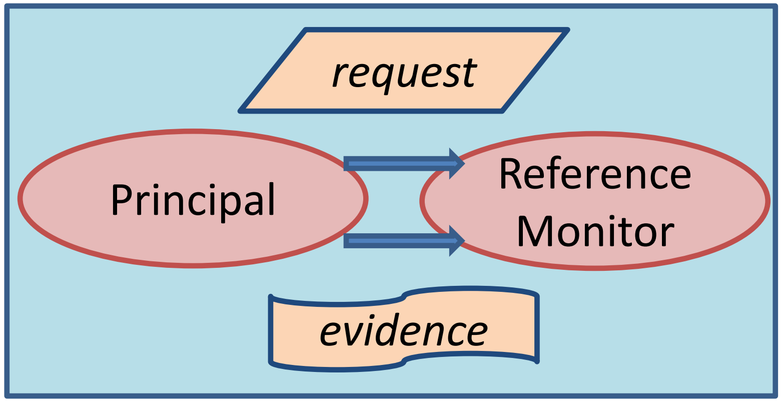
pull: Reference monitors gather evidence, with help from others.



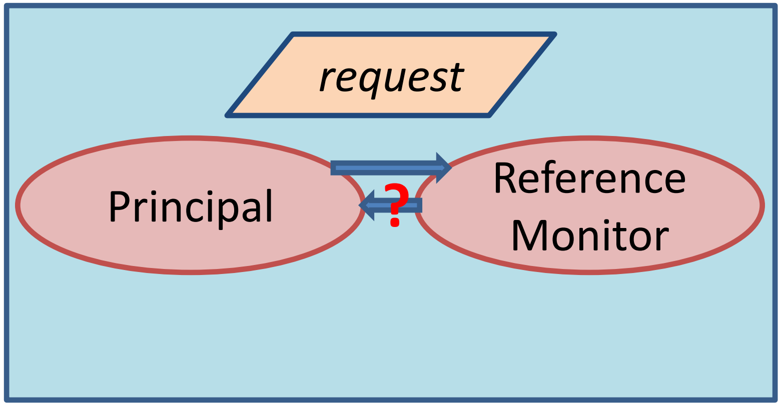
push vs. pull

- The reference monitor relies on proofs of identity, the access policy, and other evidence.
- It can gather this evidence by two methods:

push: Principals present evidence with requests



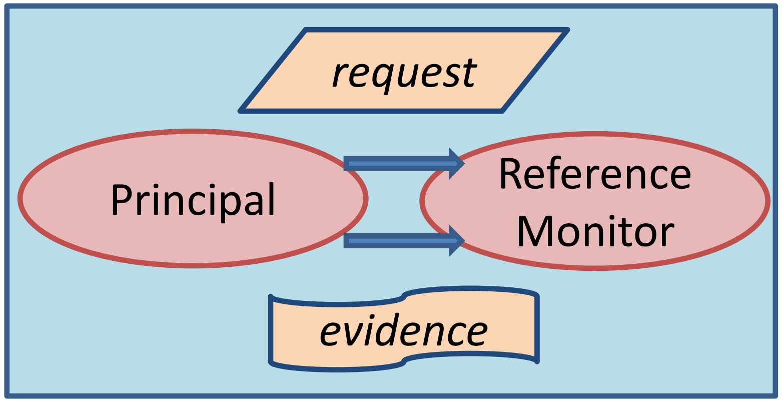
pull: Reference monitors gather evidence, with help from others.



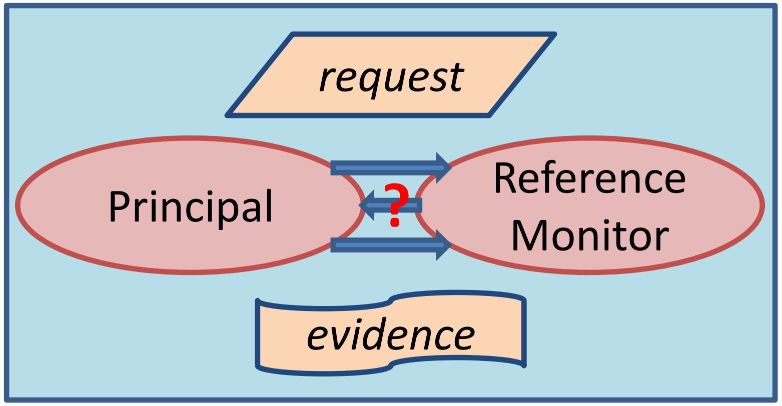
push vs. pull

- The reference monitor relies on proofs of identity, the access policy, and other evidence.
- It can gather this evidence by two methods:

push: Principals present evidence with requests



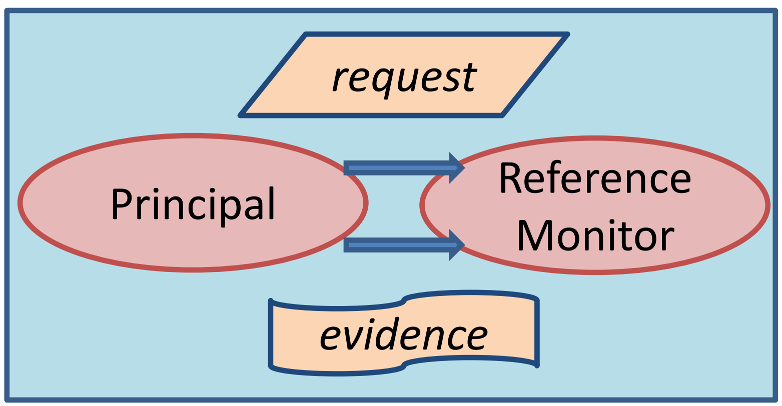
pull: Reference monitors gather evidence, with help from others.



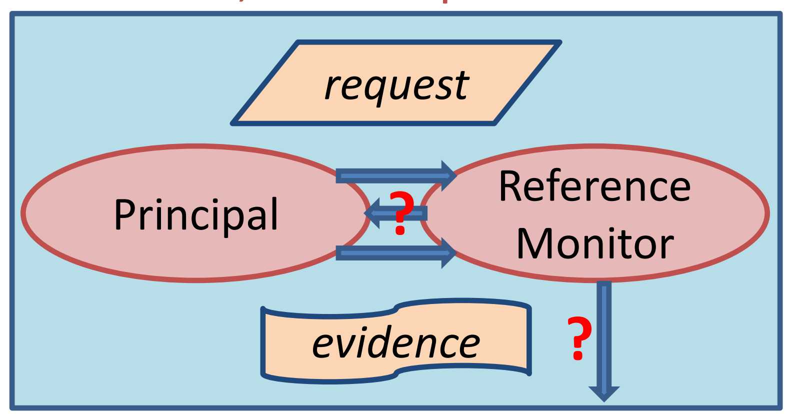
push vs. pull

- The reference monitor relies on proofs of identity, the access policy, and other evidence.
- It can gather this evidence by two methods:

push: Principals present evidence with requests



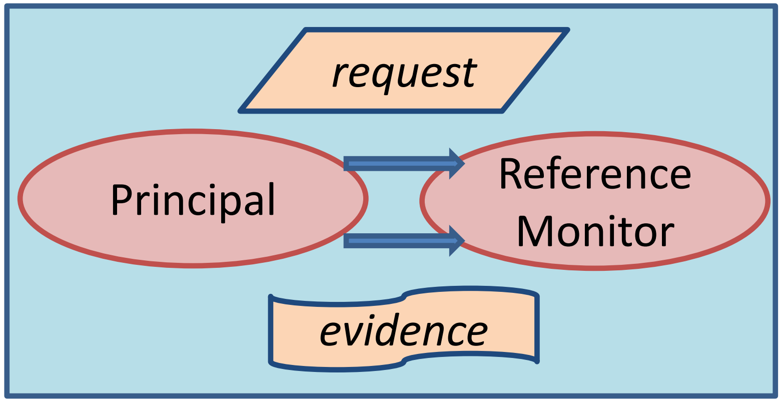
pull: Reference monitors gather evidence, with help from others.



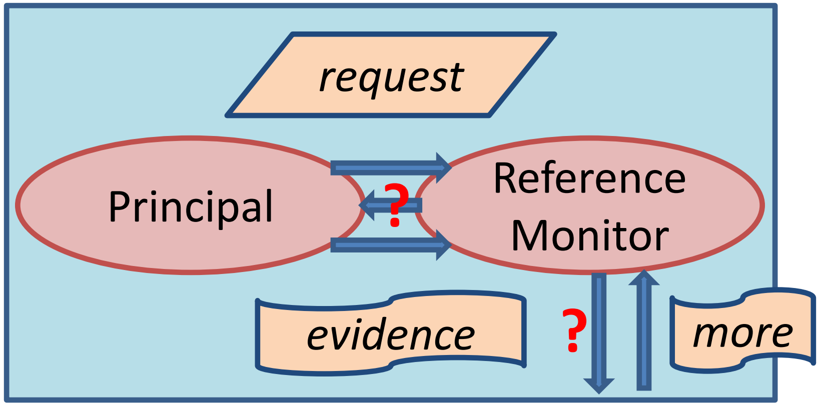
push vs. pull

- The reference monitor relies on proofs of identity, the access policy, and other evidence.
- It can gather this evidence by two methods:

push: Principals present evidence with requests



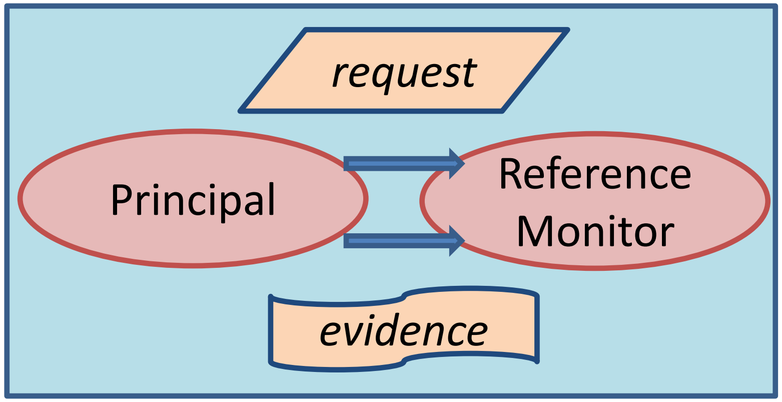
pull: Reference monitors gather evidence, with help from others.



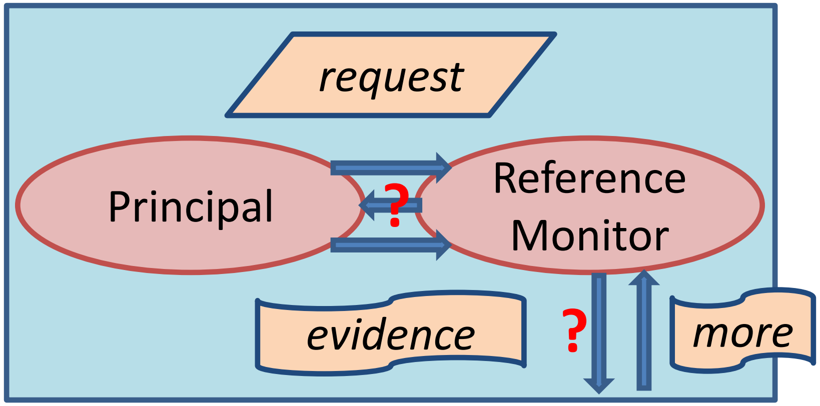
push vs. pull

- The reference monitor relies on proofs of identity, the access policy, and other evidence.
- It can gather this evidence by two methods:

push: Principals present evidence with requests



pull: Reference monitors gather evidence, with help from others.



- Concerns: completeness, efficiency, privacy.

Embellishments and complications

Principals

Principals may be

- users,
- programs,
- computers,
- origins (in browsers),
- their combinations,
- …

On principals

The notion of principal varies (dangerously) across systems and abstraction layers.

For example, one should not confuse

- IP addresses (e.g., 118.214.218.135),
- domains (e.g., whitehouse.gov),
- the computers at those addresses,
- the people who control the computers.

Some further elaborations

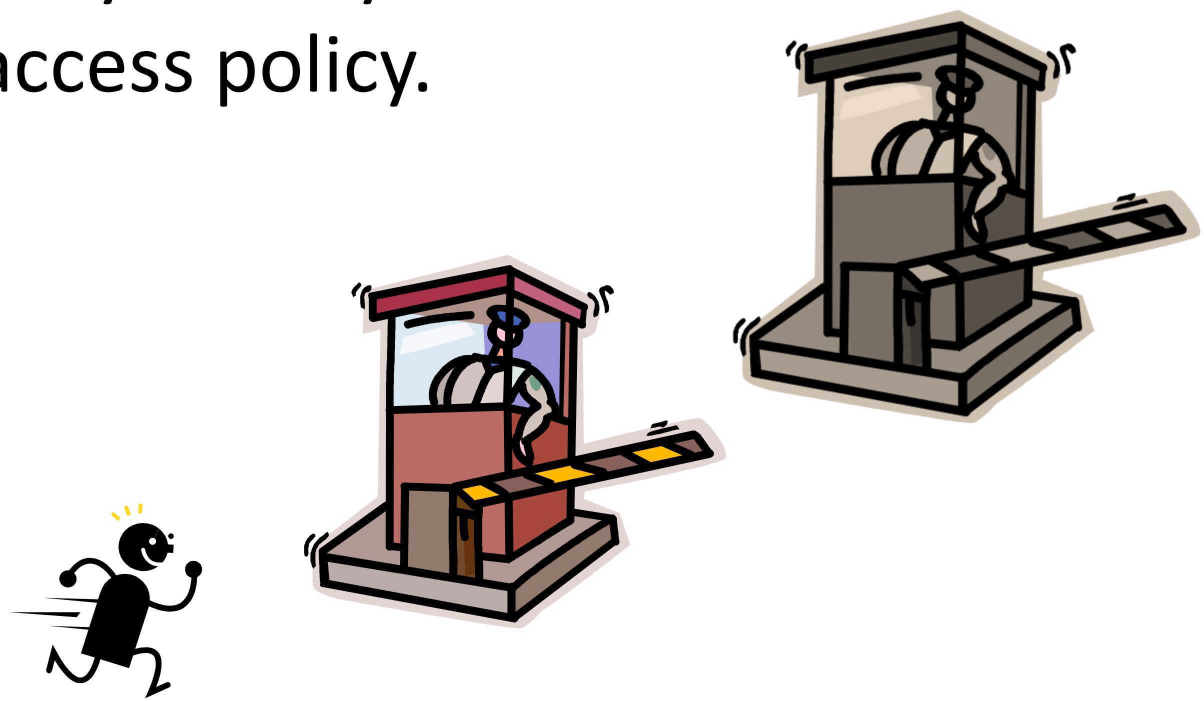
- Joint requests
- Groups
- Roles
- Negation
- Delegation
- Programs (discussed in the next lecture)

Conjunctions

- Sometimes a request should be granted only if it is made jointly by several principals.

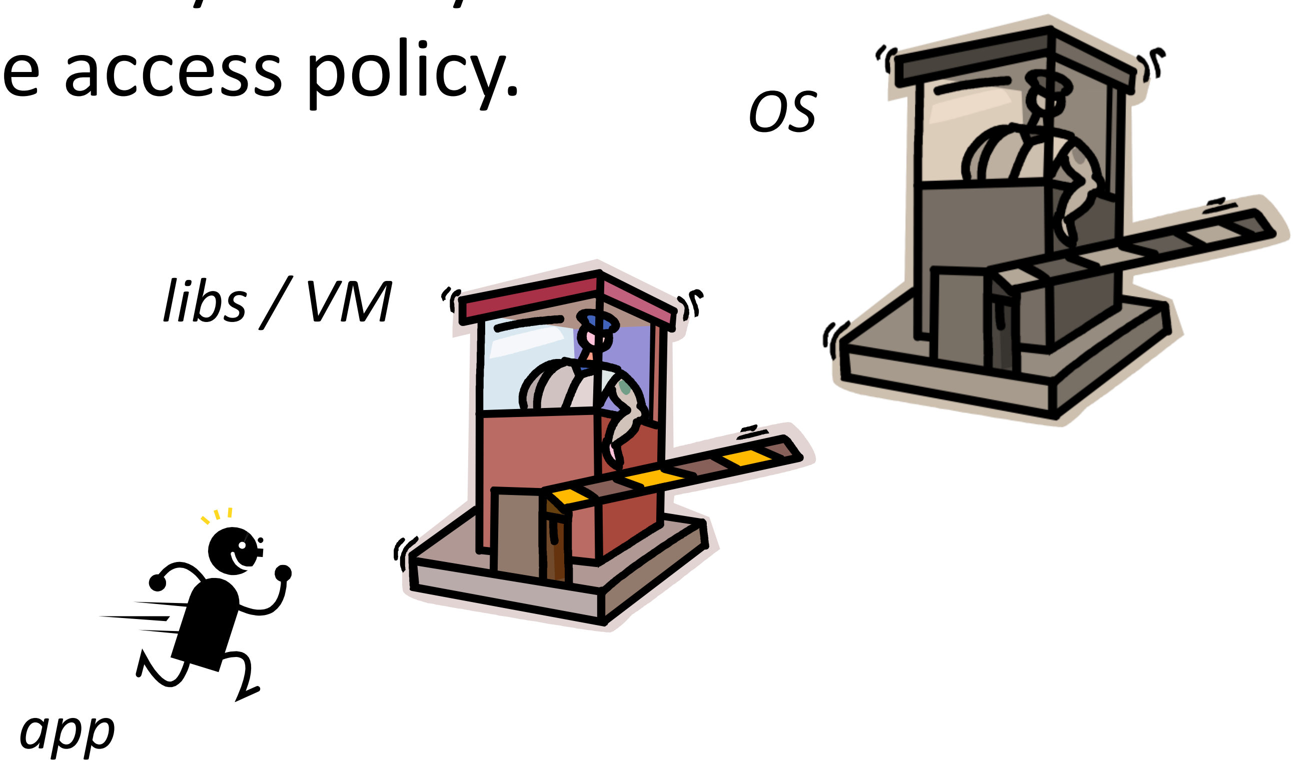
Conjunctions

- Sometimes a request should be granted only if it is made jointly by several principals.
- A conjunction may or may not be made explicit in the access policy.



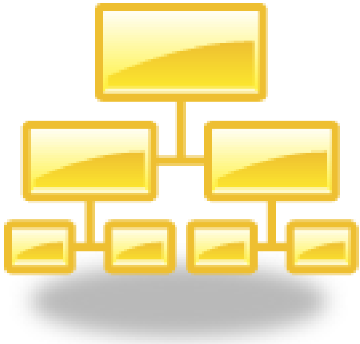
Conjunctions

- Sometimes a request should be granted only if it is made jointly by several principals.
- A conjunction may or may not be made explicit in the access policy.



Groups and roles

- Principals can be organized into groups.
- Principals can play roles.
- These groups and roles may be used as a level of indirection in access control.
  - E.g., any member of a group G may access a file f.
  - E.g., anyone who can adopt the role R may then access a file f.



Groups and roles (cont.)

- Suppose that any member of group G may access file f owned by Alice.
  - G may be maintained by someone else.
  - G may change over time, without immediate knowledge of Alice.
  - f’s ACL should be short and clear.
  - Proofs of memberships resemble (are?) capabilities.
  - Access to f may be partly anonymous.
  - Still, Alice may require a proof of identity at each f access, for auditing.

ACL for f (owned by Alice)
G

Members of G (owned by admin)
Alice
Bob
Charlie

On objects

Similarly, objects may include

- disk blocks,
- files,
- database tables, rows, and columns,
- application-level records, like calendar entries.

Picking objects is also an important part of designing an access control system.

On operations

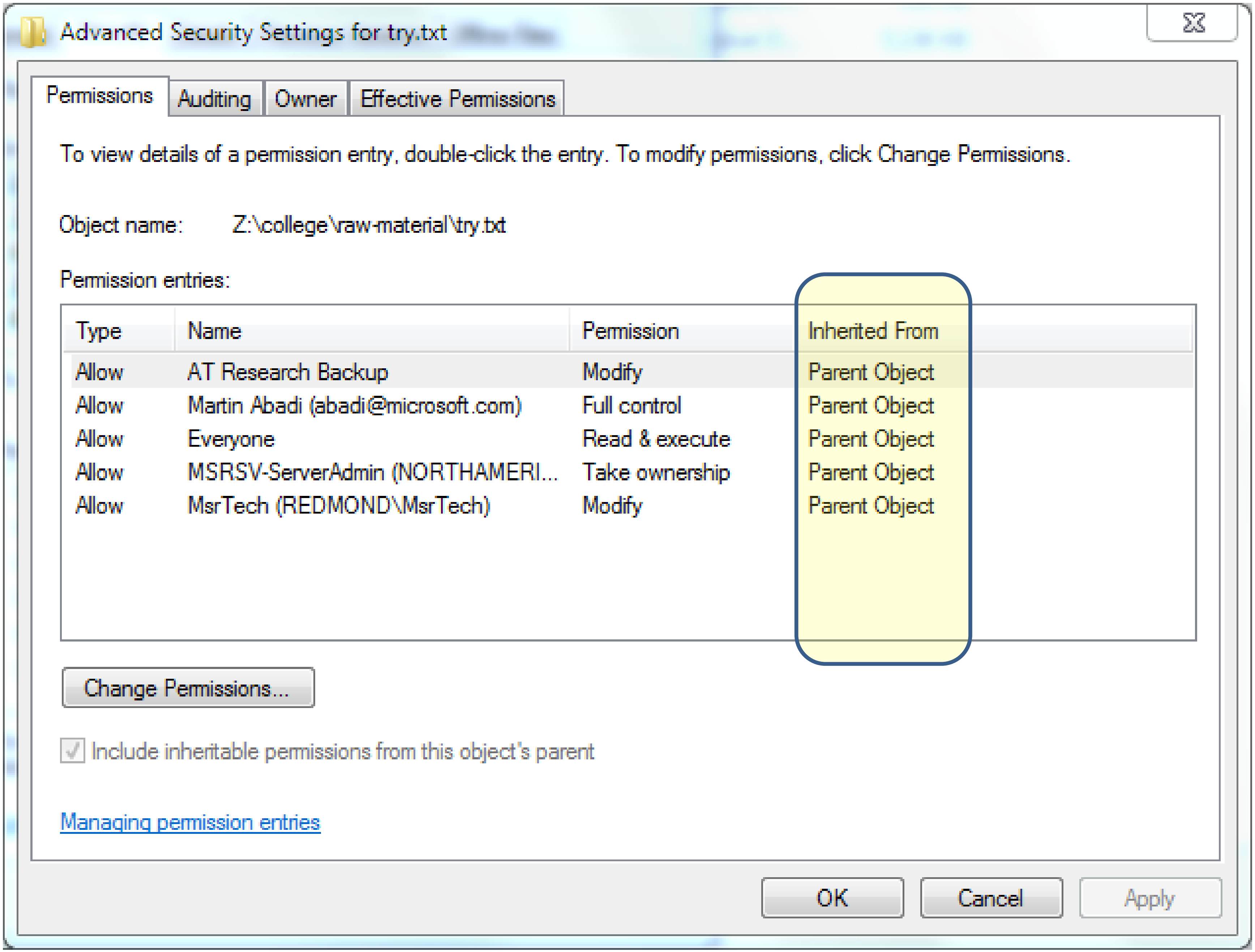
Similarly, too, there are important choices in defining operations.

In particular, sometimes “small” operations should be bundled to form “bigger” ones.

- E.g.,
  - read a patient's record,
  - write a log record (for auditing).
- A principal may be allowed to do a “big” operation but not each of its components.

More on objects and operations

- Objects and operations may also be put in groups, e.g.,
  - all company files,
  - all write operations (e.g., append) on an object.
- Moreover, some policy may be automatically inherited from object to object.


Advanced Security Settings for try.txt
Permissions
Auditing
Owner
Effective Permissions
To view details of a permission entry, double-click the entry. To modify permissions, click Change Permissions.
Object name: Z:\college\raw-material\try.txt
Permission entries:
Type
Name
Permission
Inherited From
Allow AT Research Backup Modify Parent Object
Allow Martin Abadi (abadi@microsoft.com) Full control Parent Object
Allow Everyone Read & execute Parent Object
Allow MSRSV-ServerAdmin (NORTHAMERI... Take ownership Parent Object
Allow MsrTech (REDMOND\MsrTech) Modify Parent Object
Change Permissions...
Include inheritable permissions from this object's parent
Managing permission entries
OK
Cancel
Apply

Design choices

- Principals, objects, and operations should have the “right” granularity and be at the “right” level of abstraction
  - for ease of understanding,
  - to avoid giving away too much privilege.

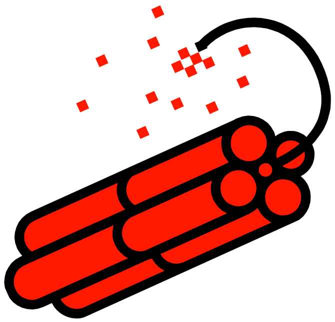
The principle of least privilege

[Saltzer and Schroeder, 1975]

Every program and every user of the system should operate using the least set of privileges necessary to complete the job.

Common dangers

- Access control can be insufficient or irrelevant
  - when it is implemented incorrectly,
  - when the underlying operations are implemented incorrectly,
  - when the policy is wrong,
  - when it is circumvented.

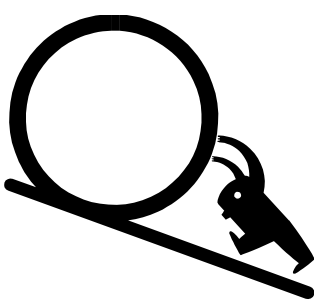


Further issues

- Many characteristics of distributed systems make access control harder:
  - size,
  - faultiness (e.g., revocations may get lost),
  - heterogeneity (e.g., of communication channels and of protection mechanisms),
  - autonomy, lack of central administration and therefore of central trust,
  - …
- Access control seems difficult to get right.

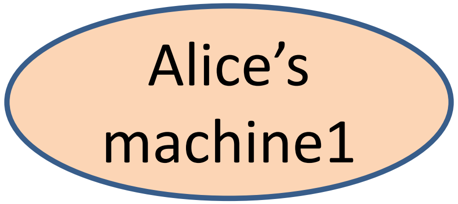
The *snowball effect*

- An illustration of the consequences of bad policies (particularly in distributed systems).
- Not a new problem, but still a problem.
- With a recent precise formulation and some research [Dunagan, Zheng, and Simon].



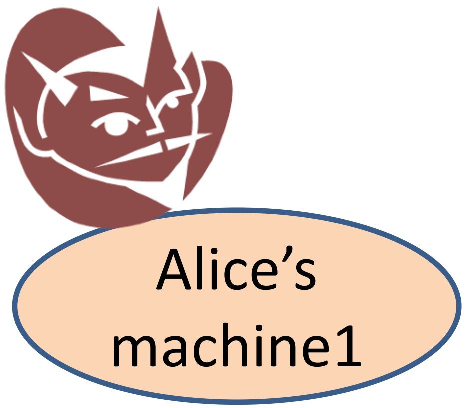
The snowball effect

An illustration of the consequences of bad policies (particularly in distributed systems):



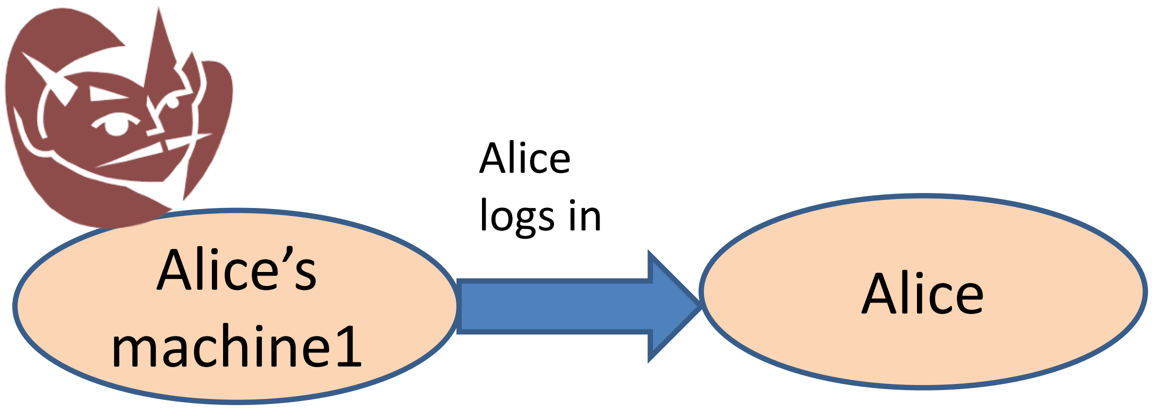
The snowball effect

An illustration of the consequences of bad policies (particularly in distributed systems):



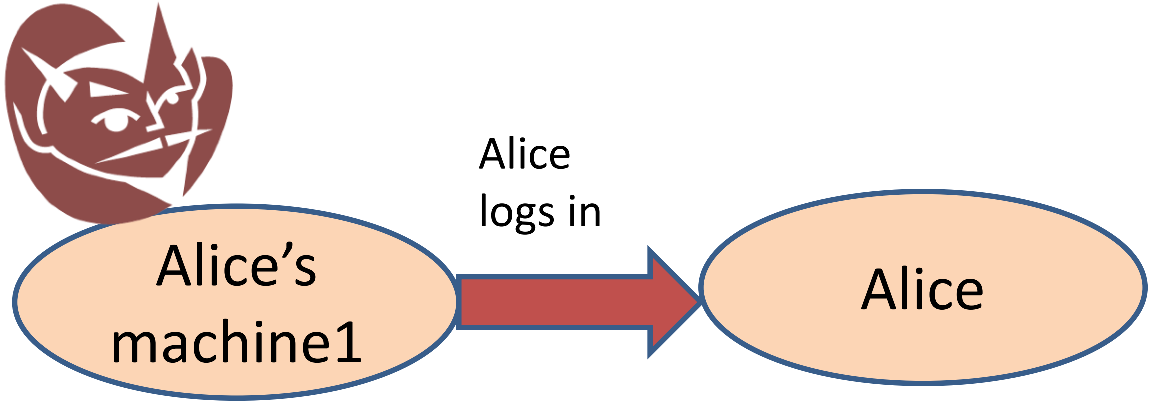
The snowball effect

An illustration of the consequences of bad policies (particularly in distributed systems):



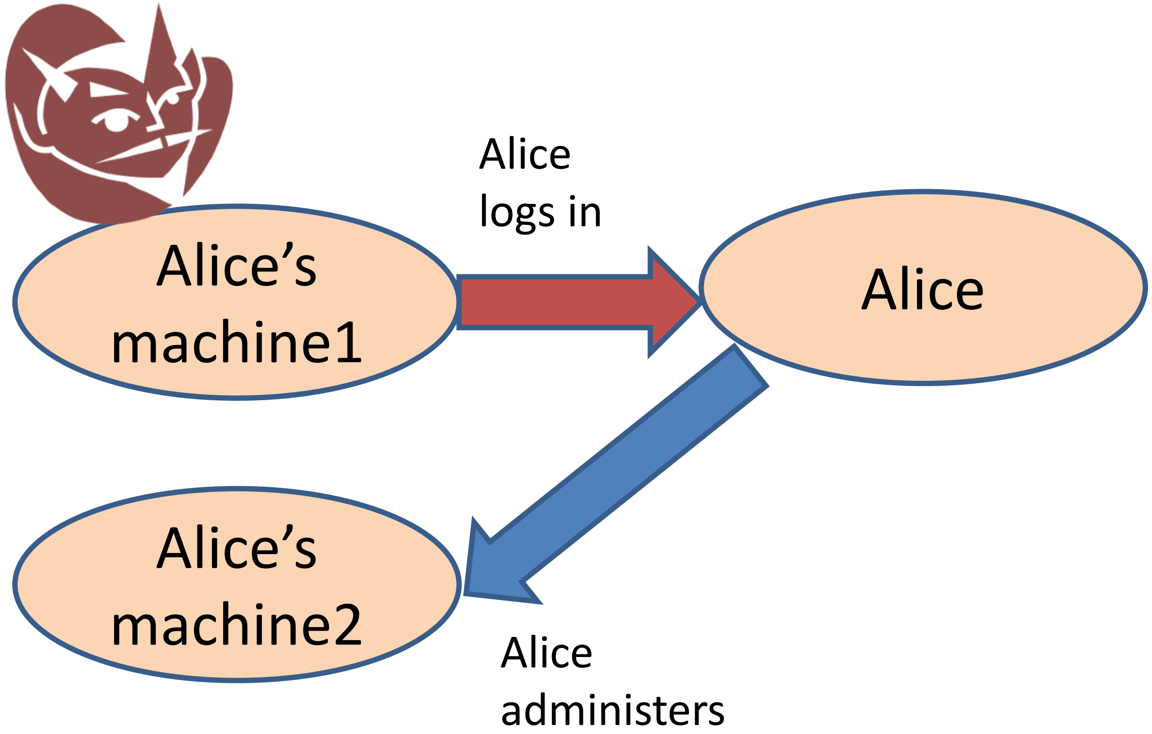
The snowball effect

An illustration of the consequences of bad policies (particularly in distributed systems):



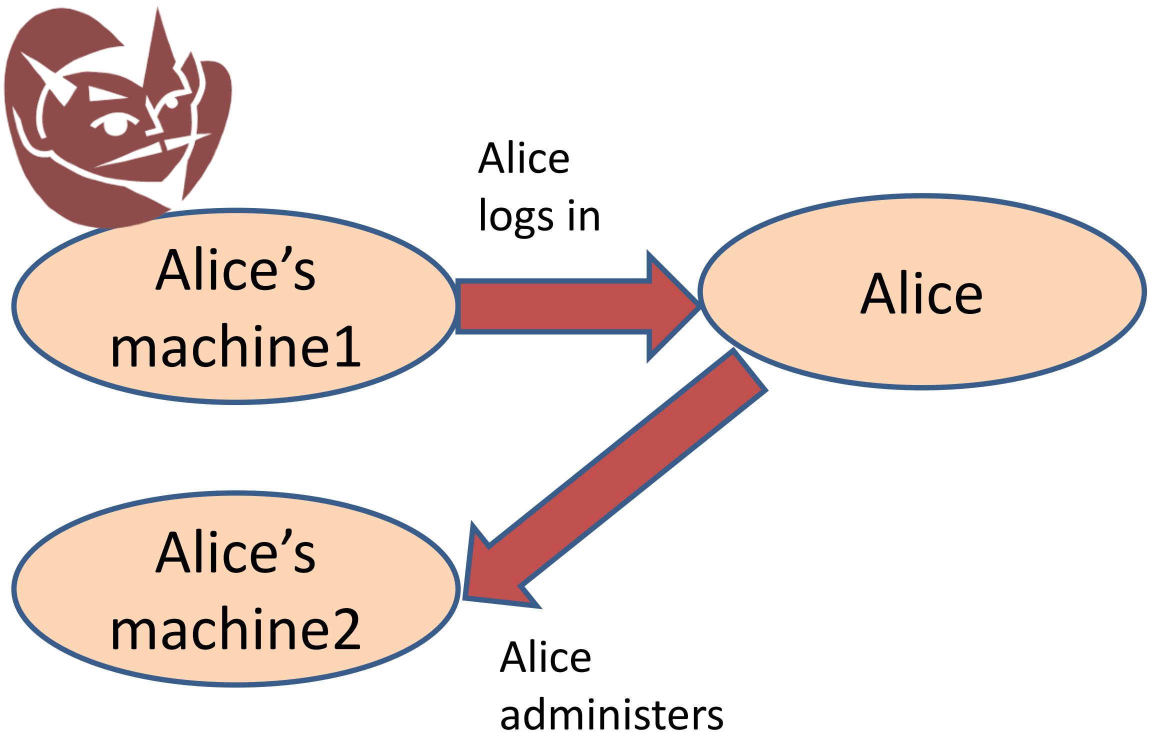
The snowball effect

An illustration of the consequences of bad policies (particularly in distributed systems):



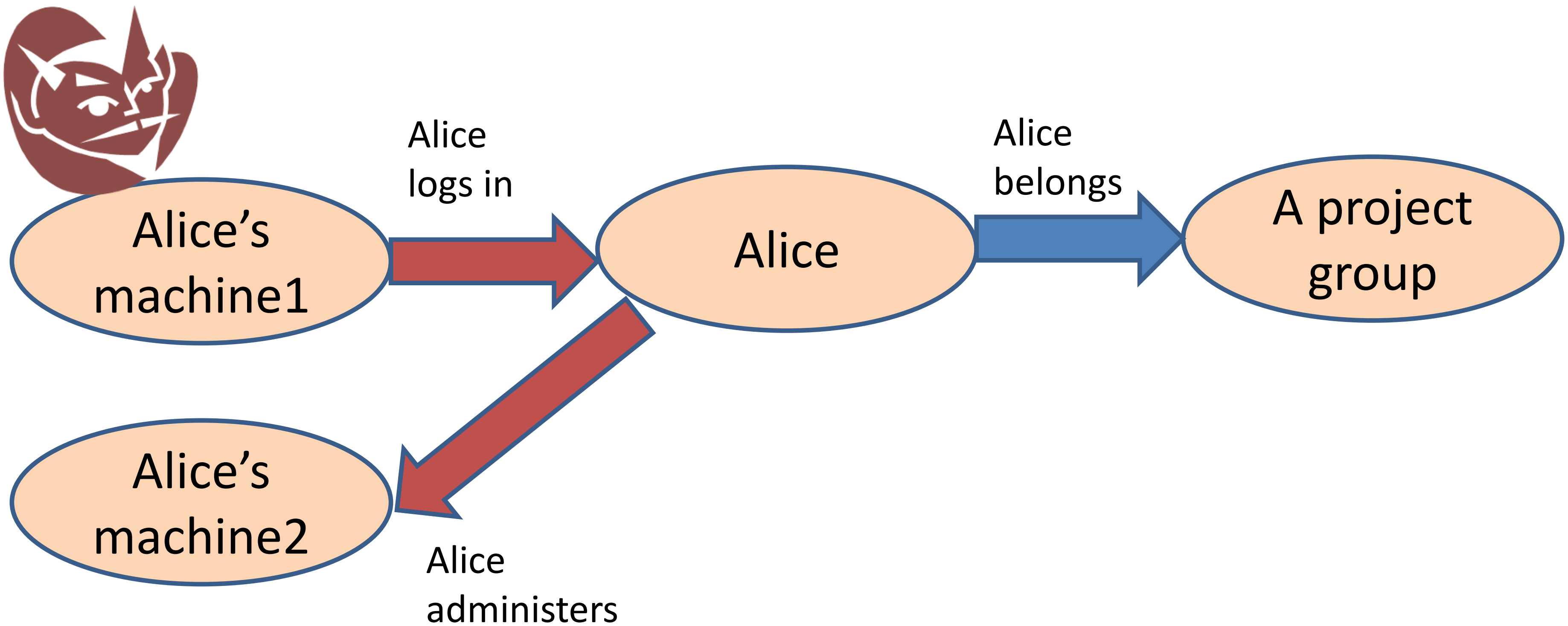
The snowball effect

An illustration of the consequences of bad policies (particularly in distributed systems):



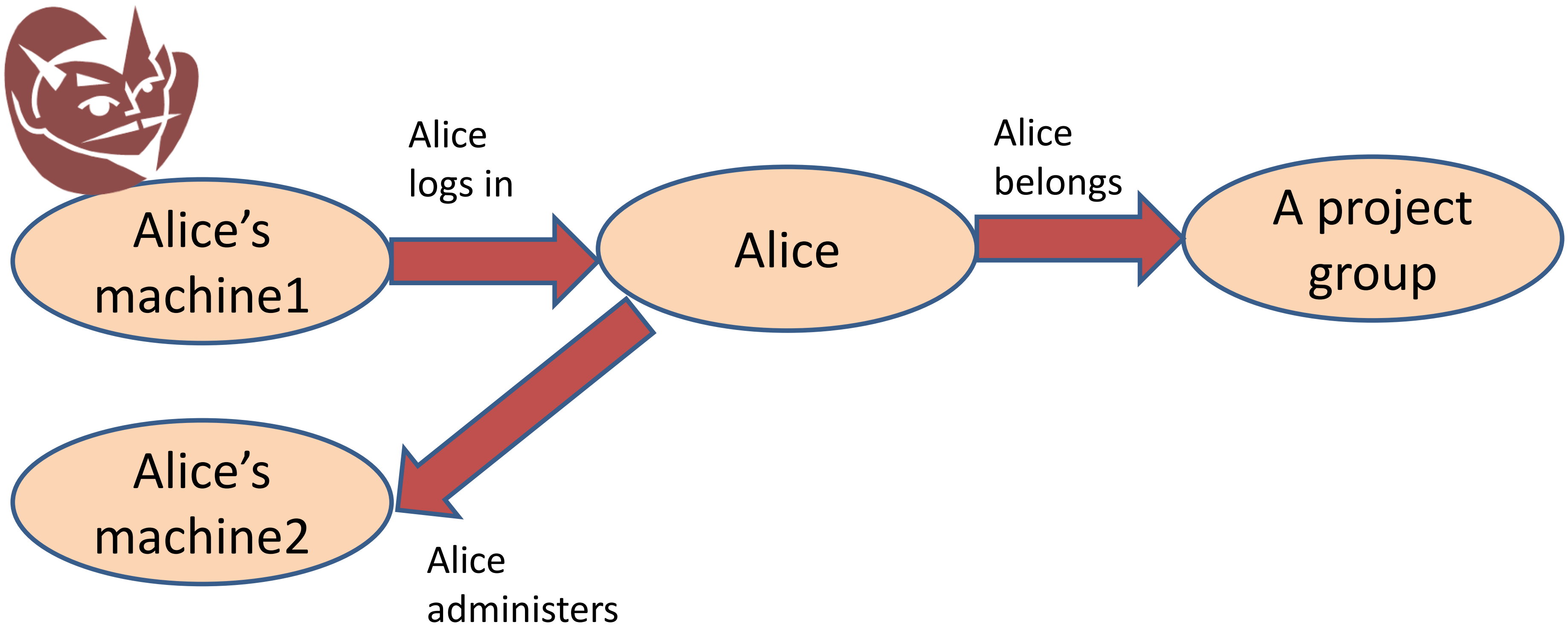
The snowball effect

An illustration of the consequences of bad policies (particularly in distributed systems):



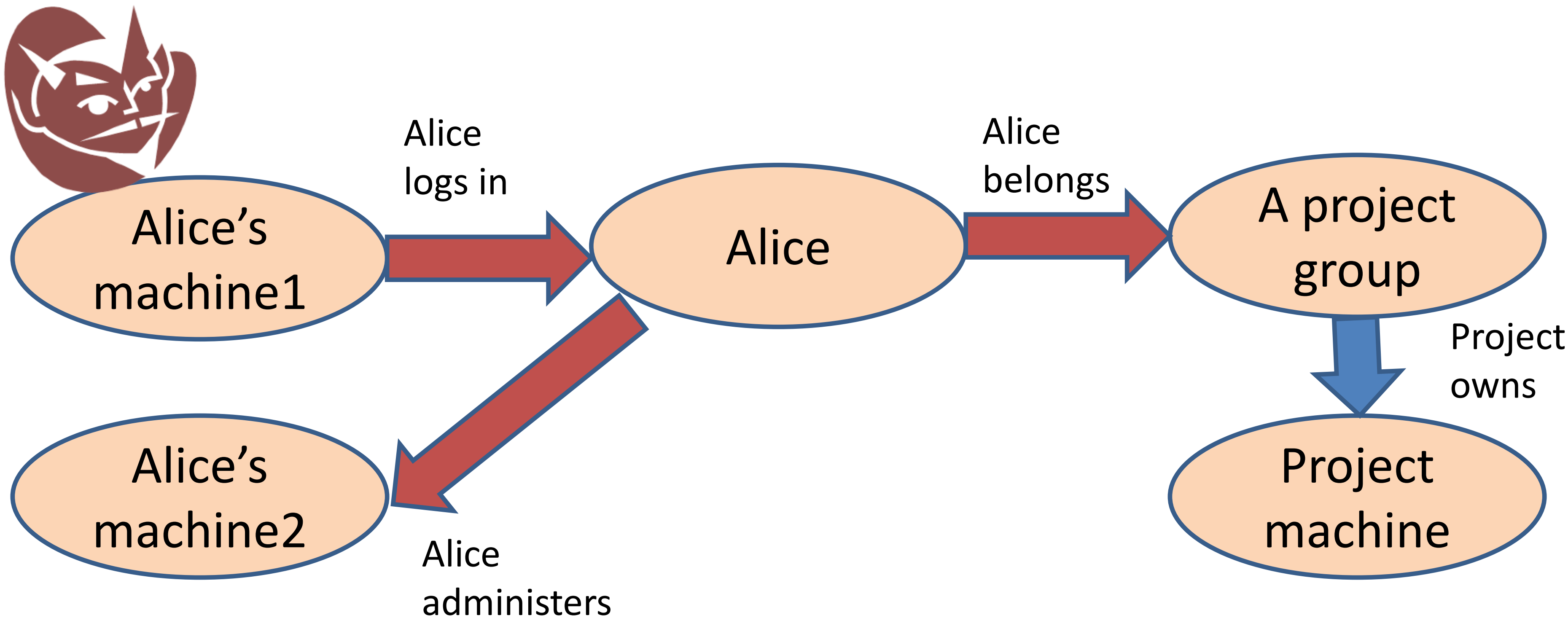
The snowball effect

An illustration of the consequences of bad policies (particularly in distributed systems):



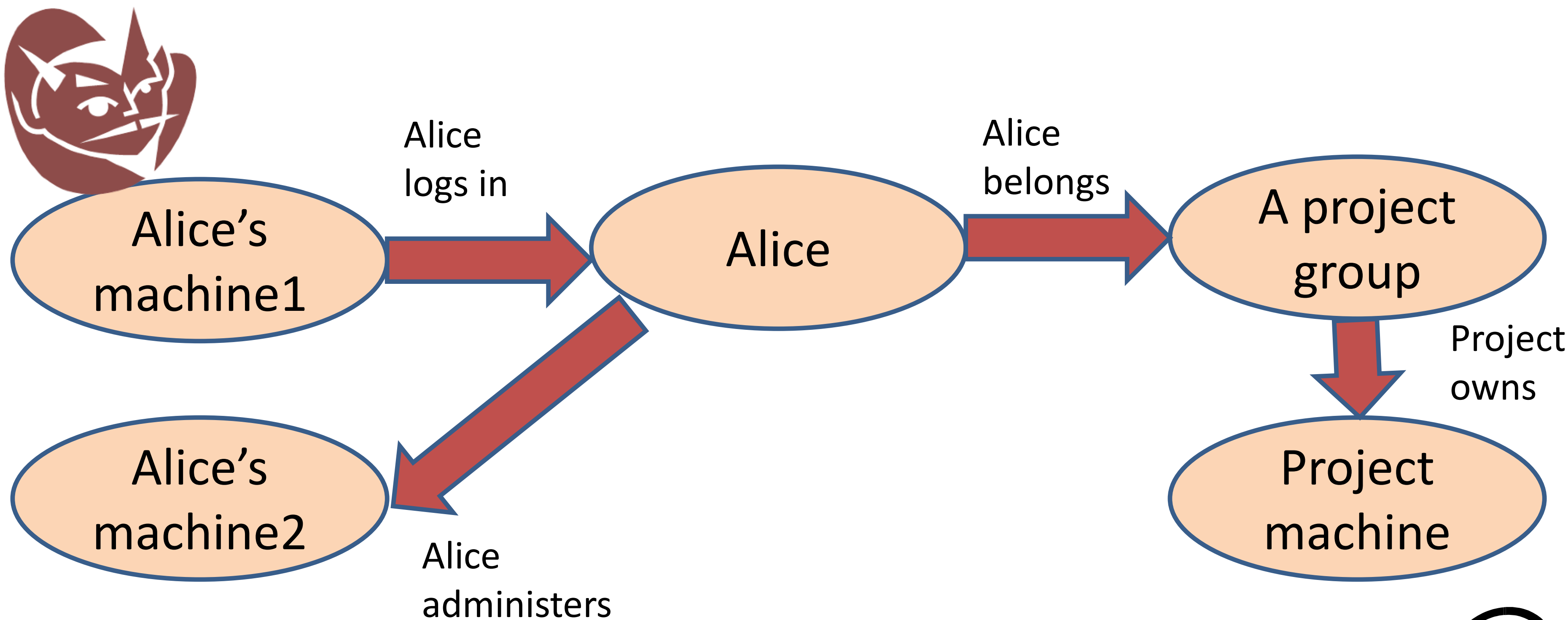
The snowball effect

An illustration of the consequences of bad policies (particularly in distributed systems):



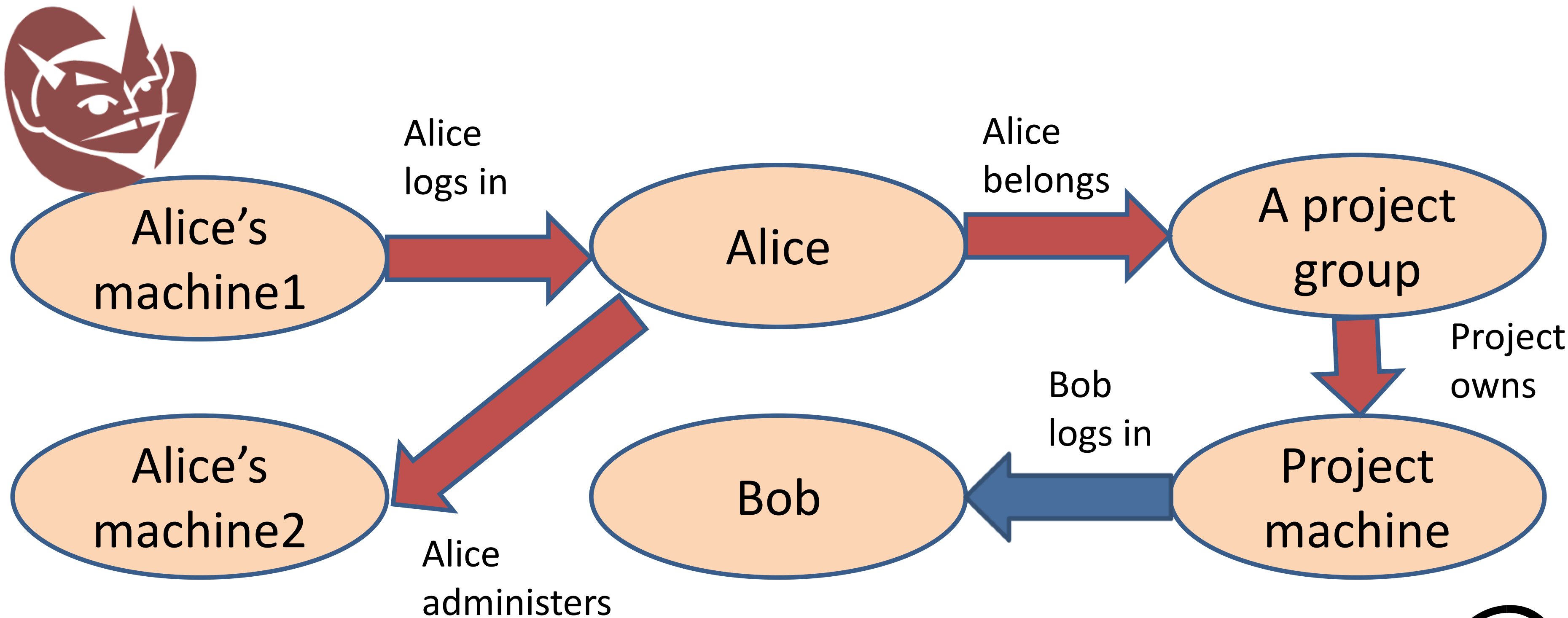
The snowball effect

An illustration of the consequences of bad policies (particularly in distributed systems):



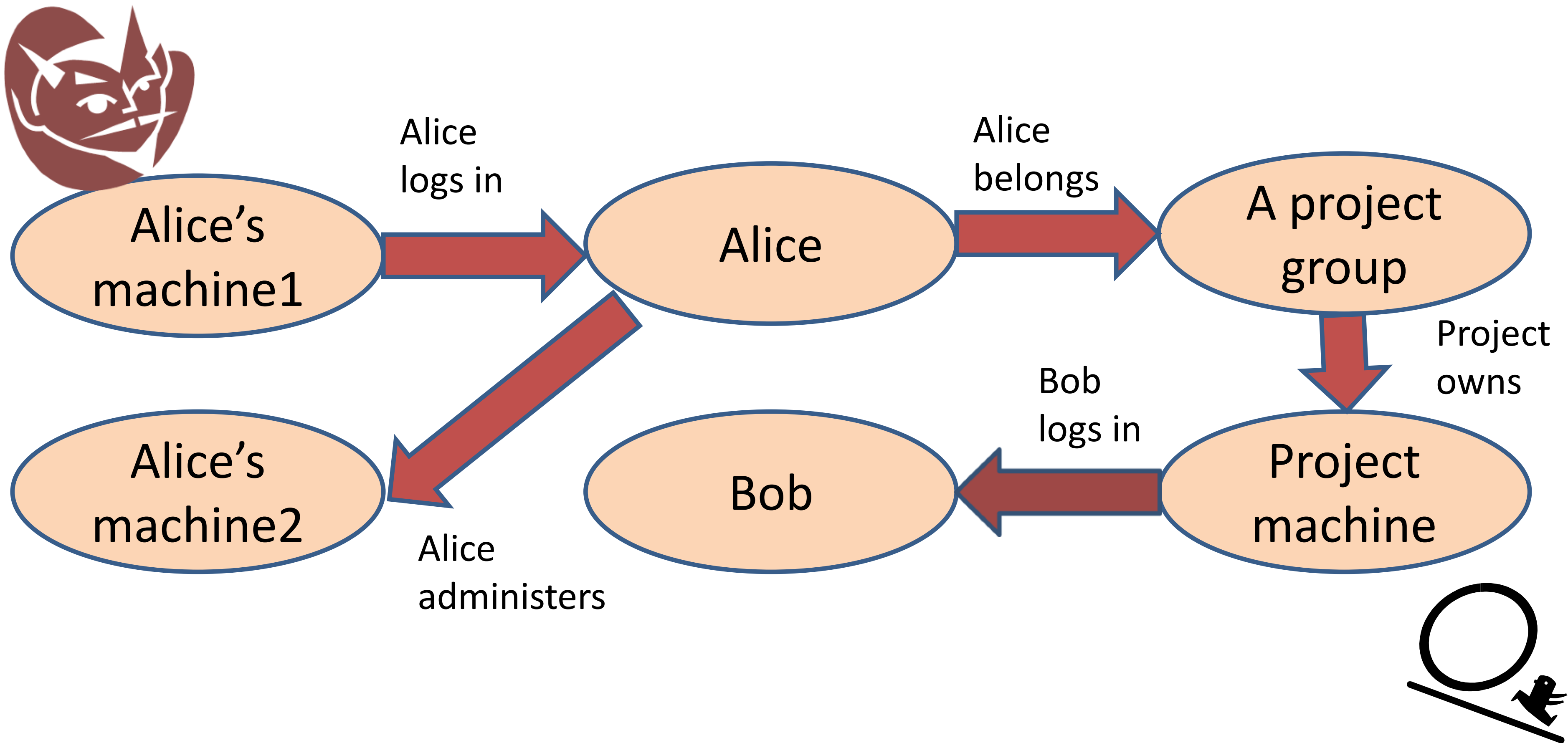
The snowball effect

An illustration of the consequences of bad policies (particularly in distributed systems):



The snowball effect

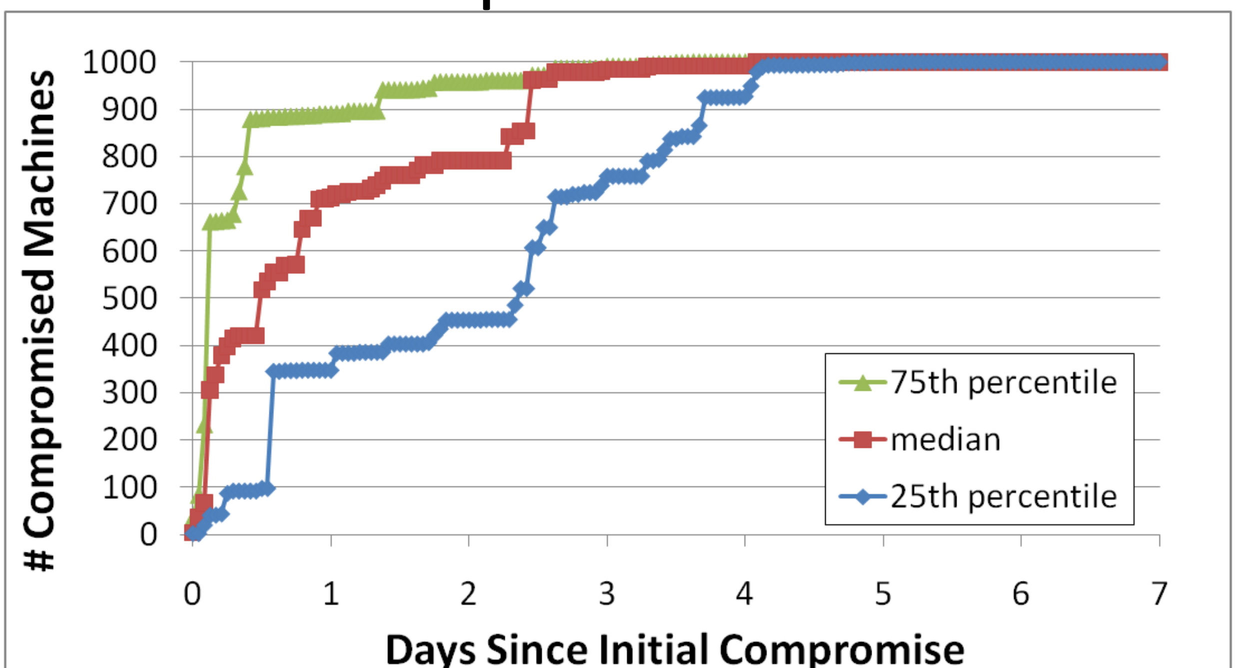
An illustration of the consequences of bad policies (particularly in distributed systems):



Snowball experiment

[Dunagan, Zheng, and Simon]

- Over 1 week, observe "log in", "administer", and "member" relations in a system.
- Then compute the effects of a single random initial compromise.

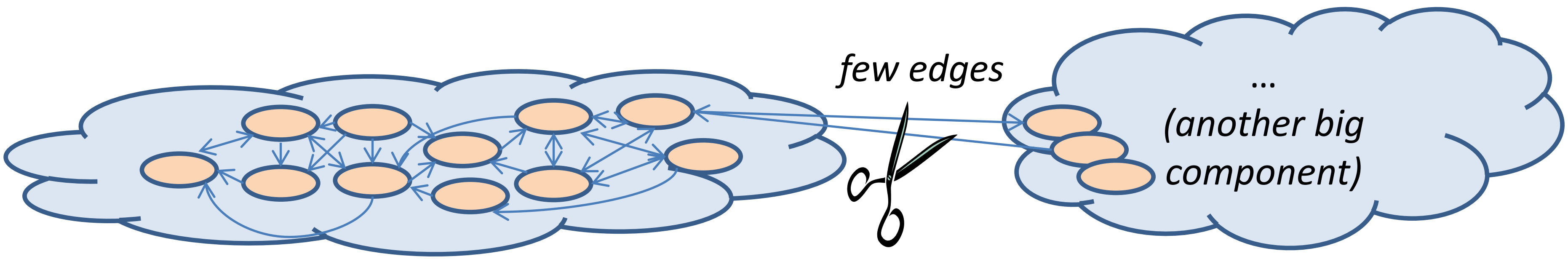


Cutoff at 1,000 for confidentiality reasons.

In an organization with ~100K accounts and ~200K machines.

Defenses

- Having analyzed the relations in a system, one may try to remove some of them.
  – The functioning of the system requires many of these relations!
  – Dunagan et al. find good candidates in *sparse cuts*.

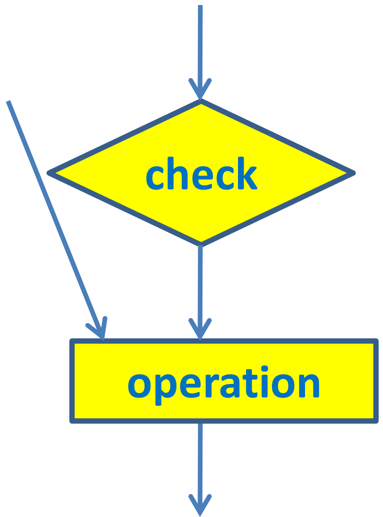


- We can also use stronger building blocks.
  – E.g., making it harder for a compromised machine to impersonate its users.

Circumventing access control

Sometimes the reference monitor does not protect all important objects and operations, for example because of

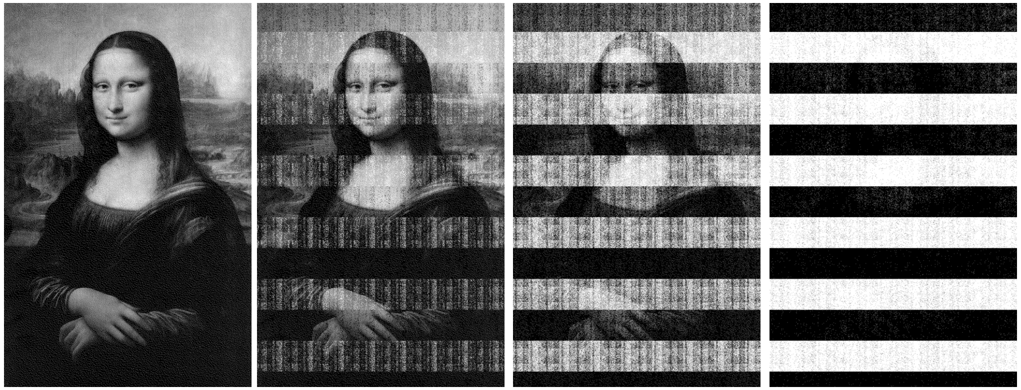
- hostile platforms (e.g., for DRM systems),
- control-flow subversions (as we will see),
- race conditions,
- data recovery from memory or disks,
- side channels.



Data recovery from memory

- Memory does not lose data as soon as it is disconnected!
- An attacker must be able to access the memory physically, find secrets in it, and do some error correction.

Cold RAM chips (-50°C).

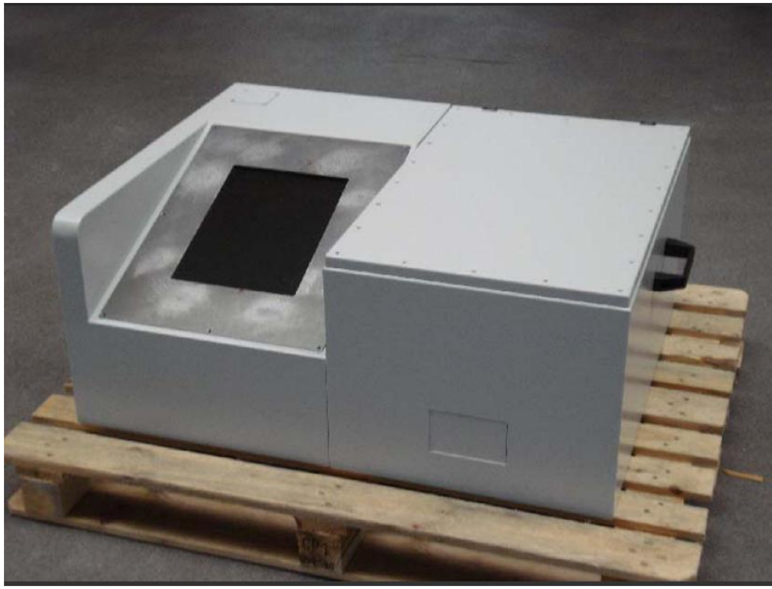


5 secs. *30 secs.* *60 secs.* *5 mins.*

Source: J. A. Halderman et al.
http://citp.princeton.edu/memory/media/

“Tempest” in Dutch voting (2006)

- A character in the name of a party caused some voting-machine displays to switch refresh frequencies.
- *The resulting radio emissions were different!*
- This could let someone outside a voting booth identify the party’s name.



Source: B. Jacobs and W. Pieters

A prototype tempest-shielded vote-printer, with touch screen and protected tray for the printed vote; almost 100kg.

Some reading

- Ross Anderson’s book *Security Engineering*.
- Butler Lampson’s paper “Computer Security in the Real World”.
- “Heat-ray: Combating Identity Snowball Attacks Using Machine Learning, Combinatorial Optimization and Attack Graphs”, by Dunagan, Zheng, and Simon.
- “Electronic Voting in the Netherlands: from early Adoption to early Abolishment” by Jacobs and Pieters.
- “Lest We Remember: Cold Boot Attacks on Encryption Keys”, by Halderman et al.

(See also the seminar.)

Séminaire

	**John Mitchell** (Stanford)	*16 mars*		**David Pointcheval** (CNRS)	*27 avril*
	**Ron Rivest** (MIT)	*23 mars*		**Adi Shamir** (Institut Weizmann)	*4 mai*
	**Andrew Myers** (Cornell)	*30 mars*		**Leslie Lamport** (Microsoft)	*11 mai*
	**Butler Lampson** (Microsoft)	*6 avril*		**Véronique Cortier** (CNRS)	*18 mai*