Willkommen zur Vorlesung

Sicherheit:
Fragen und Lösungsansätze

im Wintersemester 2012 / 2013

Prof. Dr. Jan Jürjens

TU Dortmund, Fakultät Informatik, Lehrstuhl XIV

Vorlesungswebseite (bitte notieren):

http://www-jj.cs.tu-dortmund.de/secse/pages/teaching/ws12-13/sfl/index_de.shtml

Themen der Vorlesung

Part I: Challenges and Basic Approaches
1) Interests, Requirements, Challenges, and Vulnerabilities
2) **Key Ideas and Combined Techniques**
Part II: Control and Monitoring
3) Fundamentals of Control and Monitoring
4) Case Study: UNIX
Part III: Cryptography
5) Fundamentals of Cryptography
6) Case Studies: PGP and Kerberos
7) Symmetric Encryption
8) Asymmetric Encryption and Digital Signatures with RSA
9) Some Further Cryptographic Protocols
Part IV: Access Control
10) Discretionary Access Control and Privileges
11) Mandatory Access Control and Security Levels
Part V: Security Architecture
12) Layered Design Including Certificates and Credentials
13) Intrusion Detection and Reaction

Key ideas for technical security enforcement mechanisms

- ***redundancy***
  enables one
  to infer needed information,
  to detect failures and attacks and
  to recover from such unfortunate events

- ***isolation***
  prevents unwanted information flows or interference

- ***indistinguishability***
  makes maliciously planned observations
  appear random or uniformly expected
  and thus useless

Redundancy: important examples

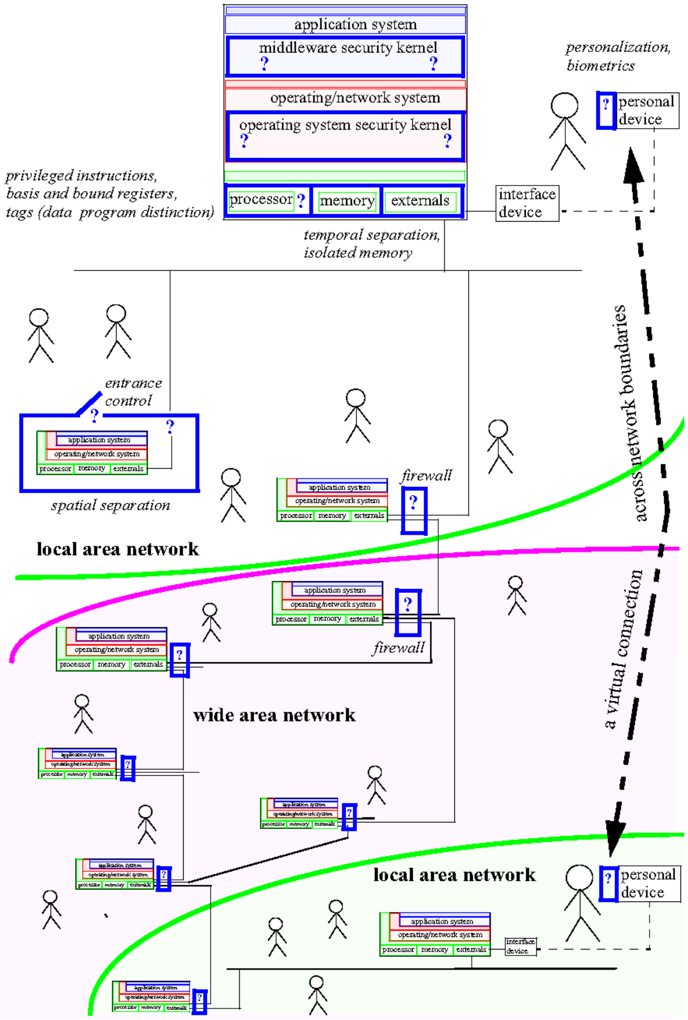
- spare equipment and emergency power
- recovery copies for data and programs
- deposit of secrets
- switching networks with multiple connections
- fault-tolerant protocols:
  - *infer* a hidden original state from observations and auxiliary redundancy and *reconstruct* it accordingly
  - *abort* a failing operation and *restart* it from a saved or reconstructed previous state, or even to *redo* a completed operation
  - take a *majority vote* regarding the actual outputs of computations performed independently and in parallel
- error-detecting and error-correcting codes
- cryptographic pieces of evidence

Isolation

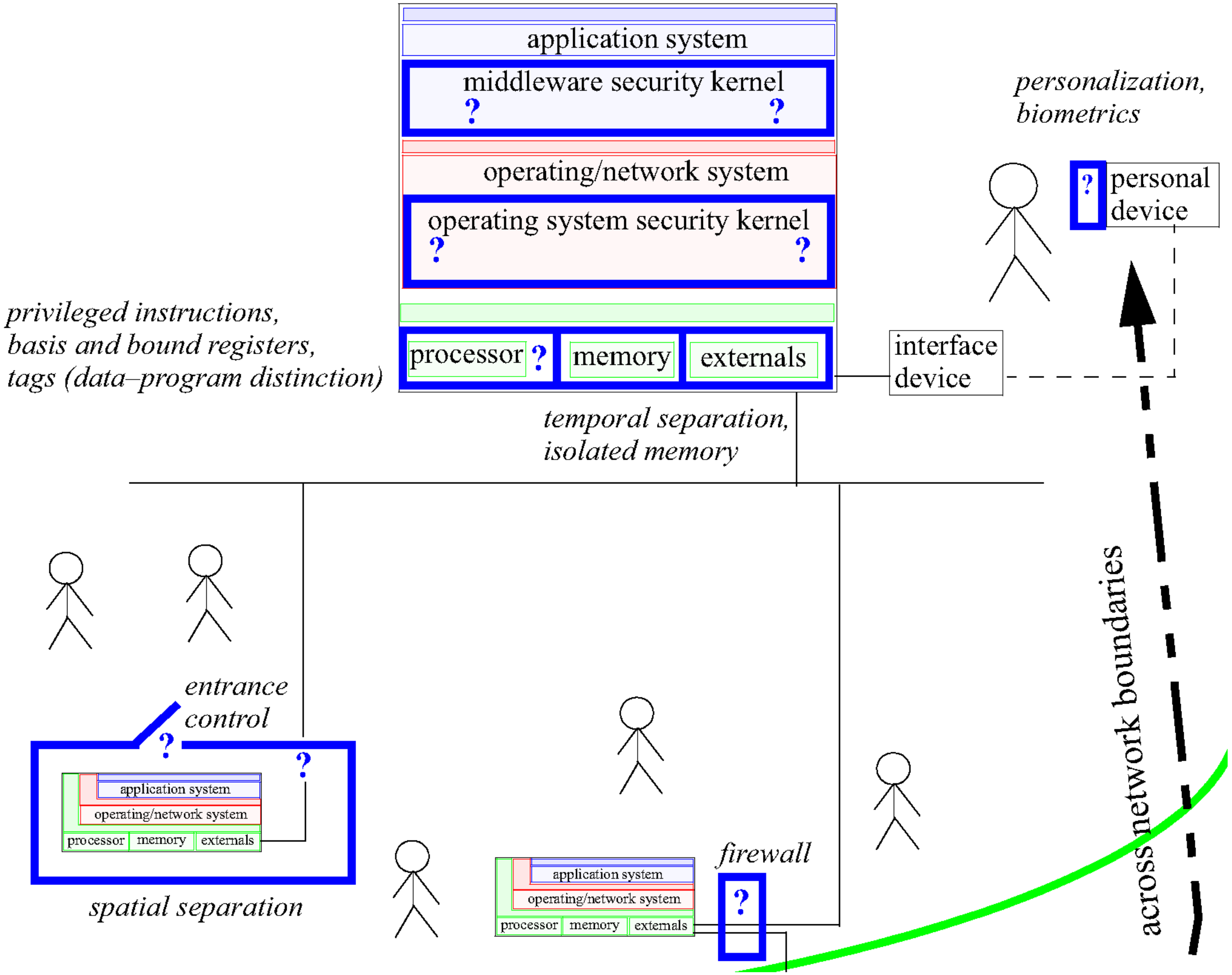
- **physical/programming-based** isolations
  requiring explicit *access decisions* at runtime,
  in order to enable the restricted usage of the isolated components
  according to declared *permissions*

- **virtual cryptographic** isolations
  employing more implicit access decisions
  based on the distribution of secret keys

Physical / programming-based isolations: a global view



Physical / programming-based isolations: a local view



Spatial separation and entrance control

- *spatially separate* an autonomously operated, *stand-alone* computing system in a dedicated closed room with locked doors (and windows)
- operate an effective *entrance control* enabling only *authorized individuals* to enter and then to (unrestrictedly) use the system
- may suffer from serious threats:
  - authorized individuals might not match the interests, owing to organizational weaknesses or unresolved conflicts
  - two or more authorized individuals might (unrestrictedly) interfere and collaborate
  - an (unrestrictedly) authorized individual might misuse the trust for unexpected and unwanted goals
  - the entrance control might fail, and some unauthorized individual might then (unrestrictedly) exploit the system

Temporal separation and isolated memory

- several participants can share a computing system
  either *strictly in sequence* or *overlapping in time*
- the participants might then interfere,
  when the processes executed on behalf of them access common memory
- if sharing is done strictly in sequence, after finishing a job,
completely *erase* all memory contents,
  i.e., re-establish an agreed *normal state*,
  maintained as an *invariant* of any usage of the computing system
- if sharing is done so that there is overlapping in time,
  adapt the notion of a normal state and take additional measures:
  - ensure that the allocated *process spaces*
    (containing programs to be executed, runtime stacks, heaps, etc.)
    always remain strictly isolated:
    one process can never access memory locations
    currently reserved for a different process
  - ground these measures on *physical tamper-resistant* mechanisms

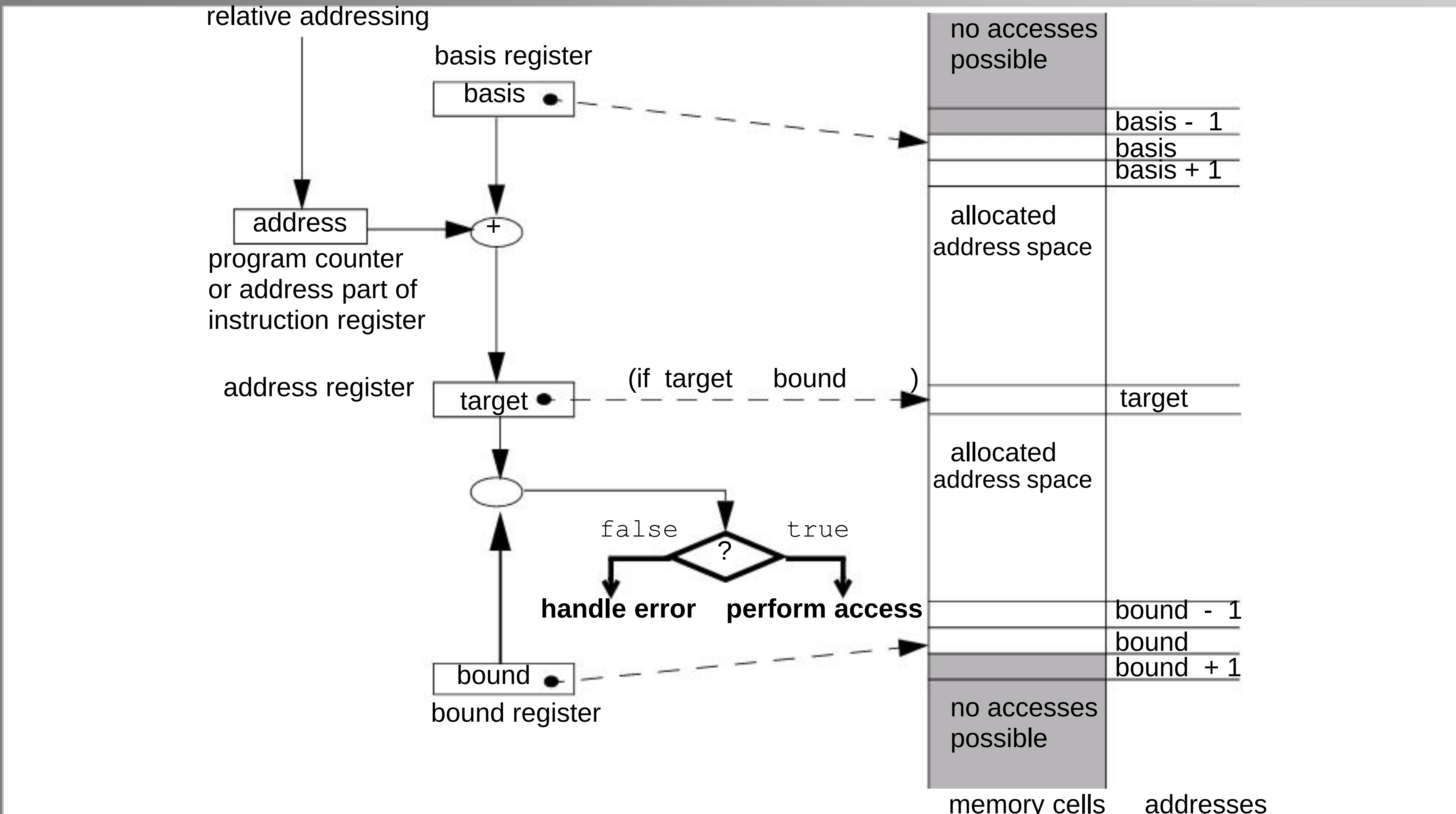
Memory protection and privileged instructions

- *memory protection* physically restrict memory accesses with respect to
  - addresses and
  - the mode of the operation requested

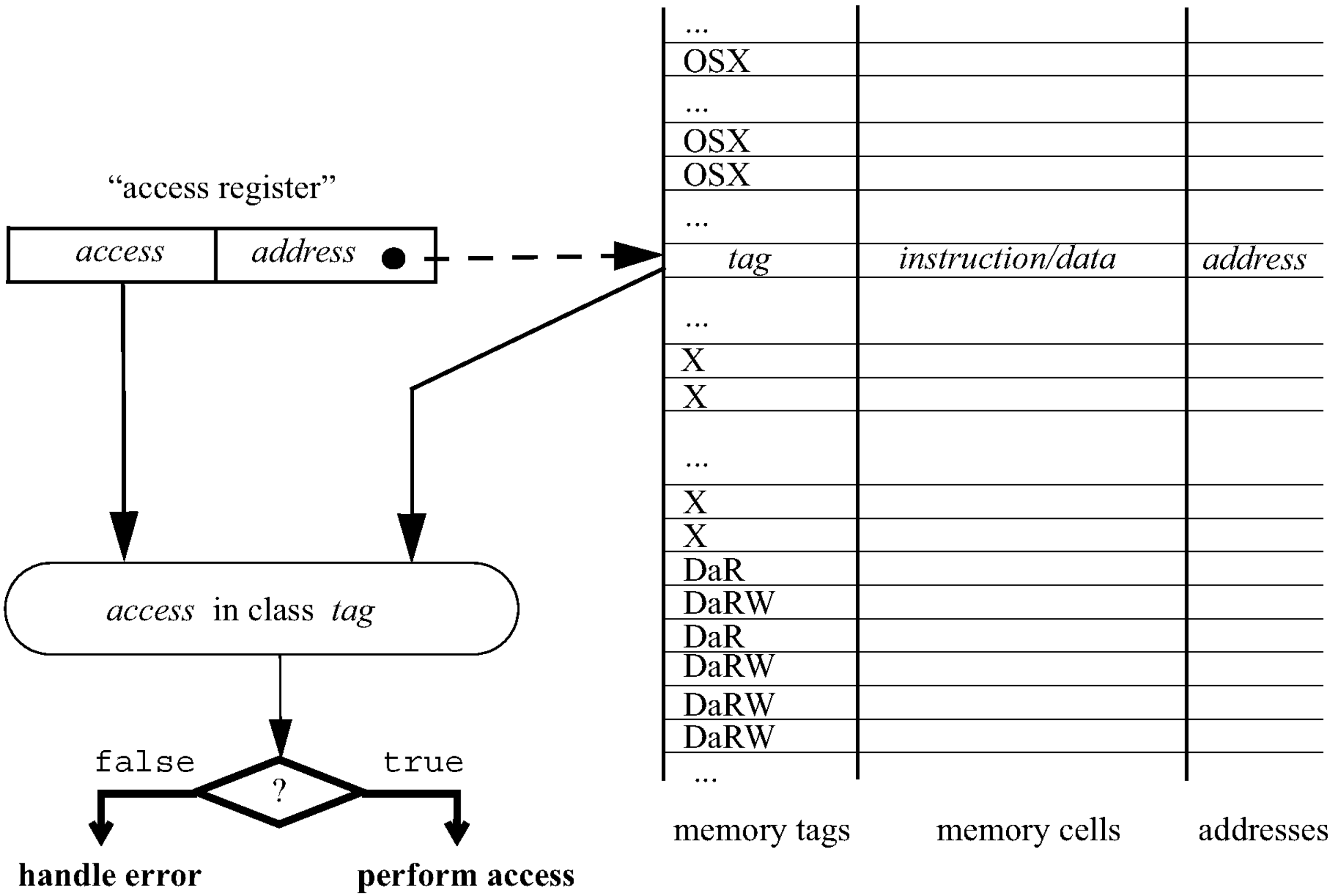
- ensured behavior of the processor's instruction interpreter:
  if the next instruction must be fetched from a memory location *address* or
  a machine instruction of the kind *instr* = [*operation*, *address*] is considered,
  then the request is actually executed
  iff
  a specific *protection condition* is satisfied

- a protection condition might depend on
  - the process,
  - the activity requested and
  - the address referred to

Basis register and bound register



Memory tags



"access register"
access
address
tag
instruction/data
address
...
OSX
...
OSX
OSX
...
...
X
X
...
X
X
DaR
DaRW
DaR
DaRW
DaRW
DaRW
...
access in class tag
false
?
true
handle error
perform access
memory tags
memory cells
addresses

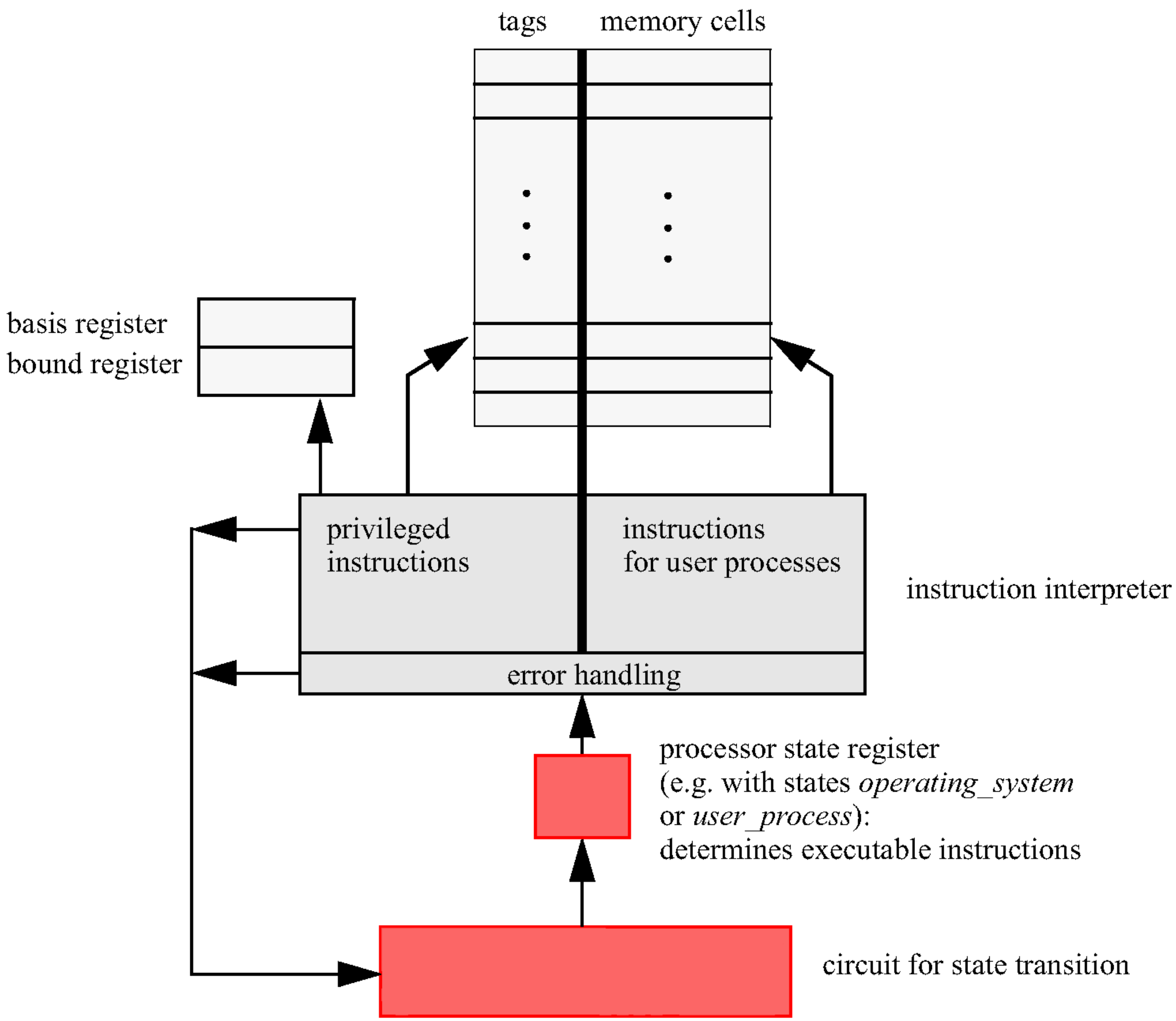
Tags as usage classes: examples

- *read* access to an executable *instruction*
  (fetching into the instruction register)
  by any *user process* or by special *operating system processes*
- *read* access to *arbitrary data*
  (loading into a data register)
  by any *user process* or by special *operating system processes*
- *write* access with *arbitrary data*
  (storing from a data register)
  by any *user process* or by *special operating system processes*
- *read* access to *data of a specific type*
  (e.g., integer, string, address or pointer),
  which has to be suitably recognized by the context or other means
- *write* access with *data of a specific type*
  (e.g., integer, string, address or pointer),
  which has to be suitably recognized by the context or other means.

Basis register and bound register versus memory tags

	Basis register and bound register	Memory tags
Extra memory	2 registers	linear in the size of memory
Operational overhead	assigning the registers; calculating and comparing addresses during memory accesses	assigning the memory tags; checking conformance during memory accesses
Abstraction layer of separated items	dynamically allocated address spaces	instances of types known to the processor
Granularity	more coarse (according to the memory requirements of dynamically generated, active items)	more fine (according to the size of instances of static types)
Protection goal primarily achieved	isolation of active items for avoiding unintended sharing of memory	isolation of instances of types for avoiding unintended usage
Coordination with higher layers	relative addressing, as usually employed	mapping of more application-oriented types to usage classes denoted by tags
Deployment	widespread, mostly together with other mechanisms of indirect addressing	seldom, mostly only in a simple variation

Privileged instructions



Further isolation mechanisms

- separate process spaces
- object-oriented encapsulation
- security kernels
- stand-alone systems
- separate transmission lines
- security services in middleware
- firewalls
- cryptographic isolation

Indistinguishability

- blurs specific informational activities
  by making them *indistinguishable* from random or uniformly expected events

- thus prevents an unauthorized observer to infer
  the details or even the occurrence of a specific activity

- might be achieved by employing
  - *randomness* or
  - *standardized behavior*

Indistinguishability by randomness

some explicit *randomness* is generated,
and then the specific activity considered
has this randomness *superimposed* on it

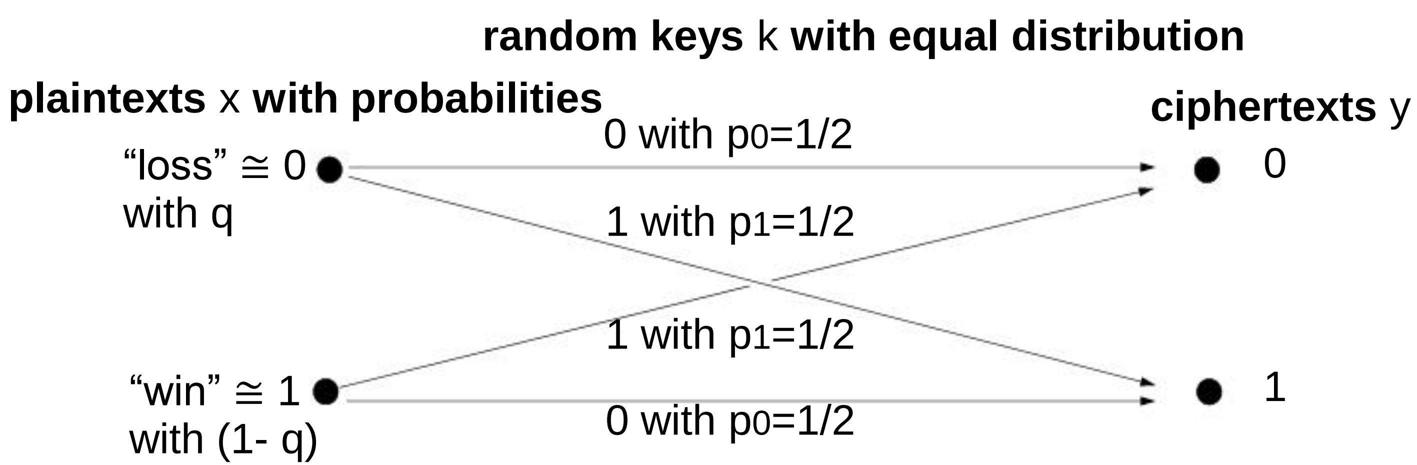
such that the activity appears (sufficiently) random itself

used in cryptography: the *secret key* is randomly selected
from a very large number of possibilities,

and the randomness of the secret key is transformed
into (some sufficient degree of) randomness
of the activity to be protected

Example for superimposing randomness: encryption

- two possible plaintexts: 0 with probability q
  1 with probability 1 - q
- source of randomness: two equally distributed keys, 0 and 1, with probability 1/2, independently of the plaintext
- the randomness of the keys is then superimposed on the plaintexts:



$$\text{Prob}[y=0] = \text{Prob}[x=0]\cdot\text{Prob}[k=0]+\text{Prob}[x=1]\cdot\text{Prob}[k=1]$$
$$= (q + (1 - q)) \cdot 1/2 = 1/2$$

Encryption: indistinguishability of plaintexts

described in terms of a mental experiment:

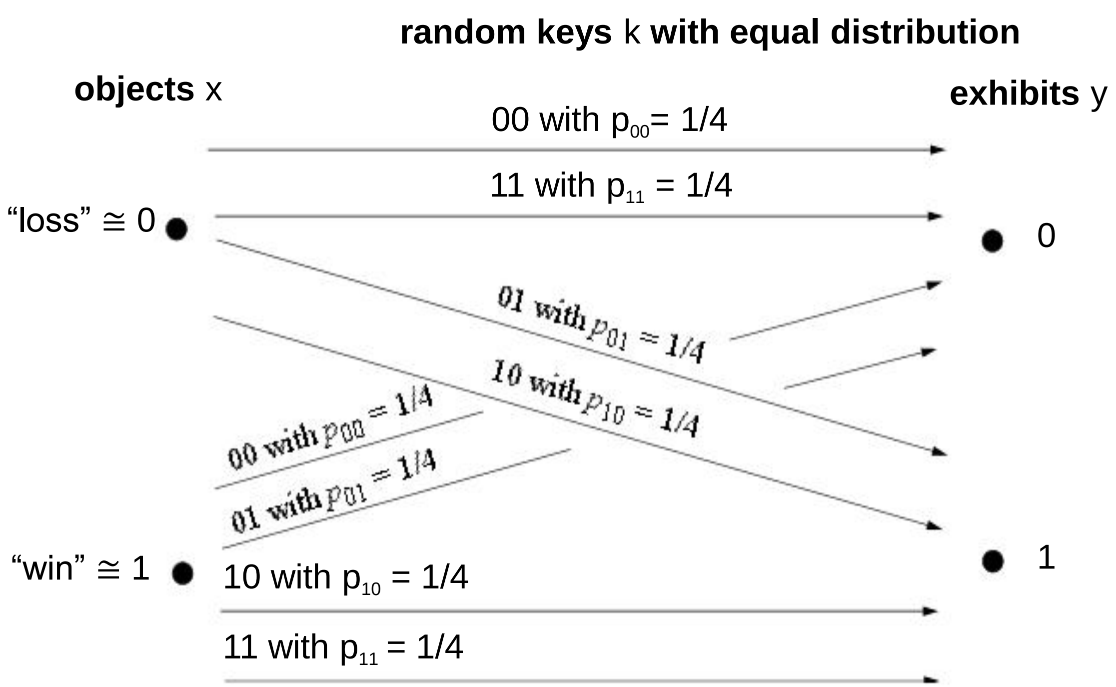
- attempt: construct an efficient *accepting device*
  that discriminates (hidden) plaintexts
  on the basis of observing (visible) ciphertexts

- insight: such a device *cannot* exist:
  an observed ciphertext does not contain
  any information about the underlying plaintext,

  thus this plaintext and the alternative one
  remain completely *indistinguishable*

Example for superimposing randomness: authentication

- two possible objects, 0 and 1
- source of randomness: four equally distributed keys, 00, 01, 10 and 11, each of which is used with probability 1/4, independently of the object
- the randomness of the keys is then superimposed on the objects:

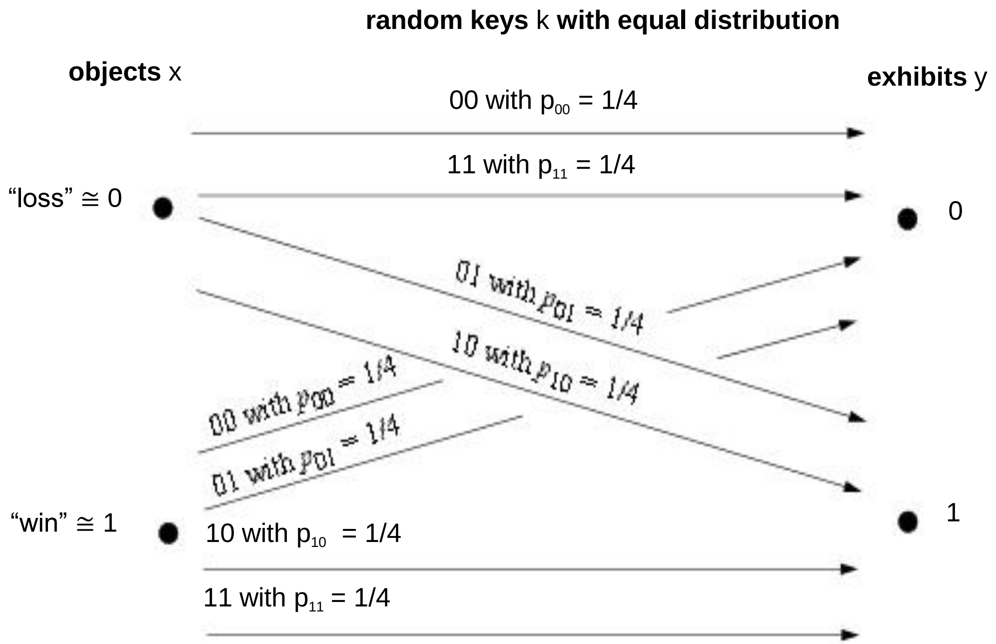


Authentication: indistinguishability of exhibits

- suppose the exhibit 0 for the event “loss” is known
- then, either key 00 or key 11 has been secretly used:

  these keys still map the event “win” onto either exhibit, 0 or 1,

  which are thus *indistinguishable*
  regarding their acceptance on the basis of the pertinent secret key



Indistinguishability by standardized behavior

a suitably designed *standardized behavior*,
possibly consisting just of dummy activities, is foreseeably produced,

and then the specific activity considered is hidden
among the foreseeable behavior,
for instance by replacing one of the dummy activities

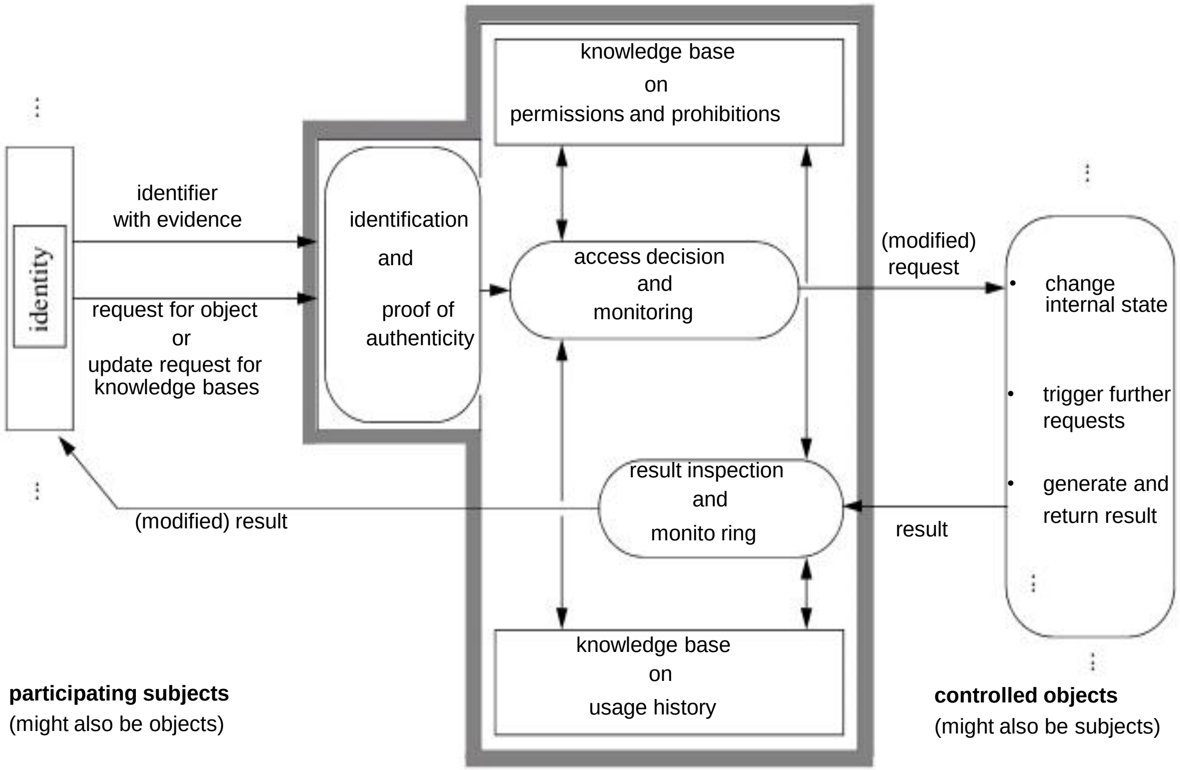
Hiding among standardized behavior: examples

- **non-observable activities**
  hiding the points in time of *sending* a message
  by pretending to be *uniformly active*:
  - participant actually communicate with some partner:
    prepares a corresponding document,
    appropriately adds the final destination of the communication,
    pads the document with some additional material until it has the expected length,
    envelops all data,
    waits for the next agreed point in time, and
    then sends the final message to the intermediate address used as a postbox
  - participant wants no "real activity":
    just sends a *dummy message* of the expected length
- **brokers and blackboards**
  employing a sort of fixed intermediate postbox to hide
  the sources and the final destinations of communications
- **group activities**
  authorizing group members to act on behalf of the community
  but without revealing the actor's *identity* to observers outside the group

Combined techniques: overview

- ***control*** **and** ***monitoring*:**
  identifiable agents can have *access rights* granted and revoked, and
  access requests of authenticated agents are intercepted by *control components*
  that decide on allowing or denying an actual access
- ***cryptography*:**
  secrets are generated and kept by agents:
  the secrets are exploited as cryptographic *keys*,
  distinguishing the key holder so that
  that agent is enabled to execute a specific operation in a meaningful way,
  in contrast to all other agents
- ***certificates*** **and** ***credentials*:**
  digitally signed digital documents (*digital legitimations*),
  conceptually bind *properties* that are relevant for access decisions
  to specific agents, which are denoted only by *public keys*
  (here, a public key is understood as a suitable reference to
  a *private* (*secret*) cryptographic *key* held by the agent considered)

Local control and monitoring



control and monitoring component

- cannot be bypassed
- (virtually) isolates participating subjects from controlled objects
- is based on physical isolation (indicated by the gray frame)
- decides on requests and results and possibly modifies them

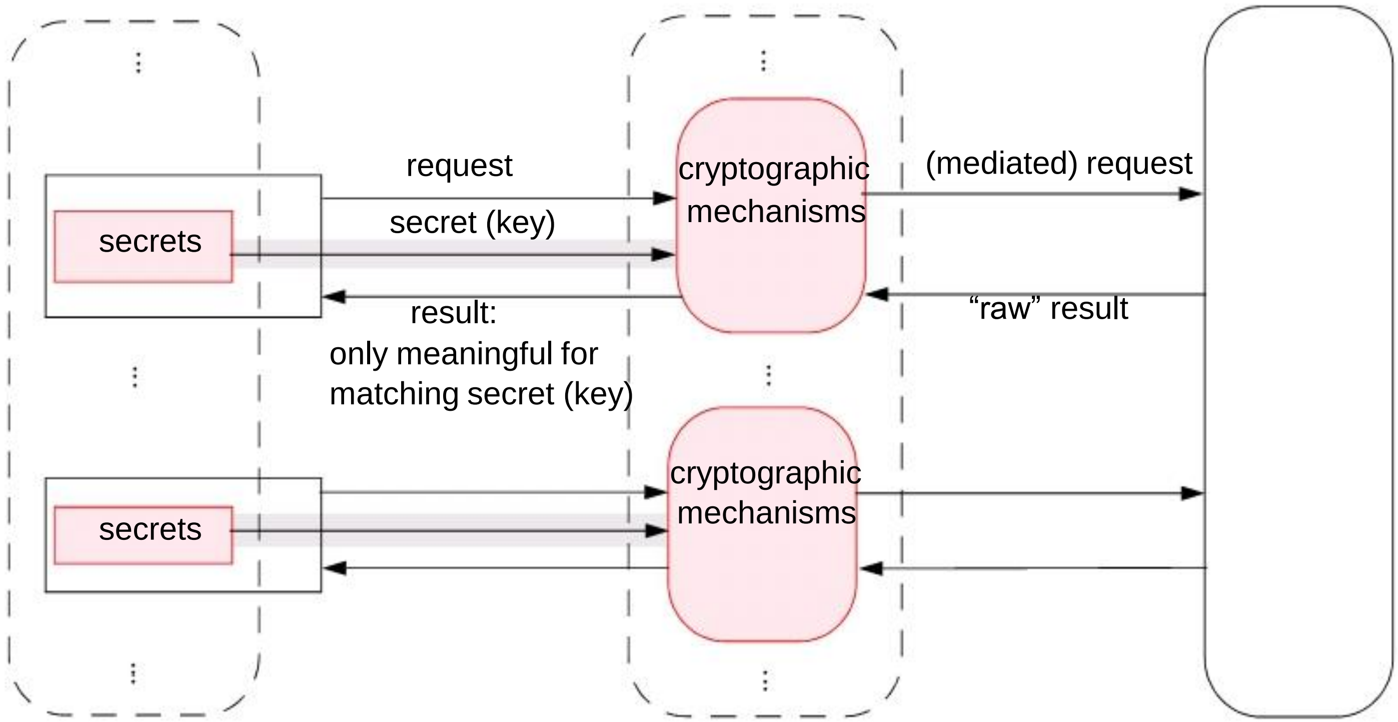
Access Control

- One of the main security mechanisms is **access control**, which ensures that only legitimate parties have access to a security-relevant part of the system.
- An important mechanism for controlling access to protected resources is the concept of **role-based access control**. In order to keep permissions manageable, especially in systems with a large or frequently changing user-base, they are not directly assigned to users.
- Instead, users can have one or more **roles** often related to their function within an organisation, and then permissions are assigned to roles.
- Sometimes, access control is enforced by **guards**: in the case of the **Java Security Architecture**, **guard objects** control access to protected objects; similarly for the **access decision objects** in *CORBA.*

Cryptography

conceptually: (cryptographic)
knowledge base on permissions
(and prohibitions)

conceptually: (cryptographic)
control component

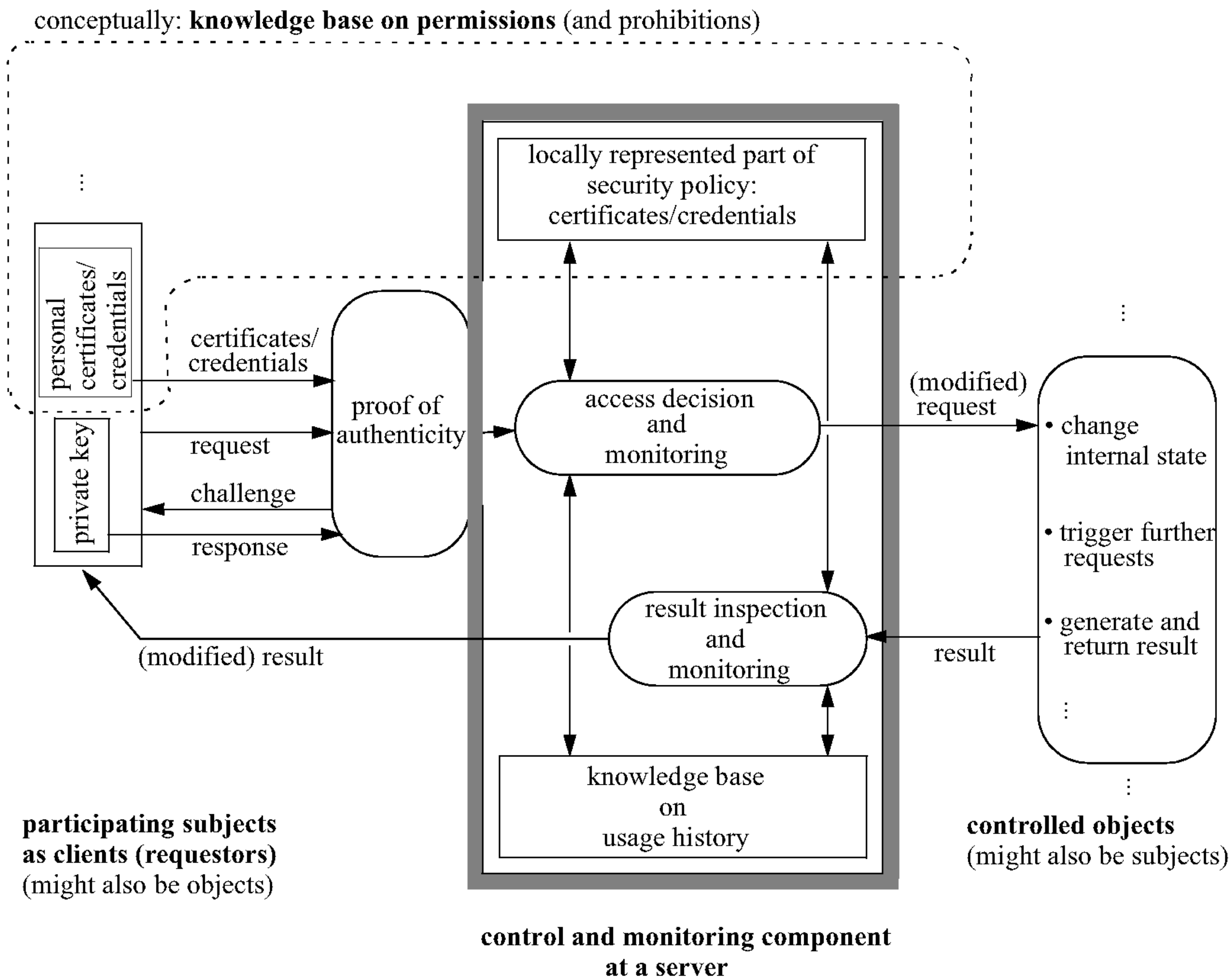


participating subjects

- generate, store and employ secrets
- exploit physical isolation
  (indicated by the gray areas)

controlled objects

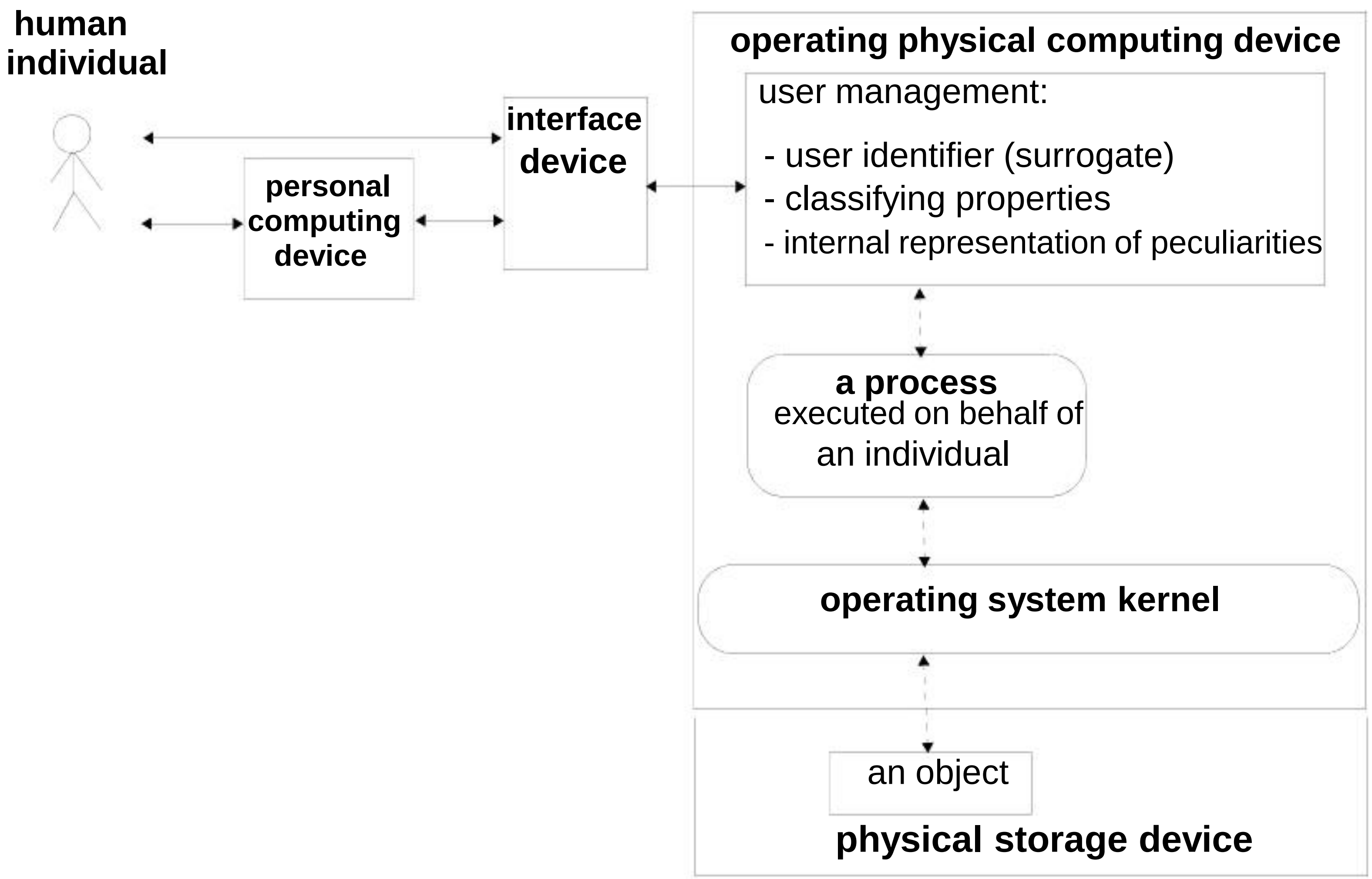
Certificates and credentials



Participants and objects involved

- a *human individual*
- a (physical) *personal computing device*
- a (physical) *interface device*
- a *physical computing device*
  (with a *processor* as its main component,
  and running an *operating system* and other *system software*)
- a *process*
- an *operating system kernel*
- a (physical) *storage device*
- a (virtual application) *object*

Local identifiers: participants and their local connections



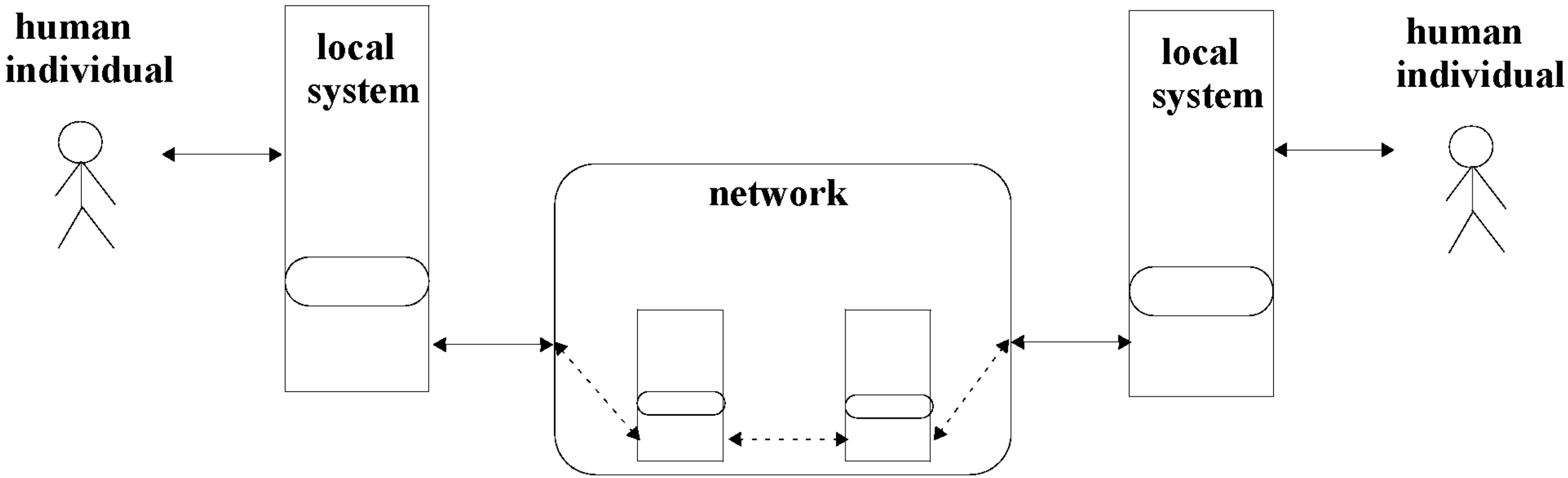
The fiction of an overall "connection"

conceptual perception:
an *individual* is permitted (or prohibited) to perform an *action* on an *object*

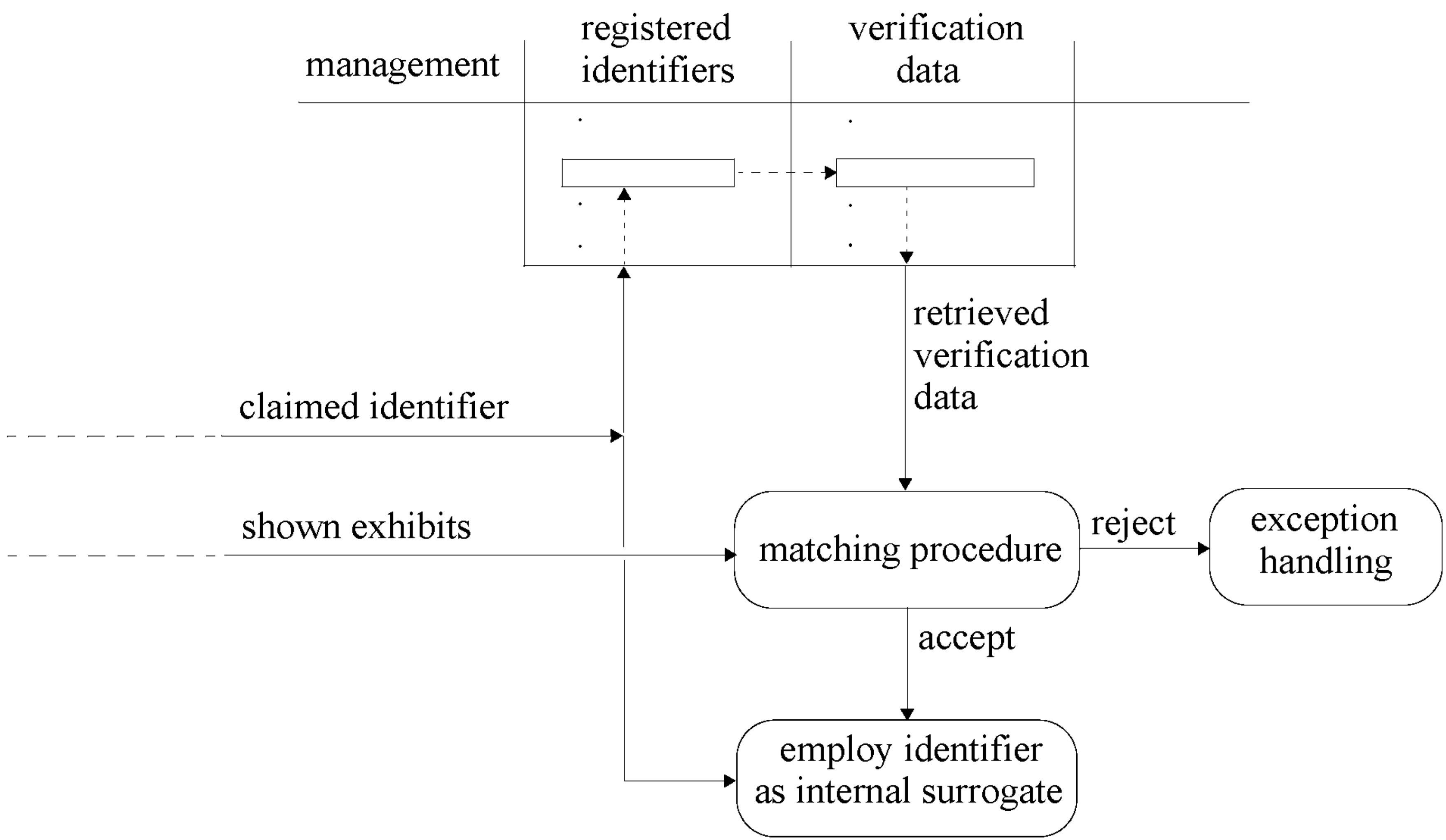
actual requirement:
the "natural identity" of a human individual
must be appropriately reflected along the chain of local connections,
ensuring that the *messages* involved are directed as expected, in particular:

- between the human individual and the interface device:
  either directly or with the help of a secure *personal computing device*
- between the interface device and the physical computing device:
  a secure *physical access path*
- between one process and another local process:
  secure *process communication*
- between a process and the local storage:
  a secure *operating system kernel*

Global identifiers: virtual end-to-end connections



Provisions for authentication and proof of authenticity



Peculiarities of human individuals: examples

- **individual knowledge:**
  - password, passphrase
  - PIN (personal identification number)
  - personal data
  - historic data
  - (discretionarily selected) cryptographic key
  - random number (nonce)
- **physical possession:**
  - smartcard
  - personal(ized) computing device
- **biological characteristics (biometrics):**
  - fingerprints
  - eye pattern,
  - genetic code
  - speech sound
- **individual (reproducible) behavior:**
  - pattern of keyboard striking

Peculiarities of physical devices: examples

- tamper-resistant, physically implanted serial number
- tamper-resistant, physically implanted cryptographic key
- discretionarily selected cryptographic key
- random number

Properties of verification data: informal version

- **(strong) correctness:**
  an exhibit presented is accepted
  iff
  it is authentic for the claimed identifier

- **(extended) unforgeability:**
  knowing the verification data alone
  should *not* enable one to produce any matching exhibits

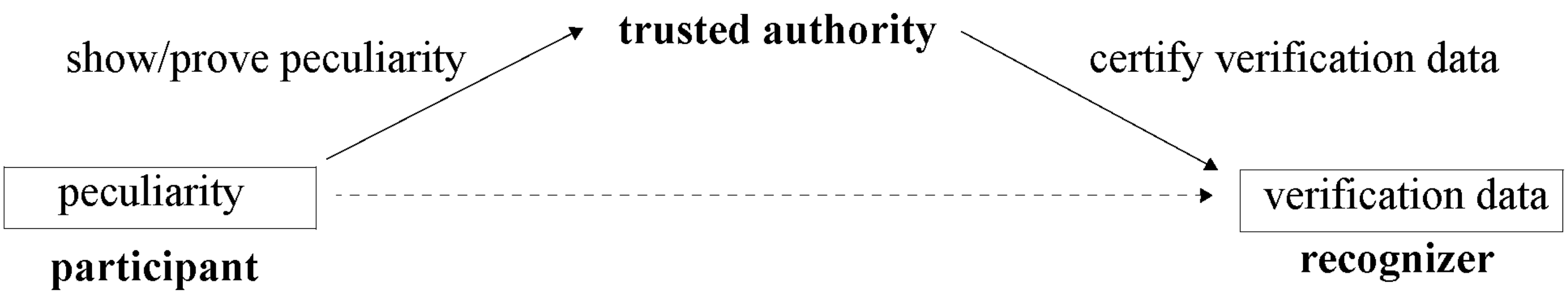
Some contributions of cryptography

- by applying ***encryption***,
  any verification data can be persistently stored in encrypted form,
  such that only the recognizing system can exploit the verification data

- by applying asymmetric ***cryptographic authentication***,
  a participant's given peculiarity can be made
  to consist of a private (secret) *authentication* or *signature key*,
  and the corresponding public *test key* serves as the verification data

- by applying a collision-resistant ***one-way hash function***,
  a (digital encoding of any) peculiarity is mapped to a *hash value*
  serving as stored verification data;
  later on, the peculiarity can be shown as an exhibit,
  whose hash value is recomputed and compared with the stored value

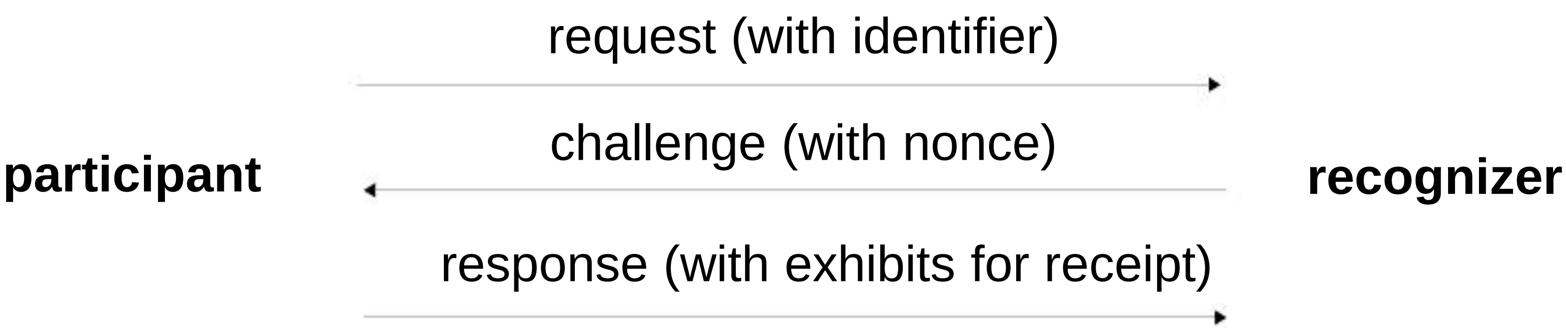
Issue of authentic verification data: trusted authorities



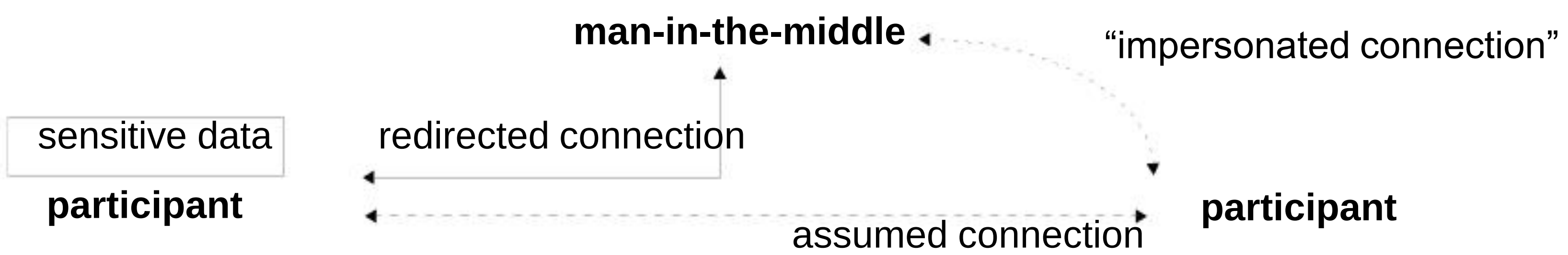
Security requirements: Freshness

- A message is **fresh** if it has been created during the current execution round of the system under consideration (for example, during the current protocol iteration) and therefore cannot be a replay of an older message by the adversary.
- A **nonce** is a random value that is supposed to be used only once (hence the name), for example to establish that a certain message containing a recently created nonce is itself freshly constructed.

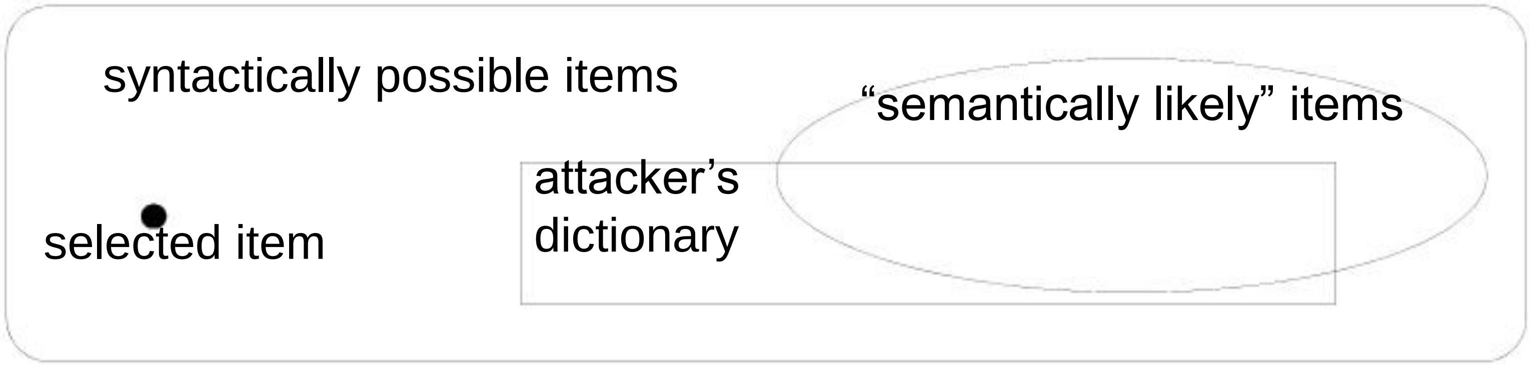
Issue of freshness: challenge-response procedures



Issue of malicious redirection by man-in-the-middle



Issue of malicious guessing or probing: carefully chosen exhibits



Permissions and prohibitions: the need for a layered approach

- participants by themselves, or
  some distinguished participants acting on behalf of the others,
  specify and *declare* the wanted permissions and prohibitions

- declarations are then (hopefully) appropriately *represented*
by the means of the computing system and inside it

- representations are (hopefully) efficiently *managed* there,
  both for *decisions* on actual requests for an operational option and for *updates*

- decisions are effectively *enforced*, i.e.,
  (hopefully) exactly those requests are successfully executed
  that have been declared permitted, and,
  accordingly, none of those that have been declared prohibited

Specification of permissions and prohibitions: some guidelines

- alignment with the environment
- least privileges according to need-to-know or need-to-act
- separation of roles
- purpose binding
- separation of privileges

Requirements and mechanisms reconsidered

security interests:

- *availability*: requested data/action returned/executed in a timely manner
- *integrity*: an item's state unmodified, or its modification detectable
- *authenticity*: claimed origin of data or action recognized as correct
- *non-repudiation*: correct origin of data or action provable to third parties
- *confidentiality*: information kept secret from unauthorized participants
- *non-observability and anonymity*: activities kept secret
- *accountability*: activities traceable to correct origin

key ideas for security mechanisms:

- *redundancy*: adding additional data or resources to enable needed inferences, detect failures and attacks, or recover from them
- *isolation*: separating items to disable information flows and interferences
- *indistinguishability*: hiding data or activities by letting them appear to be random samples of a large collection or uniformly expected

Combined techniques reconsidered

- **local control and monitoring:**
  - *identity*-based
  - identification and proof of authenticity
  - permissions as access rights
  - control of intercepted requests and results
  - monitoring of overall behavior
- **cryptography:**
  - *secret*-based
  - encryption, (cryptographic) authentication including digital signatures, anonymization, randomness, one-way hash functions, timestamps
  - more advanced protocols built from these blocks
- **certificates and credentials:**
  - *property*-based
  - features of local control and monitoring applied to requests that are accompanied by digitally signed assignments of security-relevant properties to public keys

Interests and enforcing mechanisms: summary (part 1)

Interest	Redundancy	Isolation	Indistinguishability	Control and monitoring	Cryptography	Certificates and credentials
Availability	provisionally multiplying (sub)objects or generating auxiliary objects to reconstruct lost or corrupted objects	attributing distinguishing identifiers or characterizing properties confining threatening operations in the context of integrity		granting access rights for enabling permitted operations (and confining them as far as they are threatening) detecting and reconstructing losses and corruptions while intercepting requests and results	generating and distributing secrets (keys) for enabling permitted operations	issuing documents about properties for enabling permitted operations (and confining them as far as they are threatening)
Integrity	provisionally generating auxiliary objects to detect modifications	confining operations on objects to dedicated purposes generating distinguishing secrets	making exhibits appear randomly selected for preventing forgeries	specifying prohibitions for rejecting or confining threatening operations	detecting unwanted modifications of objects	specifying prohibitions for rejecting or confining threatening operations
Authenticity	adding exhibits derived from a distinguishing secret	attributing distinguishing identifiers generating distinguishing secrets	making exhibits appear randomly selected for preventing forgeries	recognizing a requestor by identification and proof of authenticity	recognizing a requestor or actor by verifying cryptographic exhibits	challenging a requestor and verifying cryptographic exhibits in responses

Interests and enforcing mechanisms: summary (part 2)

Interest	Redundancy	Isolation	Indistinguishability	Control and monitoring	Cryptography	Certificates and credentials
Non-repudiation	adding cryptographic exhibits in the form of digital signatures derived from a distinguishing secret	generating distinguishing secrets	making exhibits appear randomly selected for preventing forgeries		proving an actor responsible by verifying cryptographic exhibits in the form of digital signatures	assigning provable responsibility to issuers of documents by verifying cryptographic exhibits in the form of digital signatures
Confidentiality		confining operations on objects to dedicated purposes	making data appear randomly selected from a large collection of possibilities	specifying prohibitions for rejecting or confining threatening operations	prohibiting gain of information by encrypting data	specifying prohibitions for rejecting or confining threatening operations
Non-observability/ anonymity			hiding activities in a large collection of possibilities	untraceably mediating requests and results	superimposing randomness	issuing documents about properties referring to public keys (rather than identities)
Accountability	adding cryptographic exhibits in the form of digital signatures or similar means derived from a distinguishing secret	attributing distinguishing identities generating distinguishing secrets		logging and analyzing intercepted requests and results	proving an actor responsible by verifying cryptographic exhibits in the form of digital signatures or similar means	logging and analyzing intercepted requests and results