Concepts of Distributed Systems 2006/2007

Security

Johan Lukkien

System security

- Protection of electronic assets agains misuse

  or more precisely,

- Regulating access to electronic assets according to some policy
  - *policy*: set of allowed and disallowed actions
  - *security mechanisms*: can be regarded as enforcing the policy

Johan J. Lukkien, j.j.lukkien@tue.nl
TU/e Computer Science, System Architecture and Networking

nFAQ - 1

- Q: Why is security so much more important on the Internet than in 'life before the net' ?
- A: A criminal has become a first-class citizen
  - *ease*: sit back at home while braking in
  - *automation*: your computer does the work
  - *tooling & information*:
    - advanced tools are available as 'condensed knowledge' - no experts required
    - information about internal workings of systems can be easily retrieved

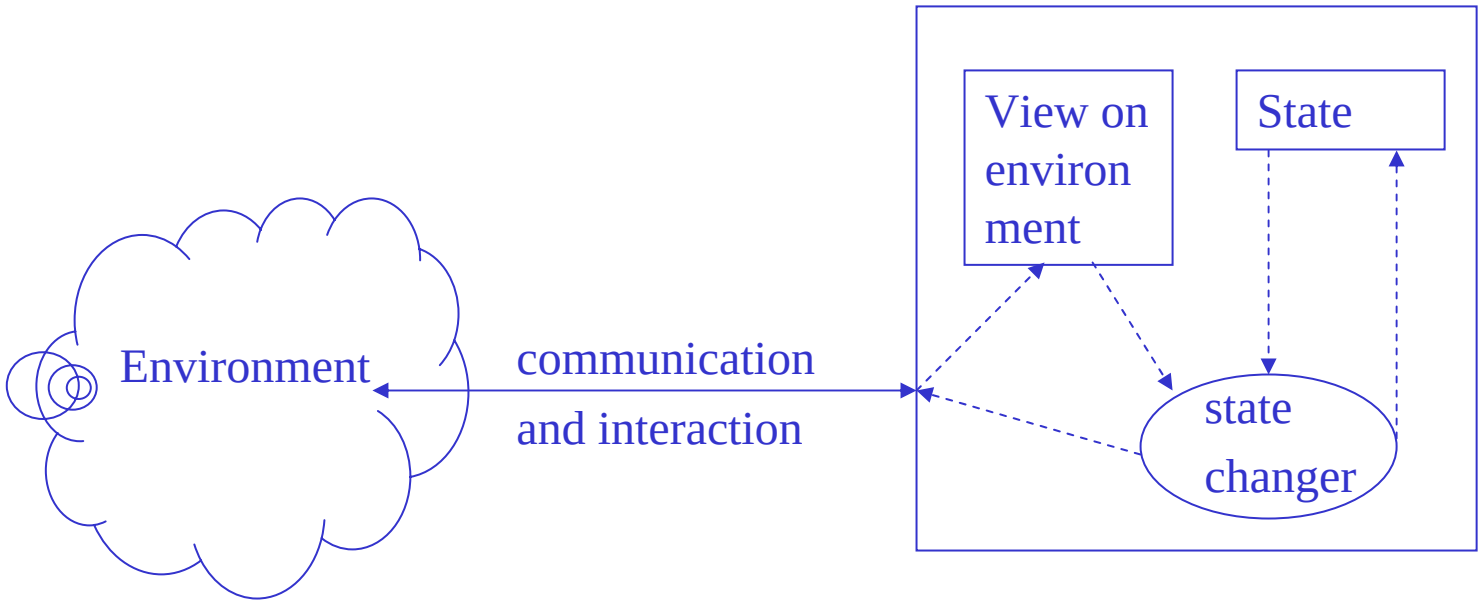
nFAQ - 1

- Q: Why is security so much more important on the Internet than in 'life before the net' ?
- A: Digital systems evolution
  - *openness*:
    - can reach –almost- any machine
    - information can be overheard upon passing
  - *predictability*: almost all machines on the net run Windows / Linux
  - *magnitude*: expected gain = probability of success * # trials
  - *storage capabilities & advanced mining*: large amounts information can be stored for interpretation

nFAQ - 2

- Q: What are essential causes of security dangers?
- A: The concepts of
  - *openness*: information and actors are visible and reachable
  - *black boxes*: internal state influenced by communication
  - *hidden assumptions*: assumed but not enforced behavior
  - *malicious exploitation*: trick a black box into doing something

Black box schematic



- The black box changes its state and interacts with the environment based on the old state and the view on the environment

Black box

- Abstraction of a functional block (a computer, a software component, a human, ...)
  - separation between *internal* and *external* (environment)

- Any useful 'black box' is characterized by:
  - **communication capabilities**
    - including observation of its environment
  - a **state** consisting of
    - an internal part and
    - a part representing its view on the environment
  - the **ability to change**
    - the state and
    - the environment (or just act upon it)
  - a **set of assumptions** and **dependencies**

Examples

- Computer
  - *view on environment*
    - represented inside programs through communication and input devices
  - *internal state*
    - consists of data in memory and on disks

- Human being
  - *view on environment*
    - through senses
  - *internal 'state'*
    - knowledge obtained over time

Hidden assumptions

- *Assumptions: the conditions under which a certain system configuration or configuration change is operating correctly*
  - usually obvious at the time
  - obsolete or dangerous as a result of system evolution

- Examples:
  - the lack of security in the original design of the IP protocol stack
  - maxima and minima
    - string size passed to an application
    - number of running processes
  - PC configuration for operation in a trusted domain
  - Examples from construction, industry .....

Hidden assumptions

- *Human*: I can rely upon the National Bank protecting my account with my own bank
  - however, only Eur 20.000 is covered
- *Computer*: I am operated behind a NAT/firewall in a home network, disk partitions can be open
  - however, then it is taken to a LAN party
- *Software component*: using the keyboard for obtaining users credentials and credit card info is safe
  - however, sypware stores each typed character to pass it to some machine on the Internet

Malicious exploitation

- Exploitation focusses on talking the black box into doing something
  - reveil state
  - act upon the environment
  - put resources on the line
    - work, storage, money

- This is done via the communication capabilities, manipulating the internal state *as well as the view on the environment*

Overview

- Common notions
- Architectural aspects
- Mechanisms for authentication and secrecy
  - encryption basics, public & private keys
  - protocols for secure channels
  - signatures
- Access control
- Security management
  - key distribution
  - delegation

Extra-functional properties

- Recall definition of service
  - the *quality attribute* describes extra-functional properties
    - e.g. availability, reliability, security, performance
  - *security* is a quality that relates to the protection of service providers and users against malicious or ignorant parties
    - this includes the protection of their (digital) property, assets

- Extra-functional properties are *emerging* properties
  - addressed largely in the system architecture ('a property of the system as a whole')
  - therefore, just security *mechanisms* don't lead to a secure system
  - and it is common to speak about the security *architecture*

Security properties

- *Confidentiality*: disclose information only to authorized parties. Decomposed into
  - *authentication* – determine identification
  - *authorization* – obtain access rights
  - *secrecy* – protect communication and system assets against eavesdropping

- *Rightfull access*: only authorized parties can access information and resources
  - just authentication and authorization

- *Privacy*: control of (personal) information
  - information cannot leave a system
  - aggregation and destruction of information
  - (explicitly) controlling information flows

Security properties

- *Integrity*: absence of unauthorized or malicious modification
  - protect communication and assets against tampering
- *Repudiation* issues
  - *non-repudiation*: party cannot deny having sent or received a message
  - *plausible deniability*: receiver of message knows sender sent it but cannot prove it
- *Anonimity*
- *Availability*: system is accessible and ready to use
  - protect against access blocking
  - protect against being forced into spending resources

Threats (to destroy security properties)

- Interception
  - e.g. listening in (compromise) and subsequently misuse
- Interruption
  - e.g. destroy information; block services
- Modification
  - injection of hazardous transactions
- Fabrication
  - make up new info
- Replacement
  - act as someone else
    - e.g. reflection - man in the middle

Attacks

- Attacks: mechanism that realizes a particular threat
  - a threat is decomposed into a list or tree of attacks
  - protection against the threat means ruling out the attacks
  - generally, not possible to fully protect against a threat, as there are many attacks

- A system can be made secure against an *attack model* (sometimes: threat model)
  - attack model: (ordered) collection of possible attacks

Attack examples

- Examples
  - eavesdrop communication line (interception)
    - decomposed into: getting physical access, line access, perhaps more
  - password gets stolen (induces most threats)
    - decomposed into e.g. getting access to keyboard, eavesdrop typing
  - replace message content (modification)
  - throw atomic bomb (interruption)
    - detailed into the activities of some benign governments
  - replay (replacement)
  - run active content (induces most threats)

Notions

- ***Trust***: the level as to which a client considers a system to live up to expectations
  - typically: to be secure, to perform operations, to manage data
- ***Trust anchor***: a starting point for trust
- ***Security Policy***: specification of allowed and prohibited actions
  - **whitelisting** specifies what is allowed
  - **blacklisting** specifies what is not allowed
  - Question:
    - which one is preferred? and why?
- ***Credential***: a verifiable object attesting some security statement
  - the statement usually includes references to users, access rights etc.

Mechanisms

- ***Mechanisms***: how to achieve policies
  - encryption: data integrity, protection, authentication
  - passwords, biometry: authentication
  - access level: authorization
  - auditing: check what clients do; monitor
  - watermarks and signatures: non-repudiation
  - 'cookies': avoid some DOS attack

- Question:
  - what about the above mechanisms with respect to white- or blacklisting?

- **Note***: Any introduced mechanism changes the nature of the attacks*

Additional requirements on the mechanisms

- Resource friendly
  - amount of memory, computation
  - #messages
- Perfect forward secrecy
  - cannot unveil information later, even if communicating partners have been compromised
    - note: information recorded today might be crackable by 2010
- Revocation
  - in case of lost password, digital key
- Transparent user control
- Attack traceability
  - observe, notify, record and track attacks

Example policy (Globus, GRID package)

- Several administrative domains
- Within domains: local security policy
- Global operation: known initiator in each affected domain
- Operations between entities in two domains: mutual authentication
- Global authentication takes over priority from local one
- User rights can be delegated
  - to a program, or system
- Credential sharing is possible within same domain

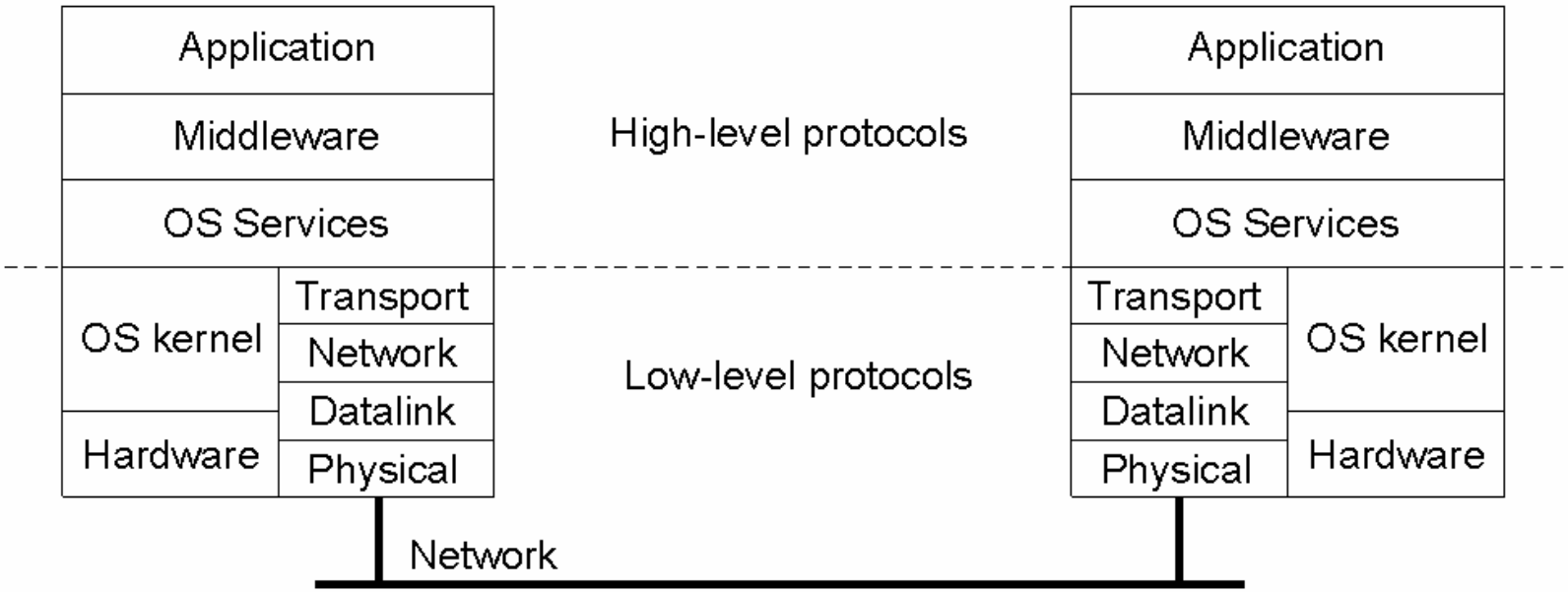
Overview

- Common notions
- Architectural aspects
- Mechanisms for authentication and secrecy
  - encryption basics, public & private keys
  - protocols for secure channels
  - signatures
- Access control
- Security management
  - key distribution
  - delegation

Security and architecture

- Security is a *design issue*
  - security pervasive in architecture
    - single failure destroys all ('emerging property')
    - not monotonic with respect to refinement
- Based on a model of the attacks
- Focus of control: determines mechanism
  - data and resource protection
    - e.g. maintain invariants to ensure integrity
  - protect against unauthorized or invalid operations
    - whitelisting, blacklisting
  - focus on role
    - system manager, user, programmer, ...

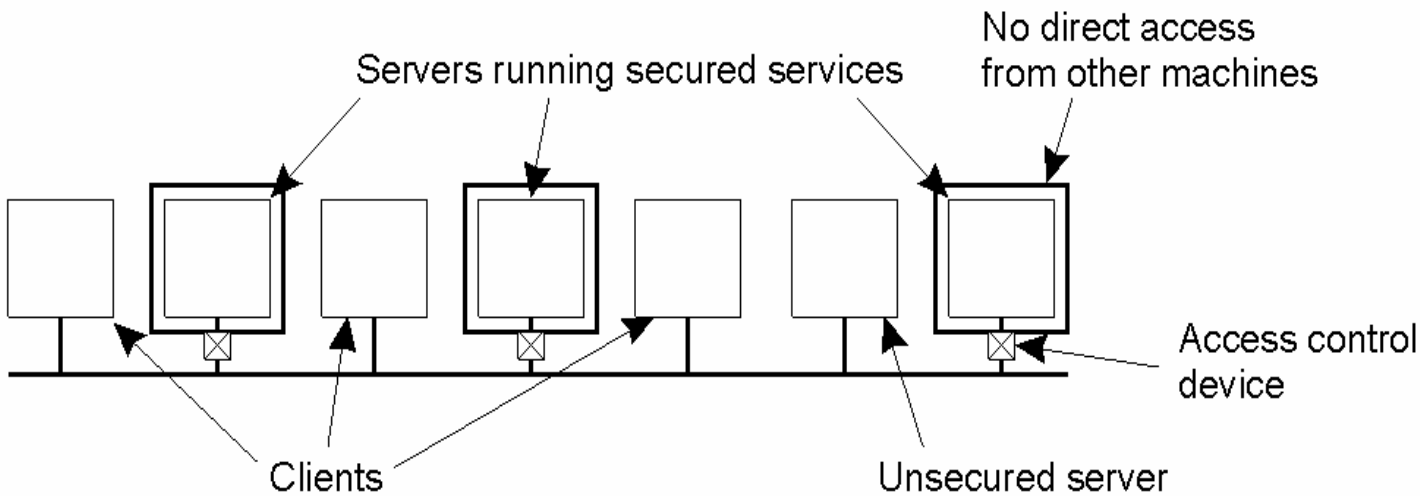
Layering


Application
Middleware
OS Services
OS kernel
Transport
Network
Datalink
Hardware
Physical
High-level protocols
Low-level protocols
Application
Middleware
OS Services
Transport
Network
Datalink
Physical
OS kernel
Hardware
Network

Layering

- When using a specific layer:
  - trust the lower ones to work properly [trust anchor]
  - or put in an encryption strategy yourself [and get your own trust anchors]
    - but: an attack at an unprotected layer 3 may cause interruption at layer 4
    - e.g. SSL/TLS vs. IPSEC
- Middleware
  - derives trust from trust assumptions and trust anchors
- Trusted computing base
  - set of mechanisms to enforce a policy
    - “the stuff that must work”
  - can include physical measures, e.g, separating a file server
    - forbid access to critical services

Reduced access...



Overview

- Common notions
- Architectural aspects
- Mechanisms for authentication & secrecy
  - encryption basics, public & private keys
  - protocols for secure channels
  - signatures
- Access control
- Miscellaneous
  - firewalls
  - security management
    - sandbox, delegation, ...

Applying cryptography: basic idea

- Protect information by applying a function to it
- Use *one-way* or *trap-door* cryptographic functions for which just *finding* the inverse is computationally infeasible
- Two applications
  - encryption: used for authentication and secrecy
    - encryption *method* public; parameterized by key
  - hashing: used for authentication

Keys

- Notation
  - $C = E_K(P)$, $P = D_{K'}(C)$
    - *ciphertext C* is the result of encrypting *plaintext P* using key *K*; *P* is obtained by decoding with key *K'*
    - *D* and *E* may be identical procedures but do not need to be

- Symmetric: share key
  - $P = D_K(E_K(P))$
  - e.g. DES

- Asymmetric: public/private
  - $P = D_{Kd}(E_{Ke}(P))$
  - example: RSA encryption

$K_{A, B}$	Secret key shared by A and B
K_A^+	Public key of A
K_A^-	Private key of A

RSA – finding the inverse w.r.t. exponentiation

- Setup:
  - $n = pq$, with p and q primes
  - e relatively prime to $\varphi(n) = (p - 1)(q - 1)$
  - d inverse of e in Z **mod** $\varphi(n)$
- Keys:
  - Public key: $KE = (n, e)$
  - Private key: $KD = d$
- Signature:
  - Message M in Z **mod** n
  - Signature $S = M^d \bmod n$
- Verification:
  - Check that $M = S^e \bmod n$

- Setup:
  - $p = 5$, $q = 11$, $n = 5 \cdot 11 = 55$
  - $\varphi(n) = 4 \cdot 10 = 40$, $e = 3$
  - $d = 27$ $(3 \cdot 27 = 81 = 2 \cdot 40 + 1)$
- Keys:
  - Public key: $KE = (55, 3)$
  - Private key: $KD = 27$
- Signature:
  - $M = 51$
  - $S = 51^{27} \bmod 55 = 6$
- Verification:
  - $S = 6^3 \bmod 55 = 216 \bmod 55 = 51$

Encryption and hashing

- Encrypted communication
  - infeasible:
    - given C and P, determine K such that $C = E_K(P)$
    - given P and K, determine K' such that $E_K(P) = E_{K'}(P)$
    - (decode without key)

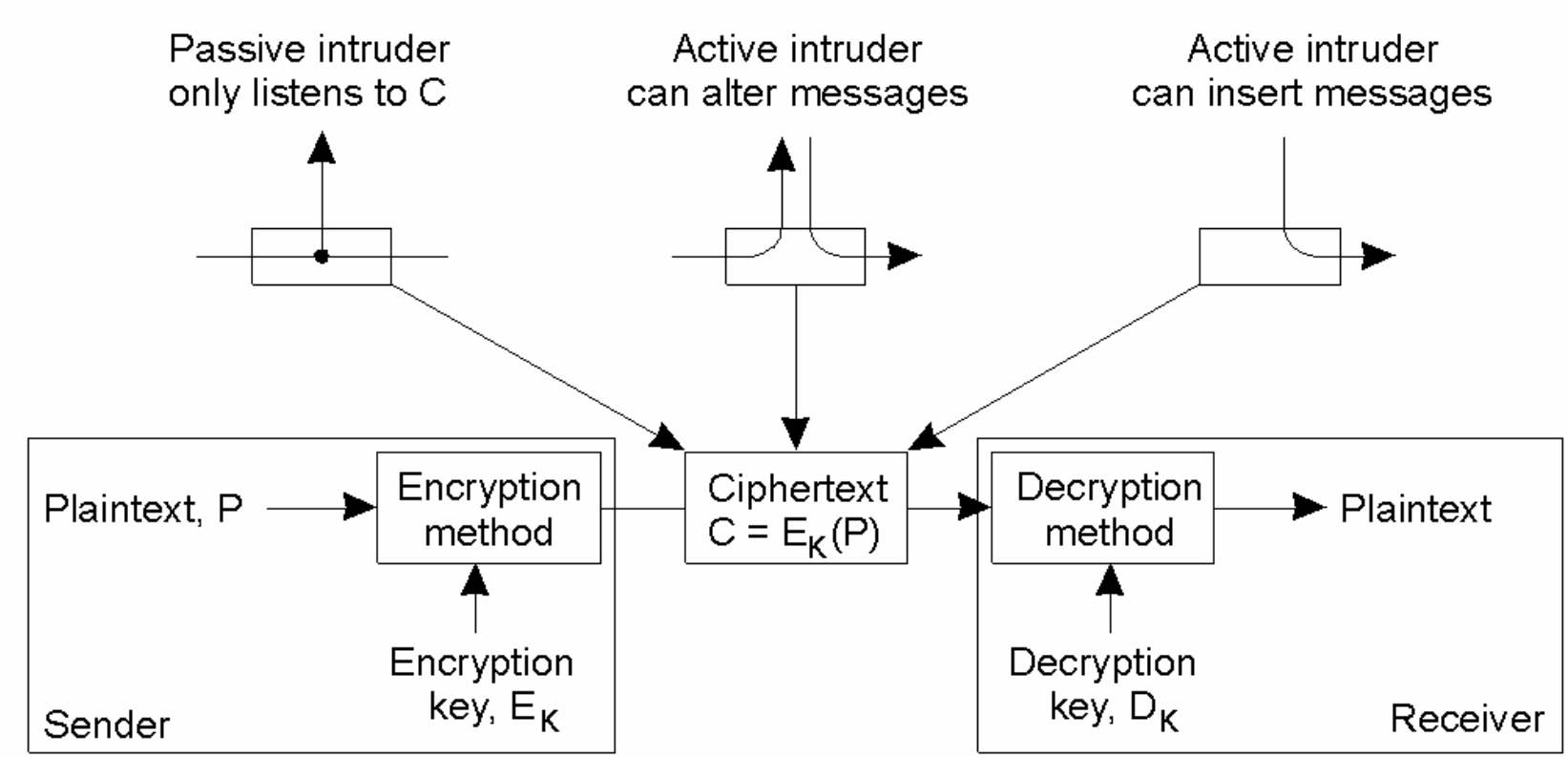
- Hashing: map message m into a fixed but sufficiently large domain through an injection H
  - $H(m)$ (fixed size *digest*, e.g. MD5, SHA-1)
  - infeasible:
    - (strong collision resistance) find m, m' such that $H(m) = H(m')$
      - note: MD2, MD4 and MD5 have been broken in this respect
    - (weak collision resistance) given m, $H(m)$, find (non-trivial) m' such that $H(m) = H(m')$

What can be done with this?

- Authentication
- Secrecy
- Integrity
- Non-repudiation
- Plausible deniability

Example: interception and modification

- Protect against interception and modification using symmetric keys



Example: interception and modification

- Separate authentication and protection with asymmetric keys
  - Make sure only intended destination (Bob) understands
    - encode using Bob's public key (must be the encoder)
  - Authentication: make sure destination knows it's you (Alice) that sent it
    - encode using Alice's private key (A's public key is the decoder)
  - Both: secure channel

Symmetric vs. Asymmetric

- Generally, symmetric schemes are more efficient
  - though asymmetric schemes may choose one of the keys to represent a simple computation
    - e.g. RSA with public key fixed to 2^16+1 (needs 17 computational rounds)

Applying encryption

- Stream ciphers
  - xor the data stream with a sequence of pseudo random numbers derived from shared seed (key)
    - RC2/4
  - fast, efficient
    - just need a good random generator
  - more vulnerable for substitution attacks

- Block ciphers
  - encrypt fixed sized blocks into same sized blocks
    - Data Encryption Standard
    - AES, Advanced Encryption Standard
      - Rijndael algorithm

Some attacks

- Substitution
  - replace a known part of a stream cipher with a new part
    - *C xor m' = C xor m xor m xor m'*
    - *C xor m*: observed; *m*: known; *m':* replaced part;
    - example: *m* is the amount to be paid
- 'Deduction'
  - learn from identical blocks
- Hence
  - don't want to have same input blocks encrypted same
  - don't want to reveal encryption of known parts
- Solution: chaining
  - xor with previous *encrypted* block (to avoid drastic effects of loss)
  - need an *intialization vector* for the first

Overview

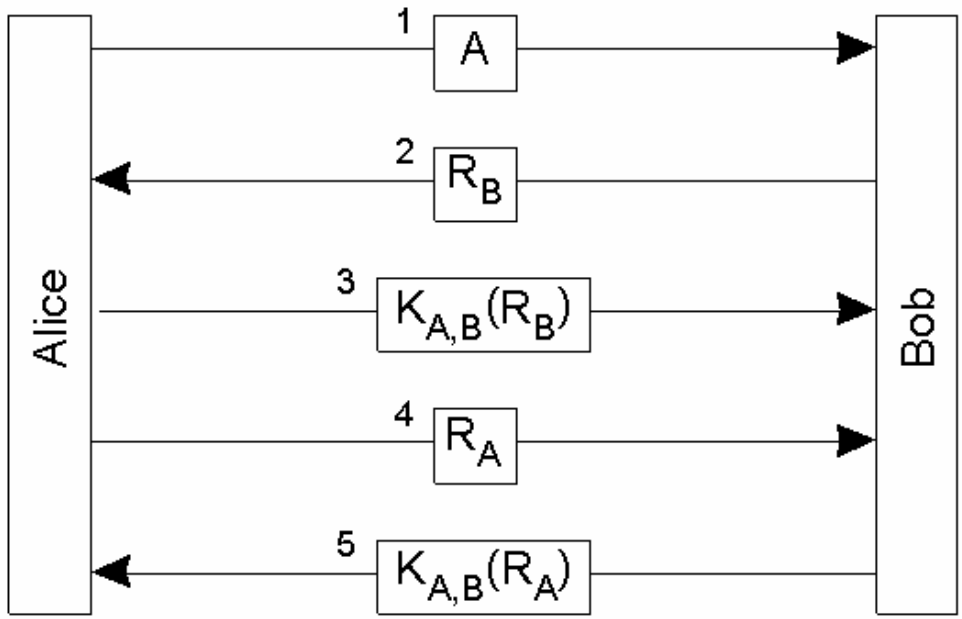
- Common notions
- Architectural aspects
- Mechanisms for authentication and secrecy
  - encryption basics, public & private keys
  - protocols for secure channels
  - signatures
- Access control
- Security management
  - key distribution
  - delegation

Secure channels

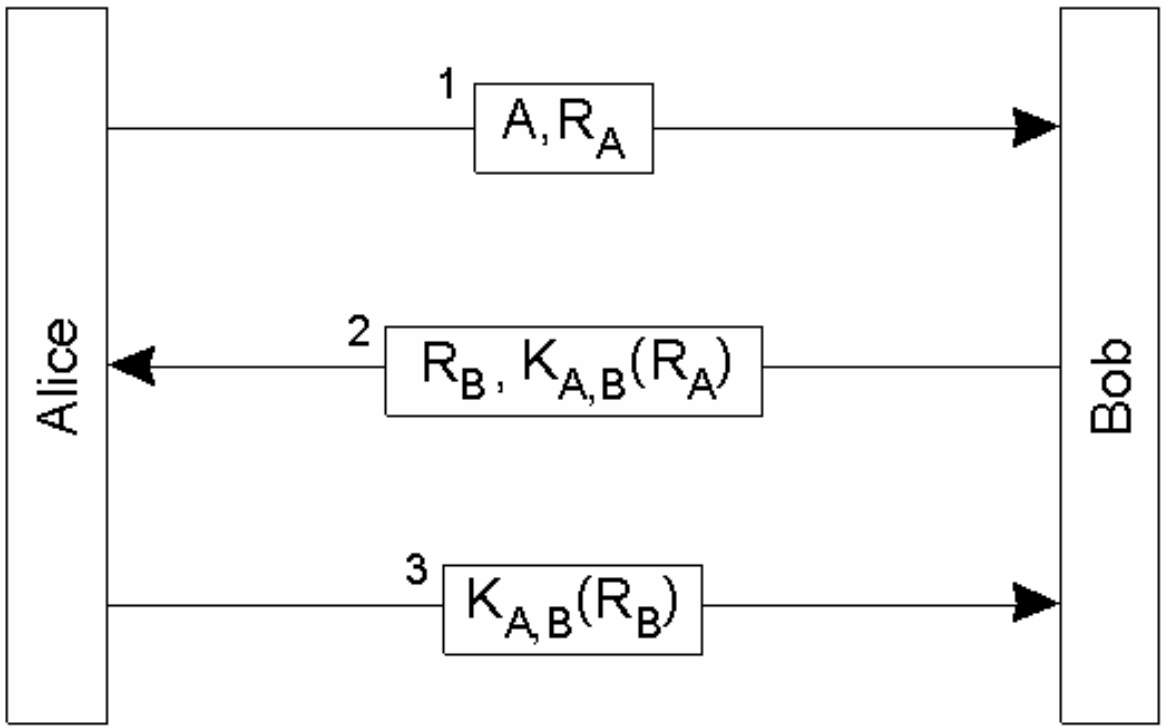
- Authentication
  - know where you're talking to
- Integrity
  - no tampering with the data
- Confidentiality
  - no listening in
- Non-repudiation
  - both sides know and can prove to third parties where the messages came from
- Attacks include the cryptographic mechanisms
  - reflection (reflect challenges), replay, use stale keys, act as if (masquerade)

Authentication using shared key ("shared secret")

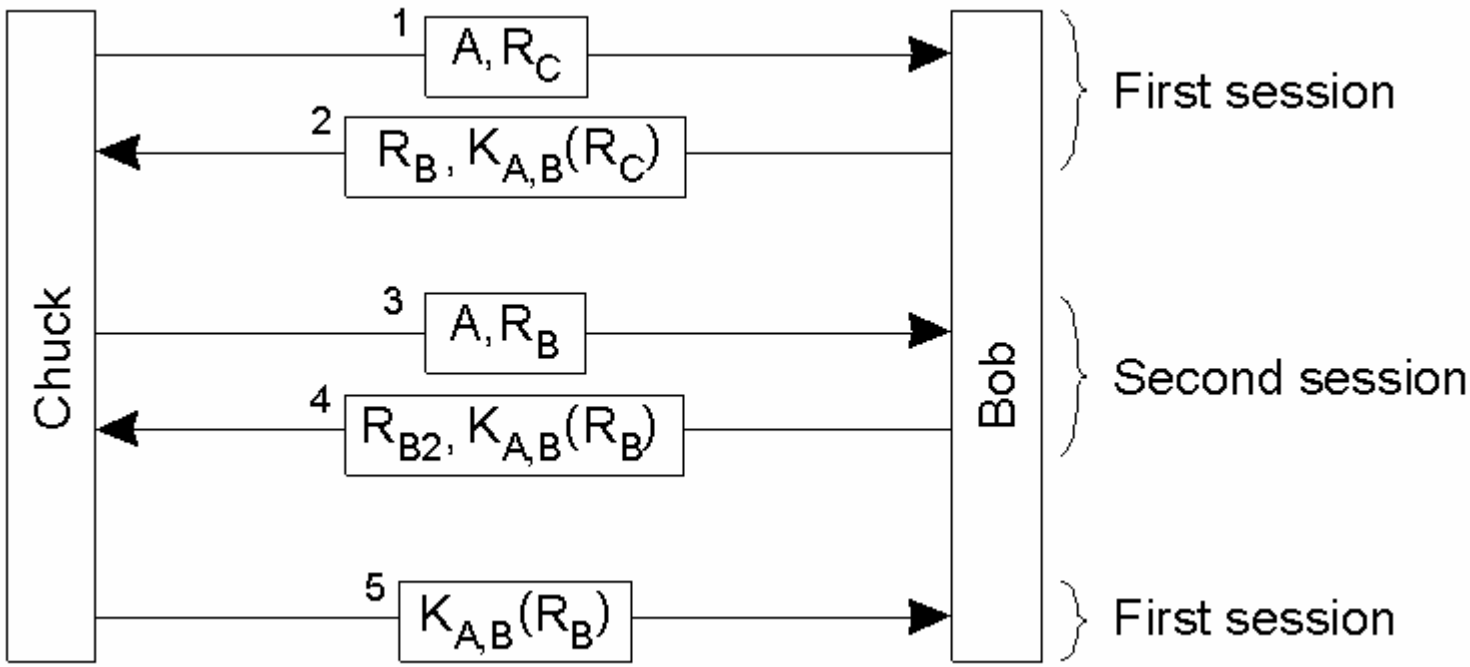
- Challenge-response
- Challenge is a random number, used only once (*nonce*)



More efficient?

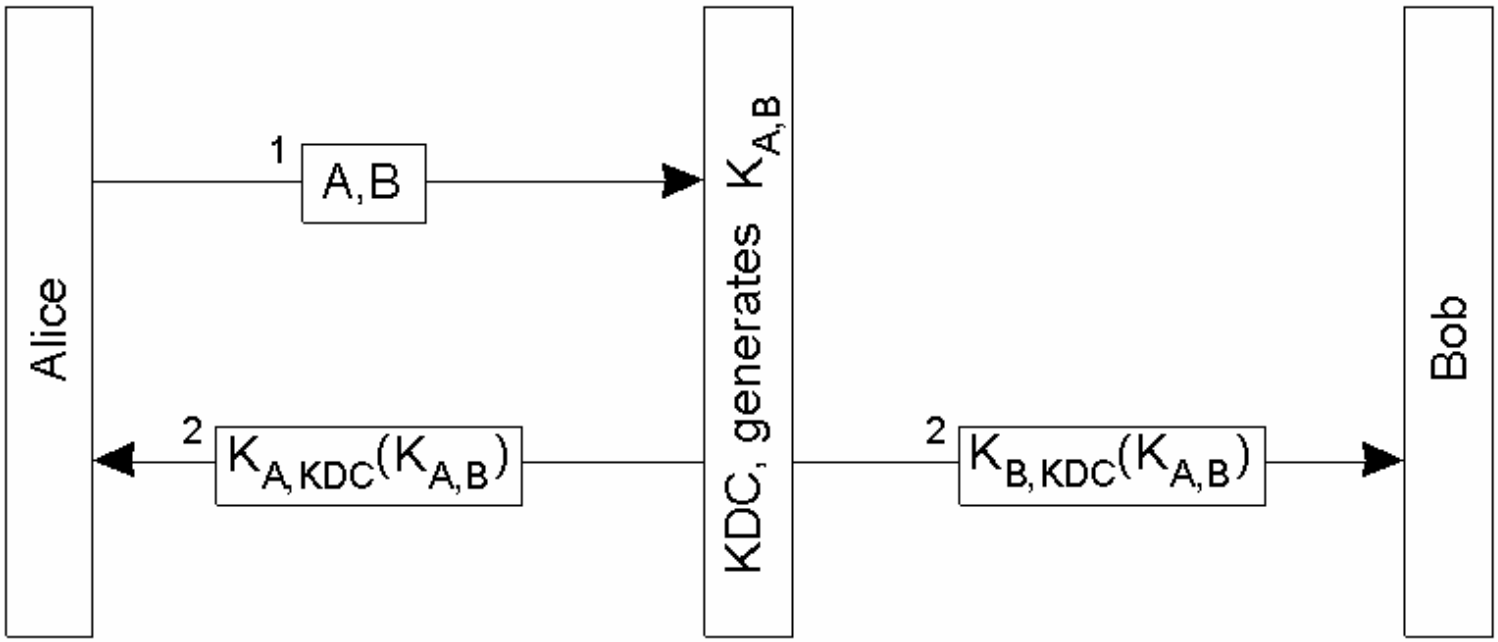


Reflection...


Chuck
Bob
1
A, R_C
2
R_B , K_A,B(R_C)
3
A, R_B
4
R_B2, K_A,B(R_B)
5
K_A,B(R_B)
First session
Second session
First session

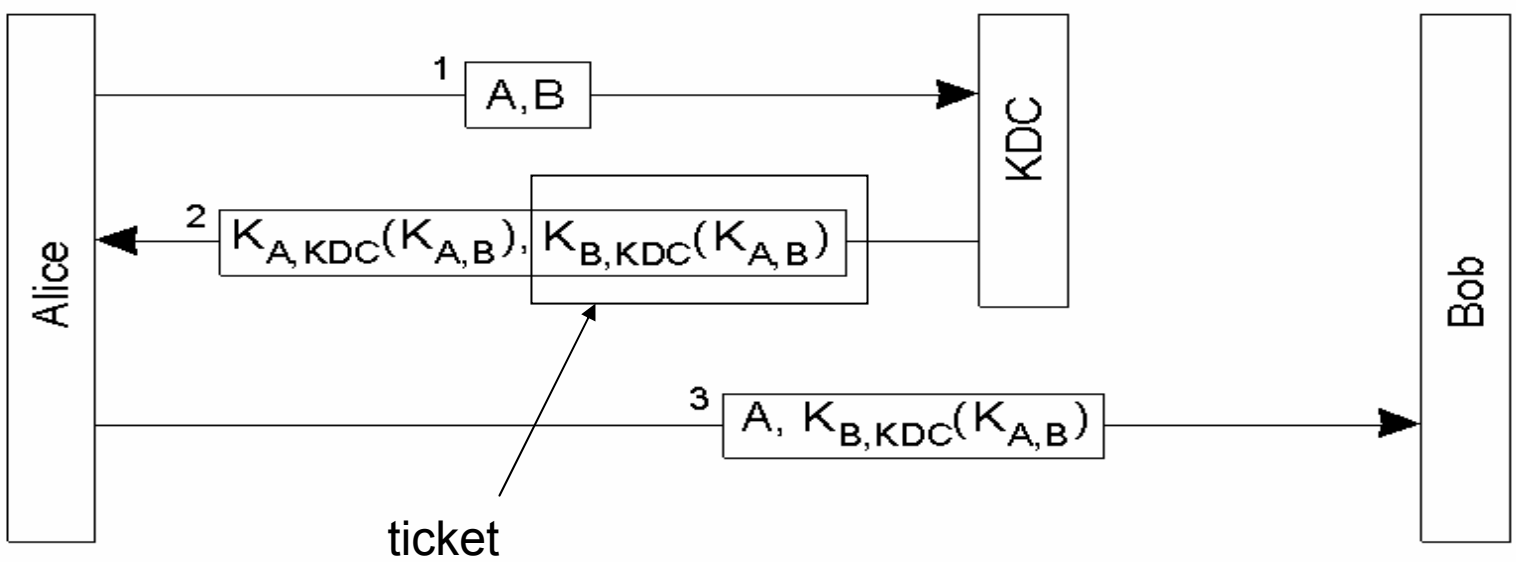
Key distribution center

- Avoid quadratic number of keys for shared keys
- Generate keys on demand by trusted party

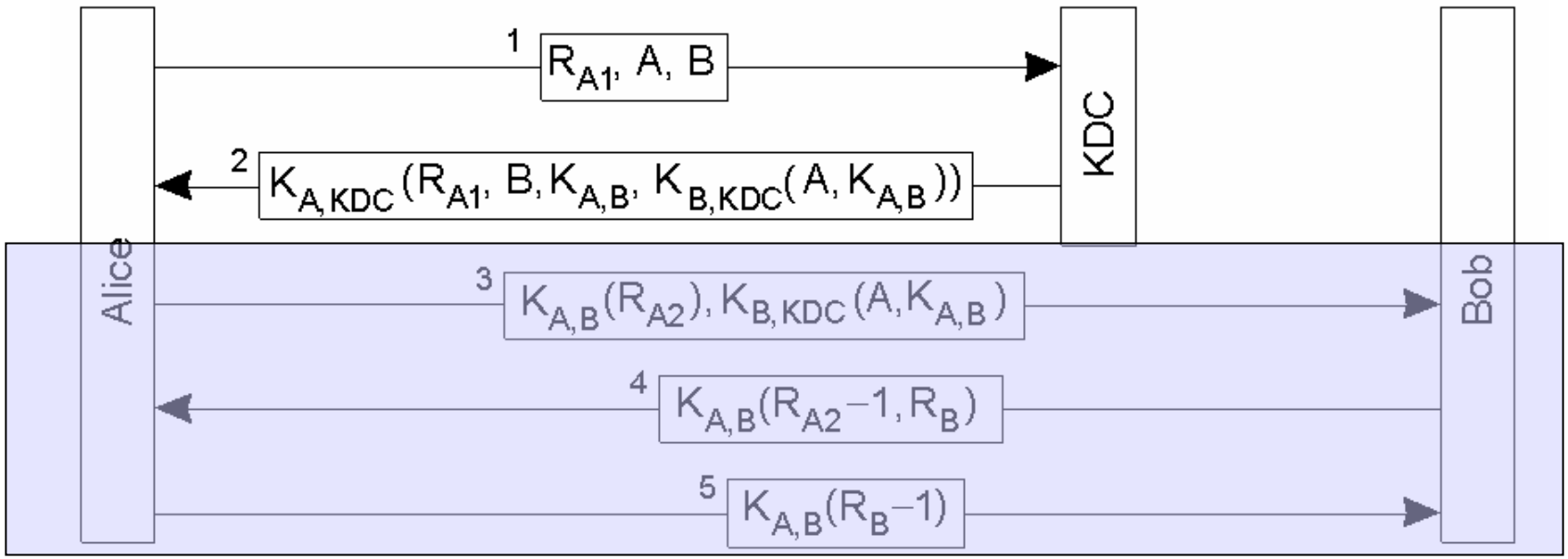


Solve order problems

- Let Alice arrange it – send *ticket* to Alice
- Too simple:
  - several replay attacks possible
    - e.g. if a $K_{A,B}$ has been stolen or Chuck replaces Bob
  - no authentication of Alice

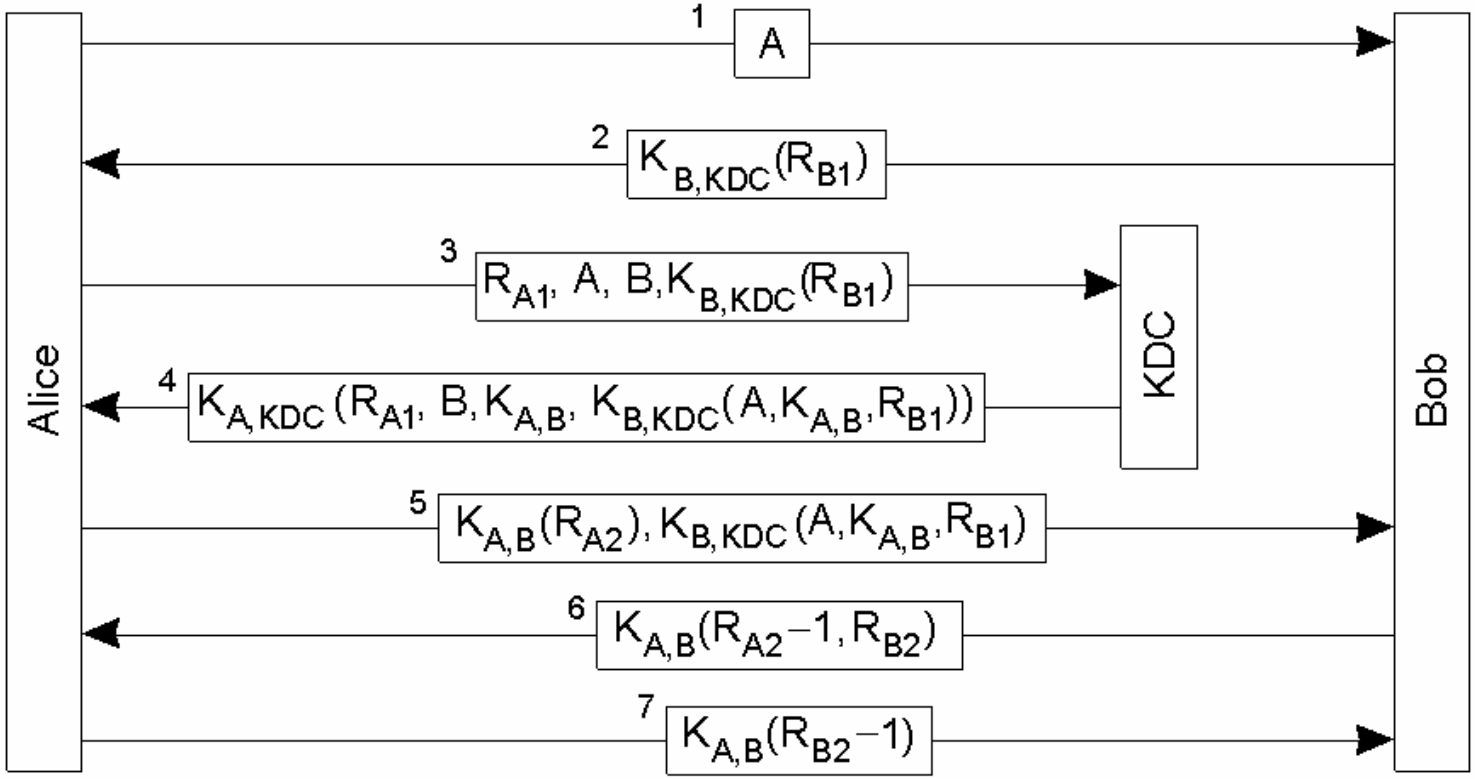


The Needham-Schroeder authentication (symmetric key)



- Challenge R_{A1}: a *nonce* – random number used only once
- Why Bob into the reply and Alice in the ticket?
- Must the ticket to Alice be encrypted?
- Why subtract 1 from the 2nd and 3rd nonce?
- Weakness: stealing an old $K_{A,B}$

Adapted N-S protocol

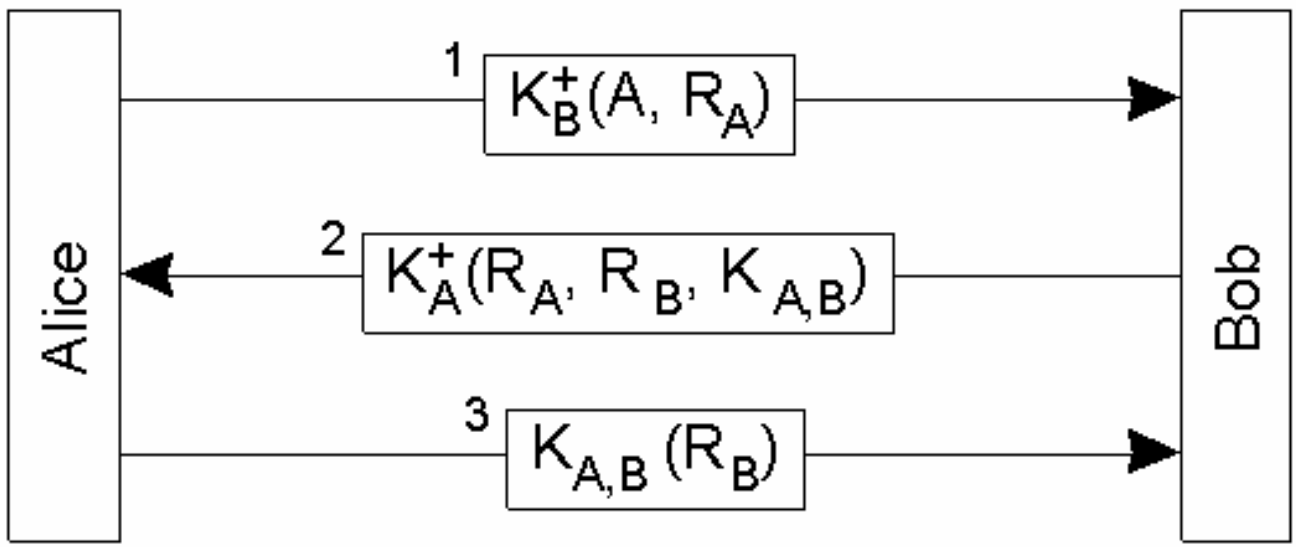


Problems with KDC

- Does not scale to millions
  - need hierarchical organization
    - Kerberos takes a few steps
  - single point of attack
    - DOS or stealing database of keys
    - fault tolerance support extends threats to copies

Public key authentication (Needham-Schroeder '78)

- Used for deciding on a session key
- Needs guarantee that the used public key are indeed Alice's and Bob's
  - otherwise, man-in-the-middle attack (Lowe, '96)



Some rules of thumb

- Assume all communication can be overheard
- Valuable information only to known parties
- Avoid solutions that require parties to do essentially the same as before
  - e.g., repeating a message, re-using a message, encrypting the same message
- Make challenges unique; don't re-use
- Avoid to let a party operate as an 'encryption server' or invest work (DOS)
  - let the one that seeks communication do the first encryption step
    - question: what about the previous protocols?
  - let the steps that lead to refusal be cheap for the server and expensive for the intruder
    - e.g. simple initial check (certain 'cookies;)

Rules of thumb (cnt'd)

- Do not encrypt the same message twice with same result
- Connect series of communications that belong together
  - not predictably (not: serial number)
  - protected
- Use session keys for the actual communication rather than the authentication keys
  - limit wear & tear
    - establishing authentication keys is expensive
  - avoid replay
  - limits the risk of revealing *old* sessions
    - forward secrecy
- Realize yourself that mechanisms have assumptions
  - choice depends on requirements!

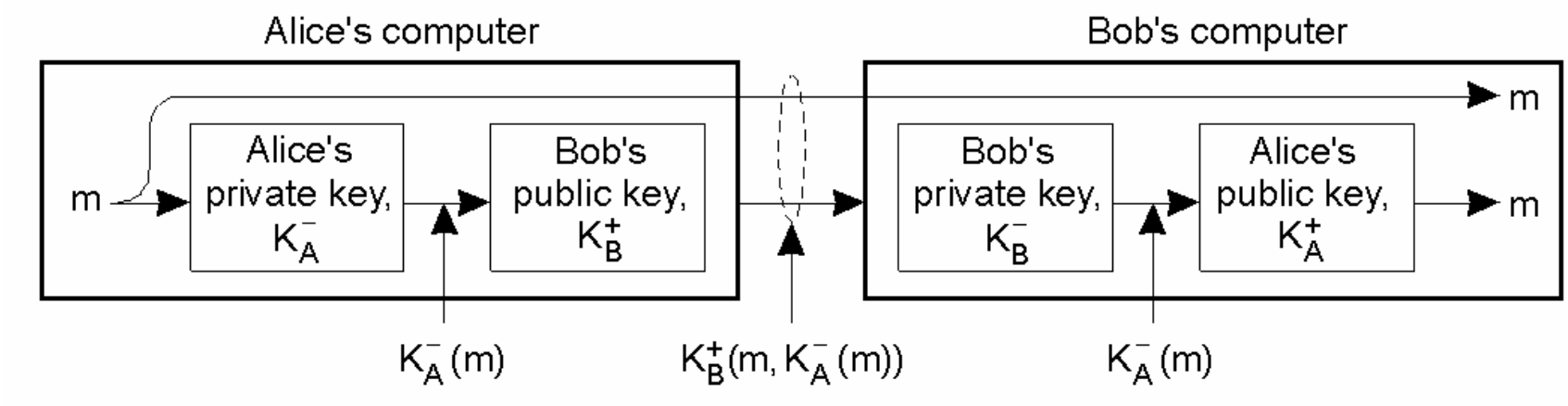
Overview

- Common notions
- Architectural aspects
- Mechanisms for authentication and secrecy
  - encryption basics, public & private keys
  - protocols for secure channels
  - signatures
- Access control
- Security management
  - key distribution
  - delegation

Digital signature

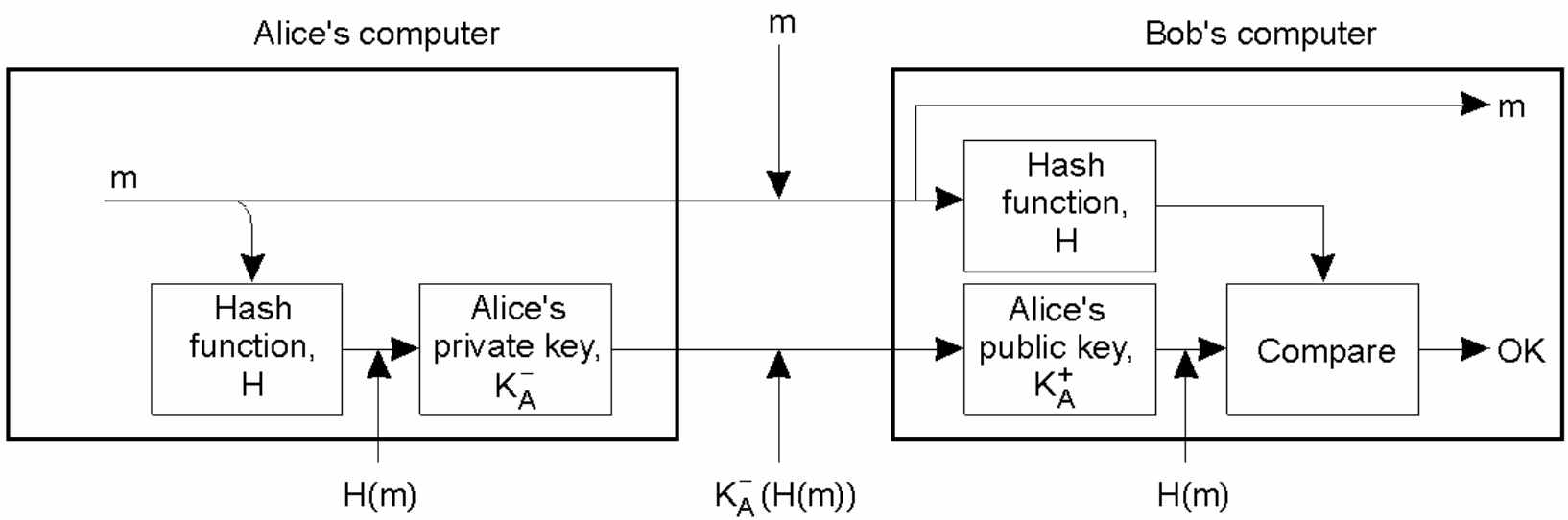
- Integrity may go beyond the communication
  - you may want that a *message itself* can be proven to originate from Alice
    - Alice cannot deny having sent it
    - Bob cannot change it
  - the message may be a (public) key
- Combine a message with a signature that cannot be removed
- Notice: this separates secrecy and authorization

Digital Signatures



- Enough?
  - changed or stolen keys?
  - message length – use a hash function

Improvement: use a hash function



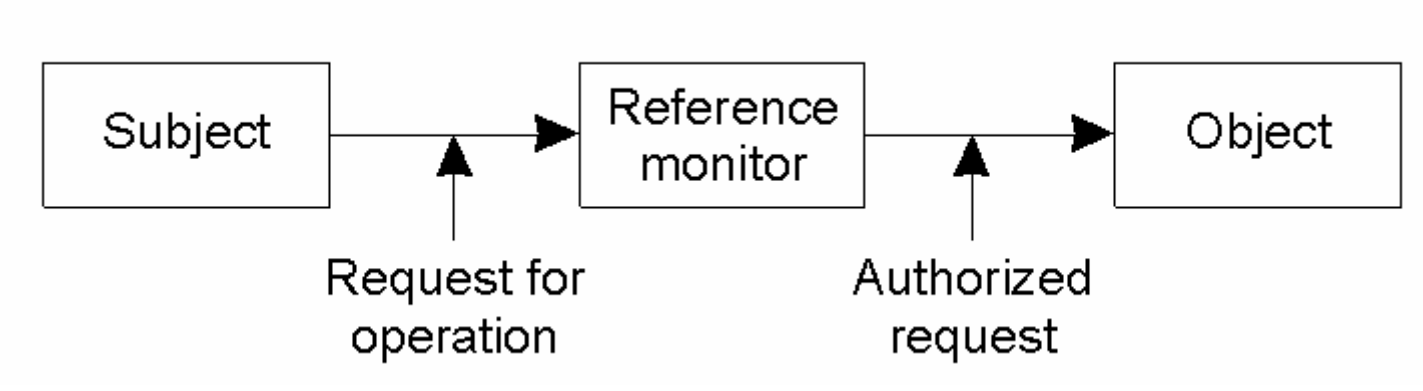
- Note: *H* must be resilient to message extension
  - not: *H(m++m') = easy_function (H(m), H(m'))*

Overview

- Common notions
- Architectural aspects
- Mechanisms for authentication and secrecy
  - encryption basics, public & private keys
  - protocols for secure channels
  - signatures
- Access control
- Security management
  - key distribution
  - delegation

Access control

- Terminology

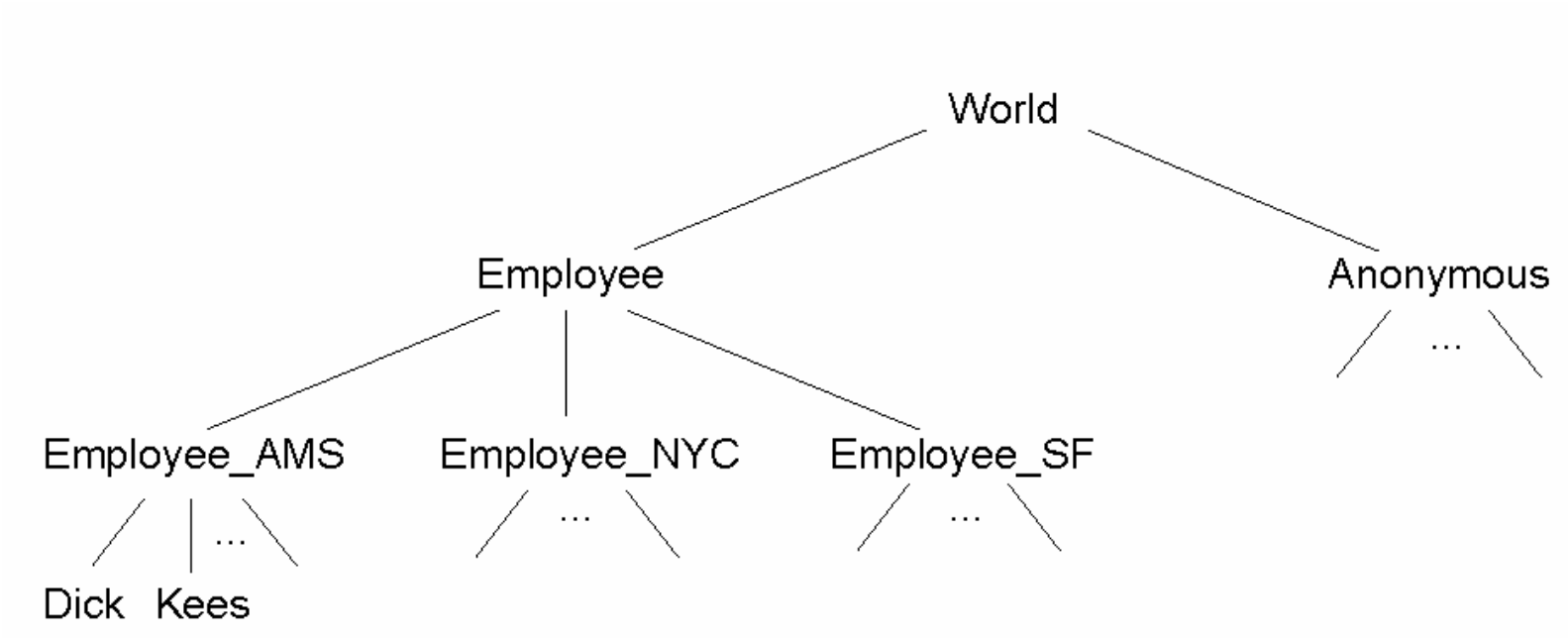


- *Access control*: enforce (verify) access rights
- *Authorization*: grant access rights
- *Authentication*: verify identity

Access Control Matrix

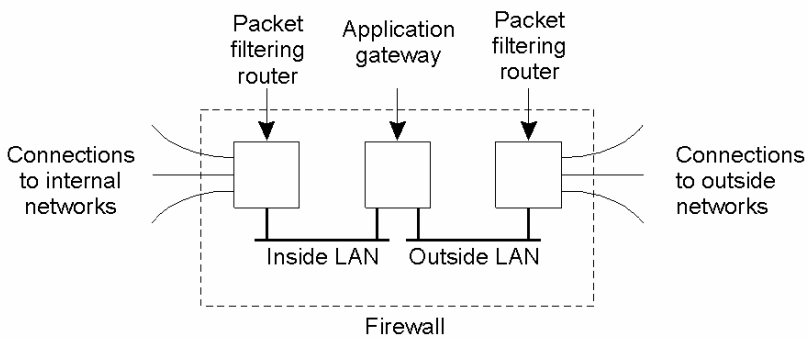
- *ACM [s,o]:* list of operations permitted by *subject* s on object *o*
- **Capability**: *an (unforgeable) string specifying access rights of a particular object-subject pair (special credential)*
- *ACM* implemented
  - as a subject list per object
  - as a *capability* list per subject ("ticket"); protected by signature
- Can improve by hierarchy
  - e.g. groups of users, per location etc.

Protection Domains



Firewalls

- Some resources and services are rather accessible
  - e.g. mail, ftp, ...
- Even with protection, attacks can be annoying
- Hence, limit the *communication*
- Special reference monitor:
  - low-level filtering
  - drop packets
  - inspect contents
  - proxy



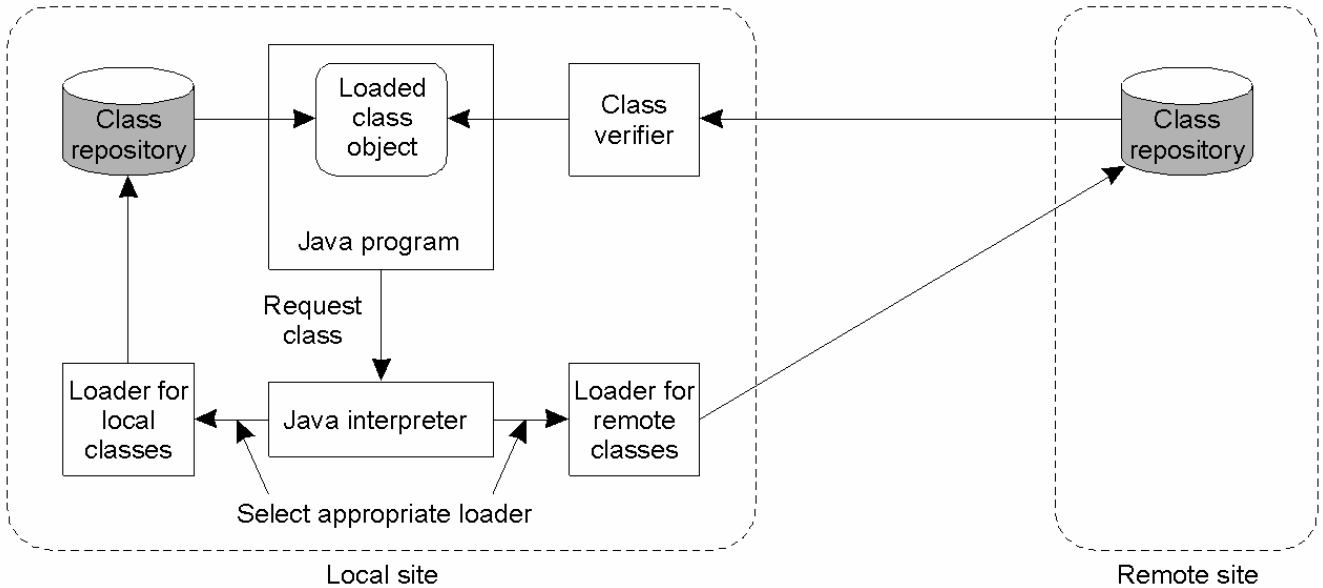
Security and mobile code

- Protect host system against malicious code
  - agents
  - downloads on request, e.g. applets
    - sandbox (code isolation), playground (dedicated machine)

- Protect mobile code against malicious hosts
  - agents
    - read-only state: signed by owner
    - append-only log: add public-key encrypted messages that contain checksum – records tampering
    - selective revealing: data available for specific servers using public-key encryption
  - not fully possible

The sandbox

- Make it *fully checked* to access vital resources at target
  - fully checked: all resource access through a reference monitor
  - limits application domain

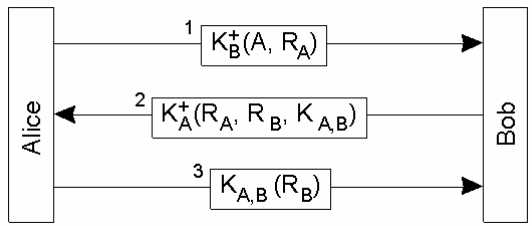


Overview

- Common notions
- Architectural aspects
- Mechanisms for authentication and secrecy
  - encryption basics, public & private keys
  - protocols for secure channels
  - signatures
- Access control
- Security management
  - key distribution
  - delegation

Security management

- How to obtain (initial) keys?
  - symmetric: transmit using out-of-band methods
    - e.g. regular mail
  - public keys must be paired with owner
    - certificate: (like credential) a document (statement) signed by an authority (principal)
      - example: capability
      - moves trust to public key of authority

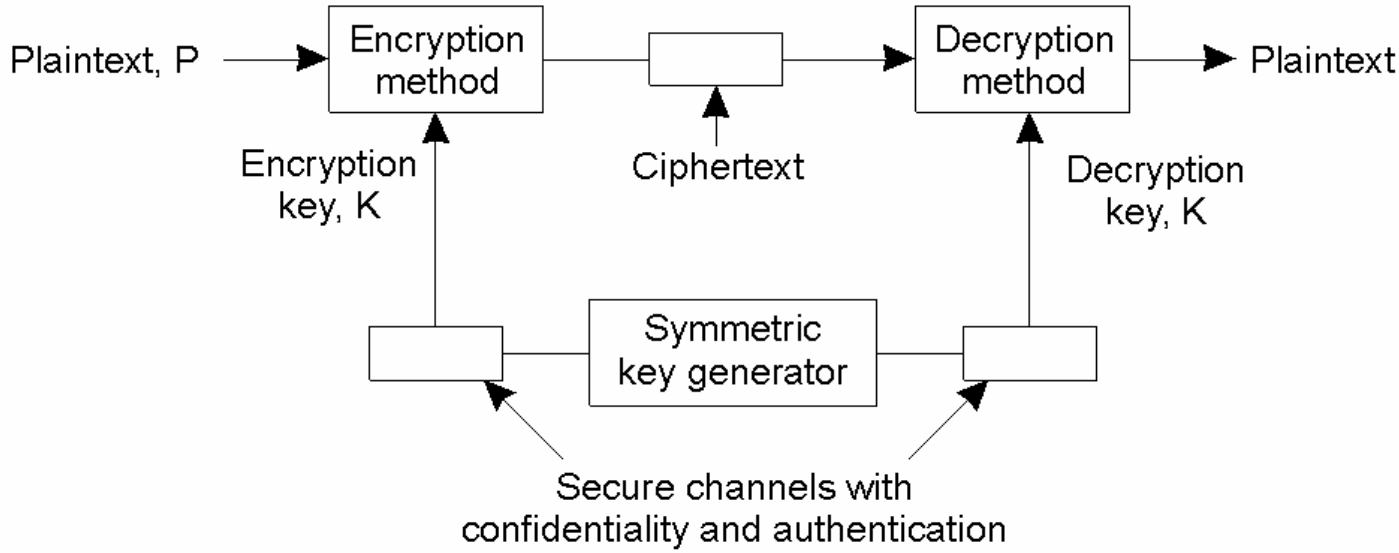


- Manage a group of trusted servers
  - e.g. member addition

- Management of authorization and access control
  - construction and management of capabilities
  - access right *delegation*
  - construction *and revocation* of certificates

Distribute a secret key

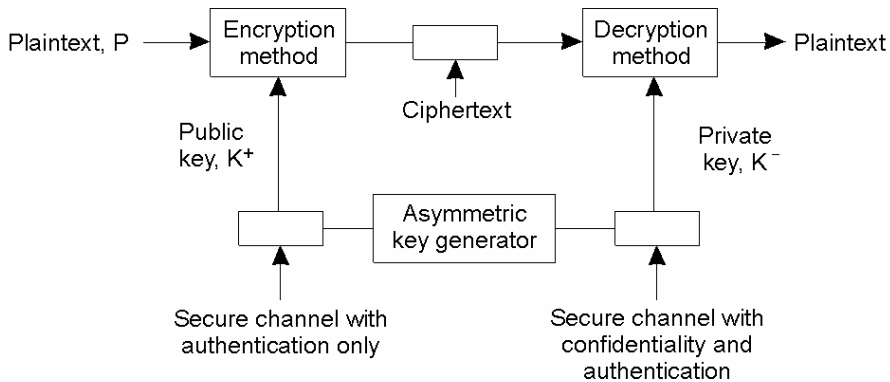
- Needs confidentiality and authentication



(a)

Distribute a key

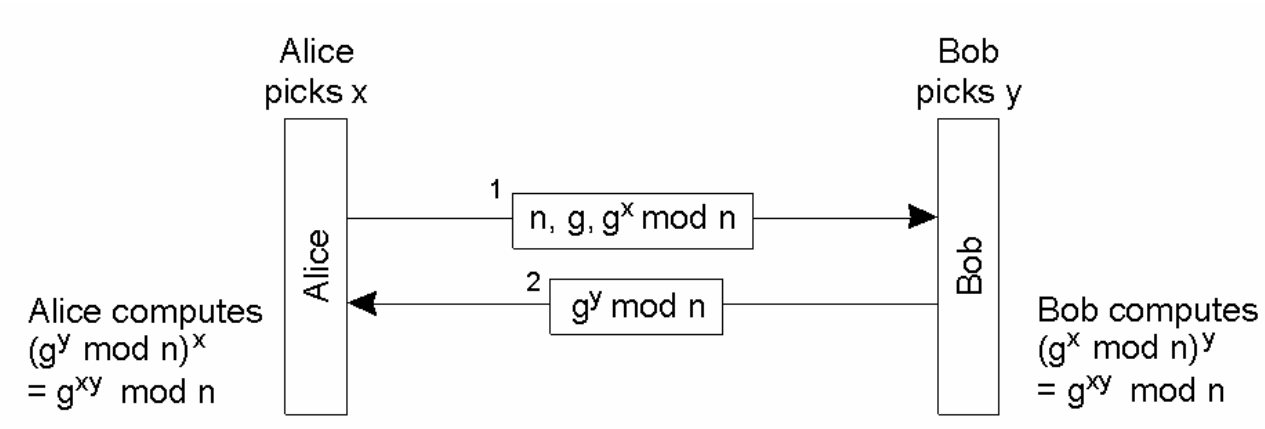
- Public: requires just authentication
  - public-key *certificate*
    - contains identifier of associated party (user, authority, ...) and public key
    - is signed
    - needs to be revoked, or has a lease
- Private: confidentiality and authentication



(b)

Diffie-Hellman key exchange

- *g* and *n* are large, and public

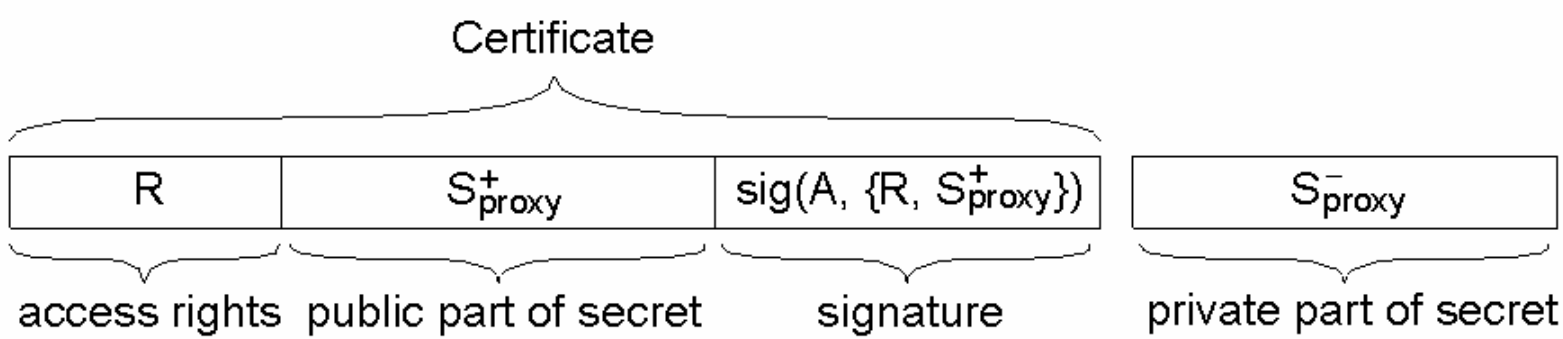


- Alice: public: g^x mod n, private: x
- Shared key: g^{xy} mod n
- Susceptible to man-in-the-middle attack
  - though protecting D-H with PKI leads to a solution of perfect forward secrecy

Delegation

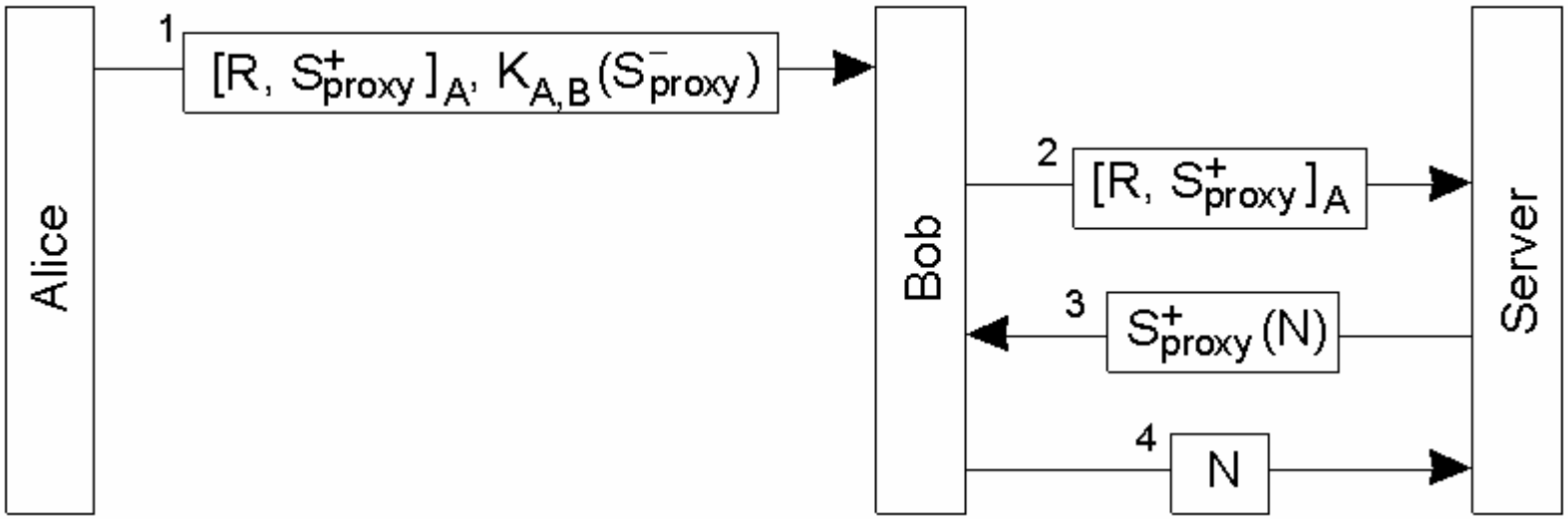
- Ask a server to act on behalf of you
  - e.g. a printer

- Proxy, encoding rights
  - certificate saying: “proxy has these rights”
  - asymmetric key, just for this purpose



Delegation

- Using a proxy to delegate and prove ownership of access rights.



Example systems

- Systems: Kerberos, Sesame, COCA
- Electronic payment
  - electronic money
  - privacy
- Transport Layer Security (derived from Secure Socket Layer)
- IP-secure
  - framework, leaving encryption, authentication and hashing functions open