DI-FCT-UNL
Segurança de Redes e Sistemas de Computadores
Computer Networks and Systems Security

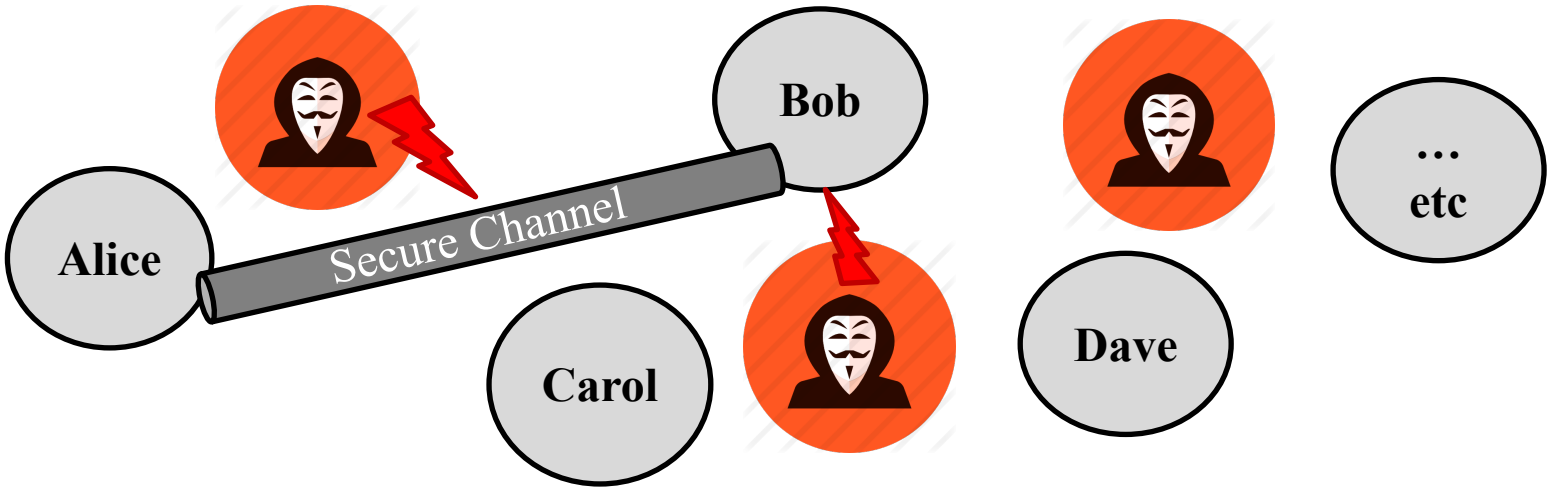
Mestrado Integrado em Engenharia Informática
MSc Course: Informatics Engineering
1st Sem., 2020/2021

Key Distribution: Models and Protocols

Last lectures

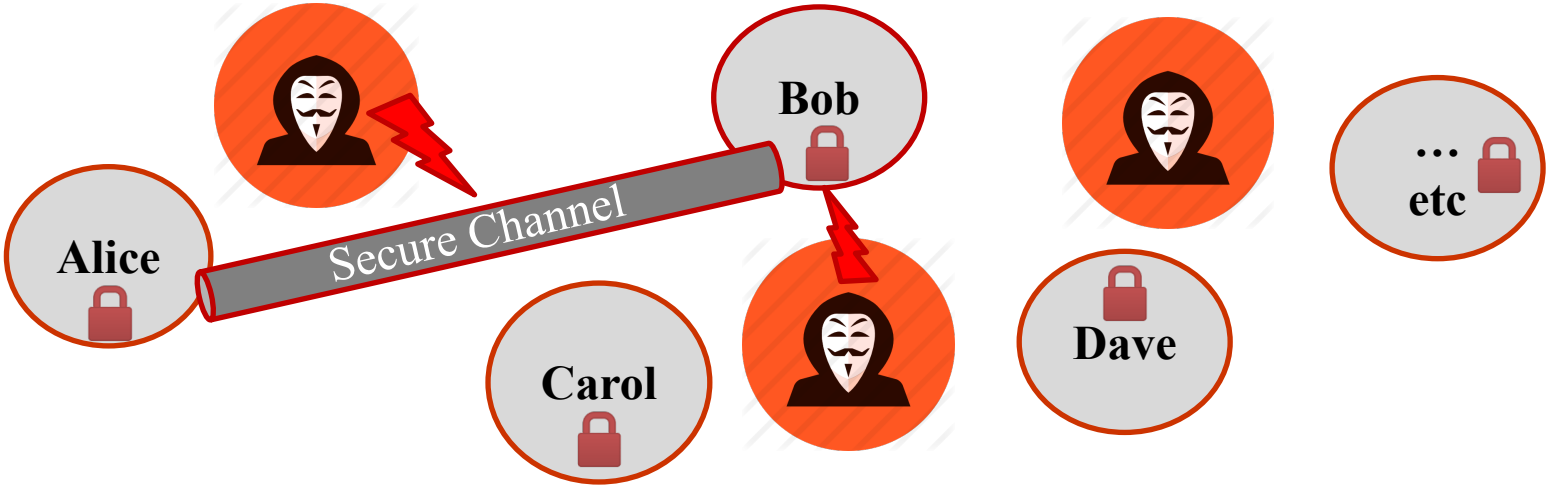
- Cryptography, Applied Cryptography
  - Symmetric and Asymmetric Cryptography
  - Secure hash functions
  - MACs and CMACs
  - Digital signatures
- Combination of cryptographic methods (algorithms, methods, constructions and techniques) to design and to build secure channels supporting secure communication protocols
  - Correct assumptions to address Security and Performance trade-offs
- … But we need to manage, distribute and/or establish **cryptographic keys** (or also other related **security association parameters**) **in a "secure way"** !

The problem …



- Alice, Bob, etc … (correct principals in endpoints) want to establish secure channels
  - Need to obtain, establish and manage, securely, cryptographic keys and/or related secrecy parameters, to setup channels with end-to-end security arguments (remember OSI X.800 security properties, services and mechanisms)
- Opponents (adversaries): want to attack communication channels
  - Target: attack the protocols used to distribute and establish keys (remember the possible attack typology in the OSI X.800 active or passive attack types), possibly acting as MiM attackers
  - Another target: attacks against the way keys are locally managed by principals

The solutions …



- Alice, Bob, etc … (correct principals in endpoints) need:
  - To use secure key-distribution protocols, resistant to attackers
    - Protocols must be designed with the required security properties, using proper mechanisms and trust-computing base assumptions
  - To manage locally keys and related security association parameters, maintained with the required secrecy conditions
    - Perimeter defenses + Isolation (or containment)
    - Key-Management security principles, good practices and enforcement means for storage and management (processing)

Strategies for key-distribution

Also applied for the secure distribution and establishment of any security association (secrecy) parameters

Possible base solutions:

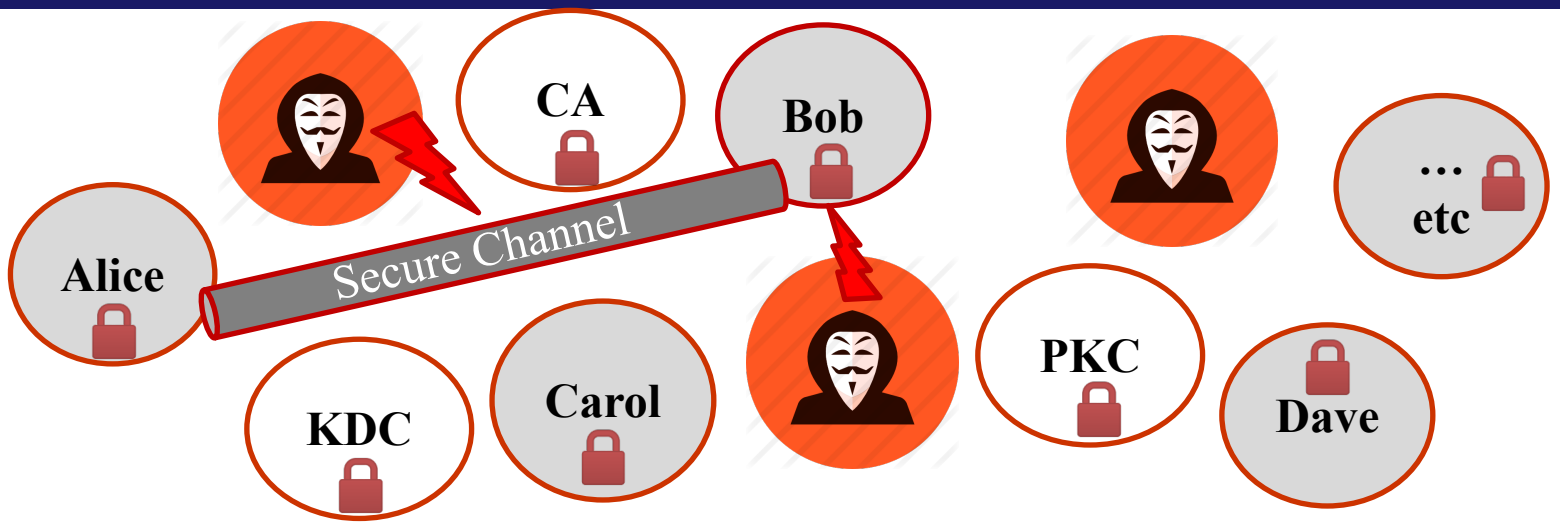
Physical distribution models (good idea ?)

- A key could be selected by A and physically delivered to B, involving a key-agreement between A and B
- A third party could select the key and physically deliver it to A and B

Key distribution protocol models for Distributed Principals (better idea ?)

- If A and B have previously used a key, one party could transmit the new key to the other, encrypted using the old key
- If A and B each have an encrypted connection to a trustable third party C, then C could deliver a key on encrypted links to A and B.

Strategies for key-distribution protocols and Trusted Third Parties



- Intermediated key-distribution protocols using Trustable **KEY-DISTRIBUTION CENTERS (KDCs)**: Particularly targeted for the use of **Symmetric Cryptographic Techniques**
- Using trustable **PUBLIC-KEY Centers (PKCs)**: Particularly targeted for the use of Asymmetric Cryptographic Techniques
- Using certified Public Keys with **certificates issued and signed by trustable Certification Authorities**

Outline

- Key Distribution Issues
- Key Distribution Models and Protocols

Outline

- **Key Distribution Issues**
- Key Distribution Models and Protocols

Key-Distribution Problem: Permanent or Master Keys

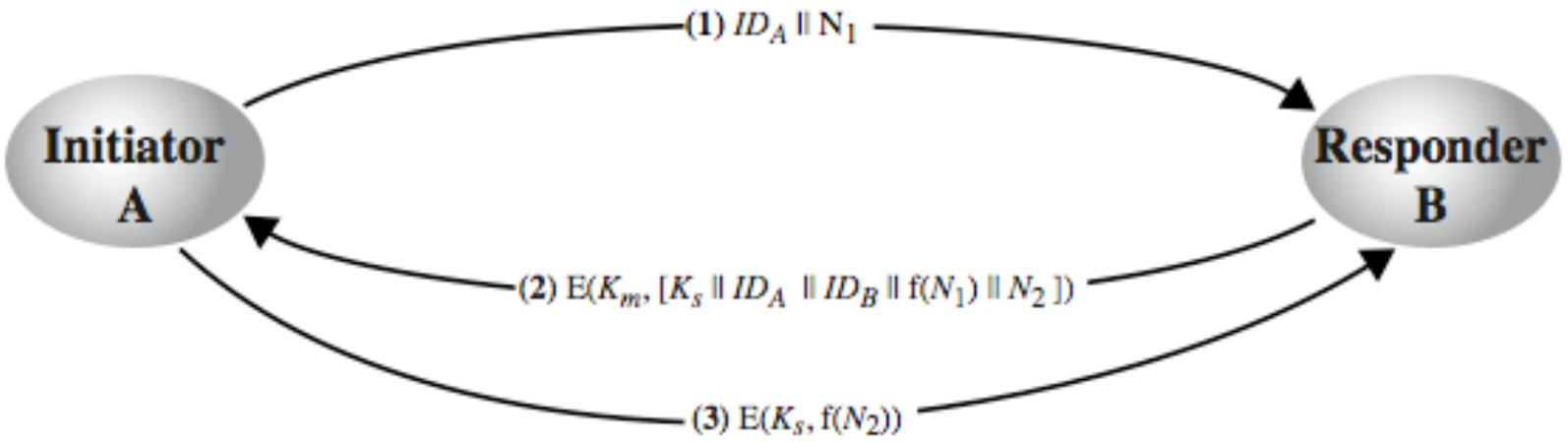
- **Permanent or master keys** are used between entities for the purpose of distributing "new or fresh" session keys
  - Used as long-time shared keys for the renegotiation of new session keys (rekeying process)
  - Examples:
    - Pre-shared symmetric keys between principals and KDCs
    - Public-keys (of Asymmetric-Keypairs) registered in PKCs
    - Certified public-keys registered and issued in public-key certificated by Certification Authorities (CAs)
  - Keep in mind: long-time duration secrets – critical to be managed, requires enforcement of security management
  - Permanent keys also can be used under "time-validity" conditions …. Problem of revocation of permanent keys !

Key-Distribution: How to securely establish session keys

- **Session keys (used as "ephemeral" or "short-duration" keys):**
  - Used as one-time pad for each message. Realistic ?
    - Establishment and rekeying for each "message" or each "encryption"
  - Used as one-time key per "session". More realistic ?
    - Establishment once per session, rekeying for each session
    - A session as a message-exchange flow between two principals
    - At the conclusion of each session, the session key will be destroyed (not used anymore)
  - One-time key per "validity time". After the limit-time the key must be revoked !
    - Can require the proper control of "synchronized time" and valid timeliness conditions
- **Depending on the use we want to establish keys or to have rekeying schemes to be secure and fast enough**

Discussion: decentralized control

- Peer-to-Peer Key Establishment Protocol



- Requires that each principal must be able to communicate in a secure way will all the other principals (implying that pre-shared master or permanent keys must exist before)

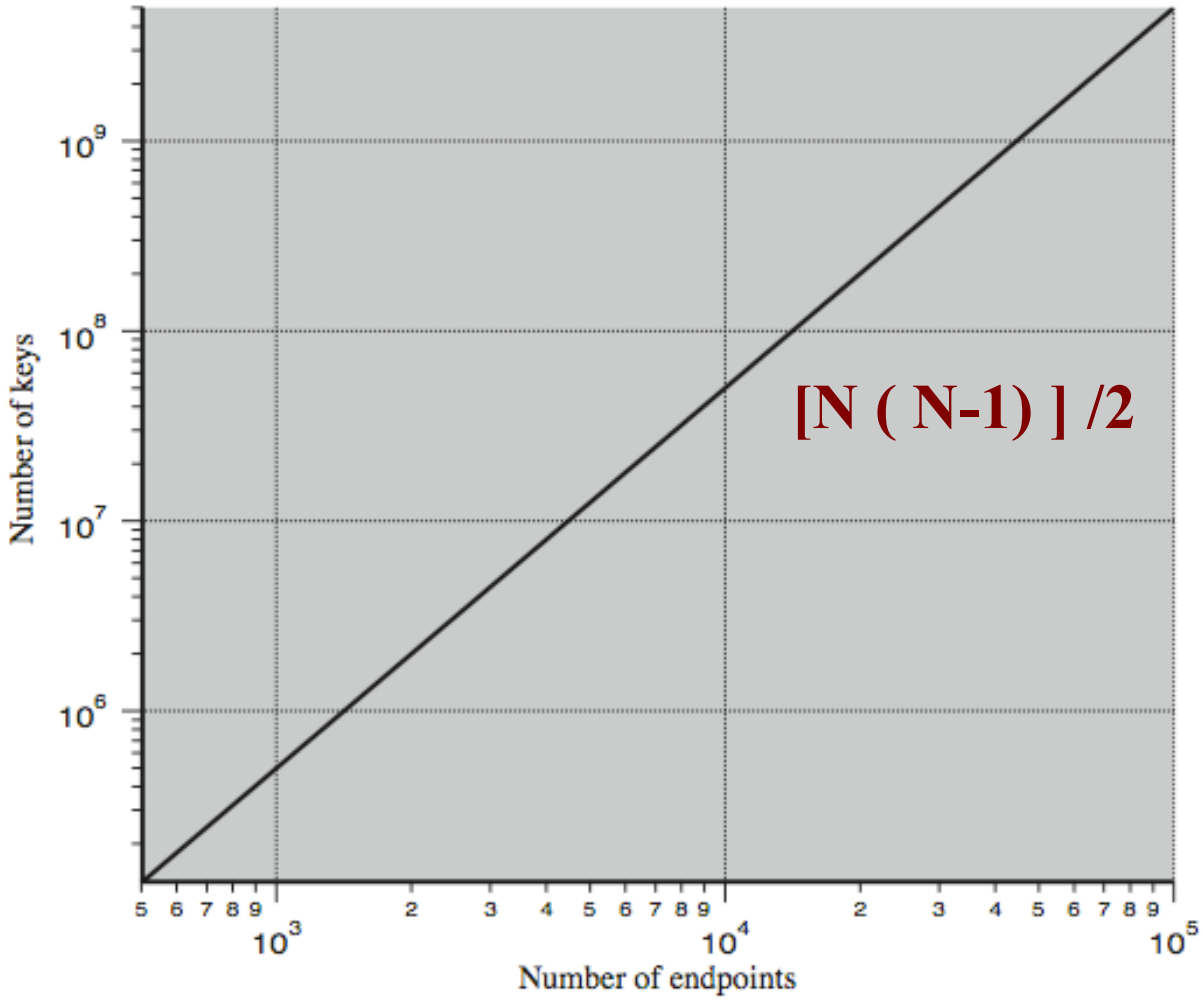
Establishment and management of symmetric keys: Scale vs. Shared Keys

Scale problem and key-management issues

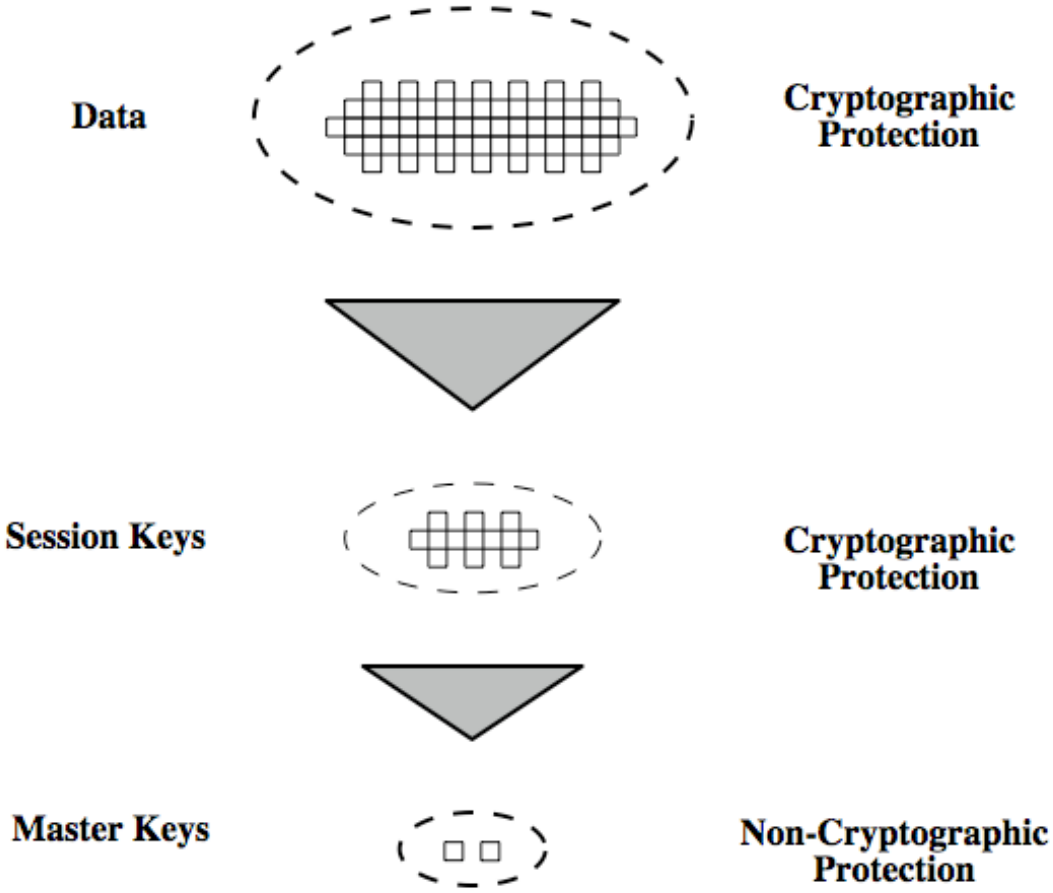
- **How many shared keys are managed by each principal ?**
  - Possibility: 1 only key, shared by everybody (not good idea)
  - One key shared by each pair of principals (pairwise keys):
    - » 2 principals: 1 key : a-b
    - » 3 principals: 3 keys : a-b, b-c, a-c
    - » 4 principals: 6 keys : a-b, a-c, a-d, b-c, b-d, c-d
    - » 10 principals: 45 keys
    - » 11 principals: 55 keys, ...
    - .....
    - » N principals: Combinations (N, 2-2) = N(N-1)/2
  - Problem: are shared keys used under "P2P or End-to-End security arguments" ?
    - What if "one of the shared keys" is compromised ... ?
    - How to retrieve loss or compromise keys ?
    - How to dismiss (revoke) compromised keys

The scale problem:
How can we solve this problem ?

O^2 complexity

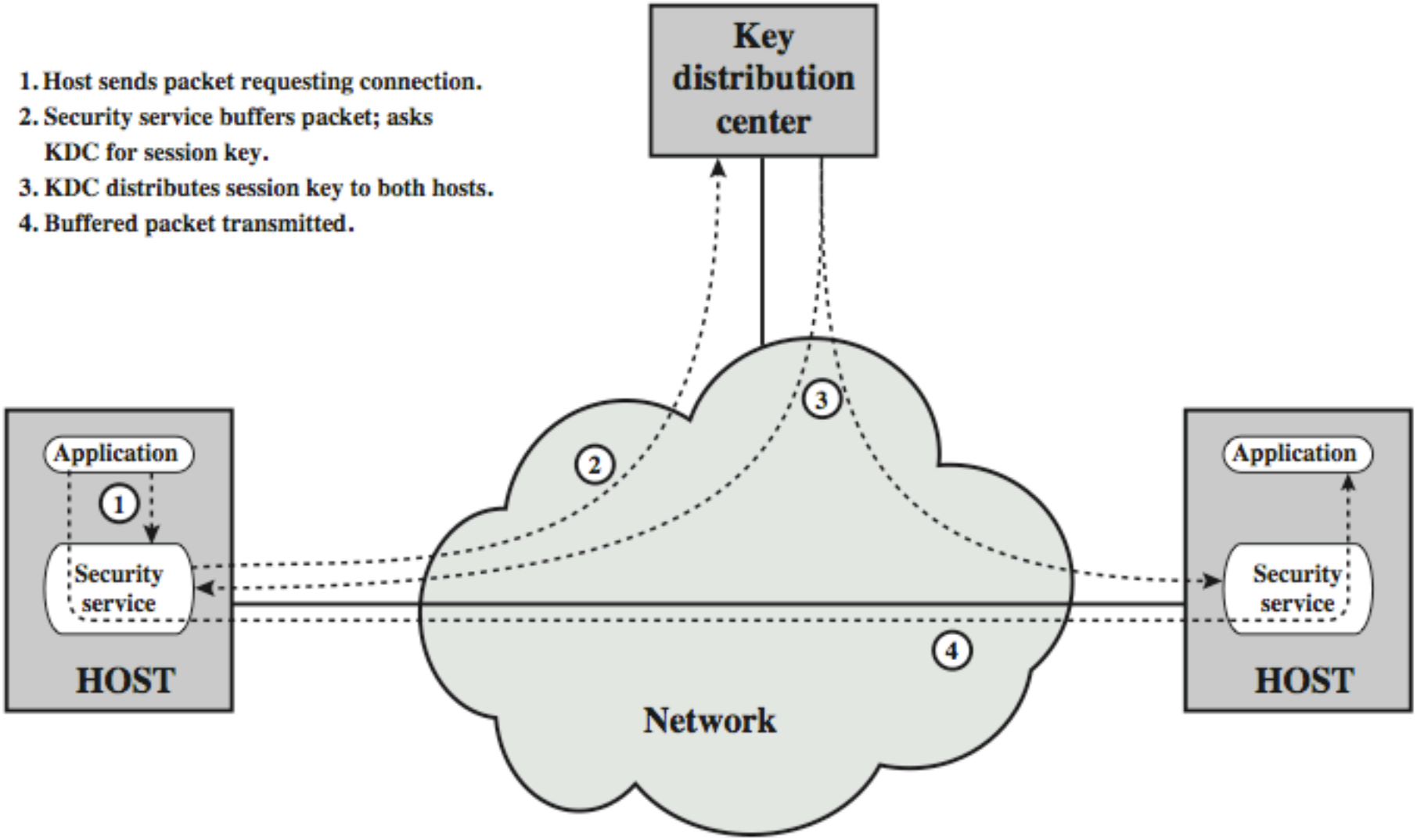


Key-Hierarchy

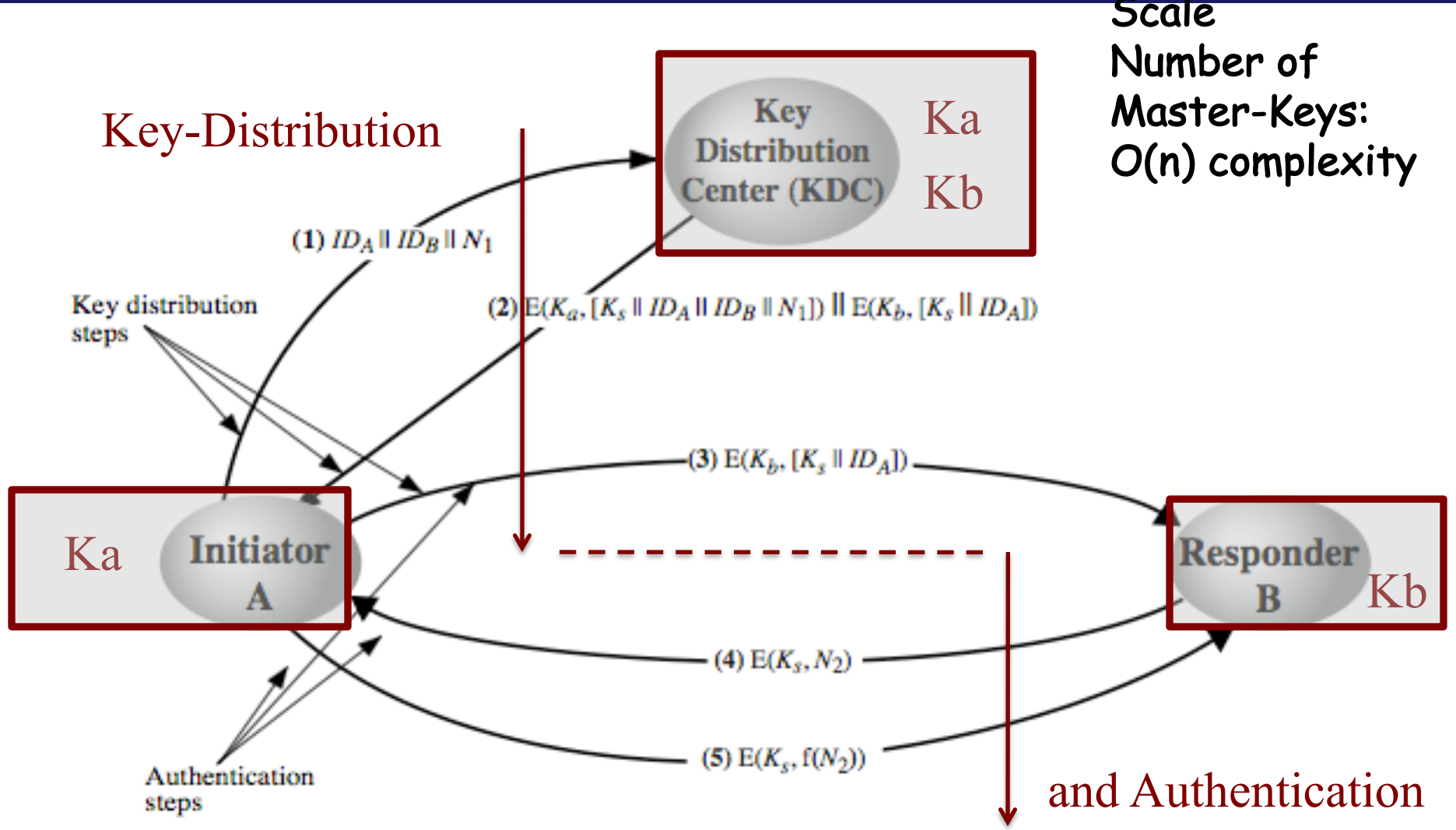


Master Keys: Shared as Long-Lasting Secrets, between Principals and Key-Distribution Centers

Key-Distribution Scenario with a KDC



Key-Distribution Scenario with a KDC



Scale
Number of
Master-Keys:
O(n) complexity

Discussion: implementation of hierarchies

- Hierarchical key control: possible implementation of more than one level for KDCs
  - Good for scalability, load-balancing, avoidance of single points of failures/attacks
  - A KDC in a hierarchy may be responsible for a "small" domain (ex., a LAN, a LAN segment, etc)
  - Secure communication between principals in different domains:
    - Local domain KDCs can ask for a key to a KDC in the next layer of the hierarchy

Discussion: lifetime of session-keys

- More frequent "rekeying" => more security
  - "hard" for brute-force and cryptanalysis attacks
- But rekeying => more overhead
  - More latency, network-traffic burden, synchronization of keys, …
- The appropriate balance is an issue

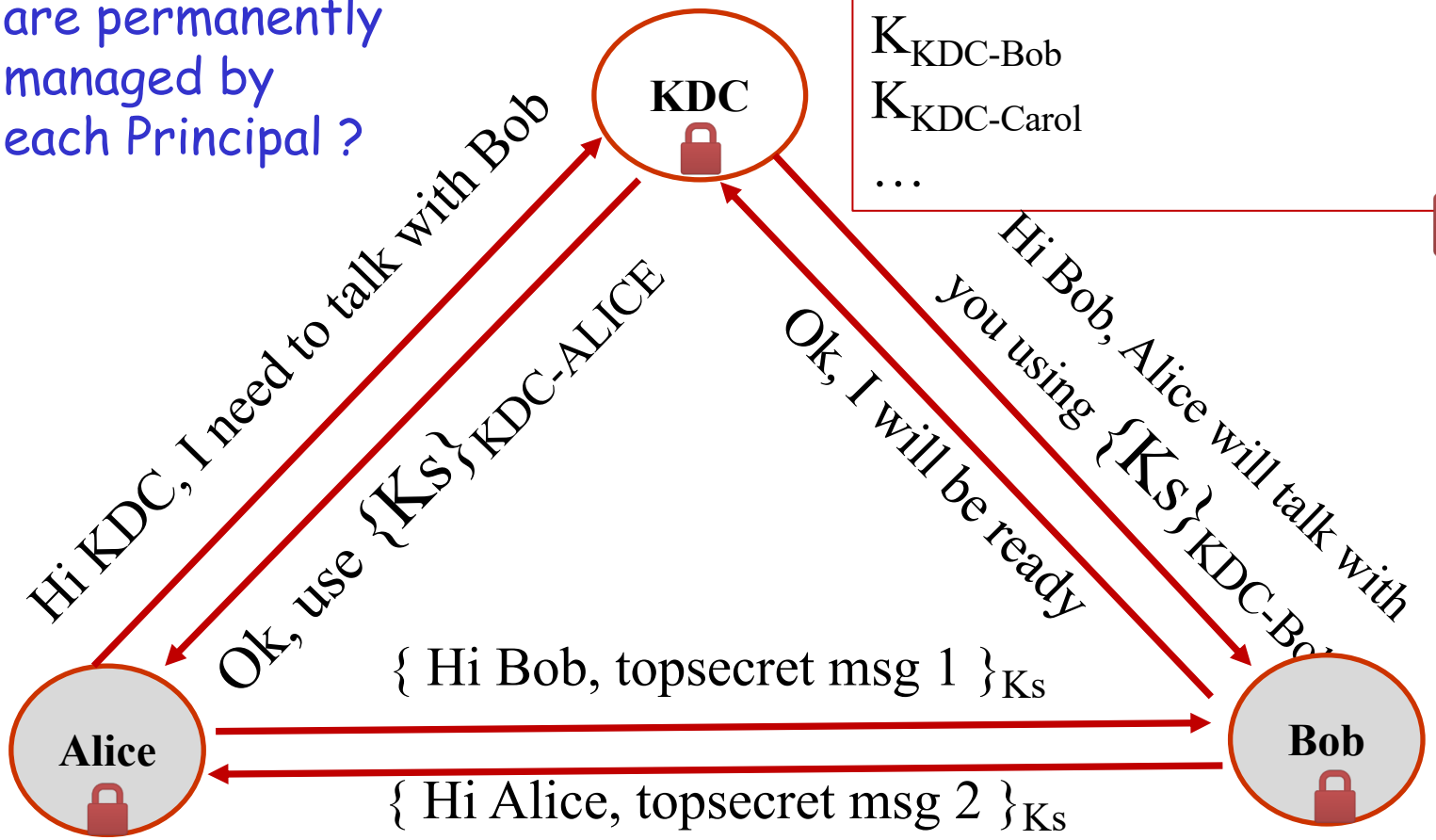
Discussion: lifetime of session-keys

- Some obvious choices:
- Connection-oriented communication: Key-distribution and establishment when connections are open, lifetime until the connection close
    - But what if connections are "long" ?
    - Ex., rekeying in each PDU sequence number cycle
    - Ex., temporal "rekeying"
    - Ex., Random-based rekeying
    - Ex., Event-based rekeying
- Connectionless communication
    - Ex., Transaction-Oriented Protocols
        - Security: New session key for each key-exchange
        - Optimistic: "fresh" session keys with a certain time to live period or a certain number of transactions
        - Criteria as in "long connections" are also possible

A Simple Symmetric key distribution model

How many keys are permanently managed by each Principal ?

Pre-Shared Permanent Keys:
$K_{KDC\text{-}Alice}$
$K_{KDC\text{-}Bob}$
$K_{KDC\text{-}Carol}$
…

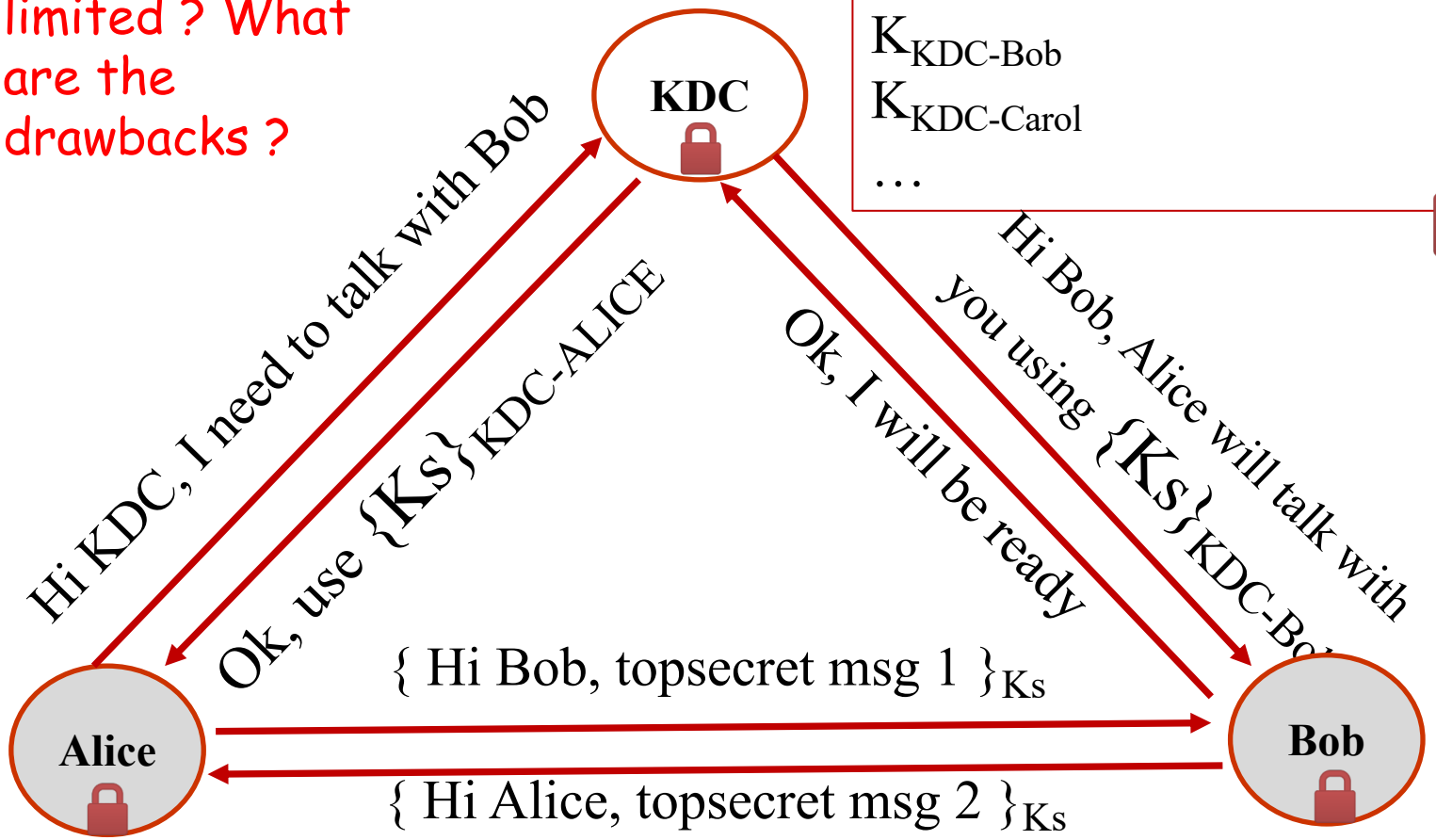


Pre-Shared Permanent Keys:
$K_{KDC\text{-}Alice}$

Pre-Shared Permanent Keys:
$K_{KDC\text{-}Bob}$

A Simple Symmetric key distribution model

Is this model limited ? What are the drawbacks ?



Pre-Shared Permanent Keys:
$K_{KDC-Alice}$

Pre-Shared Permanent Keys:
$K_{KDC-Bob}$

Problem: KDCs are central points of failure ! (and also central/preferential targets of attack)

Solution:

- Can we distribute the solution using more than one KDC (with each KDC responsible for a certain number of peers?)
  - Good to mitigate failures and possible attacks
  - Each KDC can be responsible for a certain Key Distribution Domain …
  - How to address the global distribution problem ?

- A Distributed System problem: Need to address a KDC Distribution Model
  - Can use "peering KDC models"
  - Can use hierarchical models (trees of KDCs)
  - Can replicate KDCs (under certain consistency guarantees):
    - Primary Backup (or Master Slave architectures)
    - Cluster of replicated KDC instances

Another problem: security assumptions for key generation and sharing model with KDCs

- How is the key generated ? KDC is the entity that generates the shared and used keys? Too much trust on it ?
  - No-contributive methods: when a peer obtains a key generated by another entity or another peer
  - Contributive methods: when generated with the engagement of the peers that will share and will use the key
    - How to address a contributive key-generation scheme ?
- What if no contributive generation or no key generation control ?
  - How to know that such keys were generated with secure guarantees?
  - How to know that such keys are "unique" for eth establishment process ?
  - How to be sure that the keys will be used under "non-disclosure agreement (or NDA)" guarantees ?
    - NDAs involving the entity that generated the keys
    - NDAs involving the principals sharing the keys

Another problem: How to guarantee Perfect Forward vs. Perfect Backward Secrecy conditions

- How to be sure that keys were generated and distributed, under PFS and PBS guarantees ?
  - Key-independence (in successive used keys)
  - Fast Rekeying when needed

  Even greater guarantees:
  - Contributive key-generation with fairness and reliable requirements

Another problem: We will need "FAST REKEYING" mechanisms that must be able to scale

- How to be sure that keys were generated and distributed, under PFS and PBS guarantees ?
  - Key-independence (in successive used keys)
  - Fast Rekeying when needed

  Even greater guarantees:
  - Contributive key-generation with fairness and reliable requirements

  - Can we address FAST enough contributive key generation and establishment, with PERFECT FORWARD and PERFECT FUTURE GUARANTEES ?

  Key Point: Impossible to address by using KDCs and pre-shared secrets (ex, pre-shared secrets or pre-shared secrecy parameters) !!!!!

Outline

- Key Distribution Issues
- Key Distribution Models and Protocols

A summary of solutions for our study (1)

- **Key Distribution Protocols only using KDCs and Symmetric Encryption Methods**
  - Protocol Models using only KDCs and Symmetric Cryptography
  - Possible solutions using multiple and distributed KDCs
  - Authenticated Key Distribution with Kerberos (V4, V5)
- **Key Distribution Protocols using PKCs or CAs and Asymmetric Cryptographic Methods**
  - Protocols using Asymmetric Cryptography
  - PK-INIT Protocols: Example of PK-INIT Kerberos
  - A case-study: SHP protocol in your Work Assignment (Phase 2)
  - Key Distribution Protocols using Certificated Public Keys (or X509 Public Key Certificates)

A summary of solutions for our study (2)

- **Secure Key Management Issues**
  - Secure Key-Management principles (to manage securely permanent or master symmetric keys, or to manage securely private keys)
  - Encryption devices: key location and management
- **Secure Key Establishment with ZKPs (Zero Knowledge Proofs)**

Recommended Reading (with the slides)

Suggested Readings:

Random Number Generation:
W. Stallings, Network Security Essentials – Applications and Standards, Part I – Cryptography, Chap.2 – Symmetric Encryption and Message Confidentiality (Section on: TRNGs, PRNGs and PRFs)

W. Stallings, Network Security Essentials – Applications and Standards, Part II – Network Security Applications, Chap.4 – Key Distribution and User Authentication