Kerberos

Kerberos

- Authentication protocol developed to secure campus computer facilities at MIT at the beginning of 80s

- Based on Needham-Scroeder but uses timestamp instead of nonces

- Designed for a campus this for LAN/small WAN not Internet

Kerberos

- Authentication for *user-to-server* and not *user-to-user*, it isn’t *peer-to-peer*

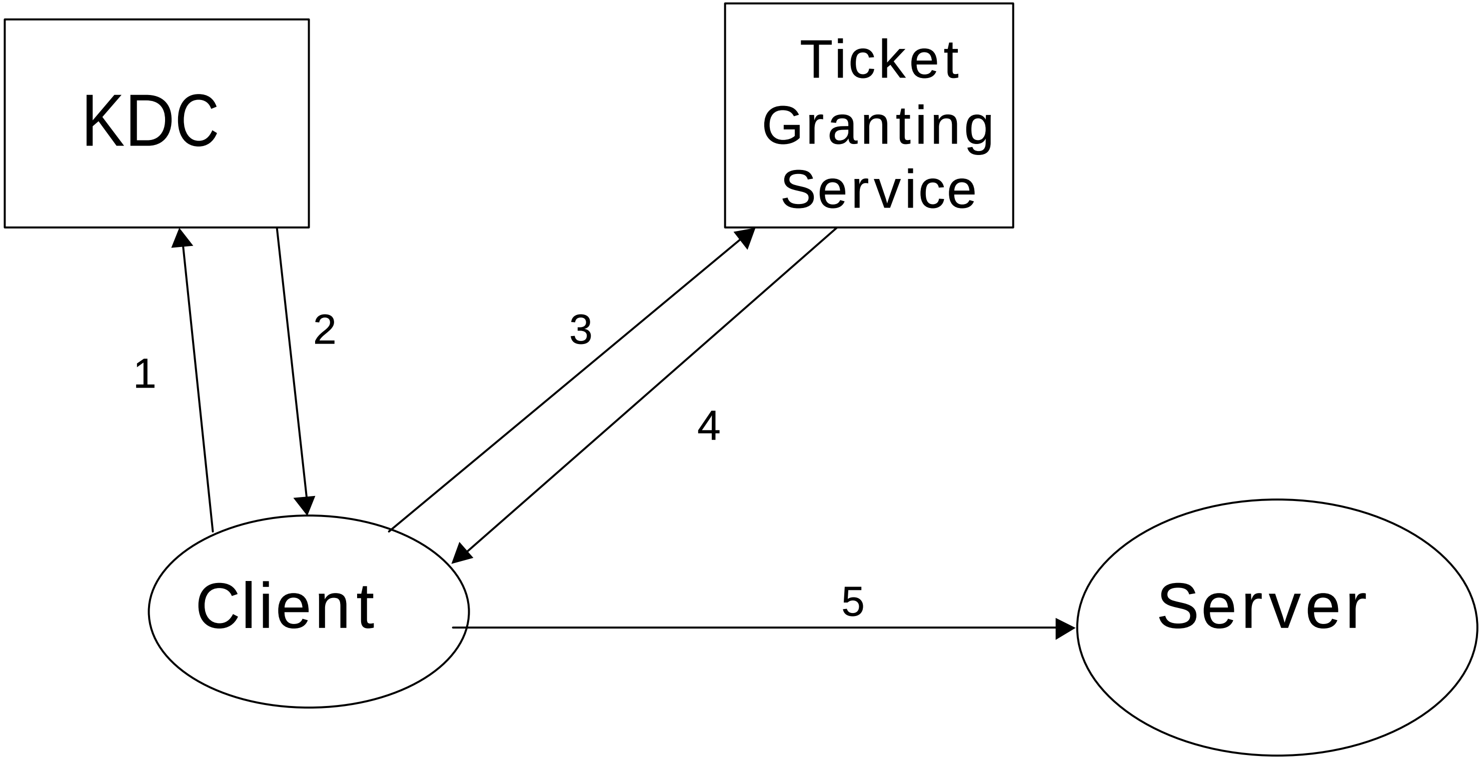
- Assumptions:
  - Public shared terminal (workstations)
  - Trusted terminal under user control for the entire session but untrusted network
  - Session start when user log-in and terminates when he logs out

Kerberos

General idea:

- user authenticates explicitely only once at the begin of the session (few hours) → single sign-on
- To access services (e.g., printer) users have to present a ticket
- One ticket for each server/type of service
- Each user has her own tickets
- A ticket can be reused for subsequent requests of the same service by the same user
- Transparent re-issuing of tickets

Kerberos



1-2: Authentication

3-4: Authorisation credentials

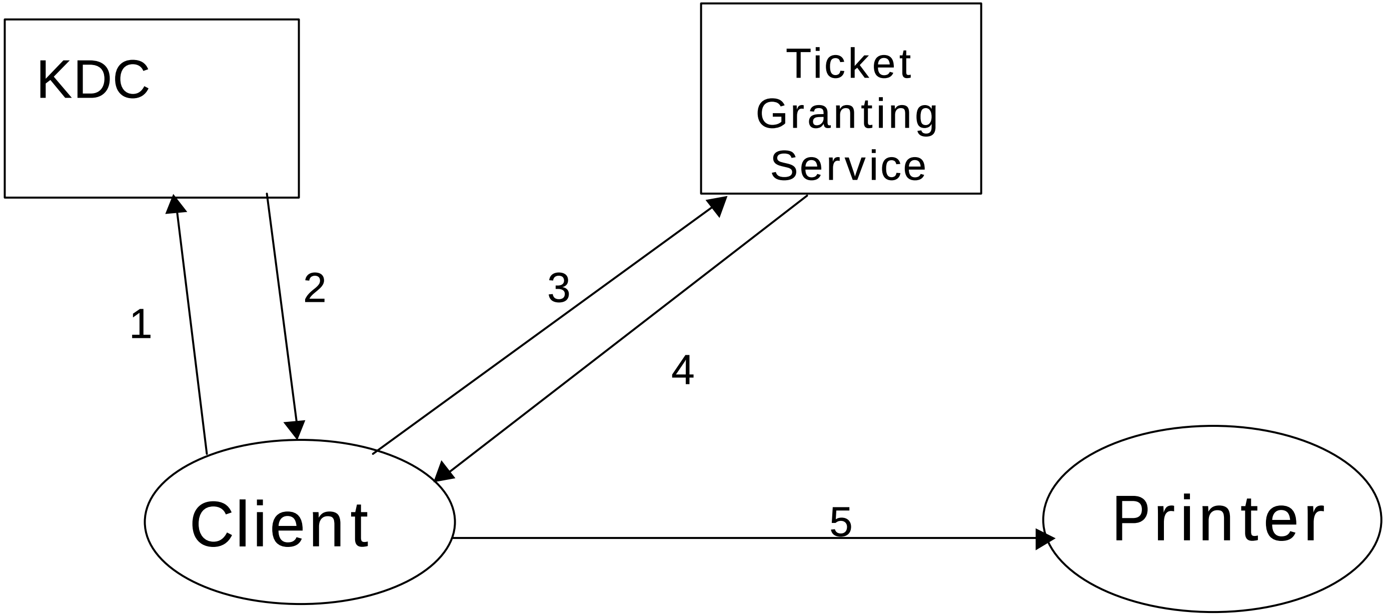
5: A&A Access

Kerberos

Among services a special one is the Ticket Granting Service (TGS) which goal is to issue tickets for other services to users

A special ticket is the Ticket-Granting Ticket (issued directly by KDC server). The TGT is the ticket users need to request services to the TGS

Kerberos



1. Request for Ticket-Granting Ticket

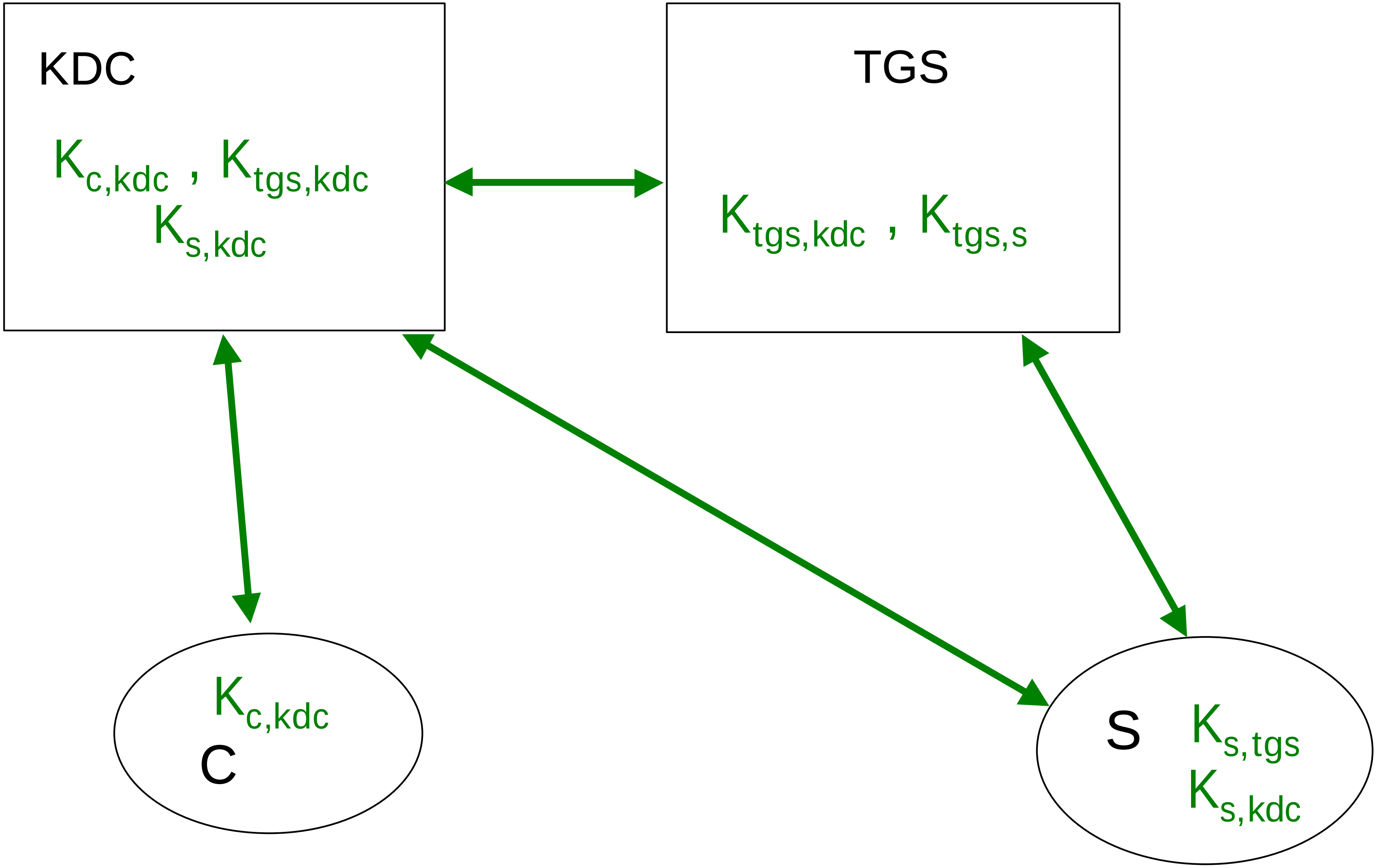
2. Ticket-Granting Ticket *(Authenticator)*

3. Request for Printer Ticket

4. Printer Ticket

5. Request for Printer

Kerberos v 4



Pre-set secure channels

Kerberos v 4

- C → KDC: C, tgs
- KDC → C: $[K_{c,tgs}, \mathrm{info}, [T_{c,tgs}]K_{tgs,kdc}]K_{c,kdc}$
- C → TGS: $[\mathit{Auth}_{c,s}]K_{c,tgs}, [T_{c,tgs}]K_{tgs,kdc}$
- TGS → C: $[K_{c,s}, [T_{c,s}]K_{s,tgs}]K_{c,tgs}$
- C → S: $[\mathit{Auth}_{c,s}]K_{c,s}, [T_{c,s}]K_{s,tgs}$

$T_{c,s} = s,[s,c,\text{network addr},\text{validity},K_{c,s}]K_{s,tgs}$

(c's ticket to use s)

$\mathit{Auth}_{c,s} = [c,\text{network addr},\text{timestamp}]K_{c,s}$

(authenticator from c to s)

$K_x = x$'s secret key shared with Kerberos

$K_{x,y} = x$'s session key for x and y $[m]K_{x,y}$= m encrypted with $K_{x,y}$

Kerberos

Kerberos uses to types of credentials:

ticket ($T_{x,y}$) and authenticathor ($Auth_{c,s}$)

Tickets are the credentials for accessing services. Each ticket is referred to a specific service and it has an expiration date. Within its validity, a user can re-use a ticket as many times as he wishes in order to authenticate himself to the related service.

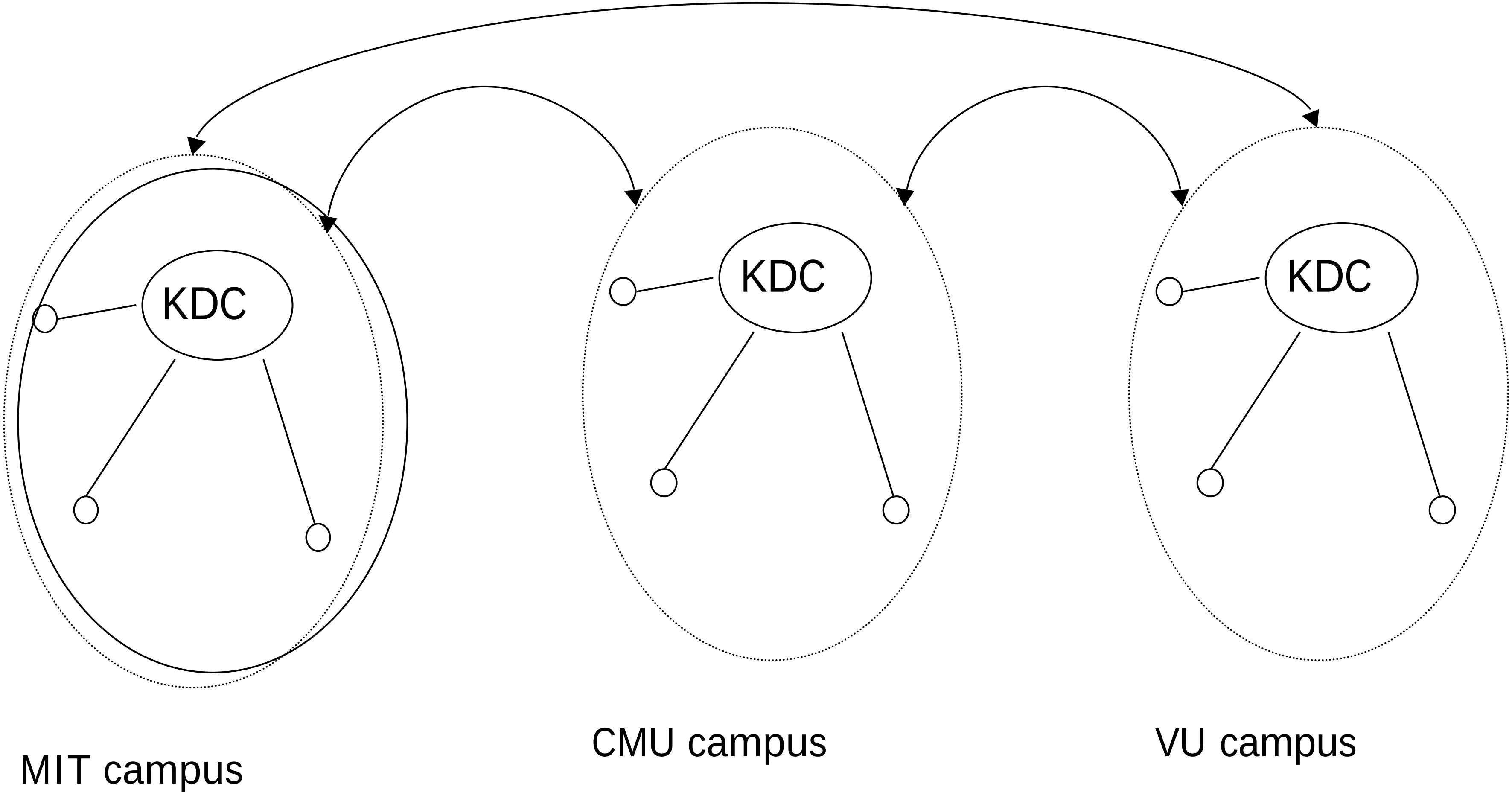
Kerberos

Authenticators are credentials used to access to a single instance of the service. They are valid only once.

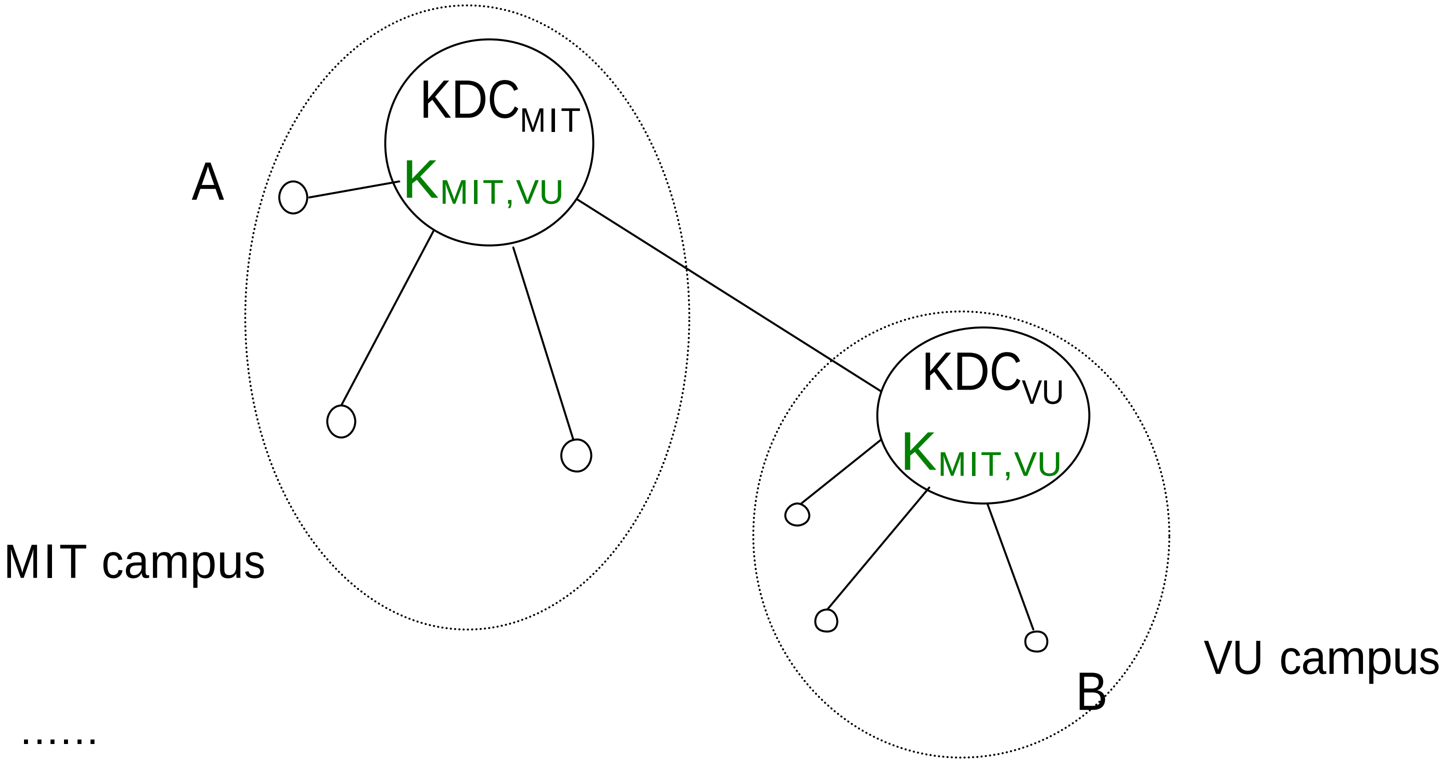
Authenticators are generated directly by the user. It is possible to have many different authenticators within a lifespan of a single ticket.

Multiple domains: realms

Realms



Multiple domains: realms



......

$A \rightarrow KDC_{MIT}$: A, KDC_{VU}

$KDC_{MIT} \rightarrow A$: credential to VU

$A \rightarrow KDC_{VU}$: Ticket Request A@MIT, B@VU

$KDC_{VU} \rightarrow A$: credential to B

$A \rightarrow B$: request

Problems

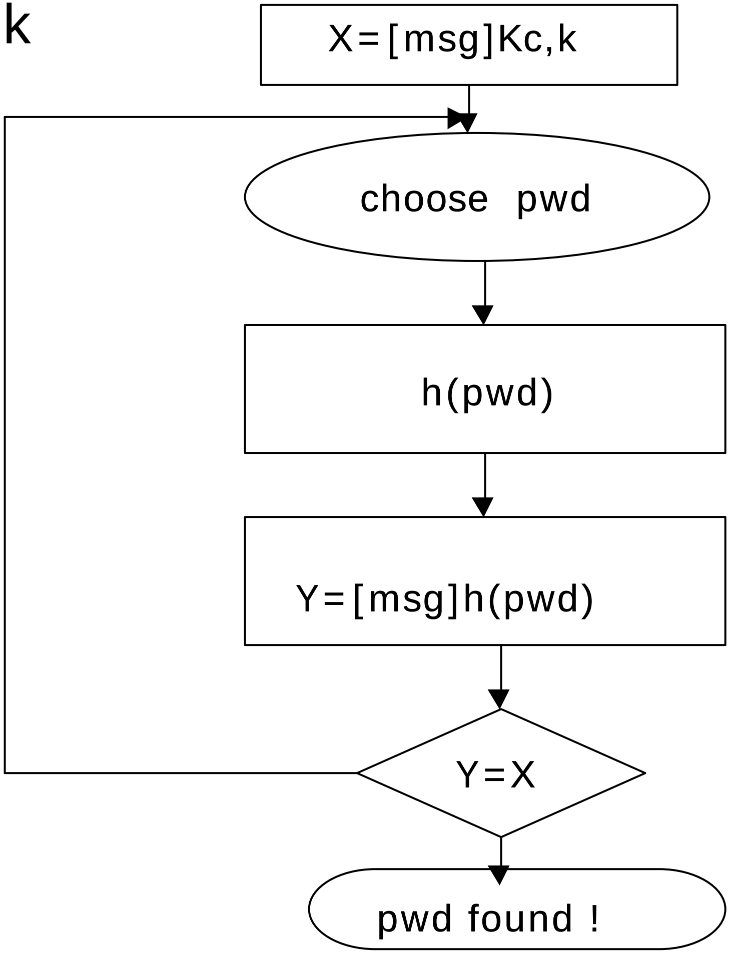
- Replay attacks possible because

  lifetime window of the authenticator → *storage*
- Problem with multiple domain and cross-realm tickets
- Dependence on a *trusted* Time Service
- No defense against malicious code
- Dictionary attack: password as initial shared secret between user and kerberos

$$K_{c,kdc}=h(PWD_{utente})$$

Problem: dictionary attack

$X=[msg]K_{c,k}$



Dictionary attack

- C → KDC: c, tgs
- KDC → C: $[K_{c,tgs}, info, [T_{c,tgs}]K_{tgs,kdc}]K_{c,kdc}$

Info: $Name_{TGS}$, Timestamp, Lifetime

All useful information with enough entropy to allow a dictionary attack

Partial solution with v5:

$$K_{c,kdc} = h(PWD_{utente}, salt_{WS}, timestamp)$$

Kerberos v 5

- Delegation of rights
  - Ticket with different network address
  - Forwardable and proxiable tickets
- Ticket lifetime
  - Renewable and postdated tickets
  - Key versioning
- Wider choice of crypto algorithms
- Optimization
  - No double encryption
- Prevention of dictionary attack

- C → KDC: C, tgs
- KDC → C: $[K_{c,tgs}, [T_{c,tgs}]K_{tgs,kdc}]K_{c,kdc}$
- C → TGS: $[\mathit{Auth}_{c,s}]K_{c,tgs}, [T_{c,tgs}]K_{tgs,kdc}$
- TGS → C: $[K_{c,s}\ , [T_{c,s}]K_{s,tgs}]K_{c,tgs}$
- C → S: $[\mathit{Auth}_{c,s}]K_{c,s}, [T_{c,s}]K_{s,tgs}$

v4

- C → KDC: C, tgs
- KDC → C: $[K_{c,tgs}]K_{c,kdc}, [T_{c,tgs}]K_{tgs,kdc}$
- C → TGS: $[\mathit{Auth}_{c,s}]K_{c,tgs}, [T_{c,tgs}]K_{tgs,kdc}$
- TGS → C: $[K_{c,s}]K_{c,tgs}\ , [T_{c,s}]K_{s,tgs}$
- C → S: $[\mathit{Auth}_{c,s}]K_{c,s}, [T_{c,s}]K_{s,tgs}$

v5

Kerberos v 5

- C → KDC: C, tgs
- KDC → C: $[K_{c,tgs}]K_{c,kdc},[T_{c,tgs}]K_{tgs,kdc}$
- C → TGS: $[\mathit{Auth}_{c,s}]K_{c,tgs},[T_{c,tgs}]K_{tgs,kdc}$
- TGS → C: $[K_{c,s}]K_{c,tgs} ,[T_{c,s}]K_{s,tgs}$
- C → S: $[\mathit{Auth}_{c,s}]K_{c,s},[T_{c,s}]K_{s,tgs}$

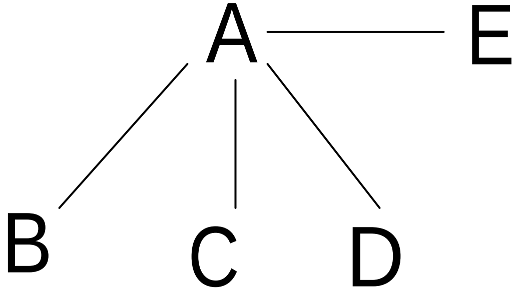
$T_{c,s} = s,[c,\text{network addr},\text{validity},K_{c,s}]K_{s,tgs}$

(c’s ticket to use s)

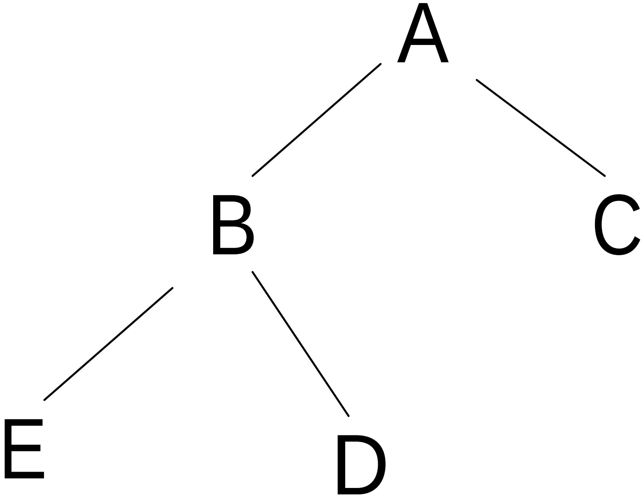
$\mathit{Auth}_{c,s}=[c,\text{timestamp},\text{key}]K_{c,s}$

(authenticator from c to s)

Hierarchy of Realms

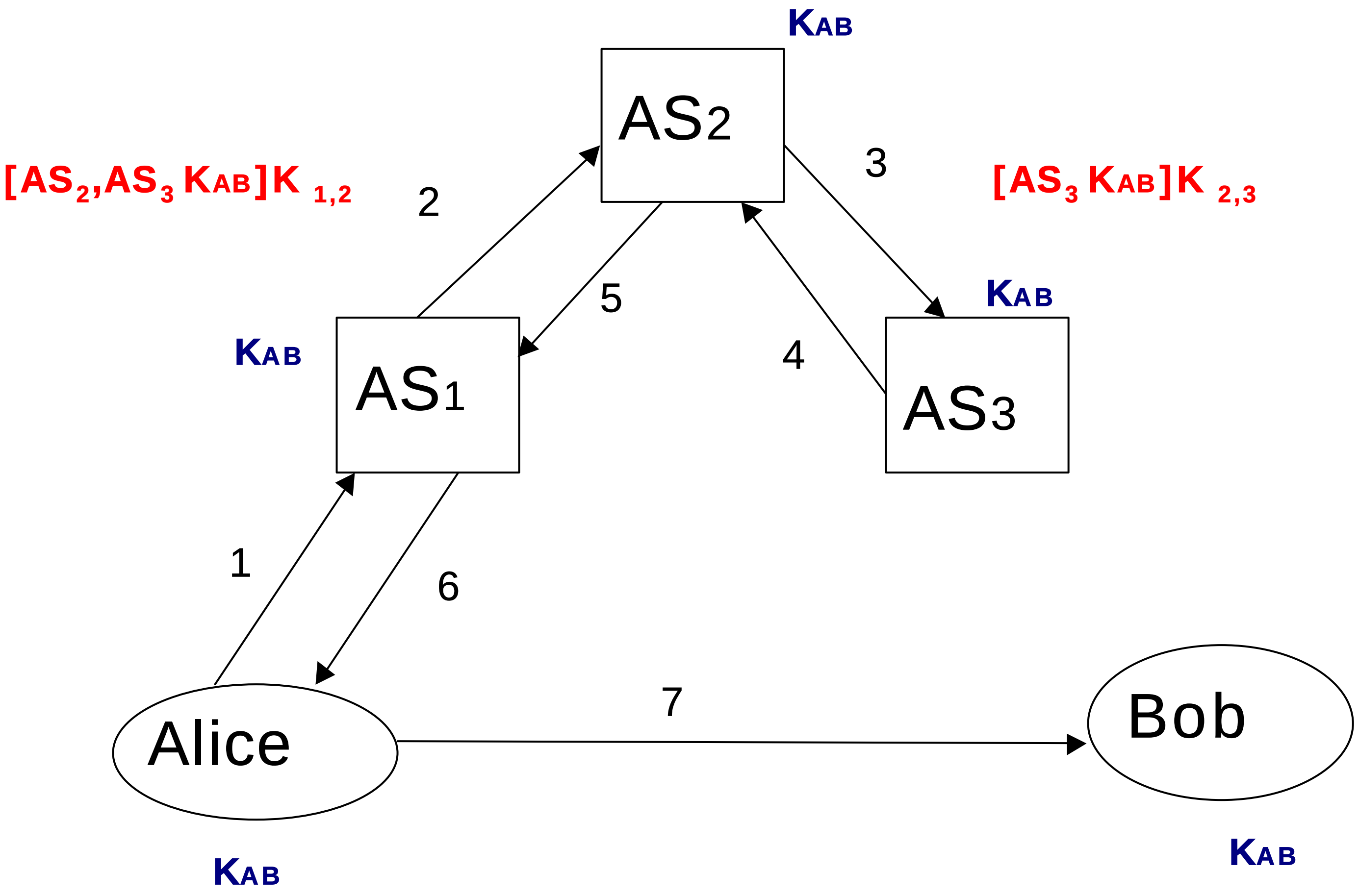


Transited field



Transitivity

Hierarchy of Realms



KAB

AS2

$[AS_2,AS_3\ KAB]K_{1,2}$

2

3

$[AS_3\ KAB]K_{2,3}$

KAB

5

KAB

4

AS1

AS3

1

6

7

Alice

Bob

KAB

KAB