

Spring 2010 – CS419

Computer Security

Vinod Ganapathy
Lecture 8

Kerberos

- Goal: Authentication
- Isolated machine, single user:
 - Physical security
- Isolated machine, multiple users:
 - Passwords
- Networked machine and services
 - Kerberos

Kerberos: Components

- User: human being requesting service
- Client: program that runs on behalf of user
- Server: provides service.
 - E.g., hosting files, printing

Kerberos authentication

- Protocol based upon Needham Schroeder with Denning Sacco modification
- Uses private key encryption.
- **Ticket**
 - Issuer vouches for identity of requester of service
- **Authenticator**
 - Identifies sender

Kerberos ingredients

- Kerberos server
- Ticket-granting server (TGS)
- Server that requires authentication
 - E.g., file server, printer service.

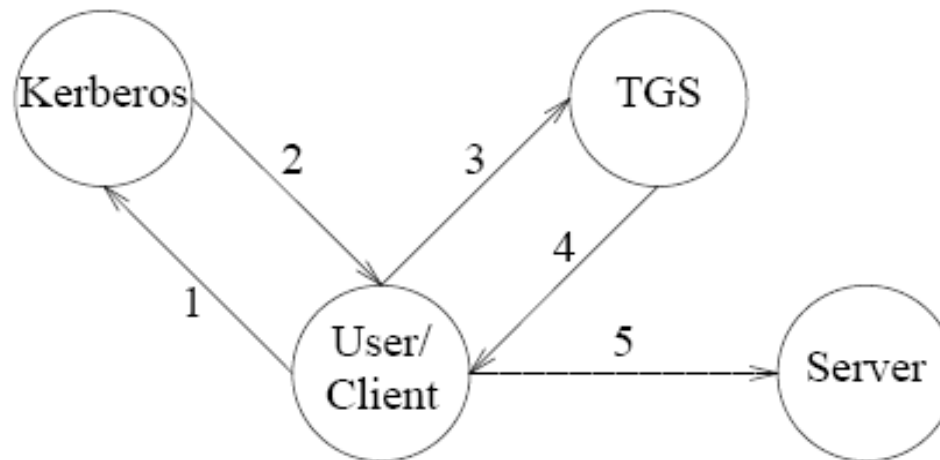
Design goals

- Protocol must be secure.
 - Replay, Man in the middle
- Reliability
- Transparent
 - E.g., not require user to enter password for each service requested
- Scalable

Idea

- User u authenticates to Kerberos server
 - Obtains ticket $T_{u,TGS}$ for ticket granting service (TGS)
- User u wants to use service s :
 - User sends authenticator A_u , ticket $T_{u,TGS}$ to TGS asking for ticket for service
 - TGS sends ticket $T_{u,s}$ to user
 - User sends A_u , $T_{u,s}$ to server as request to use s

Idea



1. Request for TGS ticket
2. Ticket for TGS
3. Request for Server ticket
4. Ticket for Server
5. Request for service

Figure 9. Kerberos Authentication Protocols.

Ticket

- Credential saying issuer has identified ticket requester
- Example ticket issued to user u for service s

$$T_{u,s} = s \parallel \{ u \parallel u\text{'s address} \parallel \text{valid time} \parallel k_{u,s} \} k_s$$

where:

- $k_{u,s}$ is session key for user and service
- Valid time is interval for which ticket valid
- u 's address may be IP address or something else
 - Note: more fields, but not relevant here

Authenticator

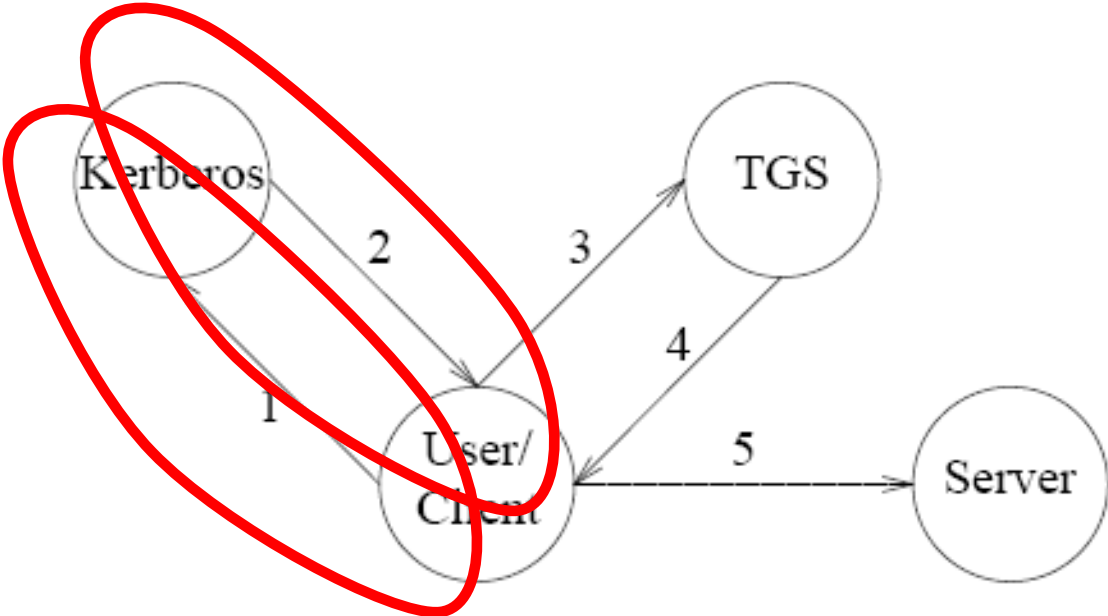
- Credential containing identity of sender of ticket
 - Used to confirm sender is entity to which ticket was issued
- Example: authenticator user u generates for service s

$$A_{u,s} = \{ u \parallel \text{generation time} \} k_{u,s}$$

where:

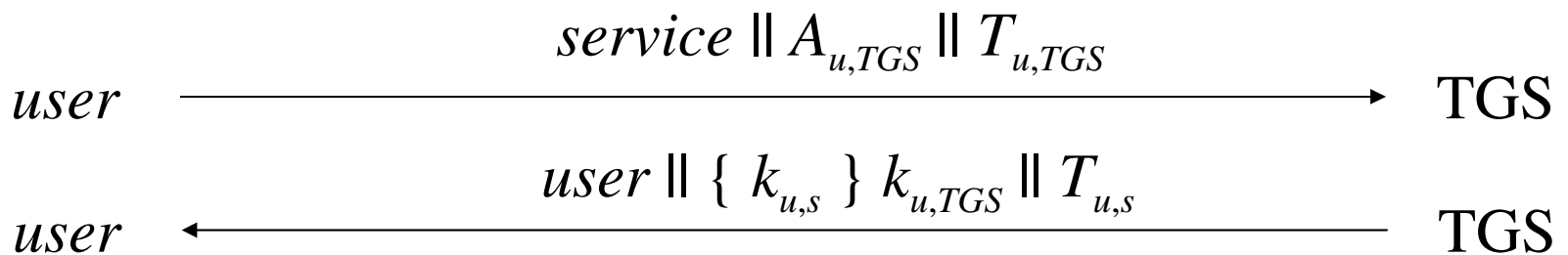
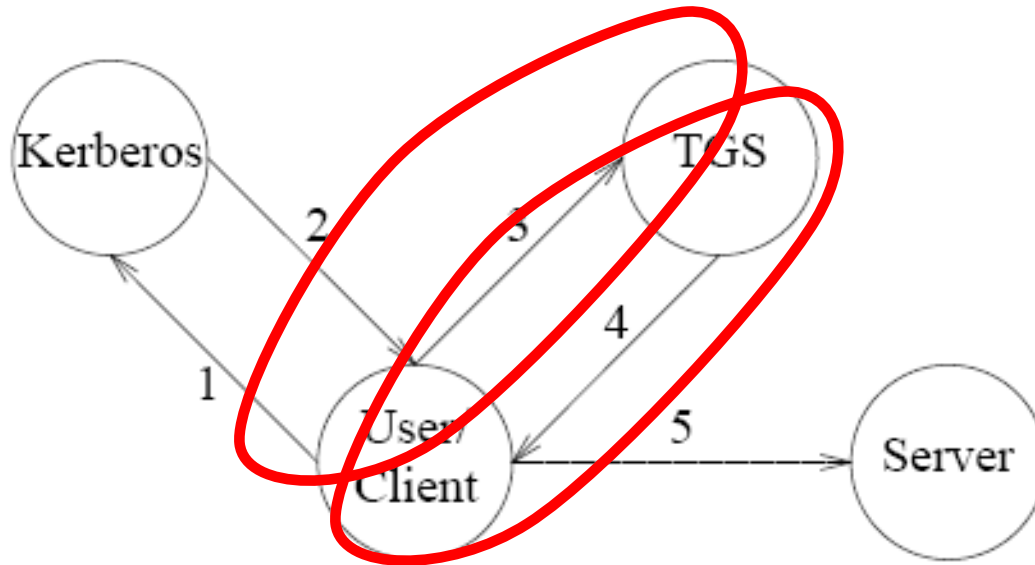
- Generation time is when authenticator generated
 - Note: more fields, not relevant here

Protocol: Ticket for the TGS

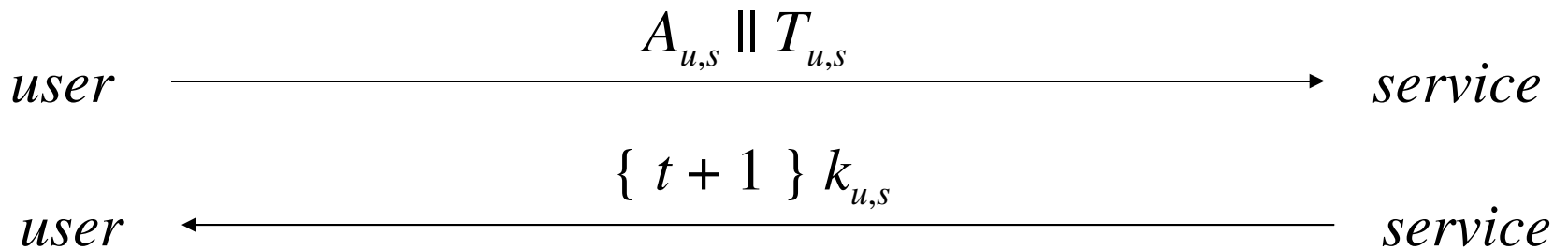
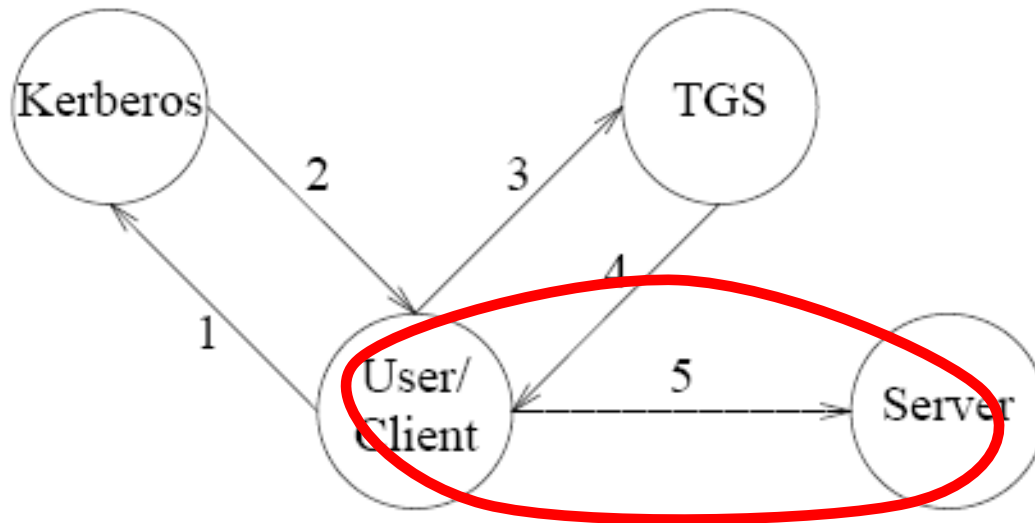


$user \xrightarrow{user \parallel TGS} Kerberos$
 $Kerberos \xleftarrow{\{ k_{u,TGS} \} k_u \parallel T_{u,TGS}} user$

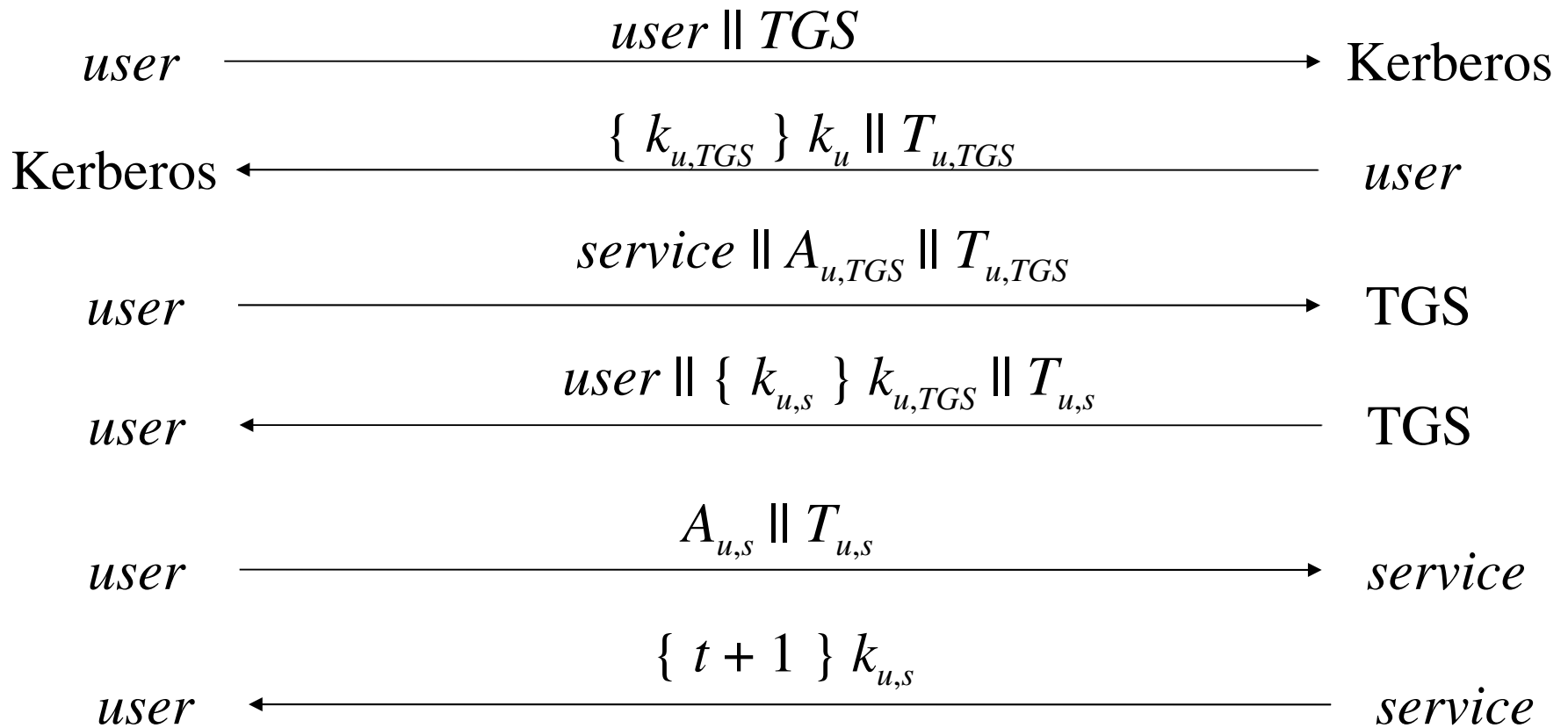
Protocol: Ticket for the service



Protocol: authenticating to service



Full Protocol



Analysis

- First two steps get user ticket to use TGS
 - User u can obtain session key only if u knows key shared with Kerberos
- Next four steps show how u gets and uses ticket for service s
 - Service s validates request by checking sender (using $A_{u,s}$) is same as entity ticket issued to
 - Step 6 optional; used when u requests confirmation

Problems

- Relies on synchronized clocks
 - If not synchronized and old tickets, authenticators not cached, replay is possible
- Tickets have some fixed fields
 - Dictionary attacks possible
 - Kerberos 4 session keys weak (had much less than 56 bits of randomness); researchers at Purdue found them from tickets in minutes

Summary and review

- Needham Schroeder
 - Goal?
 - Trusted third party?
 - Public/private key encryption?
- Problem with the original NS protocol?
- Solution?
 - Denning and Sacco
- Problem with Denning and Sacco's scheme?

Summary and review

- Public-key key exchange
 - Goal?
 - Protocol?
 - Trusted third party?
 - Man-in-the-middle attack?
- Diffie Hellman key exchange?
 - Trusted third party?
 - Public or private key encryption?