Aalto University

Network Security: Kerberos

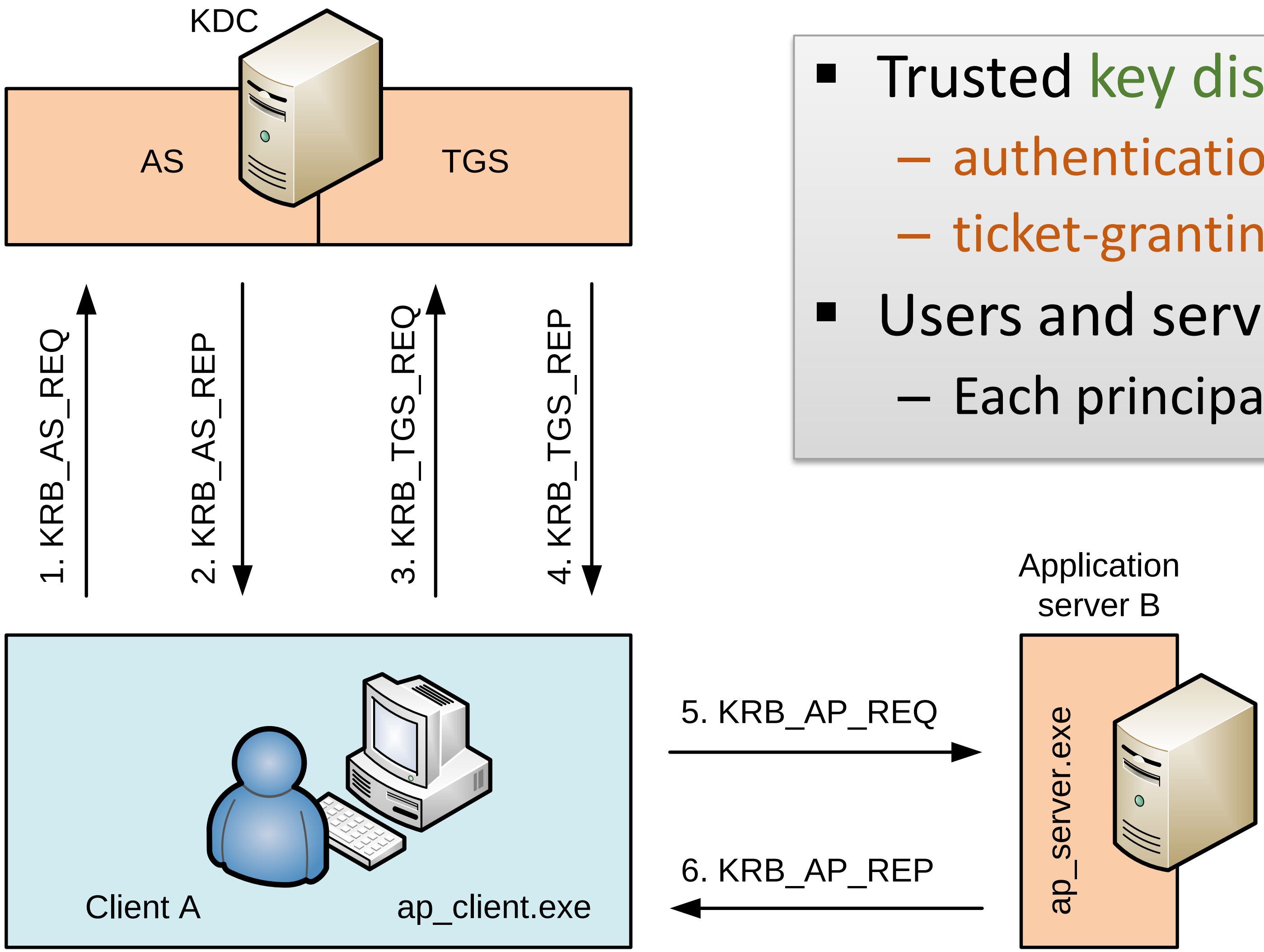
Tuomas Aura

CS-E4300 Network security
Aalto University

Kerberos

- Shared-key protocol for user login authentication
  – User passwords are the shared keys
  – Solves security and scalability problems in password-based authentication in large domains
  – Based on the Needham-Schroeder secret-key protocol
- Kerberos v4 1988- at MIT
- Kerberos v5 1993- [RFC 4120]
  – Updated protocol and algorithms
  – ASN.1 BER encoding of messages
  – Implemented in Windows 2000 and later
  – Used in intranets: university Unix systems, corporate Windows domains
  – Many extensions specified later

Kerberos architecture



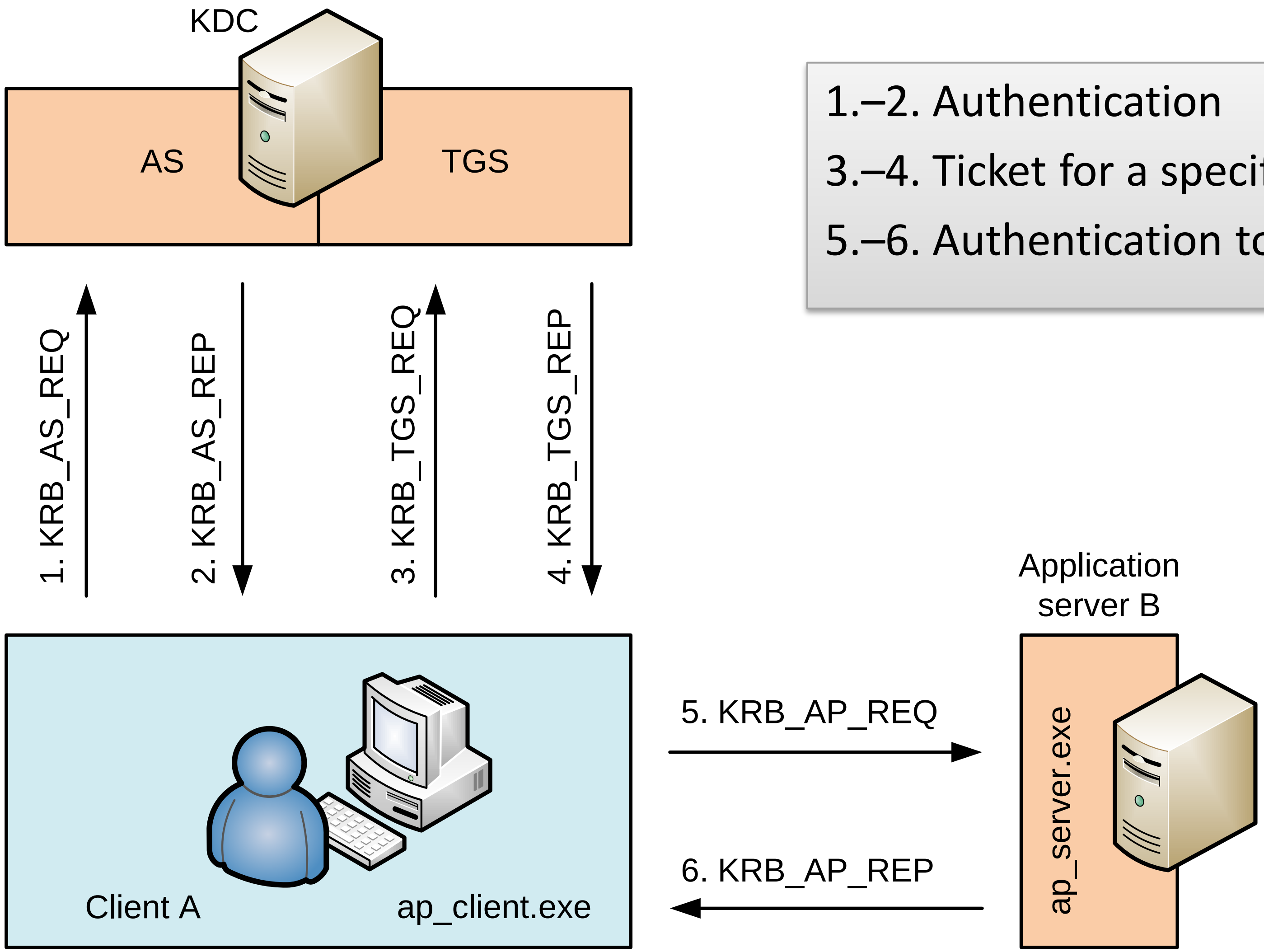
- Trusted key distribution center (KDC):
  - authentication server (AS)
  - ticket-granting server (TGS)
- Users and services are principals
  - Each principal shares a password with AS

Kerberos terminology

- Client-server computing model
  - Authentication for remote login sessions, e.g., remote shell or RPC
  - Users and services are principals
- Key distribution center (KDC)
  - Two components: authentication server (AS) and ticket-granting server (TGS)
  - Trusted by all principals to help in the key distribution
- KDC shares a master key with each principal
  - Long-term secret that is used only for initial key exchange
  - Usually derived by hashing a password [RFC3961]: password for each user and each service
- When user logs in, the workstation uses the password to obtain a ticket-granting-ticket (TGT) from AS
- When client needs to access remote services, it uses TGT to request from TGS a separate service ticket for each server

  (Note how the two-step process could be generalized to more steps)

Kerberos architecture

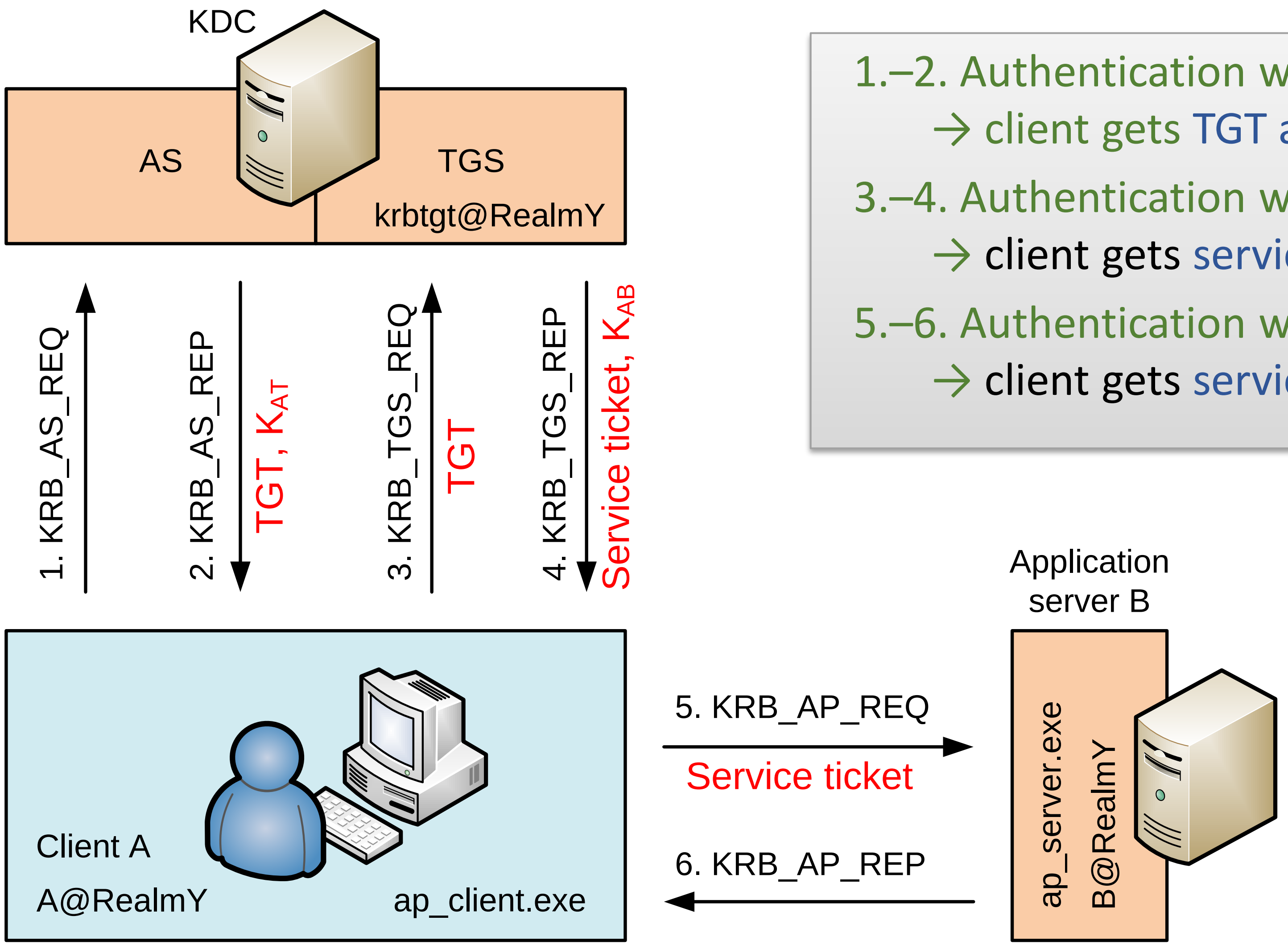


1.–2. Authentication

3.–4. Ticket for a specific service

5.–6. Authentication to the service

Kerberos architecture (details)

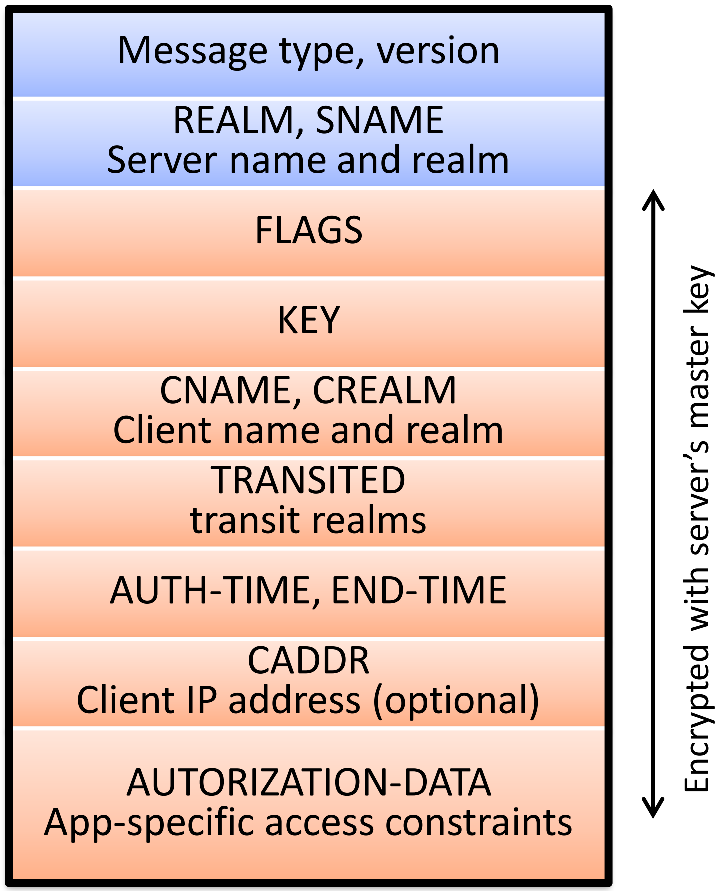


1.–2. Authentication with password
→ client gets TGT and K_{AT}

3.–4. Authentication with TGT and K_{AT}
→ client gets service ticket and K_{AB}

5.–6. Authentication with service ticket and K_{AB}
→ client gets service access

Kerberos ticket



- Same format for both TGT and service ticket
- Credentials = ticket + key
- ASN.1 BER encoding in Kerberos v5
- Encryption also protects integrity (actually encryption and a MAC)

- Flags:
  - FORWARDABLE, FORWARDED, PROXIABLE, PROXY, MAY-POST-DATE, POSTDATED, INVALID, RENEWABLE, INITIAL, PRE-AUTHENT, HW-AUTHENT
  - INITIAL flag indicates TGT

Kerberos protocol (more details)

Initial login of user A:

1. A → AS: Preauthentication, A, TGS, N_{A1}, $Addr_A$
2. AS → A: A, TGT, E_{KA} ($K_{A\text{-}TGS}$, N_{A1}, TGS, $Addr_A$)

Ticket request:

3. A → TGS: TGT, $Authenticator_{A\text{-}TGS}$, B, N_{A2}, $Addr_A$
4. TGS → A: A, Ticket, $E_{KA\text{-}TGS}$ (K_{AB}, N_{A2}, B, $Addr_A$)

Authentication to server B:

5. A → B: Ticket, $Authenticator_{AB}$
6. B → A: AP_REP

Notes:

1234) ASN.1 encoding adds type tags to all messages

Encryption mode also protects message integrity

A, B = principal names
T_x = timestamp
$Addr_A$ = A's IP addresses
K_A , K_{TGS}, K_B = master keys of A, TGS and B
$K_{A\text{-}TGS}$ = shared key for A and TGS
K_{AB} = shared session key for A and B

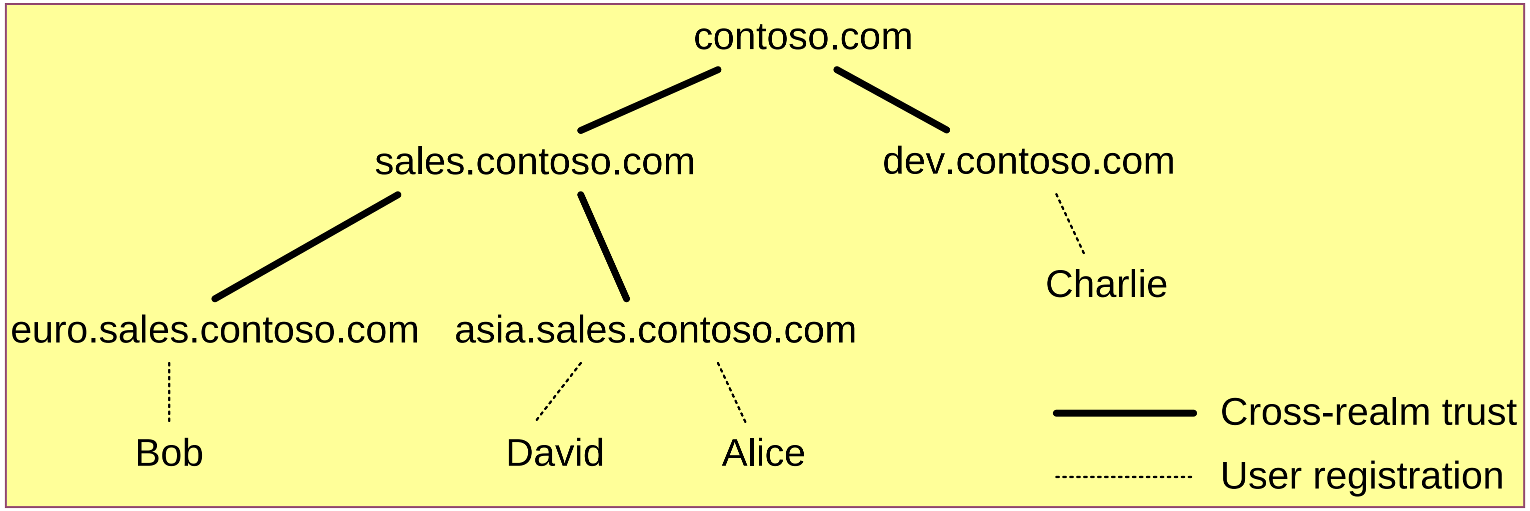
TGT = B, E_{KTGS} (INITIAL, $K_{A\text{-}TGS}$, A, T_{auth}, $T_{expiry1}$, $Addr_A$))
Ticket = B, E_{KB}(K_{AB}, A, T_{auth}, $T_{expiry2}$, $Addr_A$))
Preauthentication = E_{KA} (^{1} T_A)
$Authenticator_{A\text{-}TGS}$ = $E_{KA\text{-}TGS}$ (^{2} T_A)
$Authenticator_{AB}$ = E_{KAB} (^{3} T_A)
AP_REP = E_{KAB}(^{4} T_A)

Kerberos realms

- Users and services registered to one KDC form a realm
  - name@realm: A@X, aura@org.aalto.fi
- Cross-realm trust:
  - Two KDCs X and Y share a key: krbtgt@Y is registered in KDC X and krbtgt@X in KDC Y
  - KDCs trust each other to be honest and competent to name users in their own realms
- Cross-realm authentication:
  - Client A@X requests from TGS at realm X a ticket for TGS at realm Y
  - The ticket is encrypted for krbtgt@Y, i.e., TGS at realm Y
  - Client A@X requests from TGS at realm Y a ticket for server B@Y

- Access control can be implemented at several steps:
  - Local policy at each KDC about when to honor tickets from other realms
  - Local policy at B@Y about whether to allow access to users from other realms
  - ACLs at B@Y determine whether the authenticated users is allowed to access the particular resources
- Possible to transit multiple realms
  - TRANSITED field in the ticket accumulates the intermediate realms
  - Local policy at each server about which transited realms are ok

Realm hierarchy



- Large organization can have a realm hierarchy
  - Often the Windows domain hierarchy
  - Realms have hierarchical names, similar to internet domain names
  - Admins can add shortcut links between some or all KDCs
- Compare with X.509 certification hierarchy: what are the similarities and differences?

Password guessing attacks

- Kerberos v5 is vulnerable to password guessing:
  - Sniffed KRB_AS_REQ or KRB_AS_REP can be used to test candidate passwords → offline brute-force password guessing
  - In Kerberos v4, anyone could request a password-encrypted TGT from AS → easy to obtain material for password cracking
  - Preauthentication in Kerberos v5 prevents active attackers from obtaining material for password cracking → must sniff the TGT from the network

!

- Note: active vs. passive attacks
  - Are active attacks (spoofing, MitM) more difficult to implement than passive attacks (sniffing)? Often not!
  - Active attacks can often be initiated by the attacker while passive attacks require attacker to wait for something to be sent over the network

!

PKINIT

- Goal: take advantage of an existing PKI to bootstrap authentication in Kerberos
- Replaces the KRB_AS_REQ / KRB_AS_REP exchange with a public-key protocol
  - Public-key authentication and encryption to obtain TGT
  - Then continue with standard Kerberos → transparent to TGS and application servers
- No password; thus, not vulnerable to password guessing
- Uses DSS signatures and ephemeral DH
- Windows 2000 and later, now standardized [RFC 4556]
  - Other preauthentication methods have been added later

Using the session key

- Applications need to be modified, i.e., "Kerberized" to use Kerberos for authentication
- Applications use the session key K_{AB} in any way they want
  - KRB_AP_REQ and KRB_AP_REP may include further key material, subkeys, that are sent encrypted under K_{AB}
  - Authentication at the beginning of a session is of little value unless session data is protected with the session keys
- Kerberos provides special messages for integrity protection and encryption of session data:
  - KRB_SAFE: data, T_A, SN, $addr_A$, $addr_B$, $MAC_{KAB}(\ldots)$
  - KRB_PRIV: $E_{KAB}(data, T_A, SN, addr_A, addr_B)$
  - GSSAPI (called SSPI in Windows) provides access to these functions from applications

Delegation

- Server may need to perform tasks on the client's behalf, e.g., recursive RPC
- Delegation: client shares its TGT or service ticket and key
  - Another Kerberos message KRB_CRED for ending the encrypted credentials
- Ticket flags related to delegation:
  - FORWARDABLE flag in TGT: can request a new TGT with different IP addresses
  - PROXIABLE flag in TGT: can request service tickets with a different IP address
- Kerberos delegation is identity delegation
  - B can act as A and nobody can tell the difference → difficult to audit access
  - Other protocols delegate only access rights, so that the delegate can be identified
- Kerberos delegation is nevertheless better than sharing the user's password
  - Ticket has limited validity time
  - Ticket specifies allowed client IP addresses
  - Authorization-data field in ticket may contain app-specific restrictions

Related reading

- William Stallings. Network security essentials: applications and standards, 3rd ed. chapter 4.1; 4th ed. chapter 4.1–4.2 (Kerberos v5 only)
- William Stallings. Cryptography and Network Security, 4th ed.: chapters 14.1 (Kerberos v5)
- Dieter Gollmann. Computer Security, 2nd ed.: chapter 12.4; 3rd ed. chapter 15.4
- Kaufmann, Perlman, Speciner. Network security, 2nd ed.: chapter 14

Exercises

- How does Kerberos fix the flaw in Needham-Schroeder secret-key protocol?
- Find source code for a Kerberized client/server application (e.g., OpenSSH) and see how it accesses Kerberos services
- Why is Kerberos used on the intranets and TLS/SSL on the Internet? Could it be the other way?
- Learn about Encrypted Key Exchange (EKE) and similar password-based authentication protocols. Which problem do they solve that exists in Kerberos?