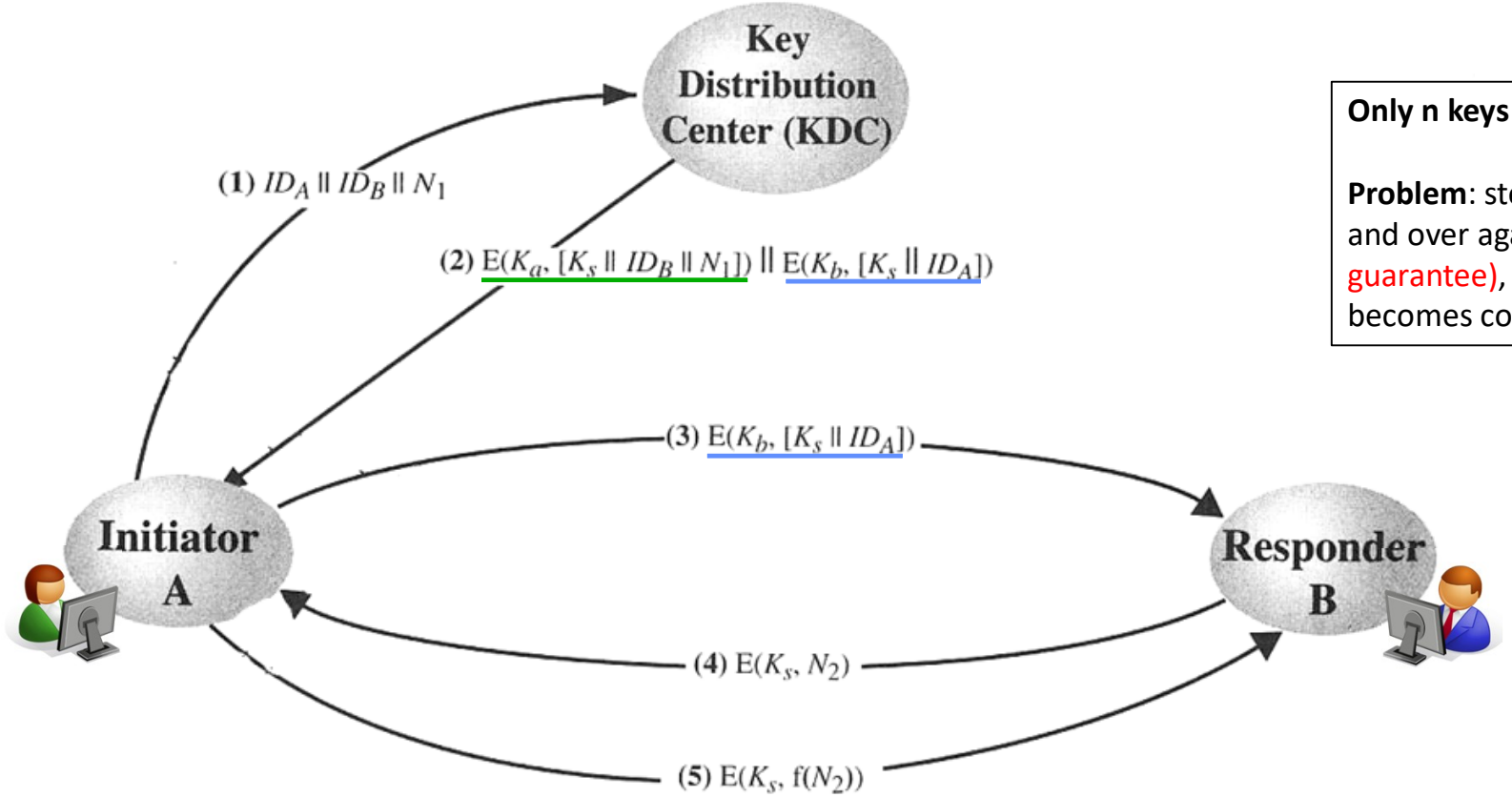
Remote authentication using encryption

Chapter 16.1 – 16.3

Key distribution with symmetric ciphers

- Public-key ciphers take time to compute
- Symmetric ciphers much faster, but requires pairwise shared keys:
  A ←→ B
  A ←→ C
  A ←→ D
  …
- $O(n^2)$ keys needed that must be distributed on forehand
- Use of a trusted third party (KDC) can solve this problem
  - KDC: key distribution center
  - Each entity only needs one key: to the KDC
- Protection against replays and MITM attacks needed:
  - Nonces can make sure communication is fresh
  - Timestamps can invalidate old “tickets” and avoid replays, see next slide (chapter 16.2)

Using a KDC with only symmetric keys



Only n keys needed!

Problem: step 3 can be replayed over and over again (no freshness guarantee), problematic if K_S becomes compromised.

CHALMERS | GÖTEBORGS UNIVERSITET

The Kerberos Authentication System

Chapter 16.3

https://web.mit.edu/kerberos

What is Kerberos?

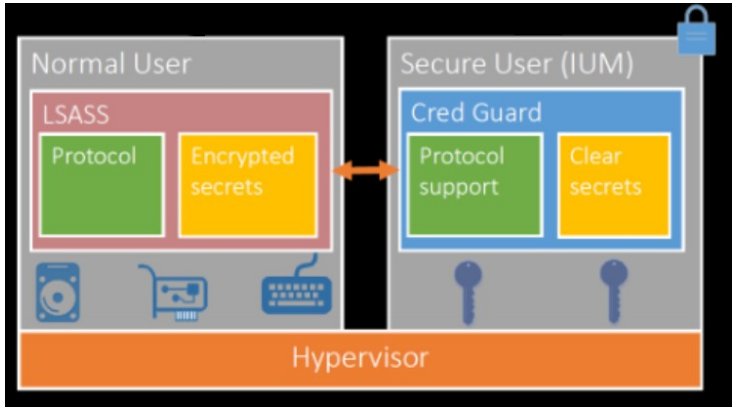
- The many-headed dog guarding the entrance of Hades in the Greek and Roman mythology
- And an AAA server guarding the network
  - Widely supported: Mac OS X and Linux/Unix systems support it
  - It's the default authentication method in Windows
- Originally developed at MIT, early '80s
  - The latest version (5) is described in RFC 1510 (1993)
  - Managed by the Kerberos Consortium (2007-): Apple, Microsoft, Google, Stanford, KTH, …
- Kerberos is a trusted third party used for authentication and authorization
- Authentication and authorization are two separate tasks
  - Although normally performed by the same server (hands out "tickets")
- Application servers only need to admit already authorized users
  - Only need to look at a "service ticket"
  - Authentication and authorization already done when they receive the ticket

Design Objectives

- Secure: An eavesdropper should not be able to impersonate a user
- Reliable: Distributed architecture, servers can back up each other
- Transparent: SSO – Single sign-on
- Scalable: Modular and distributed architecture
- It is suited for large environments:
  - No individual computers have to do authentication
  - Application servers only have to share a secret with the Kerberos server

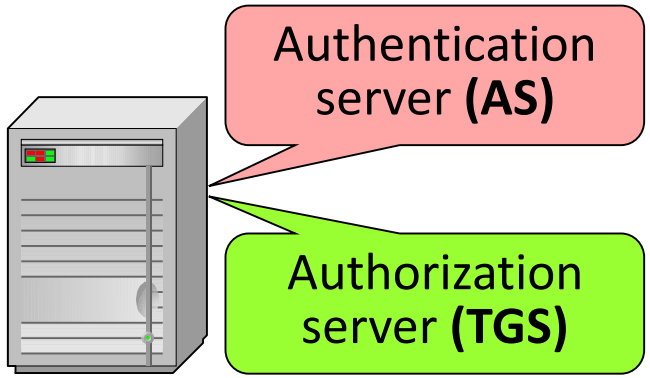
Windows authentication

- Kerberos is the preferred and default authentication system in Windows
  - Introduced with Windows 2000
  - Default in Active Directory (domain authentication)
  - Also supported by Unix/Linux/MacOS systems
- Replaced the old NTLM v2 protocol: NT LAN Manager authentication (Windows NT4, 1998)
  - Still supported
  - Challenge response authentication using HMAC-MD5
  - No support any recent cryptographic methods (AES and SHA-256)
- LSASS = Local security authority sub-system
  - Keeps credentials for single sign-on (Kerberos tickets, etc.)
  - Windows 10 LSASS process can run in its own container →



The Kerberos System

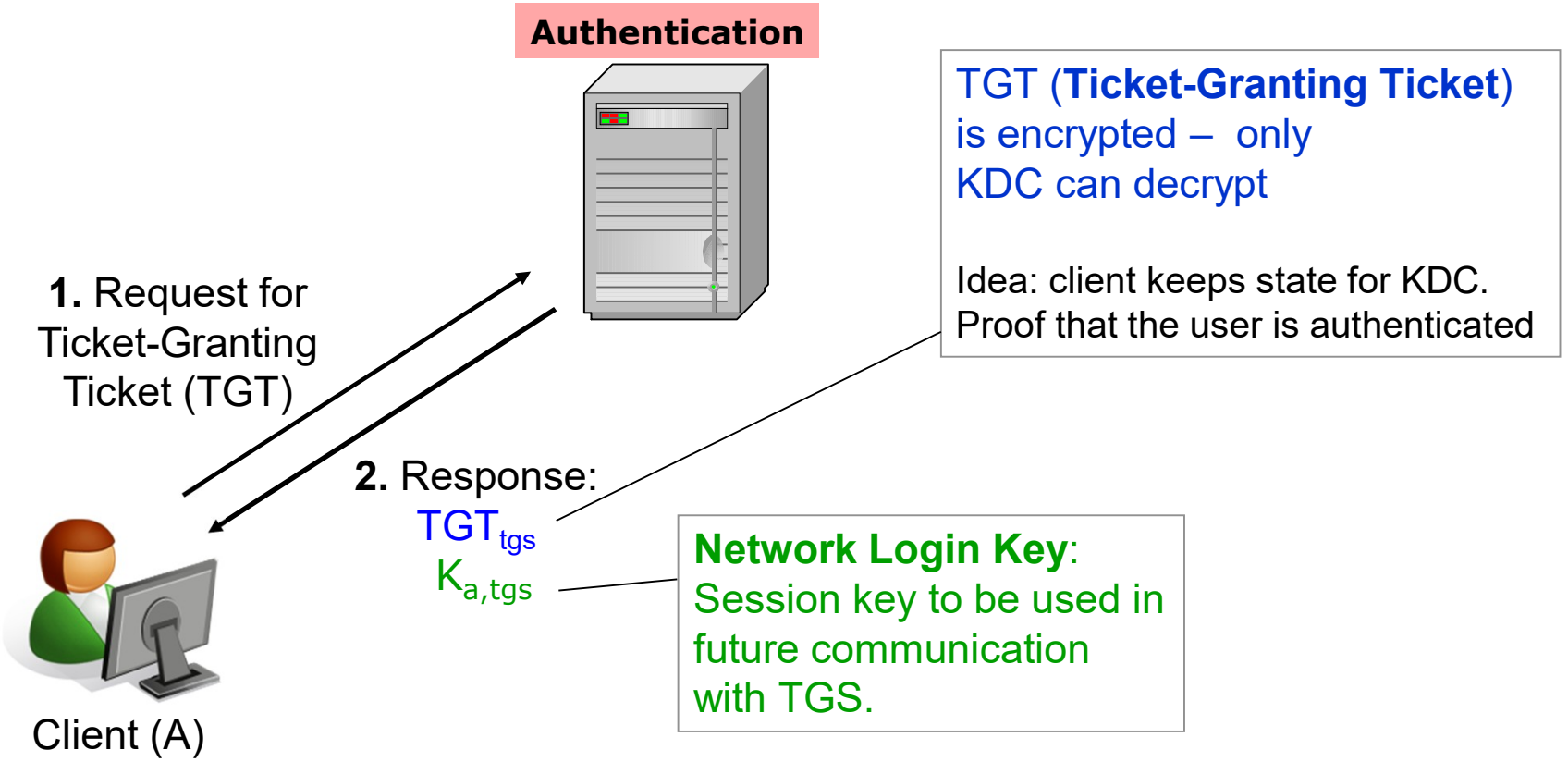
Kerberos Server:
Key Distribution Center (KDC)
Both an **AS** and **TGS**



Client (A)

Verifier (V)
(application server)

Authentication: Getting the TGT



Authentication messages

Authentication Service Exchange: To obtain Ticket-Granting Ticket

(1) A → AS: $ID_a \| ID_{tgs} \| TS_1$

(2) AS → A: $E_{K_a}[K_{a,tgs} \| ID_{tgs} \| TS_2 \| Lifetime_2 \| TGT_{tgs}]$

A cannot read but keeps state for AS

TGT_{tgs}: $E_{K_{tgs}}[K_{a,tgs} \| ID_a \| AD_a \| ID_{tgs} \| TS_2 \| Lifetime_2]$

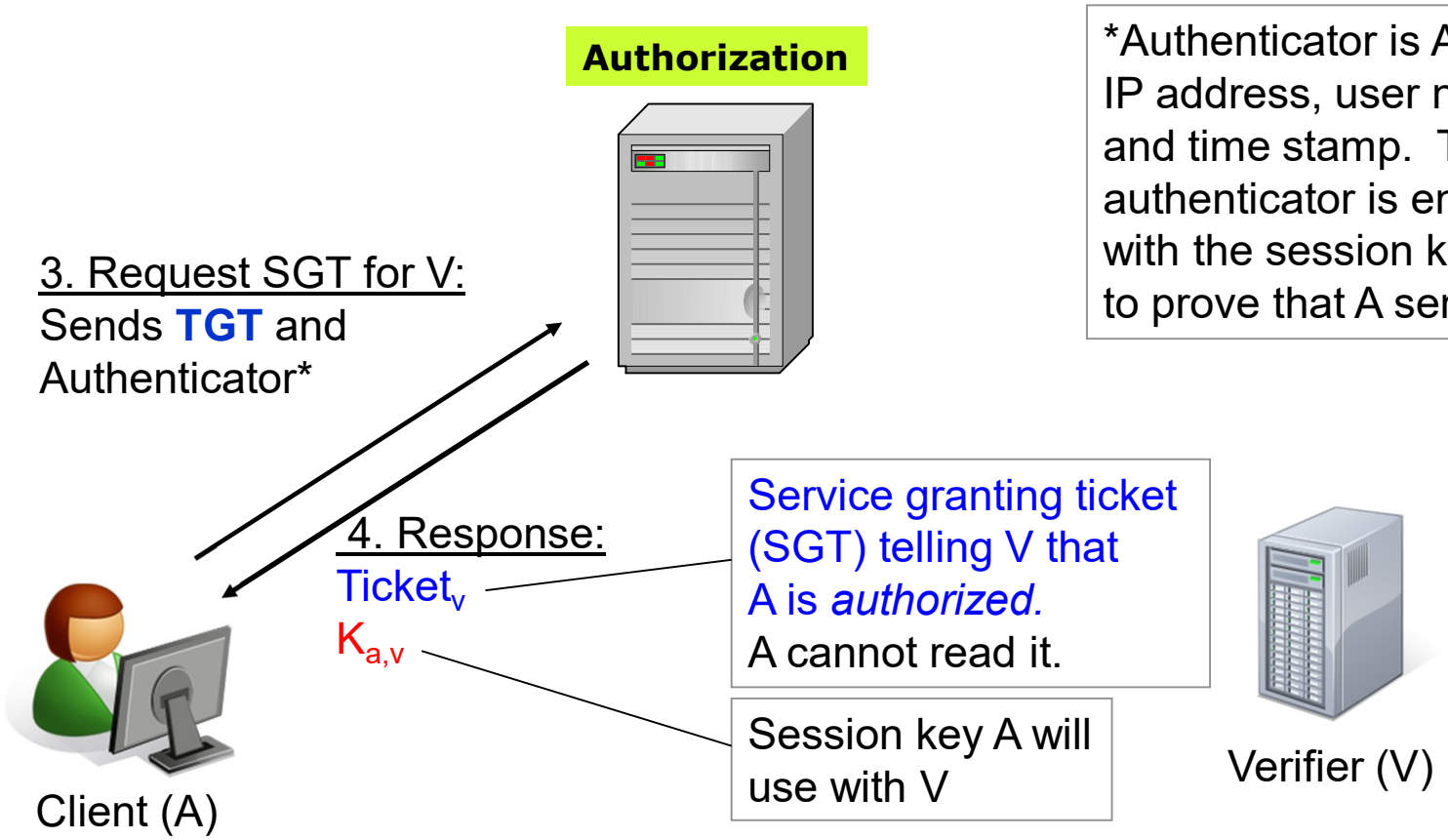
TS = Time Stamp
K_a = A's master key (next slide)
$K_{a,tgs}$ = Session key
AD_a = Address of A
K_{tgs} = Key known only by TGS
Lifetime = Lifetime of the session key

Master Keys and passwords

- The master key (K_a) is a hash of the user's password
  - The reply from AS is encrypted with K_a and contains the session key $K_{a,tgs}$
  - The master key is used as little as possible
  - It is possible to fake a request for a ticket for someone else…

    … but only the correct user can decrypt and use the ticket
- A Kerberos server may contact other servers to verify user names and passwords
  - Not only password authentication is supported
- The lifetime of the TGT must be limited
  - If it is stolen (after being decrypted), it should not be valid too long
  - If lifetime is too short, clients will repeatedly ask for new TGTs
  - Kerberos 4: lifetime max 21 hours – was too short for long running applications
- The TGT contains all state information the Kerberos server needs
  - Shows that the user is authenticated
  - The AS does not have to store information about all logged in (authenticated) users

Authorization: Getting the Service Granting Ticket



Getting the Service Granting Ticket

Ticket-Granting Service Exchange: To obtain Service-Granting Ticket

(3) A → TGS: $ID_v \| TGT_{tgs} \| Authenticator_a$

(4) TGS → A: $E_{K_{a,tgs}}[K_{a,v} \| ID_v \| TS_4 \| Ticket_v]$

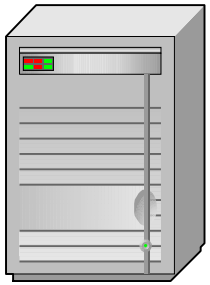
$Authenticator_a$: $E_{K_{a,tgs}}[ID_a \| AD_a \| TS_3]$

A cannot read

$Ticket_V$: $E_{K_V}[K_{a,v} \| ID_a \| AD_a \| ID_v \| TS_4 \| Lifetime_4]$

K_V = V's master key shared with the Kerberos server
ID_V dropped in version 5 since only V can decrypt message

Connecting to the verifier



5. Request for Connection: $Ticket_v$, $Authenticator_{a,v}$

6. Verifier authenticates itself:
Encrypted time stamp

7. Secure communication with $E_{Ka,v}$

Client (A)

Verifier (V)

Connecting to the verifier

Client/Server Authentication Exchange: To Obtain Service

(5) A → V: $Ticket_v$ || $Authenticator_{a,v}$

(6) V → A: $E_{K_{a,v}}[TS_5 + 1]$ (for mutual authentication)

$Authenticator_{a,v}$: $E_{K_{a,v}}[ID_a \| AD_a \| TS_5]$

Kerberos: The verifier (V)

- By looking at the Service Granting Ticket, V knows Kerberos has created it
  - It contains the session key to A, its IP address and identity
- The Authenticator proves that A also has the session key
  - It is time stamped to prevent replay attacks
- Ticket lifetime to services (V) must be limited
  - Governed by the domain security policy. Always < 10h (about a working day)
  - New tickets then needs to be generated with new session keys
- No public key/asymmetric encryption is used
  - Kerberos is fast
  - No certificates need to be distributed, although certificates can be used for user authentication
- Implements SSO – Single Sign-on
- Problem: Applications must be "Kerberized" (services need to support tickets)

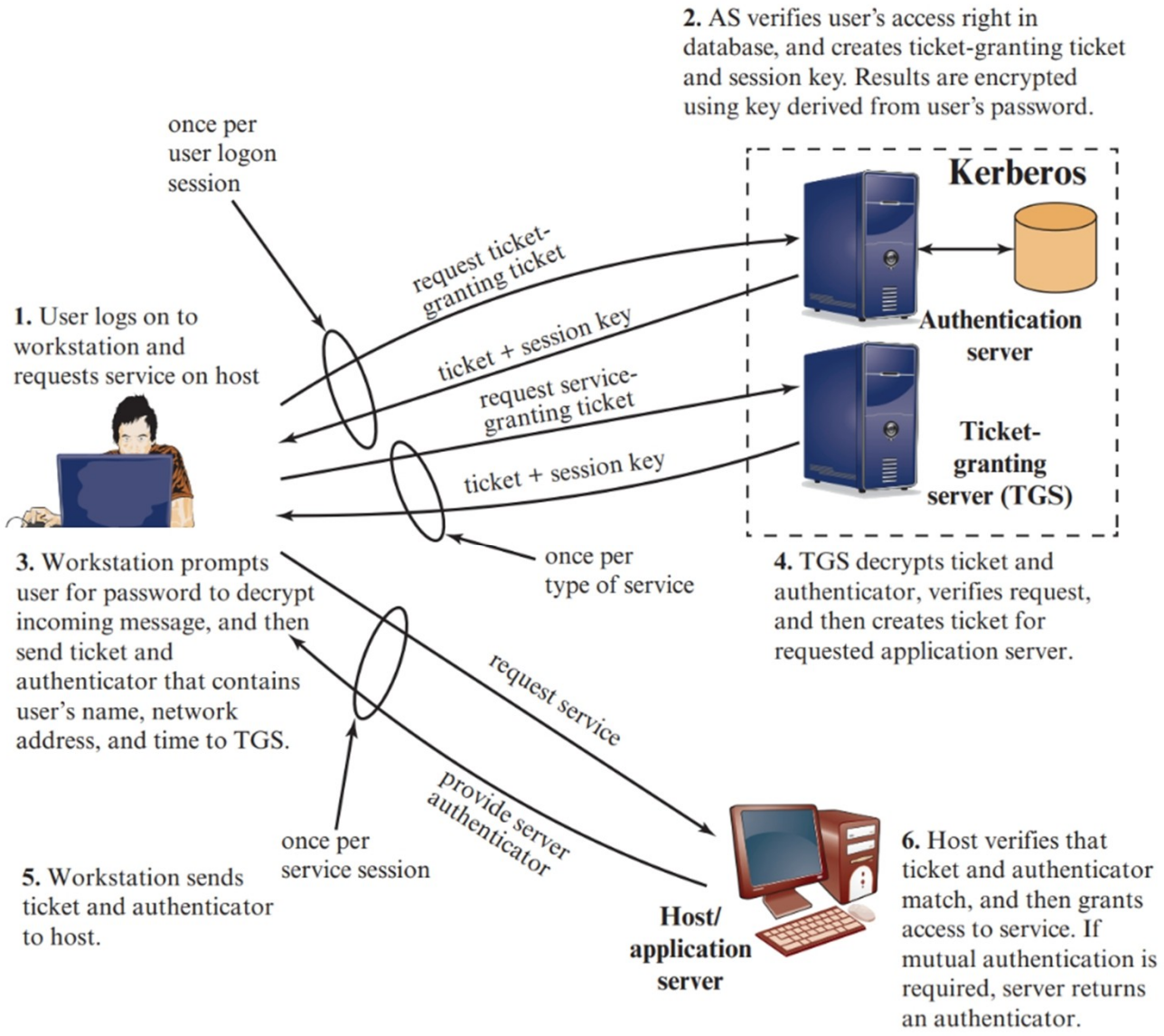


Figure 16.3 Overview of Kerberos

Kerberos Realms (Domains)

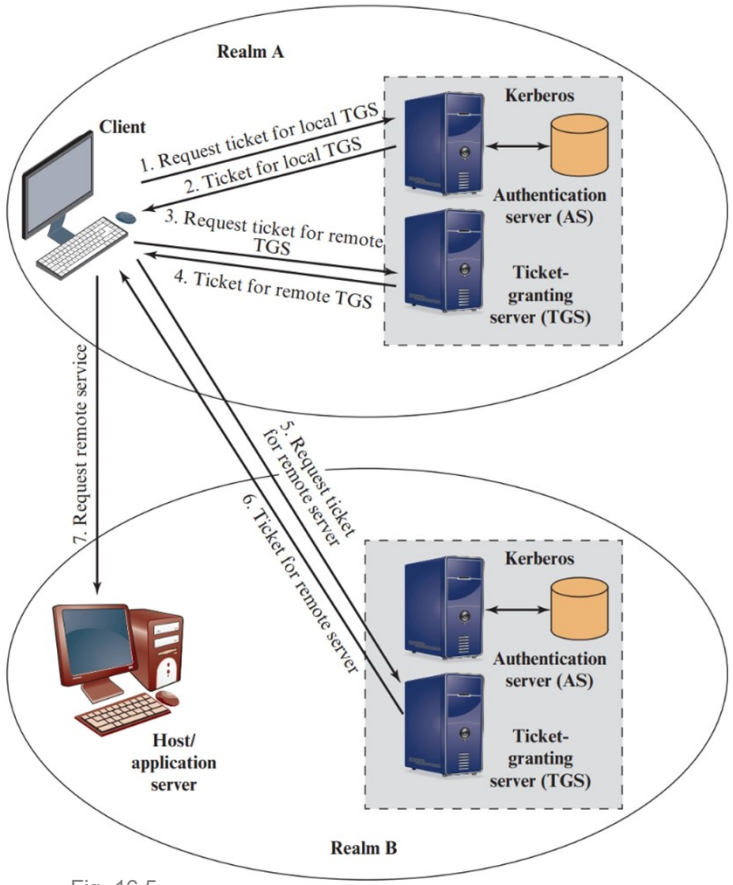


Fig. 16.5

- A realm or domain is a collection of computers that share and trust the same set of user accounts
- Inter-domain keys make it possible to create trusting domains:
  - Possible to get session tickets in other domains through foreign KDCs
  - Clients get a "referral" key from the local KDC
  - Client then contacts foreign KDC with this key
  - Foreign KDC can decrypt it with the Inter-domain key
  - The referral key shows that the client is trusted
- This can be done in multiple steps
  - Each server issues a new referral key
  - The final KDC may issue both a session key and a TGT to make sure the client can talk directly to this KDC next time

Kerberos 5 [1993]

- Uses synchronized clocks to prevent replay attacks
  - For time stamps to work properly, the clocks need to be in sync
  - Default domain policy = 5 minutes
- Version 5 enhancements:
  - Authenticators are only valid 5 minutes
  - Master key only used once and session keys are much stronger
  - Supports any encryption system such as AES (not just DES)
  - V5 now allows arbitrary TGT ticket lifetimes, max in v4 was 21 hours
- Possible to grant another entity to request tickets on behalf of us
  - The new entity will get a special TGT to request session tickets
  - Example: Web server needs to contact a database server; it cannot use its own credentials if it is working on behalf of the user

Summary

- Kerberos implements single sign-on (SSO)
- Kerberos performs both authentication and authorization
  - Authorization by the Ticket Granting server
- Two types of tickets:
  - TGT - Ticket Granting Ticket
  - SGT - Service Granting Tickets
- In a realm:
  - Kerberos server (AS) stores hashed passwords for all its users
  - Kerberos server has a secret key with each application server
  - Cross-realm trust is possible
- Microsoft Active Directory is a Kerberos implementation