Overview of Kerberos(I)

- ❑ **Network Authentication Protocol for C/S application based on symmetric cryptosystem**
- ❑ **TTP authentication service**
- ❑ **Based on secret key, single login**
- ❑ **Part of MIT's project Athena (public domain), '85**

  ; I’v been there Aug. 2000 during CHES2k
- ❑ **Components: library, data base, authentication daemon, ticket-granting service, applications**
- ❑ **Uses authenticators (for users and servers) and tickets**

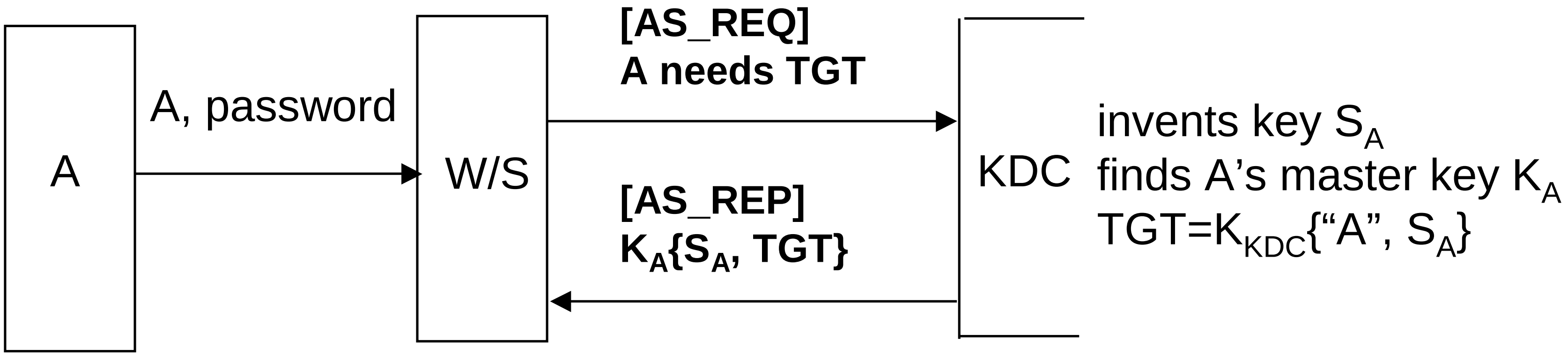
Kerberos : 지옥문을 지키는 머리**3**개 달린 개

Overview of Kerberos(II)

- ❑ Provides:
  1. authenticated messages
  2. safe messages (encrypted checksum)
  3. fully encrypted messages (encrypted telnet)
- ❑ Needs network time
- ❑ Uses one-way encryption (DES) (keys)
- ❑ Applications must be "kerbetized"
- ❑ Does not trust hosts
- ❑ V4 and V5 available

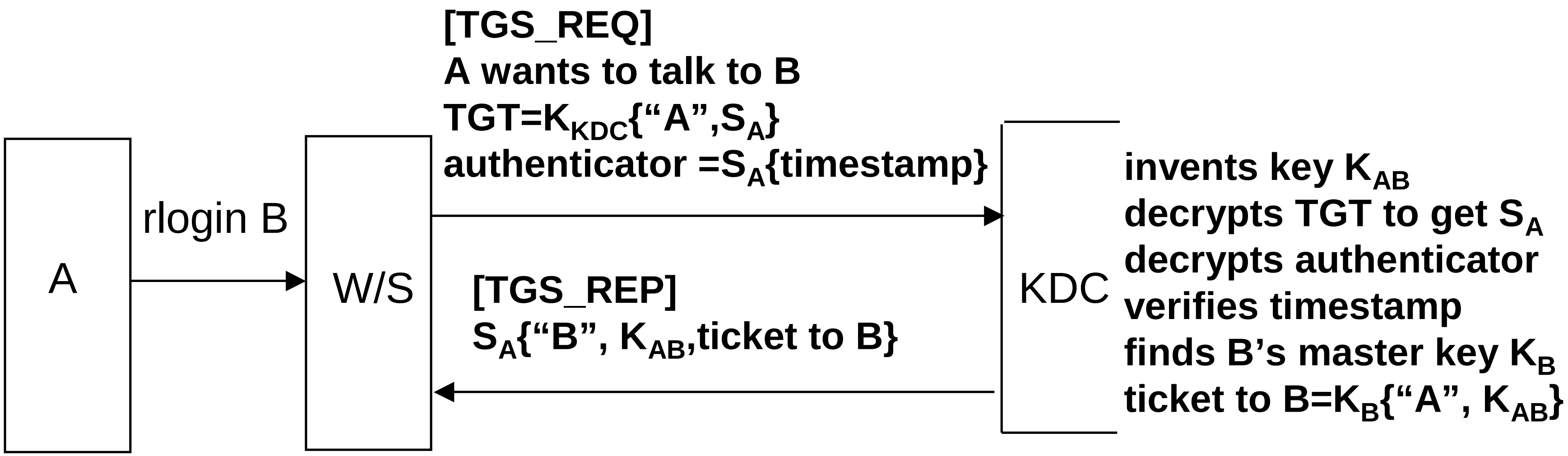
- ❑ Network Security Solution

S1. Obtaining TGT



invents key S_A
finds A's master key K_A
$TGT=K_{KDC}\{\text{"A"}, S_A\}$

S2 Getting ticket to B for A

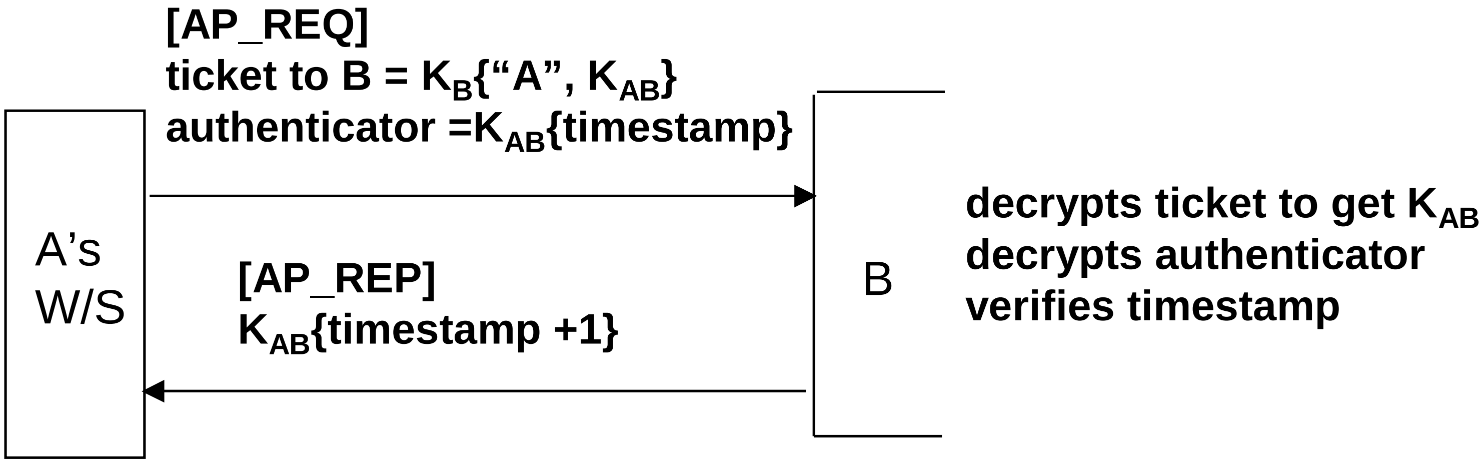


[TGS_REQ]
A wants to talk to B
TGT=K_{KDC}{"A",S_A}
authenticator =S_A{timestamp}

[TGS_REP]
S_A{"B", K_{AB},ticket to B}

invents key K_{AB}
decrypts TGT to get S_A
decrypts authenticator
verifies timestamp
finds B's master key K_B
ticket to B=K_B{"A", K_{AB}}

S3 Logging into B from A’s W/S



[AP_REQ]
ticket to B = K_B{“A”, K_{AB}}
authenticator =K_{AB}{timestamp}

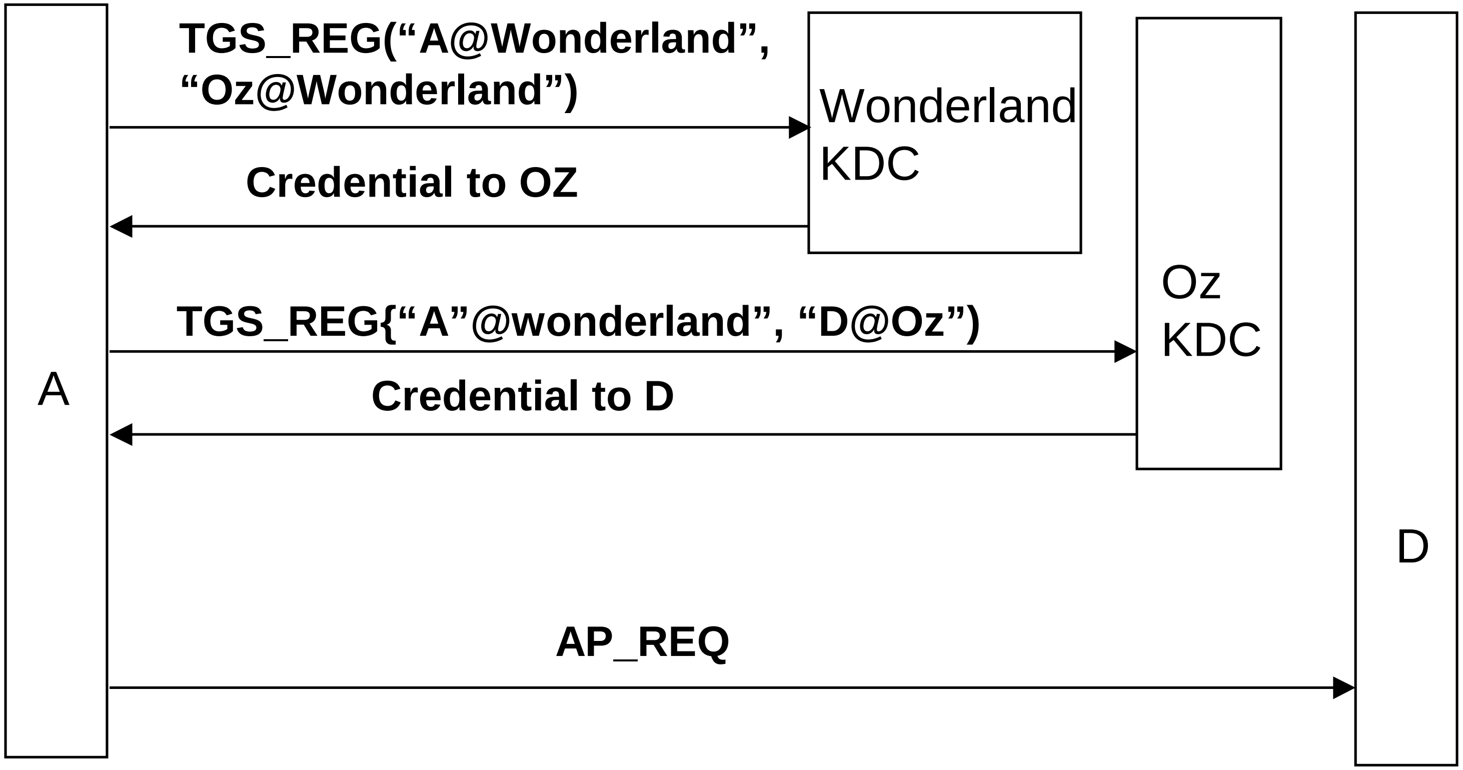
A’s
W/S

[AP_REP]
K_{AB}{timestamp +1}

B

decrypts ticket to get K_{AB}
decrypts authenticator
verifies timestamp

Interrealm Authentication



Kerberos credentials(I)

authenticator

1. name/instance/realm of the client
2. timestamp

- ❑ used only once
- ❑ generated each time client wants to use a service
- ❑ encrypted with server's session key
- ❑ inhibits replay

Kerberos credentials(II)

ticket

1. server
2. client
3. client workstation address
4. timestamp
5. lifetime
6. session key

- ❑ encrypted with server's key
- ❑ generated by TGS
- ❑ good for a single client and server

Setting up Kerberos

- get source from MIT (cygnus)
- designate secure authentication server machine
- maybe slave authentication servers
- build applications (r-utilities, login, ftp, pop, klogin, kinit, klist, kadmin)
- register principals (user, servers)
- data base is encrypted with master key
- install each server's key (/etc/servtab)

client-only easy, (PC/MAC versions)

Kerbetizing

- you can add Kerberos calls to your own client/servers
- need Kerberos data base, authenticator, ticket-granting server, and administrative programs
- can use klogin, but better if you have kerberized BSD utilities
- Kerberos calls added to login, r-utilities, NFS
- rlogin -x sets up encrypted session, every packet is encrypted

V4 implementation

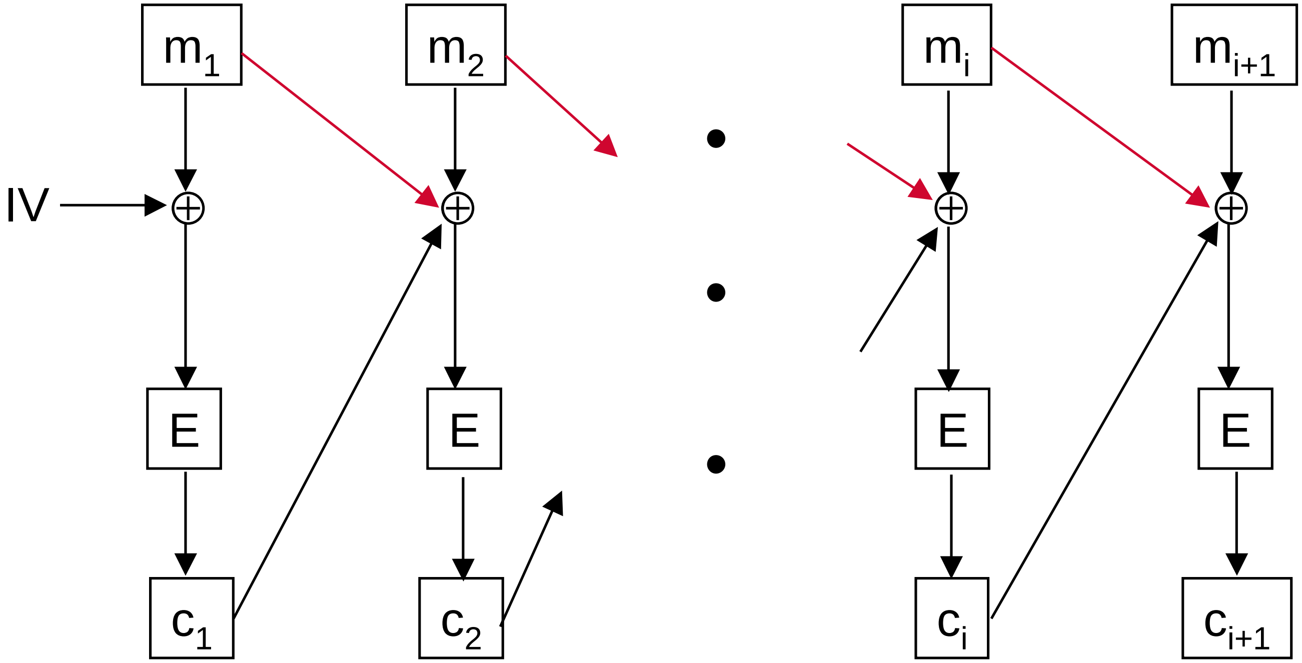
- □ typical client/server application
- □ library requests, just UDP packets
- □ Kerberos servers listening on well-known ports (88)
- □ encryption: modified DES CBC
- □ MAC: Juneman checksum on (key,msg)

Kerberos services

```
/etc/services
kerberos          88/udp    kdc    # Kerberos authentication--udp
kerberos          88/tcp    kdc    # Kerberos authentication--tcp
klogin            543/tcp          # Kerberos authenticated rlogin
kshell            544/tcp   cmd    # and remote shell
kerberos-adm      749/tcp          # Kerberos 5 admin/changepw
kerberos-adm      749/udp          # Kerberos 5 admin/changepw
kerberos-sec      750/udp          # Kerberos authentication--udp
kerberos-sec      750/tcp          # Kerberos authentication--tcp
kerberos_master   751/udp          # Kerberos authentication
kerberos_master   751/tcp          # Kerberos authentication
krb5_prop         754/tcp          # Kerberos slave propagation
kpop              1109/tcp         # Pop with Kerberos
eklogin           2105/tcp         # Kerberos encrypted rlogin

krb524            4444/tcp         # Kerberos 5 to 4 ticket xlator
```

Encryption for Privacy and Integrity



PCBC (Plaintext Cipher Block Chaining)

V5

- ❑ More functionality
- ❑ Principle names multicomponent
  - – v4 was NAME/INSTANCE/REALM(40 max)
  - – v5 : NAME/REALM
- ❑ New encodings (ASN 1.0)
- ❑ New ticket flags (delegation) and longer lifetimes
- ❑ Encryption/MAC replacement
- ❑ V5 will handle v4 requests

V5 encodings

- ❑ ASN.1 data representation (v4 : byteorder bit)
- ❑ address encoding (v4 : IPv4 only)
- ❑ selectable encryption/MAC

V5 tickets

- ❑ **proxiable TGT - can be used to request tickets for a different net address (Alice can let Bob use her printer)**
- ❑ **forwardable TGT - can be presented to a remote TGS**
- ❑ **lifetimes**
  - – **longer lifetimes (v4 : 21 hrs) (v5:start/end)**
  - – **renewable (by KDC)**
  - – **postdated (good a week from now for 2 hrs, KDC clears INVALID flag)**

V5 extensions

- ❑ MAC: DES of md5/md4/DES- CBC
- ❑ Encryption+MAC: DES + md4/md5/CRC
- ❑ Hierarchy of realms
  - – v4: principals in A to be authenticated in B, B's KDC must be registered in A's KDC

Why not?

- every network service must be modified
- Kerberos server must be physically secure
- export restrictions
- doesn't protect against Trojan horses
- off-line password attack on message from KDC to client
- if password is disclosed, eavesdropper can decrypt other tickets and spoof servers and users

Still, better than anything else.

new Kerberos features

- ❑ public key for initial authentication
- ❑ one-time password support
- ❑ Kerberos V5 RFC1510
- ❑ using Kerberos for authorization

Yaksha

- ❑ **Problems of Kerberos**
  - **AS keeps C’s secret key**
  - **On issuing ticket, user authentication only, no digital signature**
  - **Possible dictionary attack of password**

- ❑ **Ravi Ganesan, “The Yaksha Security System”, Communication of the ACM, Vol. 39, No.3, pp.55 -60, 1996**