

Authentication II

Dr. Arjan Duresi
Louisiana State University
Baton Rouge, LA 70810
Duresi@csc.lsu.edu

These slides are available at:

<http://www.csc.lsu.edu/~duresi/csc4601-07/>



- Kerberos V4
- X.509 authentication and certificates

What Is Kerberos?

- ❑ Trusted key server system from MIT.
Recommended reading:
<http://web.mit.edu/kerberos/www/dialogue.html>
- ❑ Provides centralised secret-key third-party authentication in a distributed network
 - allows users secure access to services distributed through network
 - without needing to trust all workstations
 - rather all trust a central authentication server
 - Relieve users/administrators the burden of managing potentially many accounts and passwords

Kerberos Requirements

- ❑ First published report identified its requirements as:
 - security
 - reliability
 - transparency
 - scalability
- ❑ Implemented using an authentication protocol based on Needham-Schroeder

Kerberos

- ❑ First three versions are no longer in use, we'll study versions 4 and 5
- ❑ Version 4
 - has a greater installed base
 - Simpler
 - Better performance
 - Works only on TCP/IP
- ❑ Version 5 has greater functionalities

Kerberos Deployment...

- ❑ KDCs are “physically” secured
- ❑ Kerberos libraries are distributed on all nodes with users, applications, and other Kerberos-controlled resources
- ❑ All Kerberos exchanges are protected against confidentiality and integrity attacks
- ❑ Kerberos-rized applications
 - telnet
 - rtools (rlogin, rcp, rsh)
 - Network file systems (NFS/AFS)
 - Others (pine, eudora, etc.)

Kerberos 4 Overview

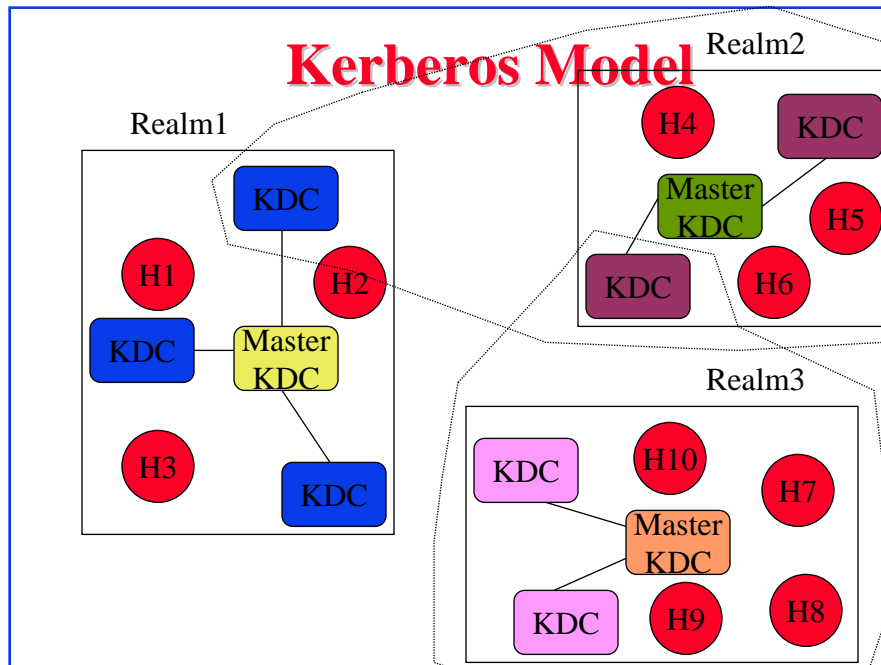
- ❑ A basic third-party authentication scheme
- ❑ Have an Authentication Server (AS)
 - users initially negotiate with AS to identify self
 - AS provides a non-corruptible authentication credential (ticket granting ticket TGT)
- ❑ Have a Ticket Granting server (TGS)
 - users subsequently request access to other services from TGS on basis of users TGT

Terms and Configuration

- ❑ Configuration
 - Alice (human) uses secure terminals with no user-related information to access resources over insecure network
 - Bob (application server) holds the resources
 - Alice uses KDC to authenticate herself to Bob
- ❑ Terms
 - *principal* – a user or resource that use Kerberos
 - *master key* – a long-term secret shared between KDC and each principal
 - KDC is also known as Authentication Server (AS) and Ticket Granting Server (TGS), has the database of secrets for the principals, encrypted for security (why?) with KDC's *master key* (that KDC shares with no one "for now")
 - *authenticator* – a timestamp encrypted by a key: proves the receiver that the sender knows the key
 - clocks in network need to be somewhat synchronized
 - servers tolerate clock skew (about 5mins)

Sessions, Keys, Tickets

- ❑ A session is the process of continuous interaction between entities
- ❑ Alice has two types of sessions
 - with KDC: session proceeds from login to logout
 - with Bob: when she needs his services, Alice may have multiple sessions with multiple application servers during login
- ❑ Keys
 - K_A - Alice's master key (shared with KDC) and derived from her password
 - S_A - session key with KDC
 - K_{AB} - session key with Bob
 - K_{KDC} - KDC's master key
- ❑ Ticket granting ticket (TGT) - a ticket that Alice's workstation during the session with KDC and presents to KDC to initiate a session with Bob
- ❑ A key + ticket to a certain server is the client *credentials* to this server
 - What credentials may Alice have?



Kerberos Realms

- ❑ A Kerberos environment consists of:
 - a Kerberos server
 - a number of clients, all registered with server
 - application servers, sharing keys with server
- ❑ This is termed a realm
 - typically a single administrative domain
- ❑ if have multiple realms, their Kerberos servers must share keys and trust

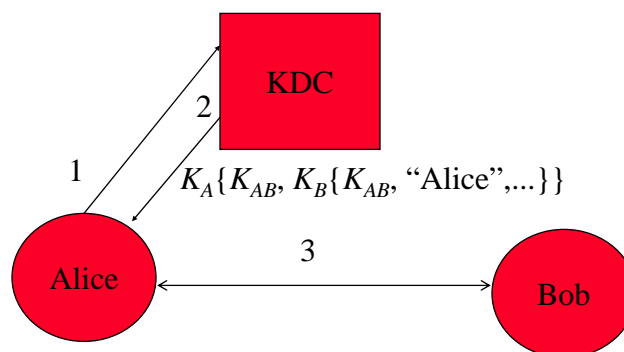
Where To Start...

- ❑ Every principal has a master (secret) key
 - Human user's master key is derived from password using DES
 - Other resources must have their keys configured in
- ❑ Every principal is registered with the Kerberos server, i.e. KDC
- ❑ All principals' master keys are stored in the KDC database, encrypted using the KDC master key

Tickets ...

- ❑ Every principal has a main shared secret with the KDC - principal's master key
- ❑ Any secure communication/access among principals must be "mediated" by KDC through *tickets*
- ❑ How would Alice talk to Bob?

Alice, Bob, and KDC



Ticket to Bob: $K_B\{K_{AB}, \text{"Alice"}, \dots\}$

Session Key and Ticket-granting Ticket (TGT)

- ❑ Messages between a host and the KDC can be protected using the principal's master key
- ❑ For every request to KDC from the principal:
 - Insists on principal retyping in the password
 - Remember the principal's password
 - Remember the principal's master key derived from the password
- ❑ All options are equally inadequate!

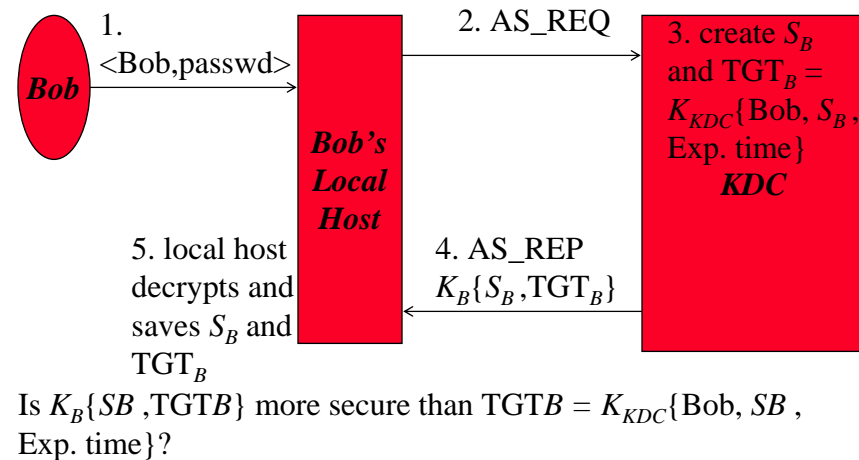
Session Key and TGT...

- ❑ To avoid potentially too much exposure to password/master key
 - At initial login, a per principal session key S_B (for Bob) is requested from KDC
 - S_B has a limited valid time period
 - A TGT for Bob is also issued by the KDC, which includes the session key S_B and Bob's identification information, all encrypted using the KDC's master key

Session Key and TGT...

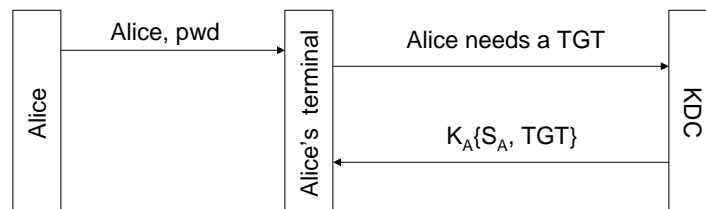
- ❑ Bob's Kerberos client (e.g., the login host) decrypts and remembers:
 - S_B , for subsequent message with KDC
 - TGT, for reminding/convincing KDC to use S_B with it as well
 - No need for remembering/storing password
- ❑ New request to KDC must include TGT in the request message
- ❑ New tickets from KDC must be decrypted with S_B

Login



Logging into Network

- ❑ Potentially Alice's workstation can keep K_A during Alice's session with KDC every time Alice needs to talk to Bob
 - how?
 - however, it is considered insecure, why?
- ❑ Solution: session key and TGT – a ticket that Alice's workstation keeps and presents to KDC for authentication, TGT contains:
 - S_A encrypted with K_{KDC} ,
 - timestamp (validity TGT is limited to a few hours)



dictionary attack is possible, how?

Login

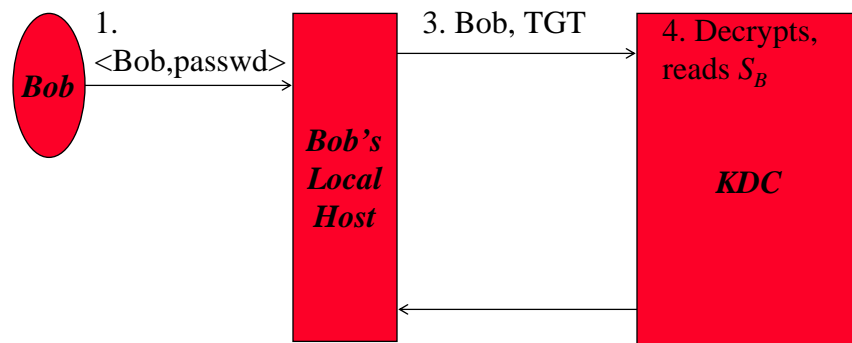
- ❑ Kerberos V4 does not prompt the user for the password until after the workstation has received the credentials from KDC
 - To keep the password for minimum time
- ❑ Kerberos V5 has the user type the password before sending the credentials
 - Make less easy to obtain information which could be used for off-line guessing
- ❑ TGT enables KDC to operate without having to remember state of operations
 - For each request, KDC sends a response and forget about it

Login

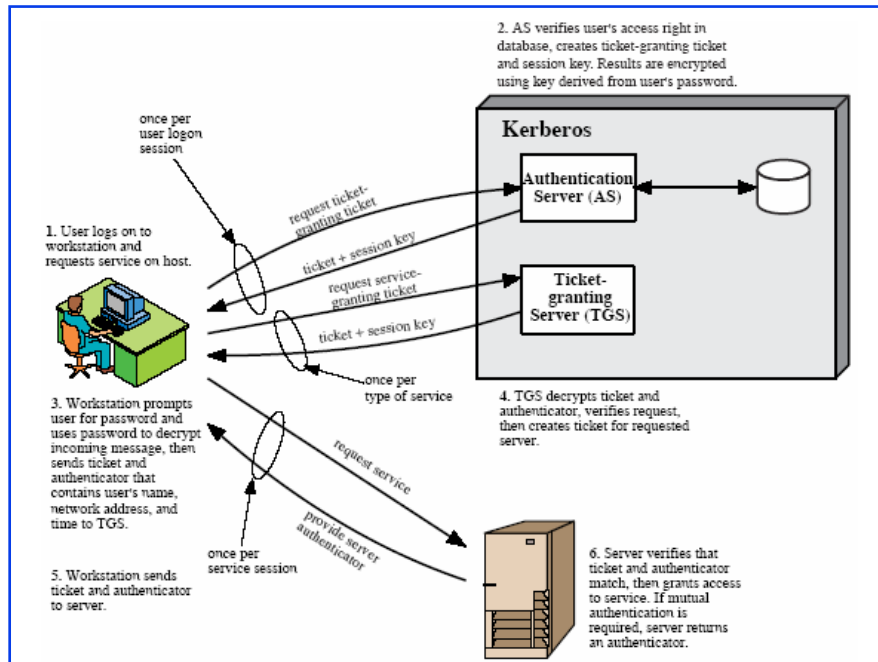
- What if the workstation generates the TGT?

Login

2 . Bob host generates $TGT = K_B\{Bob, S_B\}$

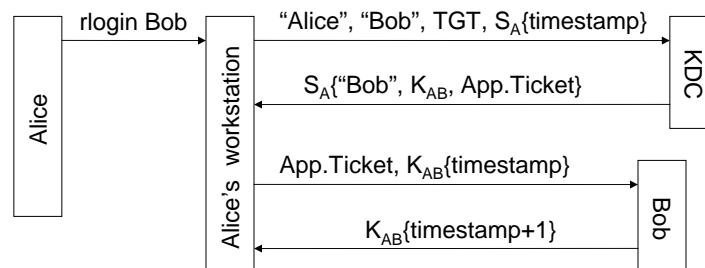


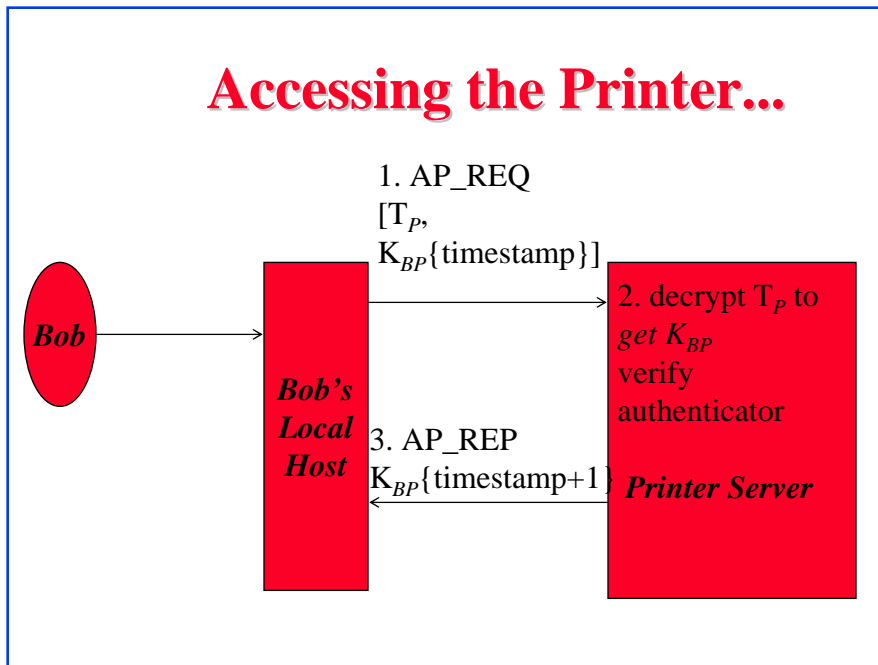
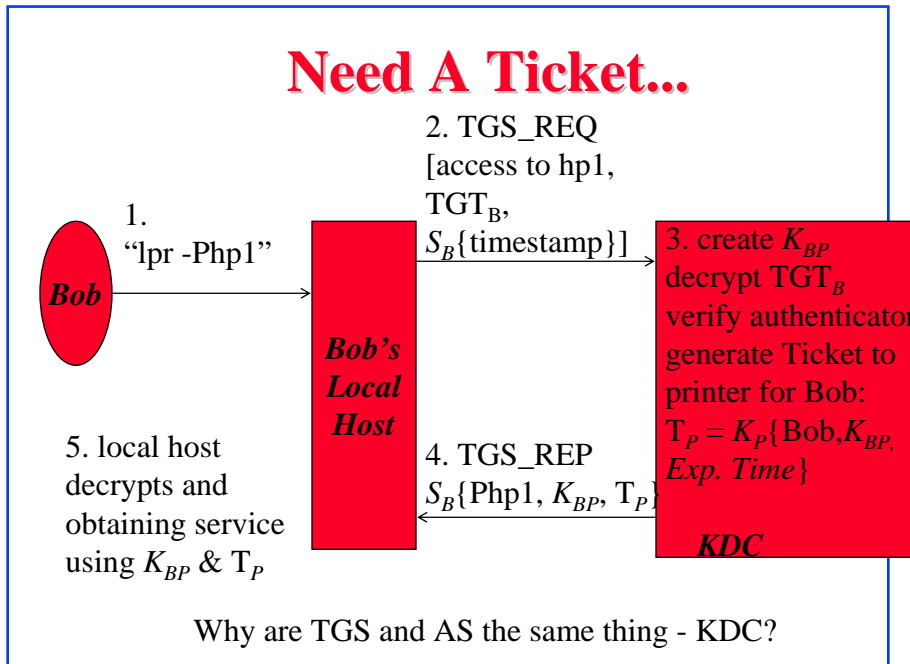
If Bob changes password and K_B - invalidates TGT



Accessing a Remote Principal

- Authenticator to KDC is not needed (why?) but makes authentication to KDC similar to authentication to Bob
- App.Ticket is encrypted with K_B , it contains :
 - "Alice", K_{AB} , time the ticket was created, lifetime (to determine when the ticket expires)
- What's the purpose of Bob's reply to Alice?
- After authentication, the traffic between Alice and Bob can optionally be encrypted and integrity protected





Authentication and Global Clock Synchronization

- ❑ Authenticator == $K_X\{\text{timestamp}\}$
- ❑ Global clock sync is implied
- ❑ Is the authenticator for TGS_REQ necessary?
 - The TGS_REP is encrypted with S_B
- ❑ What about the AP_REQ?
- ❑ Main purposes of authenticator is to avoid
 - replay of old requests to the same server
 - replay of request on one server to another (server farm, shared principal's master key)

Kerberos

- ❑ Kerberos could be used for:
 - Authentication
 - Integrity-protected
 - Encryption and integrity-protected
- ❑ The decision is a trade-off between performance vs. security

Replicated KDCs

- ❑ A single KDC – a single point of failure
- ❑ Multiple replica of KDC - availability and performance. The KDC are interchangeable, share the same master key and database
- ❑ Keeping KDC databases consistent
 - Single master KDC as the point of direct update to principals' database entries
 - Updated database is downloaded from the master to all replica KDCs
 - Periodic download or on-demand

Will It Be Effective?

- ❑ KDC dynamic state consists of outstanding TGTs and tickets.
- ❑ Kerberos puts the burden of “maintaining” them on the clients - hosts/servers/grantees.
 - Convince me that I did this for you...
- ❑ KDC is only involved in the initial “mediation” and it stays “out of the picture” once a ticket is issued.
- ❑ Only static state information is principals' database – **read-only** for all replica KDCs.

Database Content Protection

- ❑ Encryption is required for sensitive data
- ❑ Integrity of the database must be ensured
 - Installation of masqueraded master keys
 - Substitution (replay) of old databases
- ❑ Kerberos stores principals' master keys encrypted with KDC master key
- ❑ Kerberos transmits a secure hash of the database with encryption, in a separate message during downloads
- ❑ The master key for human users is derived from their password. Other resources are configured with their own master keys

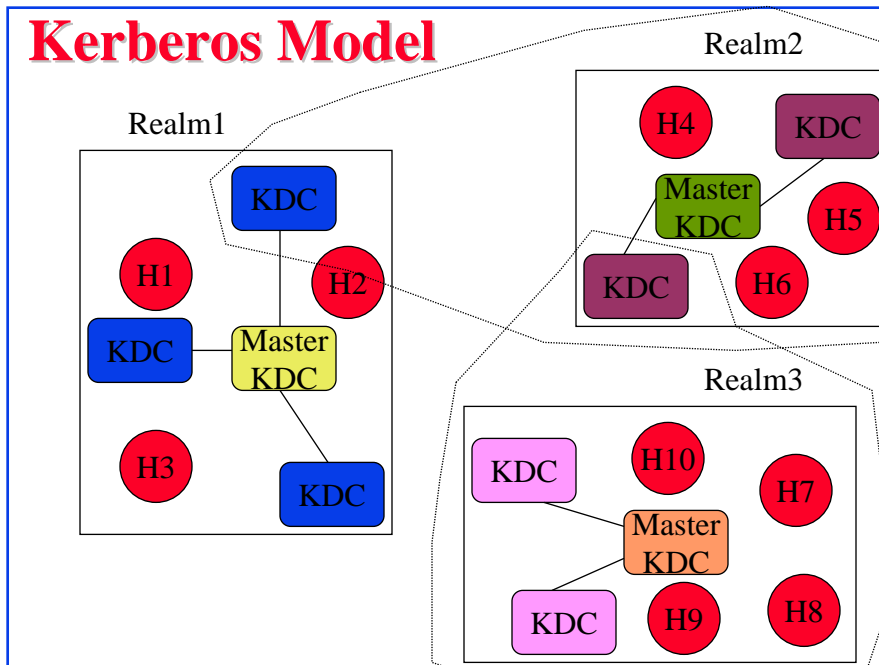
Multiple Trust Domains

- ❑ Single master KDC can only stretch so far...
- ❑ KDC asks people to put too much trust in it.
 - Should competing commercial entities use the same KDC?
 - .gov, .org, .edu etc, each having a different model of "what is more trustworthy."
- ❑ Single master KDC - greatest temptation - biggest security risk/vulnerability.
- ❑ So comes different domains or *realms*.

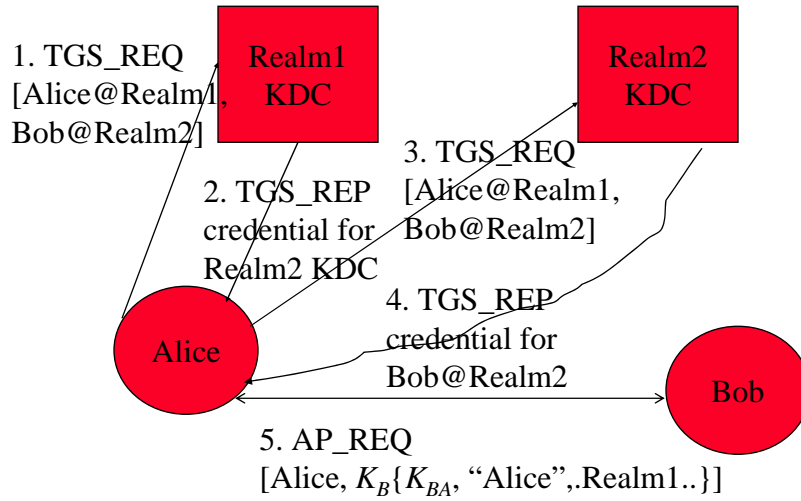
Kerberos Realms

- ❑ Single-Realm KDC is not scalable
 - hard to keep track of principals,
 - single KDC (even with replicas) is a single point of failure/attack
- ❑ Solution: multiple Realms
 - realm administrators decide what principals to allow interrealm connections
 - for Alice to connect to Bob, Alice becomes a principal in KDC_B 's realm
- ❑ Each realm has a different master KDC, with different master KDC key
- ❑ Each realm can have many replica KDCs, but all sharing the same KDC master key
- ❑ Two KDCs in different realms have different principals' master key databases
- ❑ Chains longer than two KDCs are not allowed

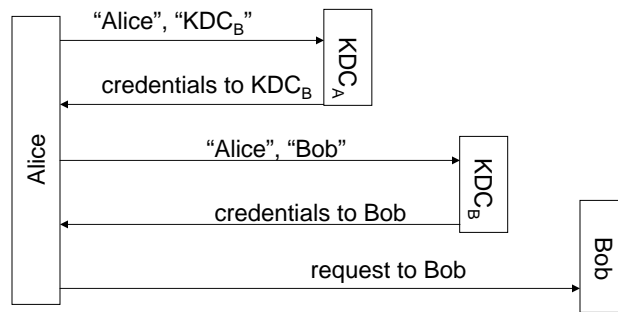
Kerberos Model



Inter-realm Authentication



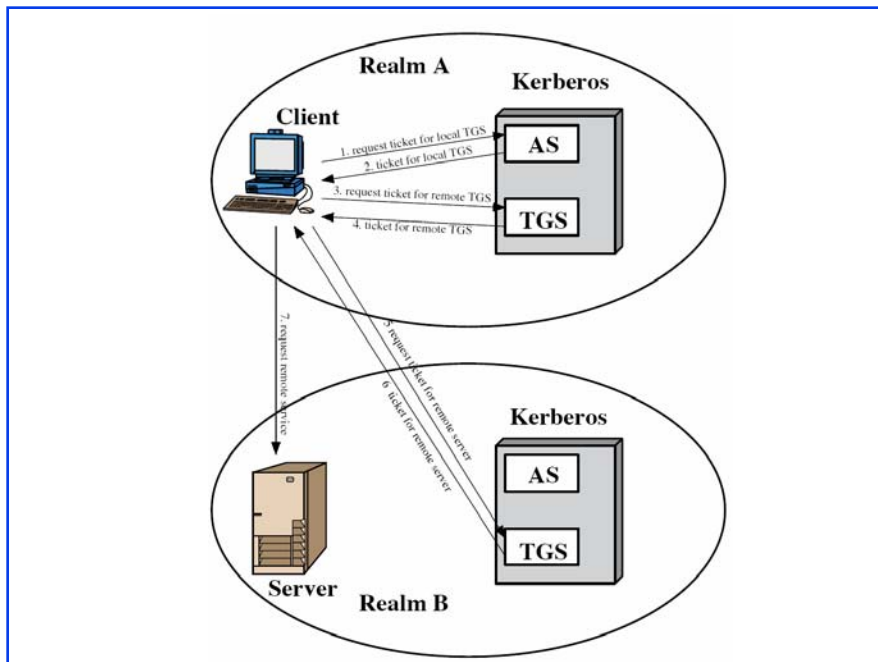
Multiple Realms



Chains longer than two KDCs are not allowed

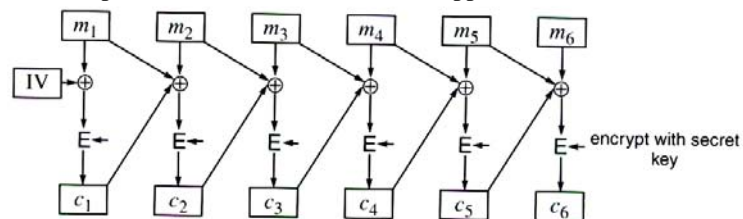
Inter-realm Authentication

- ❑ Kerberos deliberately prevents access through a chain of KDCs to avoid a rogue KDC to impersonate users from other realms
- ❑ Realms A and B share a key and realms B and C share another key
- ❑ Alice@A would like to talk to Carol@C
 - Alice can get a ticket to B, then request and get from B a ticket to C
 - When Alice ask C for a ticket to Carol the TGS_REQ will have realm field B and in the ticket Alice's realm A. C will refuse to issue a ticket for Carole because the two realms don't match
- ❑ If there are N realms, there must be $N(N-1)/2$ secure key exchange in order for all realms to interoperate



Privacy and Integrity in Kerberos

- ❑ block cipher is DES
- ❑ how to ensure privacy and integrity together?
- ❑ doing it properly requires two passes and two separate keys
- ❑ Kerberos version – plain cipher block chaining (PCBC)
 - both plaintext and ciphertext are XORed to get the next block of the pad
 - will not resynchronize if the block is lost or tampered
 - ❑ what other stream cipher has this property?
 - puts a recognizable data at the end of the message to confirm lack of tampering
 - still has problems – if two blocks are swapped, the tail checks out



Kerberos Version 5

- ❑ Developed in mid 1990's
- ❑ Provides improvements over v4:
 - Addresses environmental shortcomings
 - and technical deficiencies
- ❑ Specified as Internet standard RFC 1510

Environmental Shortcomings

- ❑ Encryption algorithm: Version 4 uses DES
 - Export restriction, doubt about security of DES
 - In version 5 any type of encryption can be used
- ❑ Network protocol: Version 4 – IP, 5 - any
- ❑ Byte order: in Version 4 – user defined, 5 – ASN.1
- ❑ Ticket lifetime: Version 4 – encoded 8 bit in units of 5 minutes – total 21 hours, might not be enough, 5 – start and end
- ❑ Authentication forwarding: not allowed in Version 4, included in 5
- ❑ Interrealm authorization: in Version 4 – interoperability among N realms requires N² relations. Version 5 supports hierarchy

Technical Deficiencies

- ❑ Double encryption of tickets to clients – waste
- ❑ Non-std mode of use: PCBC encryption - shown to be vulnerable to interchange of cipherblocks
- ❑ Session keys: Each ticket includes a session key used for authenticator and encryption. The same ticket can be used again – risk of replay attack. In Version 5: client and server negotiate a sub-session key
- ❑ Password attacks (both versions): The message from AS to client is encrypted with the client's key based on password: open to password guessing

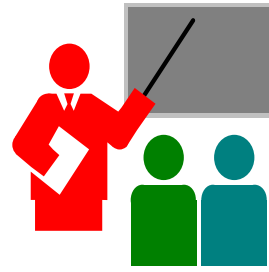
Kerberos v5

- ❑ Platform-independent specification (ASN.1 –data representation language standardized by ISO)
- ❑ allows non-IP addresses
- ❑ allows non-DES encryption
- ❑ delegation of rights – Alice may request Bob to access principals on other servers on her behalf
- ❑ hierarchy of realms
- ❑ ticket flexibility
 - renewable tickets
 - post-dated tickets (tickets valid for some time in the future)
- ❑ a set of integrity- and privacy/integrity-protection algorithms
- ❑ avoids possibility of dictionary attack by pre-authenticating the user
 - in KDC's database users are marked as specific principals whose TGT should not be given out without pre-authentication
- ❑ has public-key extensions (e.g., SESAME, Win2000)

Security of Kerberos

- ❑ It may be possible to cache and replay old authenticators. Servers might not be able to keep all tickets
- ❑ If a host can be fooled about the correct time, then an old authenticator can be used. Most network time protocols are insecure. This can be a serious problem
- ❑ Kerberos is vulnerable to password-guessing. An intruder can collect tickets and then try to decrypt them
- ❑ The most serious attack – malicious software . If Kerberos client is replaced with a version that records passwords...
- ❑ Kerberos is not in the public domain. The MIT code is freely available

Summary



- ❑ Kerberos trusted key server system
- ❑ Version 4
- ❑ Version 5