DI-FCT-UNL
Segurança de Redes e Sistemas de Computadores
Computer Networks and Systems Security

Mestrado Integrado em Engenharia Informática
MSc Course: Informatics Engineering
1st Sem., 2020/2021

Key Distribution Protocols and Models using Symmetric Cryptography

The KERBEROS Authentication and Key-Distribution System

Last slides

- Issues, strategies and initial models for Key Distribution Protocols (KDPs)
- Different models for KDPs

In these slides:

Outline

- Key Distribution Protocols and Models using Symmetric Cryptography and Key Distribution Centers
- The Kerberos System
  - Kerberos Authenticated Key-Distribution Protocol
  - Kerberos V4
  - Kerberos V5

In these slides:

Outline

- Key Distribution Protocols and Models using Symmetric Cryptography and Key Distribution Centers
- The Kerberos System
  - Kerberos Authenticated Key-Distribution Protocol
  - Kerberos V4
  - Kerberos V5

Secure Key-Distribution Protocol

- Base security conditions (depending on the adversary model definition) relates with:

  Secure channel abstractions for the protocol processing and its properties:

  - Authentication (Peer vs. Data or Message)
  - Confidentiality (Connectionless vs. Connection-Oriented)
  - Integrity (including No-Replaying Guarantees)
  - Availability (DoS, at least DoS Mitigation)
  - Non-Repudiation
  - Access-Control

  + other additional (complementary) security controls, ex:

  Routing control, Resilient Routing/Delivery, Notarization/Logging, Traffic Padding and Layering/Encapslation for Traffic Flow Guarantees

Different key-distribution protocols based only in the KDC and symmetric cryptography model

Examples of Base models for KDPs

- **Needham Schroeder and variants // We will study this one !**
- Otway-Rees
- Neuman-Stubblebine
- Yahalom
- Miller-Neuman > Kerberos, Kerberos Versions, ex. V4 and V5)
- …. Many other protocols and variants:
  - Wide Mouth Frog
  - Janson-Tsudik
  - Bellare-Rogaway
  - Who-Lam
  - Gong
  - Boyd
  - Etc …

Needham-Shroeder Model (Variant 1)

A -> KDC: A, B, Na

KDC ->A: $\{N_{a+1}, K_{AB}, B, \{K_{AB}, A, B, N_{b1}\}_{KB}\}_{KA}$

A -> B: $\{K_{AB}, A, B, N_{b1}\}_{KB}$

B -> A: $\{N_{b2}\}_{KAB}$

A -> B: $\{N_{b2+1}\}K_{AB}$ //... can piggyback 1st encrypted msg

Problem: replay attack of message 3, exploring a possible compromise of previously used Ks

How to fix it ?

Needham-Shroeder Model (Variant 1)

A -> KDC: A, B, Na

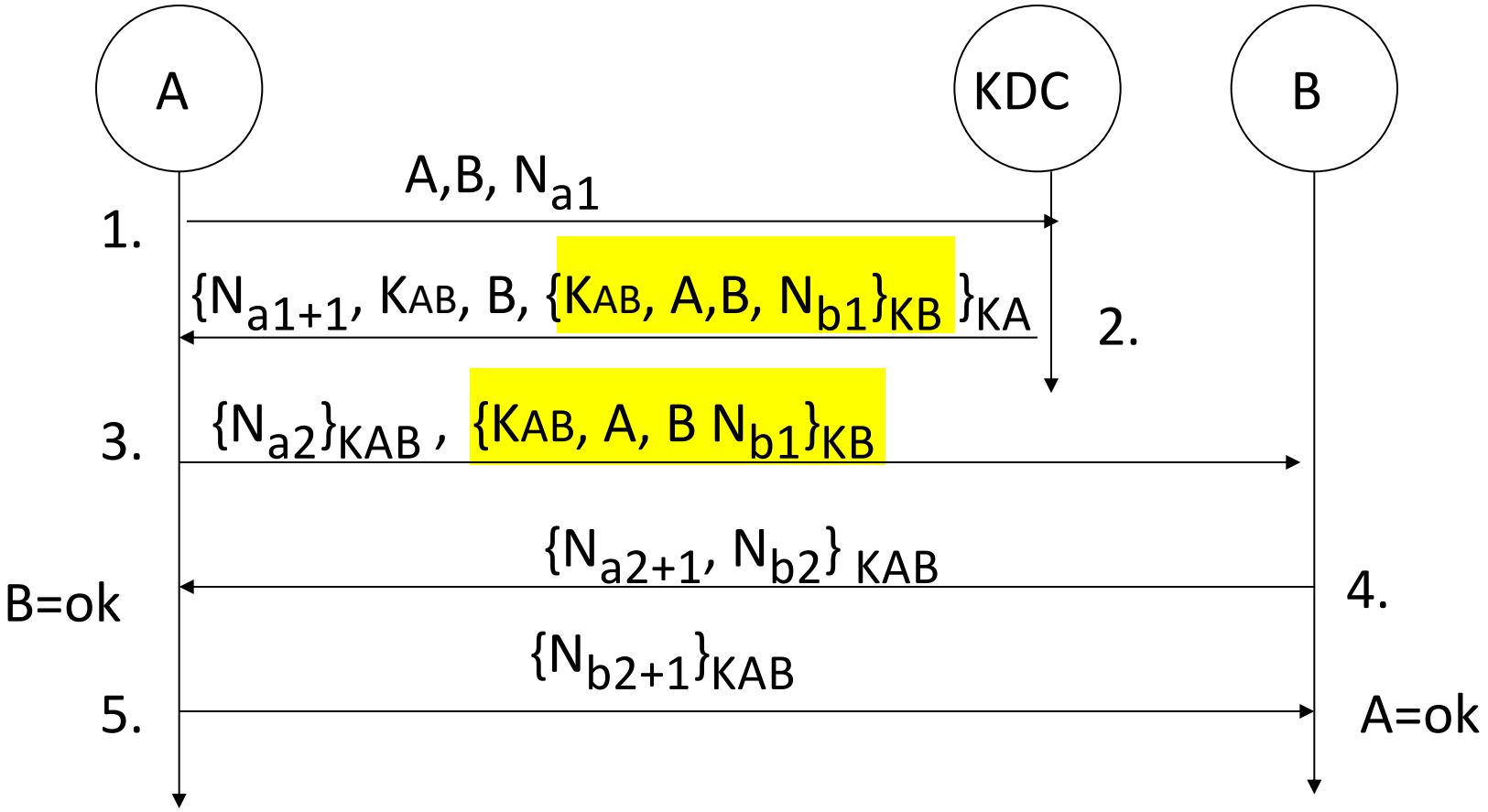
KDC ->A: $\{N_{a+1}, K_{AB}, B, \{K_{AB}, A, B, N_{b1}\}_{KB}\}_{KA}$

A -> B: $\{N_{a2}\}_{KAB}$, $\{K_{AB}, A, B, N_{b1}\}_{KB}$

B -> A: $\{N_{a2+1}, N_{b2}\}_{KAB}$

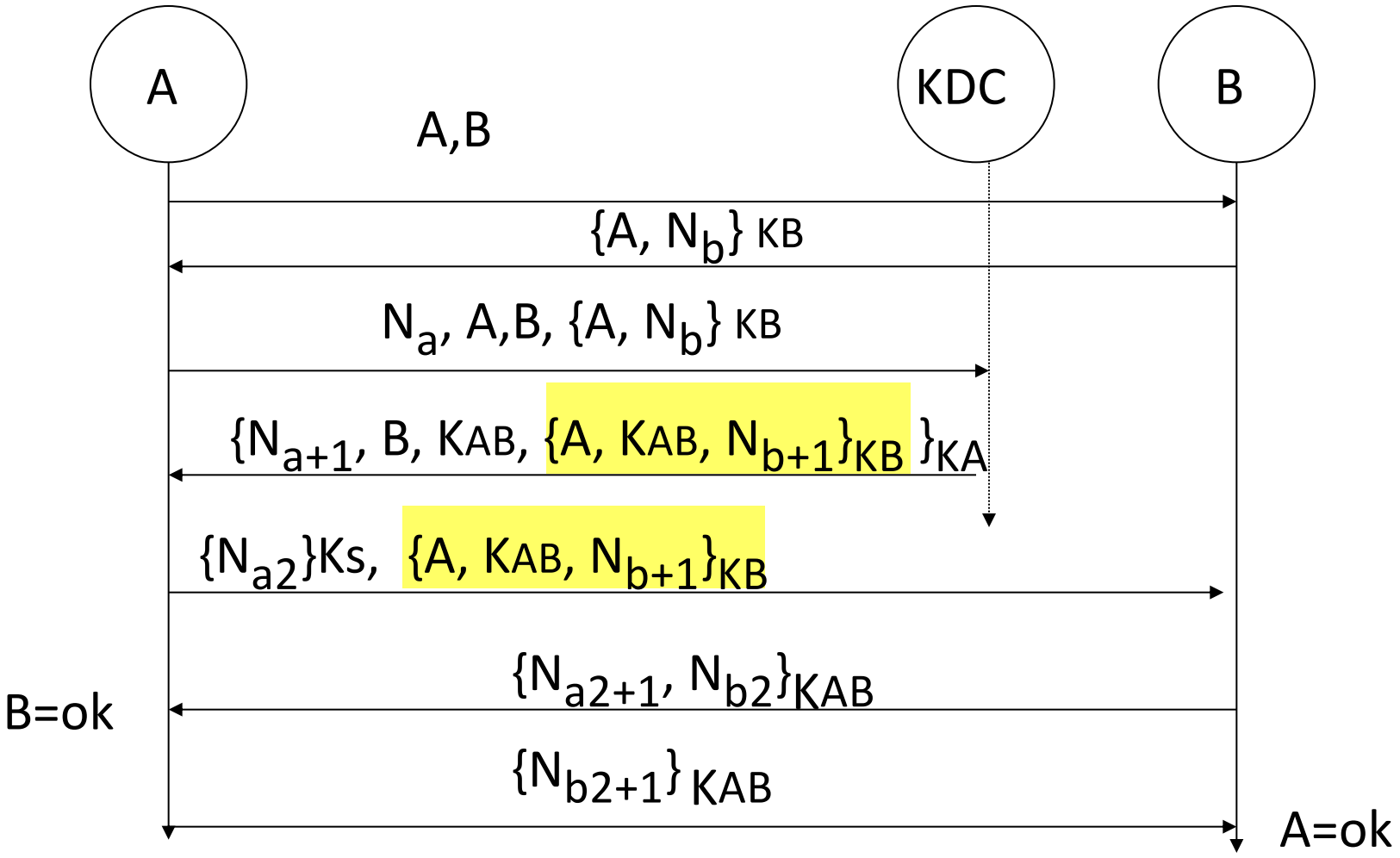
A -> B: $\{N_{b2+1}\}K_{AB}$ //.. can piggyback 1st encrypted msg

NS Protocol (timing diagram)



How can Bob have a better control about the Nb1 in the ticket ?
What if the first message to establish the setup is between A and B ?

Modelo de Needham-Schroeder (Variant)



Otway-Rees Model

A -> B: $M, A, B, \{M, A, B, N_a\}_{K<A\text{-}KDC>}$

B -> KDC:

$M, A, B, \{M,A,B, N_a\}_{K<A\text{-}KDC>}, \{M,A,B, N_b\}_{K<B\text{-}KDC>}$

KDC -> B:

$M, \{A,B, K_{AB}, N_{a+1}\}_{K<A\text{-}KDC>}, \{A,B, N_{b+1}\}_{K<B\text{-}KDC>}$

B -> A: $\{A,B, K_{AB}, N_{a+1}\}_{K<A\text{-}KDC>}, \{N_{b2}\}_{K_{AB}}$

A -> B: $\{N_{a2}, N_{b2+1}\}_{K_{AB}}$

B -> A: $\{N_{a2+1}\}_{K_{AB}}$ //.. And can piggyback encrypted msg

BAN-Yahlom Protocol Model

A -> B: A, B, N_a

B -> KDC: $B, N_b , \{A, N_{a+1}\}_{K<B-KDC>}$

KDC -> A:

$\{B,A,N_{a+1},K_{AB}\}_{K<A-KDC>} , \{B,A,K_{AB}, N_{b+1}\}_{K<B-KDC>}$

A -> B: $\{B,A,K_{AB}, N_{b+1}\}_{K<B-KDC>} , \{N_{a2}, N_{b+1}\}_{K_{AB}}$

B -> A: $\{N_{a2+1}\}_{K_{AB}}$ //.. And can piggyback encrypted msg

Neuman-Stubblebine Model

A->B: A, B, N_a

B->KDC: $B, N_b , \{A, N_{a+1}, TSb \}_{KB}$

KDC->A:

$\{B,A,N_{a+1},K_{AB},TSb \}_{KA} , \{B,A,K_{AB},TSb\}_{KB} ,N_b$

A->B: $\{B,A,K_{AB},TSb\}_{KB} , \{N_{a2}, N_{b+1}\}_{KA}$

B->A: $\{N_{a2+1}\}_{KAB}$

Neuman-Stubblebine

1. A->B: A, Na
2. B->KDC: B, Nb, $\{A, Na, Tb\}_{KB}$
3. KDC->A: $\{B, Na, Ks, Ts\}_{KA}$, $\{A, Ks, Ts\}_{KB}$,Nb
4. A->B: $\{A, Ks, Ts\}_{KB}$, $\{Nb\}_{Ks}$

- Subsequent communication avoiding the pre-phases involving the KDC:

  A->B: {A, Ks, Ts}Kb, Na' // if Ts is fresh

  B->A: Nb' , $\{Na'\}_{Ks}$

  A->B: $\{Nb'\}_{Ks}$

Neuman-Stubblebine + Rekeying

1. A->B: A, Na
2. B->KDC: B, Nb, $\{A, Na, Tb\}_{KB}$
3. KDC->A: $\{B, Na, Ks1, Ts\}_{KA}$, $\{A, Ks1, Ts\}_{KB}$, Nb
4. A->B: $\{A, Ks1, Ts\}_{KB}$, $\{Nb\}_{Ks1}$

- Subsequent communication avoiding the pre-phases involving the KDC:

  A->B: $\{A, Ks1, Ts\}_{KB}$, Na' // if Ts is fresh

  B->A: Nb' , $\{Na'+1, Ks2\}_{Ks1}$

  A->B: $\{Nb'\}_{Ks2}$

Neuman-Stubblebine + Contributive Rekeying

1. A->B: A, Na
2. B->KDC: B, Nb, $\{A, Na, Tb\}_{KB}$
3. KDC->A: $\{B, Na, Ks1, Ts\}_{KA}$, $\{A, Ks1, Ts\}_{KB}$, Nb
4. A->B: $\{A, Ks1, Ts\}_{KB}$, $\{Nb\}_{Ks1}$

- Subsequent communication avoiding the pre-phases involving the KDC:

  A->B: $\{A, Ks1, Tb\}_{KB}$, Na' // if Tb is fresh

  B->A: Nb' , $\{Na'+1, Ksxa\}_{Ks1}$

  A->B: $\{Nb', Ksxb\}_{Ks2}$ w/ Ks2=f(Ksxa,Ksxb)

  Ks2 is the new established key …

Other implementation issues (1)

- Integrity (hashing) and/or some form of MACs
  - Remember the problem of using MACs, HMACs, CMACs and "secret values", and also how to minimize possible DoS assumptions
- Flow-control and communication support (state-machine control in KDC and principals):
  - Connection-oriented
  - Connectionless-oriented
  - Concurrency issues, stateful vs. stateless considerations
  - Management of security associations and the validity of parameters
  - Rendom Nonces: control for use ONLY ONCE
  - Timestamps used as nonces: non-synchronized time

Other implementation issues (2)

- DoS issues (minimization)
  - Not immune of possible DoS attacks
  - Mitigation using MACs (ex., HMACs) or Secure Hash Functions
  - Can be enhanced with other base and complementary guarantees (as stated before)
- Processing control (heading parameters):
  - Ex., version control, size-of message, selectors for protocol-content types and message types, use of dynamically negotiated cyphersuites (or dynamic handshaking)

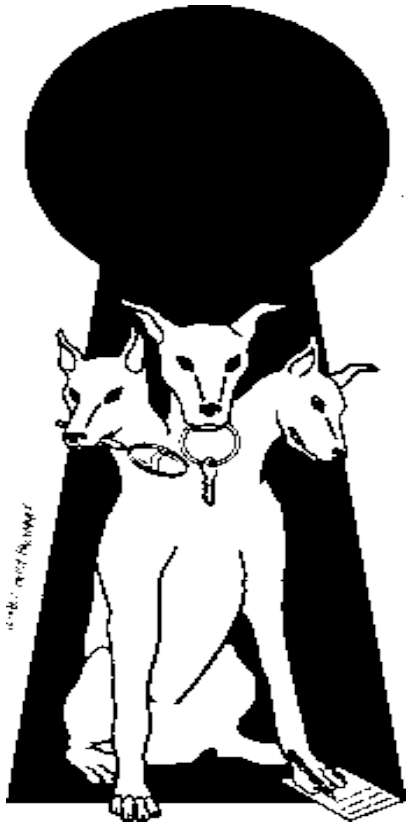
In these slides:

Outline

- Key Distribution Protocols and Models using Symmetric Cryptography and Key Distribution Centers
- The Kerberos System
  - Kerberos Authenticated Key-Distribution Protocol
  - Kerberos V4
  - Kerberos V5

KERBEROS Protocol

In Greek mythology, a many headed dog, the guardian of the entrance of Hades



- MIT, Project Athena, Steve Miller and Clifford Neuman, Oct, 1988

  Dynamic (standardization) Evolution…

  - RFC 1510 Sep/1995

  .....

  - RFC 8009, Oct/2016

  … Ongoing RFCs:

  24/Oct/2016

  30/Mar/2017

  09/Feb/2017

  30/04/2017

  30/03/2017

https://datatracker.ietf.org/doc/search/?name=Kerberos&activedrafts=on&rfcs=on

KERBEROS Evolution

- Version 5, John Kohl and Neuman
  - RFC 1510 – 1993 (V4), made obsolete by RFC 4120, 2005 (V5)
- Until 2000, MIT implementations with DES banned from exportation by the US gov.
- KTH-KRB developed by the Royal Institute of Technology, Sweden, initially from the eBones MIT version (V4)
  - After RIT released V5 (Heimdal distribution)
- Kerberos implementations from MIT freely available after 2000
- Microsoft Windows 2000 adoption of Kerberos as default authentication protocol
- 2007, Kerberos Consortium (Sun, Apple, Google, Centrify, Microsoft, MIT, Stanford Univ and other founding sponsors)
- New Kerberos improvements until now (on going evolution)

Kerberos

- Presented as an Authentication service:
  - Designed for use in a distributed environment
  - In fact: an Authentication and Key Distribution Service for Distributed Applications (C/S Model)
- Usable as a SSO Approach for Client/Server applications
  - Generic solution (Single-Sign-On philosophy)
  - "Kerberized" applications
- Separation of Authentication concerns within the multiple entities involved:
  - Clients,
  - Servers (Services):
    - Authentication and Ticket Granting Services
  - Delegation to cross different authentication domains (Kerberos Realms)

System model overview

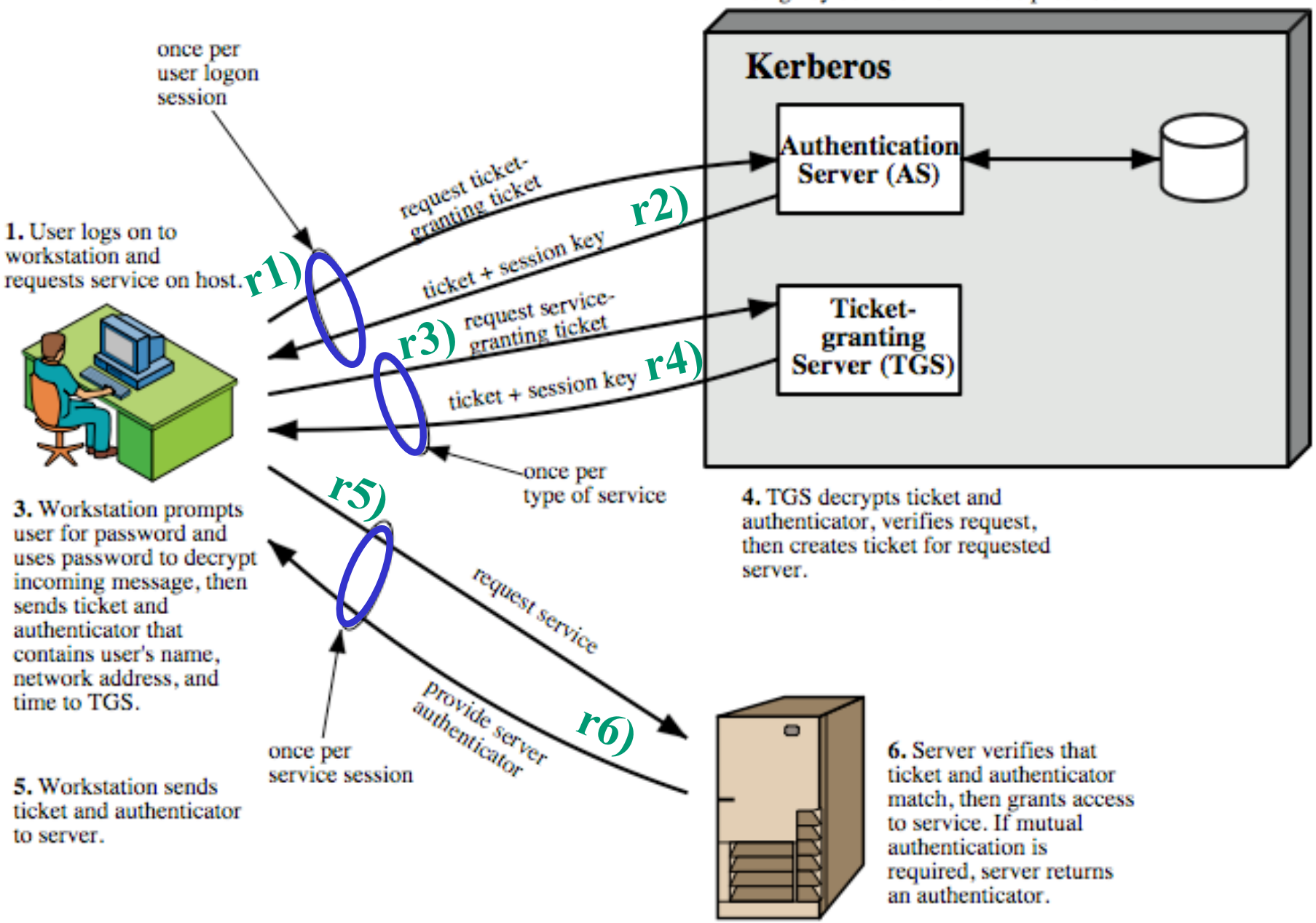
- Client-server model / mutual authentication
  - Requiring the user to prove her/his authenticated identity for each service invoked
  - Requiring that services prove their authenticated identity to clients
- Implementing an authentication protocol
  - Similar assumptions and principles as in the Neuman-Stubblebine and Needham-Schroeder protocol models (use of KDC and only symmetric cryptography model)
- Base architectural model (not one, but at least two "KDC" parts: AS + TGS):
  - **Authentication Server (AS)**
    - Users initially negotiate with AS to identify self
    - AS provides a non-corruptible authentication credential (ticket granting ticket or TGT)
  - **Ticket Granting server (TGS)**
    - Users subsequently request access to other services from TGS on basis of users TGT

Kerberos Requirements

Its first report identified requirements as:

- **Security**
  - Protection against eavesdroppers trying to impersonate users and services
- **Reliability**
  - To avoid a single point of failures/attacks
  - Reinforced for a distributed architecture
- **Transparent**
  - Transparent for users (similar to non-kerberized client applications and local logon procedures)
    - Password-based authentication in the base line
- **Scalable**
  - Support for a large number of clients and servers, in a distributed environment
    - Modular architecture, supporting possible different administrative distributed domains

System model and overview



Kerberos Dialogue and message exchanges

Rounds 1, 2: Authentication Service Exchange

Obtain ticket granting ticket from AS: ($\mathbf{Ticket}_{TGS}$ and $K_{c,tgs}$)

- Once per session (once per user authenticated logon session)

Rounds 3, 4: Ticket-Granting Service Exchange

Obtain service granting ticket from TGT: ($\mathbf{Ticket}_{v}$, $K_{c,v}$)

- For each distinct service required
- Once per final service

Rounds 5, 6 : Client/server authentication exchange

To obtain final service

- On every service request
- Once per specific service session

Kerberos Protocol (version 4)

Authentication Exhange: To obtain Ticket-Granting Ticket (TGT): $Ticket_{tgs}$

1) C → AS: $ID_c \,||\, ID_{tgs} \,||\, TS_1$

2) AS → C: $E\,(K_c\,,\,[\,K_{c,tgs}\,||\,ID_{tgs}\,||\,TS_2\,||\,Lifetime_2\,||\,Ticket_{tgs}\,])$

$Ticket_{tgs} = E\,(K_{tgs}\,,\,[\,K_{c,tgs}||\,Id_c\,||\,Ad_c||\,Id_{tgs}||\,TS_2||\,Lifetime_2\,])$

Ticket-Granting Exchange: To obtain Service-Granting Ticket (SGT): $Ticket_v$

3) C → TGS: $IDv\,||\,Ticket_{tgs}\,||\,Authenticator_c$

4) TGS → C: $E\,(K_{c,tgs}\,[\,K_{c,v}||\,ID_v\,||\,TS_4\,||\,Ticket_v\,])$

$Ticket_v = E\,(K_v\,,\,[\,K_{c,v}||\,Id_c\,||\,Ad_c||\,Id_v\,||\,TS_4\,||\,Lifetime_4\,])$

$Authenticator_c = E\,(K_{tgs}\,,\,[\,ID_c||\,Ad_c\,||\,TS_3\,])$

Client/Server Authentication Exhange: To obtain Service

5) C → V: $Ticket_v\,||\,Authenticator_c$

6) V → C: $E\,(K_{c,v}\,,\,[\,TS_5 + 1\,])$

V4 environmental shortcomings

- **Encryption system dependence (V4, only DES)**
  - Inclusion of Encryption Type Identifier
  - Encryption Keys tagged with type and length
- **Internet protocol dependence (V4, only IP)**
  - ISO network addresses, tagged with type and length
- **Message byte ordering, message representation types (V4: specific tags, specific implementation types)**
  - Lack of standardization for generic adoption (ASN.1 and BER)
- **Ticket lifetime and control (granularity issues)**
  - V4, 8 bits as units of 5 minutes
- **Authentication forwarding or delegation (no support)**
  - Forwarding client credentials from server to server, and other flexibility/adaptive issues
- **Scalable inter-domain authentication (no support)**

Other issues in Kerberos V4

- **Double Encryption (tickets encrypted twice)**
  - Messages 2 and 4, Second encryption not necessary
- **PCBC encryption mode**
  - Propagating Cipher Block Chaining
  - Not standard and vulnerable (security)
  - V5 uses CBC
- **Session keys**
  - Replaying messages from old sessions to the client and to the server
  - No rekeying possibility specified for each client/server connection
- **Password Based Attacks**

Kerberos Version 5

Developed in mid 1990's

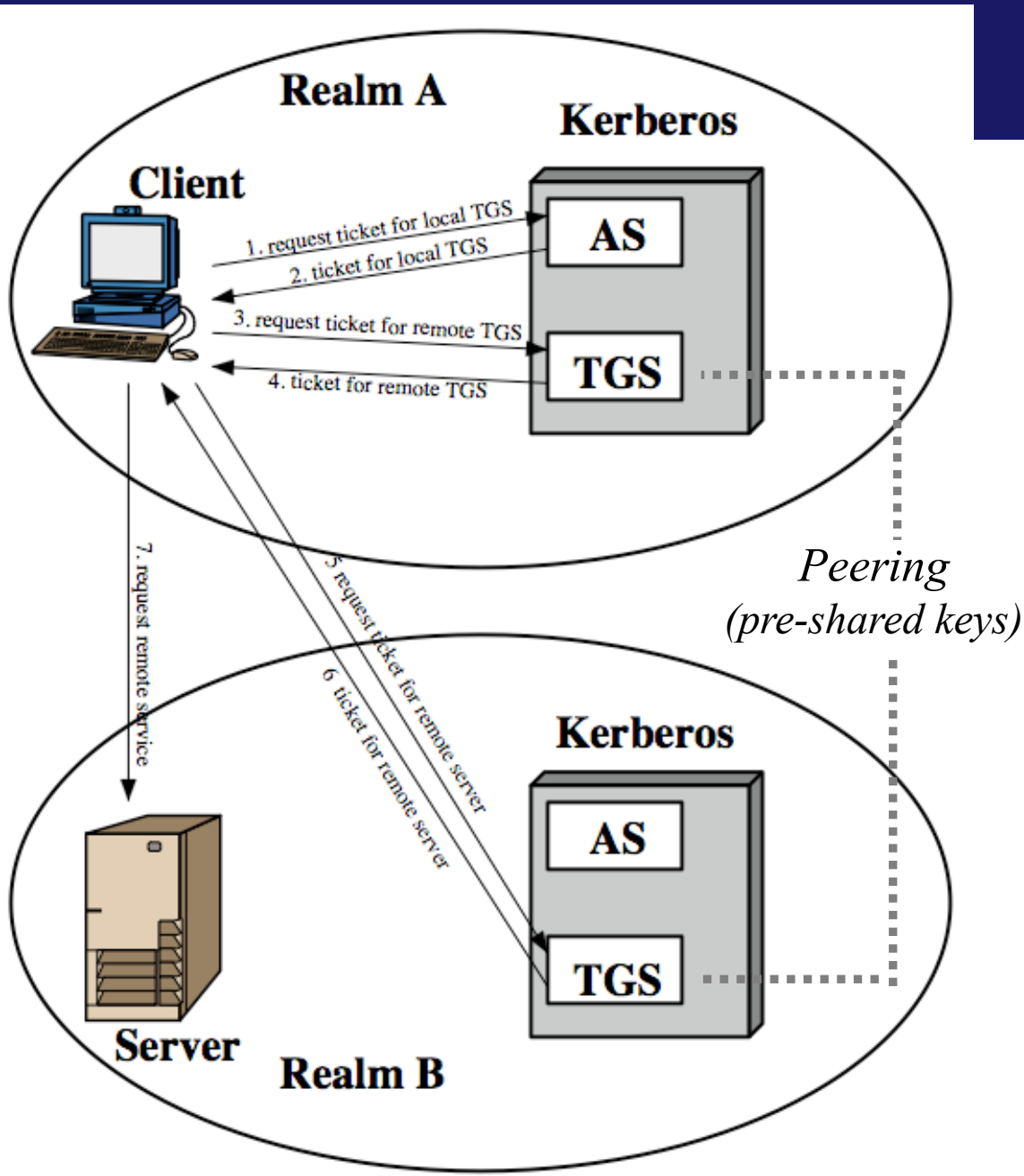
- Specified as Internet standard RFC 1510
- Provides security and performance improvements over v4
  - Solution for previous security concerns and issues
  - Addresses environmental shortcomings
    - encryption alg, network protocol, byte order, ticket lifetime, authentication forwarding, inter-realm auth
  - And overcoming some technical deficiencies, ex:
    - Double encryption, non-std mode of use, different ciphersuites and session keys, support for different network protocols, mitigation of password attacks

Kerberos Realms (introduced in V5)

- A **Kerberos Realm** in V5 environment consists of:
  - A Kerberos server (AS + TGS)
  - A certain number of clients, all registered with AS
  - Application servers, sharing keys with TGS
- A Kerberos realm is typically a single administrative domain
- To support multiple realms, their Kerberos servers must share keys and trust
  - TGS in one realm can issue TGS tickets to remote TGS in another realm
  - TGS Peering model
  - Delegation model
    - AS authenticated clients in one realm can use remote servers in another realm
    - TGS tickets issued for other TGS (in other realms)

Request for Service in Another Realm:

Multi-Realm Environment



Kerberos protocol (version 5)

(1) C → AS Options ‖ ID_c ‖ $Realm_c$ ‖ ID_{tgs} ‖ $Times$ ‖ $Nonce_1$

(2) AS → C $Realm_c$ ‖ ID_C ‖ $Ticket_{tgs}$ ‖ $E(K_c, [K_{c,tgs} \| Times \| Nonce_1 \| Realm_{tgs} \| ID_{tgs}])$

$$Ticket_{tgs} = E(K_{tgs}, [Flags \| K_{c,tgs} \| Realm_c \| ID_C \| AD_C \| Times])$$

(a) Authentication Service Exchange to obtain ticket-granting ticket

(3) C → TGS Options ‖ ID_v ‖ $Times$ ‖ ‖ $Nonce_2$ ‖ $Ticket_{tgs}$ ‖ $Authenticator_c$

(4) TGS → C $Realm_c$ ‖ ID_C ‖ $Ticket_v$ ‖ $E(K_{c,tgs}, [K_{c,v} \| Times \| Nonce_2 \| Realm_v \| ID_v])$

$$Ticket_{tgs} = E(K_{tgs}, [Flags \| K_{c,tgs} \| Realm_c \| ID_C \| AD_C \| Times])$$

$$Ticket_v = E(K_v, [Flags \| K_{c,v} \| Realm_c \| ID_C \| AD_C \| Times])$$

$$Authenticator_c = E(K_{c,tgs}, [ID_C \| Realm_c \| TS_1])$$

(b) Ticket-Granting Service Exchange to obtain service-granting ticket

(5) C → V Options ‖ $Ticket_v$ ‖ $Authenticator_c$

(6) V → C $E_{K_{c,v}}$ [TS_2 ‖ Subkey ‖ Seq#]

$$Ticket_v = E(K_v, [Flags \| K_{c,v} \| Realm_c \| ID_C \| AD_C \| Times])$$

$$Authenticator_c = E(K_{c,v}, [ID_C \| Realm_c \| TS_2 \| Subkey \| Seq\#])$$

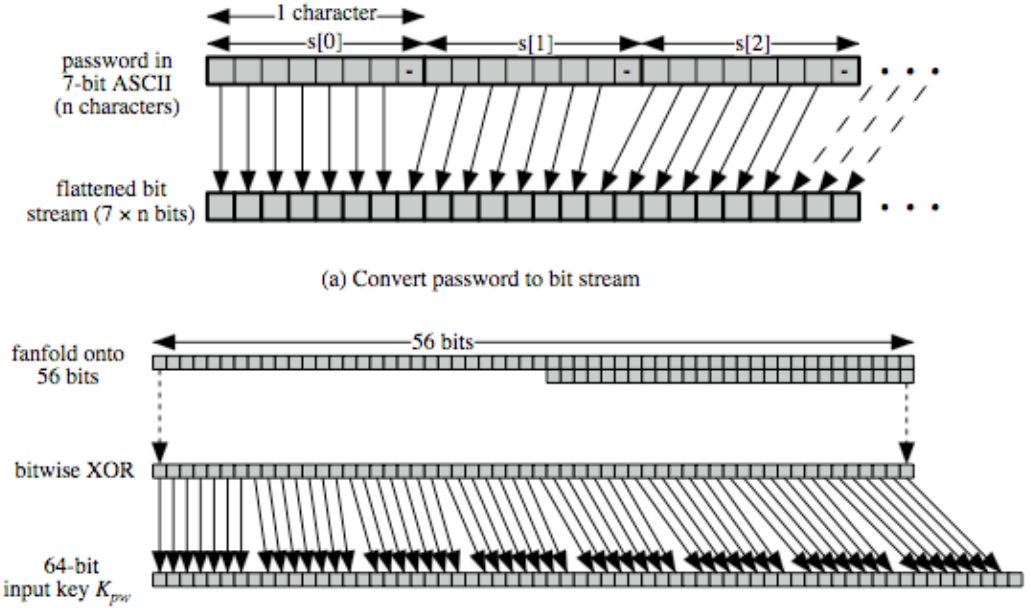
(c) Client/Server Authentication Exchange to obtain service

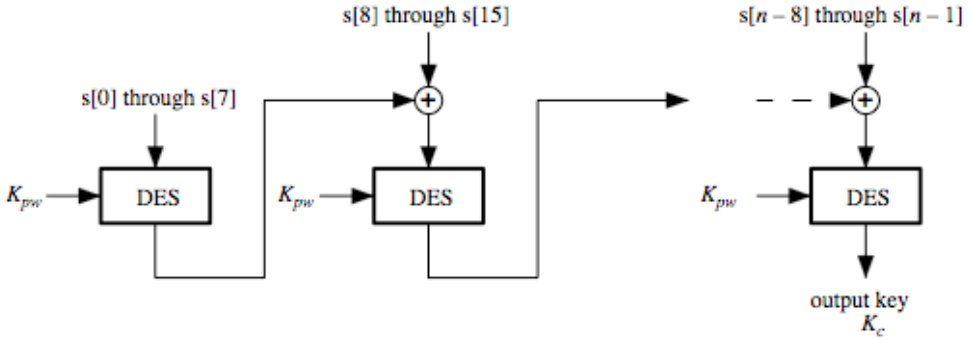
Kerberos V5 flags

INITIAL	This ticket was issued using the AS protocol and not issued based on a ticket-granting ticket.
PRE-AUTHENT	During initial authentication, the client was authenticated by the KDC before a ticket was issued.
HW-AUTHENT	The protocol employed for initial authentication required the use of hardware expected to be possessed solely by the named client.
RENEWABLE	Tells TGS that this ticket can be used to obtain a replacement ticket that expires at a later date.
MAY-POSTDATE	Tells TGS that a postdated ticket may be issued based on this ticket-granting ticket.
POSTDATED	Indicates that this ticket has been postdated; the end server can check the authtime field to see when the original authentication occurred.
INVALID	This ticket is invalid and must be validated by the KDC before use.
PROXIABLE	Tells TGS that a new service-granting ticket with a different network address may be issued based on the presented ticket.
PROXY	Indicates that this ticket is a proxy.
FORWARDABLE	Tells TGS that a new ticket-granting ticket with a different network address may be issued based on this ticket-granting ticket.
FORWARDED	Indicates that this ticket has either been forwarded or was issued based on authentication involving a forwarded ticket-granting ticket.

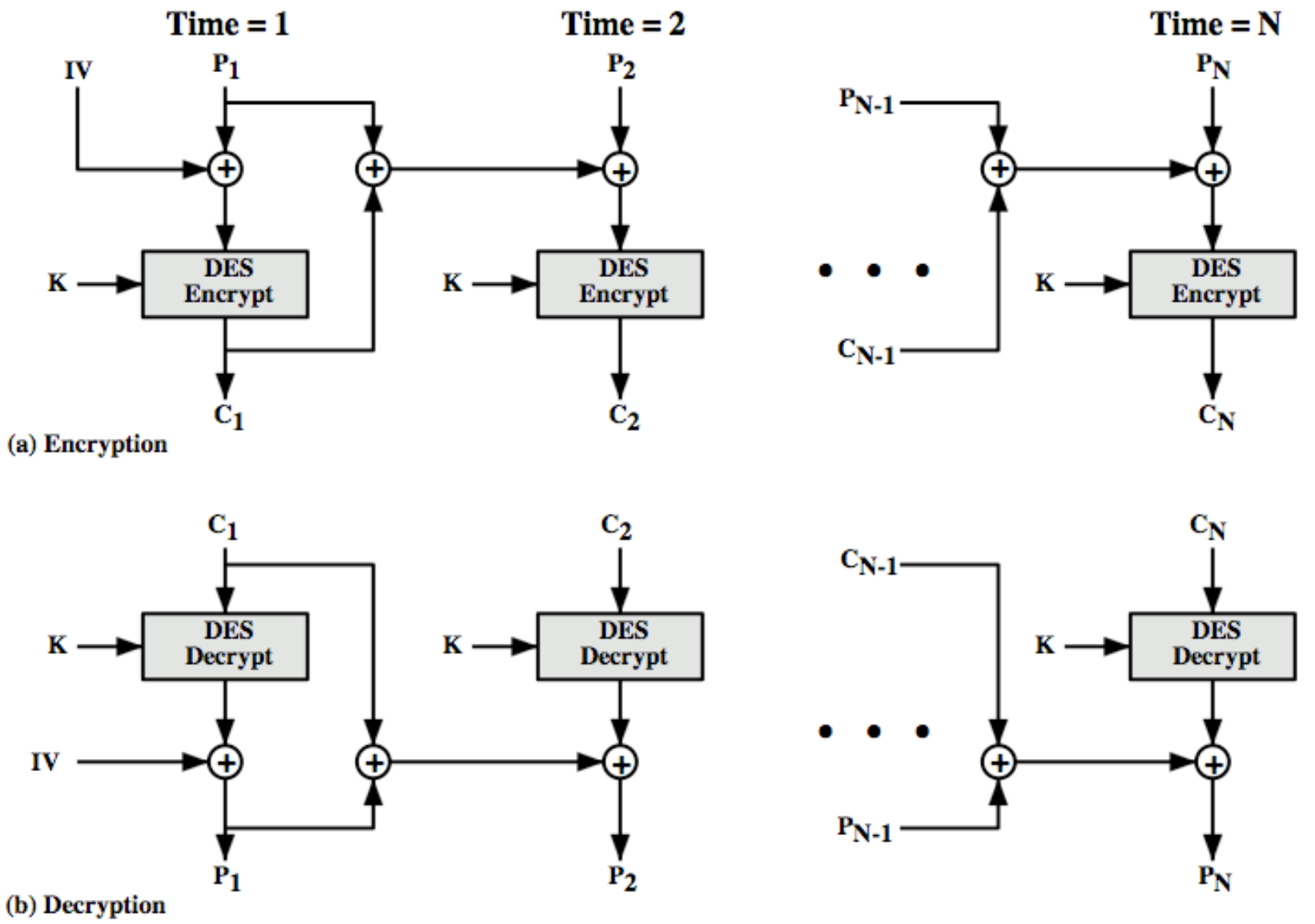
Kerberos Encryption Techniques

DES-CBC Checksum of Encryption Scheme to generate Encryption Key from the Password





PCBC Mode


Time = 1
Time = 2
Time = N
IV
P_1
P_2
P_{N-1}
P_N
K
DES Encrypt
C_1
C_2
C_{N-1}
C_N
(a) Encryption
DES Decrypt
(b) Decryption

Kerberos and its use (summary)

Two Kerberos versions:

- V4 : restricted to a single realm
- V5 : allows inter-realm authentication

- Kerberos v5: an Internet standard (initially in RFC1510, and used by many utilities that also evolved the base model)

Use of Kerberos (Practical evolved implementations and variants):

- Based on a KDC solution (divided in AS and TGS + Kerberos Realms)
- Need to have *Kerberized* applications running on all participating systems
- Problems: Password-Attacks (Key-Generation process)
- Enhancements with different variants for key-generation and use of other ciphersuites: symmetric cryptographic algorithms + Secure Hash functions + MACs or CMACs
- Some new defined variants (ex., PKINIT Kerberos)

  Combined use of Asymmetric Cryptography, Digital Signatures and Certified Public Keys

Recommended Reading (with the slides)

Readings:

W. Stallings, Network Security Essentials – Applications and Standards, Part II – Network Security Applications, Chap.4 – Key Distribution and User Authentication