Block III. Internet security

Authentication, key management and access control

Network Security

Maria Dolores Cano Banos

Contents

Introduction

- Attacks:
  - Revelation
  - Traffic analysis
  - Masking
  - Content modification
  - Sequence modification
  - Temporary modification
  - Repudiation

Encryption

AUTHENTICATION

DIGITAL SIGNATURE

- Authentication or digital signature
  - 1) Authenticator
  - 2) Authentication protocol

Contents

HASH and MAC functions

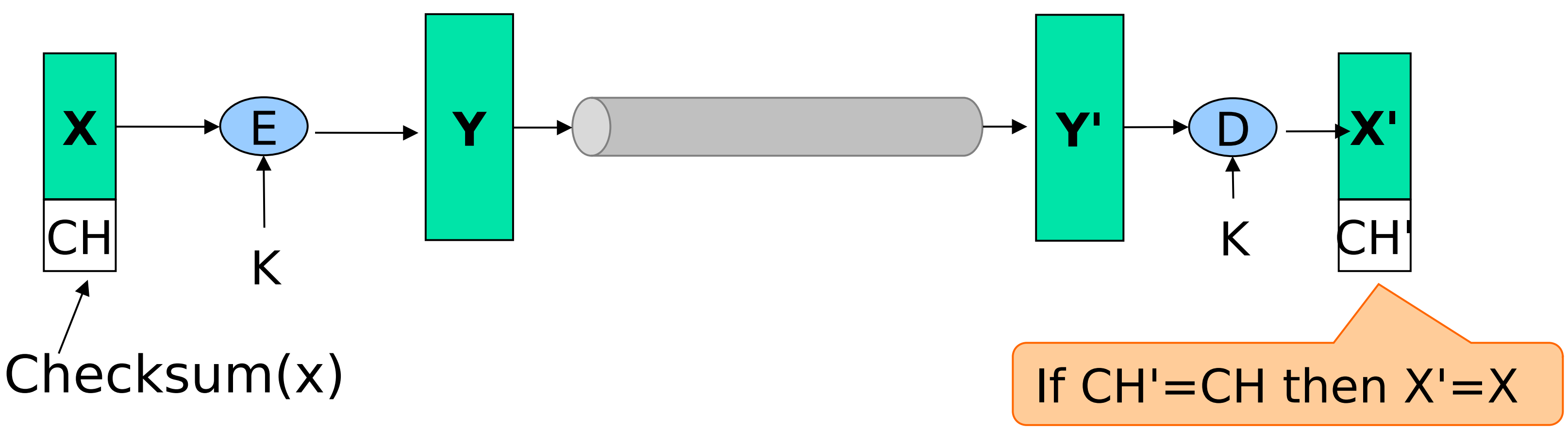
- Functions to generate an **authenticator**:
  - Message encryption
  - Message authentication code
  - Hash function

HASH and MAC functions

- **Message encryption:**

SYMMETRICAL ENCRYPTION

  - Confidentiality *
  - Authentication *
  - How to guarantee that the plain text is the original?

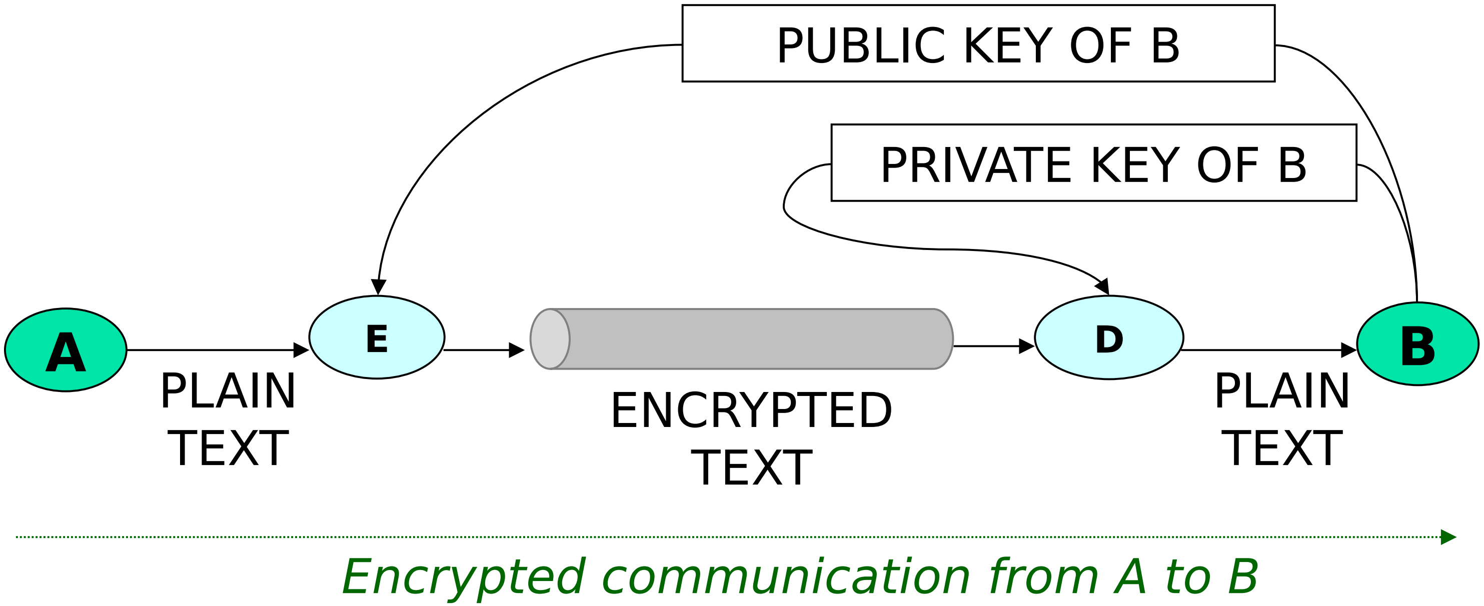


HASH and MAC functions

- **Message encryption:**

ASYMMETRIC
ENCRYPTION

  - Confidentiality *
  - ~~Authentication~~

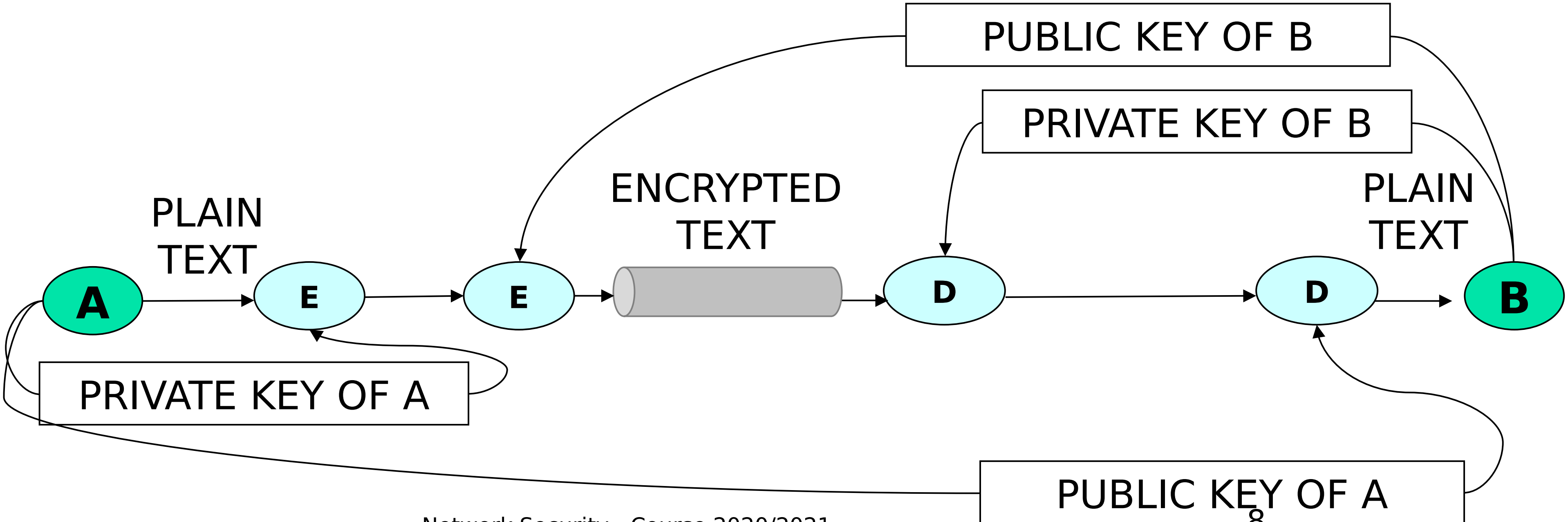


HASH and MAC functions

- **Message encryption:**

ASYMMETRIC ENCRYPTION

  - Confidentiality *
  - Authentication *
  - How to guarantee that the plain text is the original? Checksum



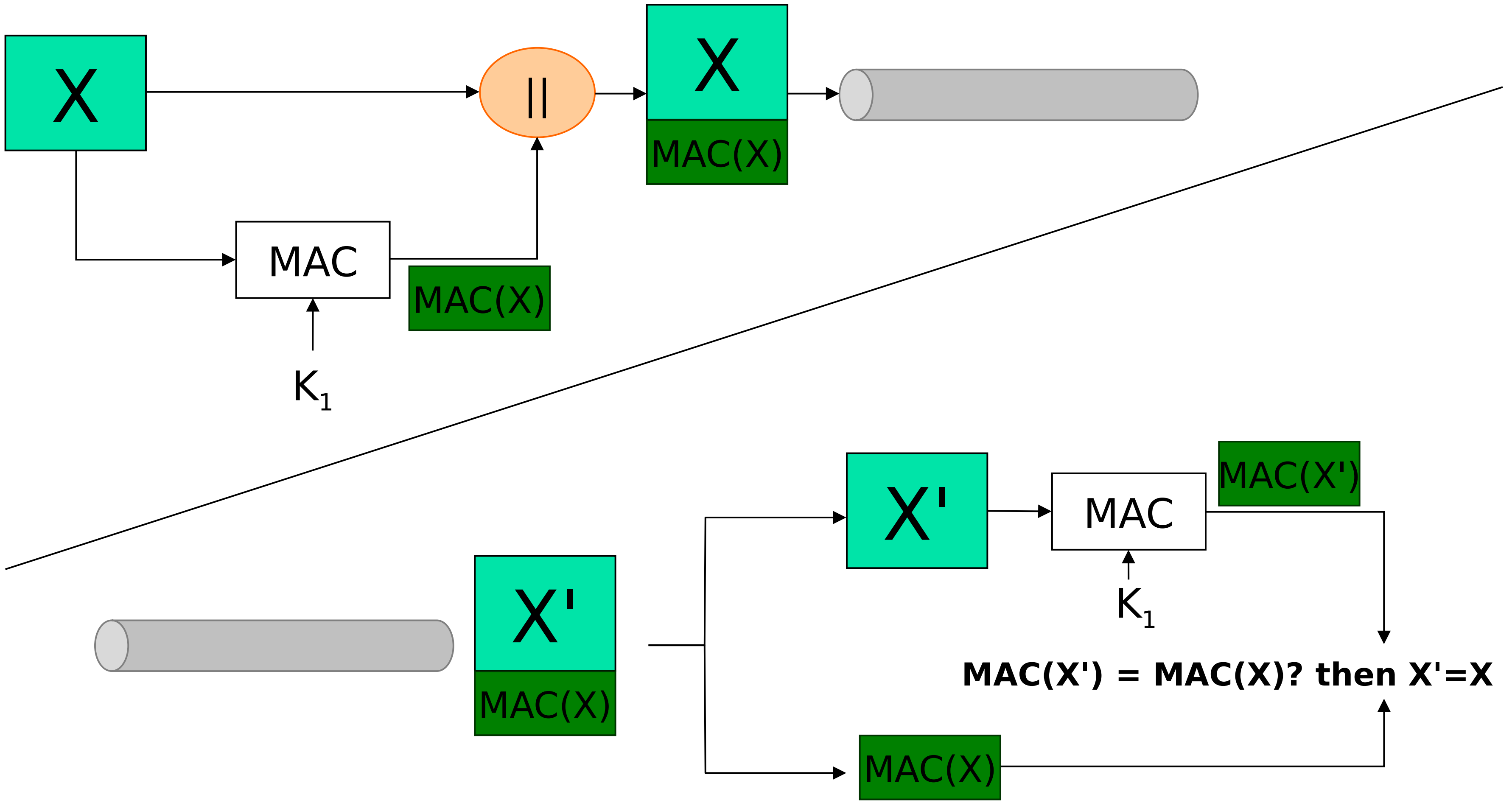
HASH and MAC functions

- **Message authentication code** (Message Authentication Code, MAC)
  - Private key
  - Fixed size data block ≡ MAC
  - It is added to the message

AUTHENTICATION

- The message was not altered
- Message comes from the true sender
- If sequence numbers are included, temporary modification by an attacker is not possible
- For there to be confidentiality, you have to encrypt plain text + MAC

HASH and MAC functions



HASH and MAC functions

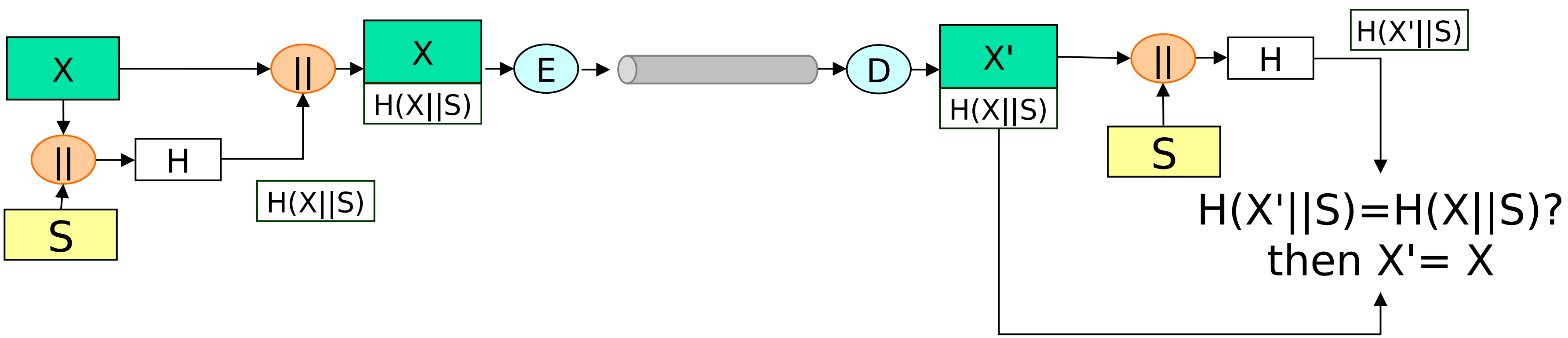
- MAC code = $C_k(X)$, if an attacker knows function C but does not know key k:
  - Computationally intractable to create a message X' such that $C_k(X') = C_k(X)$
  - For two randomly selected messages X and X 'the probability that $C_k(X') = C_k(X)$ is 2^{-n} where n is the length of the MAC
  - Let X '= f (X) be the probability that $C_k(X') = C_k(X)$ is 2^{-n} where n is the length of the MAC

HASH and MAC functions

- **Hash function**, accepts a variable size X message and generates a fixed size hash code H(X) ≡ Message Digest
  - H(X) offers error detection
- Authentication possibilities
  1. Plain text is concatenated with hash code and everything is encrypted with symmetric encryption
  2. Only the hash code is encrypted with symmetric encryption
  3. Only hash code is encrypted with asymmetric encryption (with issuer private key) ㄹㅁ digital signature
  4. Hashes is encrypted with asymmetric encryption (with issuer private key) and everything is encrypted with symmetric encryption

HASH and MAC functions

- Authentication possibilities
  5. Two communicators share a secret value S.



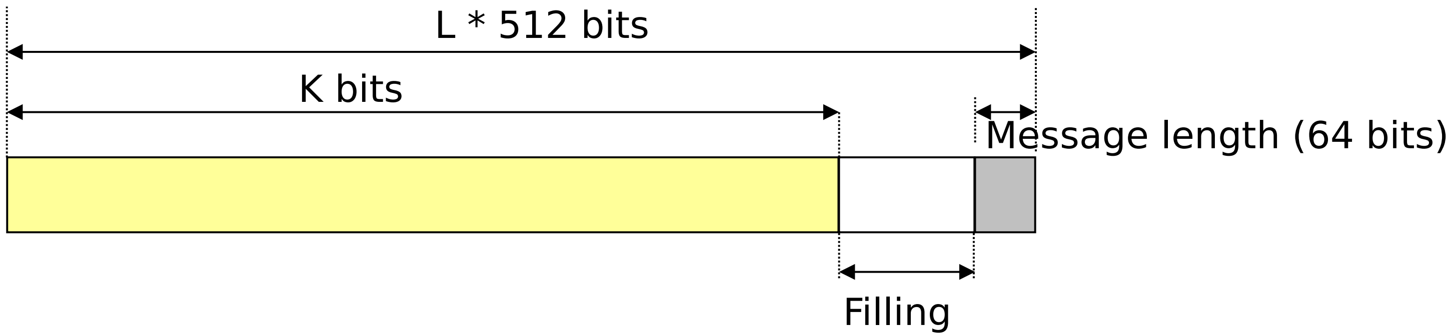
HASH and MAC functions

- Hash code h=H(X)
  - H can be used with data blocks of any size
  - H generates fixed size output
  - H(X) is easy to obtain
  - Known h, it is computationally intractable to find X such that H(X)=h
  - It is computationally intractable to find X' such that X'≠X and H(X')=H(X)
  - It is computationally intractable to find a pair (X, X') such that H(X)=H(X')

Contents

MD5

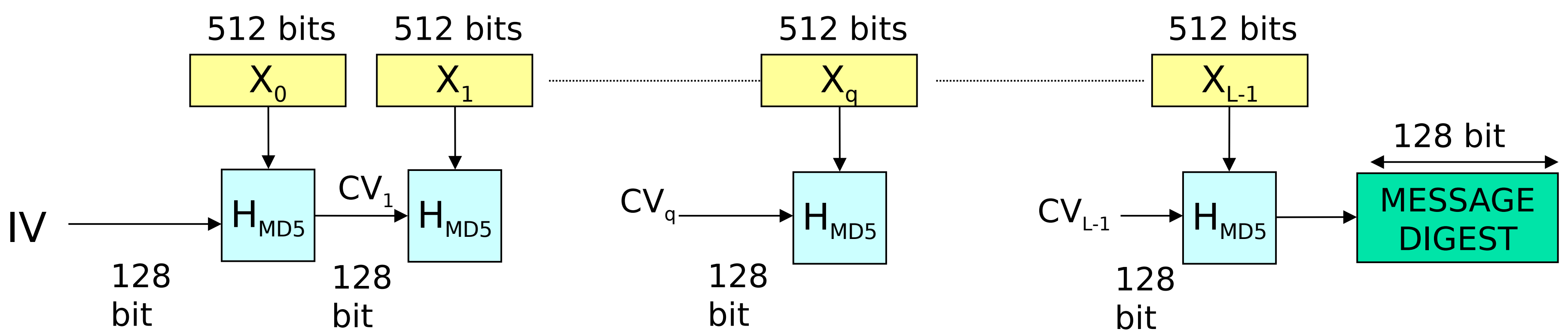
- Algorithm **Message Digest MD5**, variable-length input (in 512-bit blocks) and generates 128-bit output (message digest)
- Algorithm:
  1. Add padding bits (final length = 448 mod 512)
  2. Add original message length (64 bit)



MD5

Plain text in blocks $X_0, X_1, \ldots, X_{L-1}$ -> message length 512 * L bits = 16 * 32 * L bits = N 32-bit words (N = 16 * L)

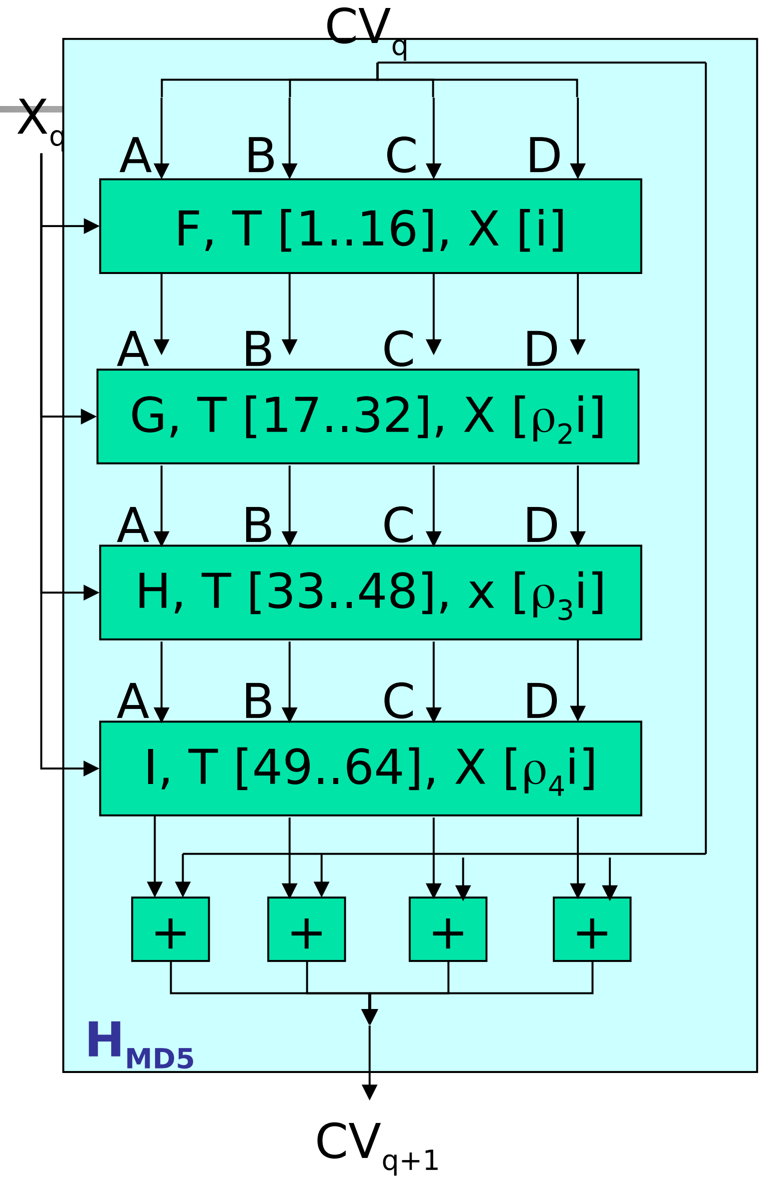
3. Initialize buffer (128 bits) = (A, B, C, D)
   1. A = 67452301; B = EFCDAB89; C = 98BADCFE; D = 10325476
4. Process plain text in 512-bit blocks: compress the message by applying H_{MD5}



5. The output of stage L is the Message Digest

MD5

- Blocks H_{MD5}
  - Four rounds each with a different logic function (F, G, H, I)
  - Round Entry: X_q; 128 bits of buffer; table T content
  - T [1..64] has 64 inputs (32 bits) obtained from the sine function: T[i]= integer part ($2^{32} \cdot$ abs (sin (i))) where i are radians



MD5

- Blocks H_{MD5}
  - Each round 16 operations
  - Operation: b ← b+((a+g(b, c, d)+X[k]+T[i]) <<< s
  - Where:
    - a, b, c, d are the four words of the buffer
    - g is one of the functions F, G, H, I
    - <<< s left circular shift of s bits
    - T [i] i-th word of T

Round	G function	G (b, c, d)
1	F (b, c, d)	(b AND c) OR (b AND d)
2	G (b, c, d)	(b AND d) OR (c AND d)
3	H (b, c, d)	b XOR c XOR d
4	I (b, c, d)	c XOR (b OR d)

MD5

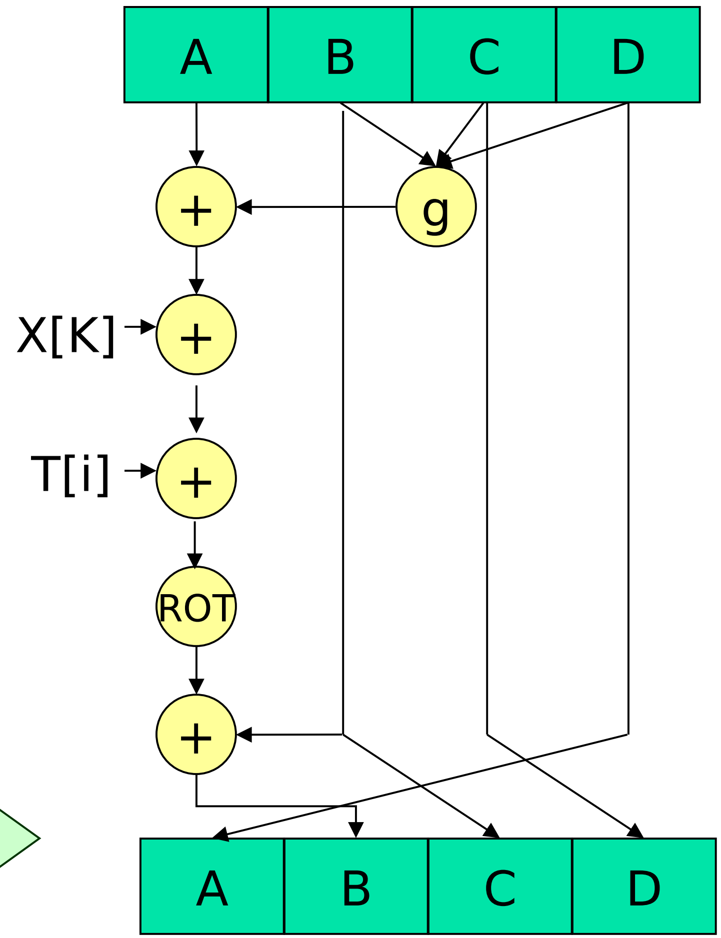
- blocks H_{MD5}
- X[k] = K-th word of block X_q

  In rounds 2, 3 and 4 ⇒

  $\rho_2(i) = (1+5i) \bmod 16$

  $\rho_3(i) = (5+3i) \bmod 16$

  $\rho_4(i) = (1+5i) \bmod 16$



Example of an iteration within a round

MD5

- Every bit of *message digest* is a function of all input bits
- Probability of finding two messages that generate the same *message digest* is of the order of 2^{64} operations
- Probability of finding a known message the message digest is of the order of 2^{128} operations
- New proposals: SHA-1 and RIPEMD-168

Contents

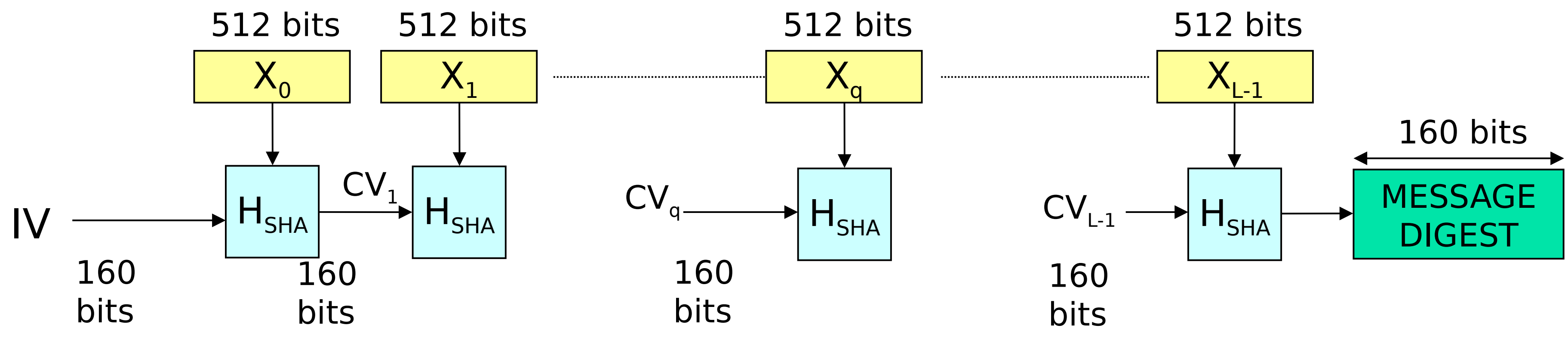
SHA

- SHA (Secure HAsh)
- Based on the MD4
- Entry ≡ maximum message length $<2^{64}$ bits (processed in 512-bit blocks)
- Message disgest ≡ 160 bits
- Overall structure similar to MD5

SHA

- Algorithm:
  - Add padding bits (final length 448 mod 512)
  - Add original message length (64 bit)
  - Initialize buffer (160 bits) = (A, B, C, D, E)

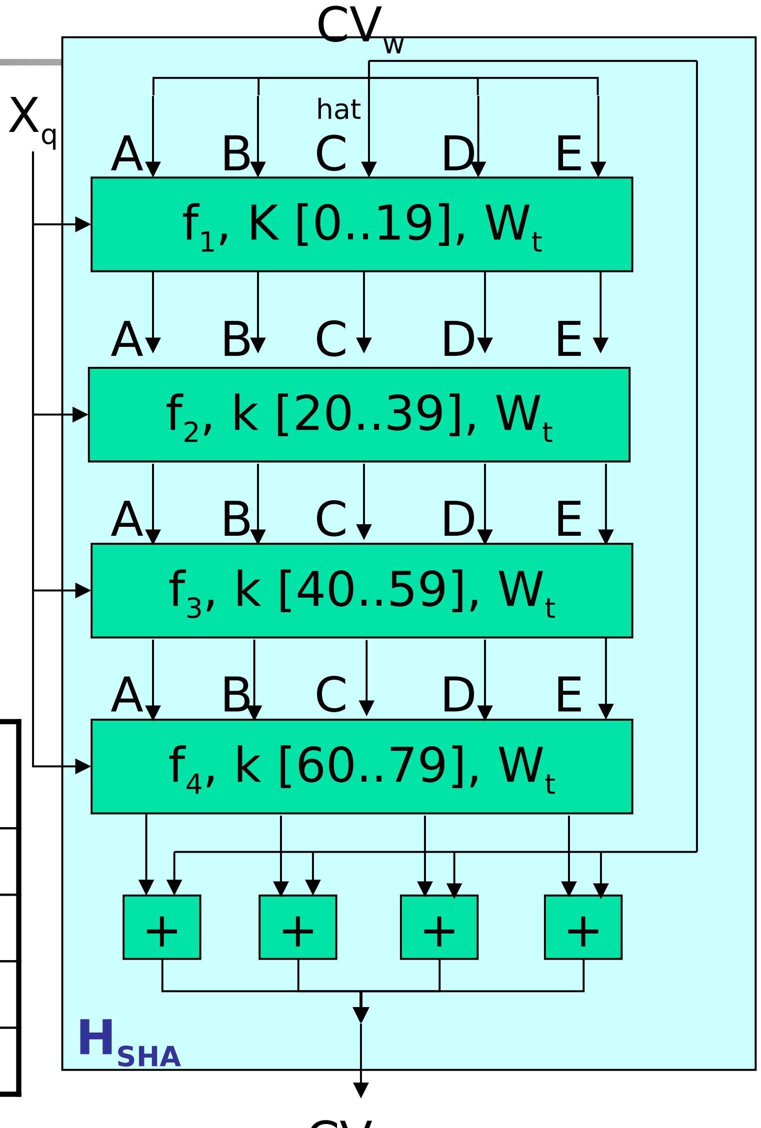
  A = 67452301; B = EFCDAB89; C = 98BADCFE; D = 10325476; E = C3D2E1F0
  - Process 512-bit blocks through H modules$_{SHA}$



SHA

- blocks H_{SHA}
  - Four rounds each with a different logic function (f_1, f_2, f_3, f_4)
  - Each round 20 operations
  - Constant K is used in each round$_t$ ($0 \le t < 80$)

Operation #	Hexadecimal	Take full part of
$0 \le t < 20$	K_t= 5A827999	$2^{30}\sqrt{2}$
$20 \le t < 40$	K_t= 6ED9EBA1	$2^{30}\sqrt{3}$
$40 \le t < 60$	K_t= 8F1BBCDC	$2^{30}\sqrt{5}$
$60 \le t < 80$	K_t= CA62C1D6	$2^{30}\sqrt{10}$



SHA

- Blocks H_{SHA}
  - Operation: $A \leftarrow (E + f(t, B, C, D) + S^5(A) + W_t + K_t$

A, B, C, D, E 32-bit words in buffer

- $t \equiv$ operation number
- $f(t, B, C, D) \equiv$ logical function of the operation t
- $S^k \equiv$ circular shift left k bits
- $W_t \equiv$ 32-bit word derived from 512-bit input

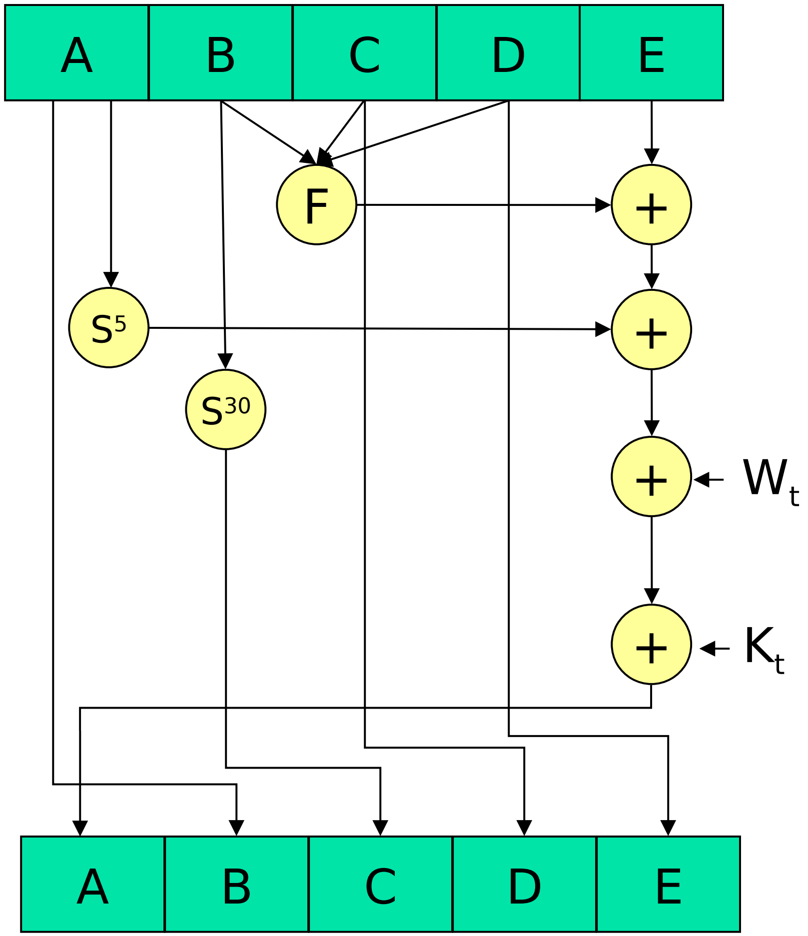
$W_t = S^1 (W_{t-16} \oplus W_{t-14} \oplus W_{t-8} \oplus W_{t-3})$

- $K_t \equiv$ additive constant

Operation #	**Function f**	**Value**
$0 \leq t < 20$	$F_1 = f(t, B, C, D)$	(B AND C) OR (B AND D)
$20 \leq t < 40$	$F_2 = f(t, B, C, D)$	B XOR C XOR D
$40 \leq t < 60$	$F_3 = f(t, B, C, D)$	(B AND C) OR (B AND D) OR (C AND D)
$60 \leq t < 80$	$F_4 = f(t, B, C, D)$	B XOR C XOR D

SHA

Example of an operation within a round



SHA

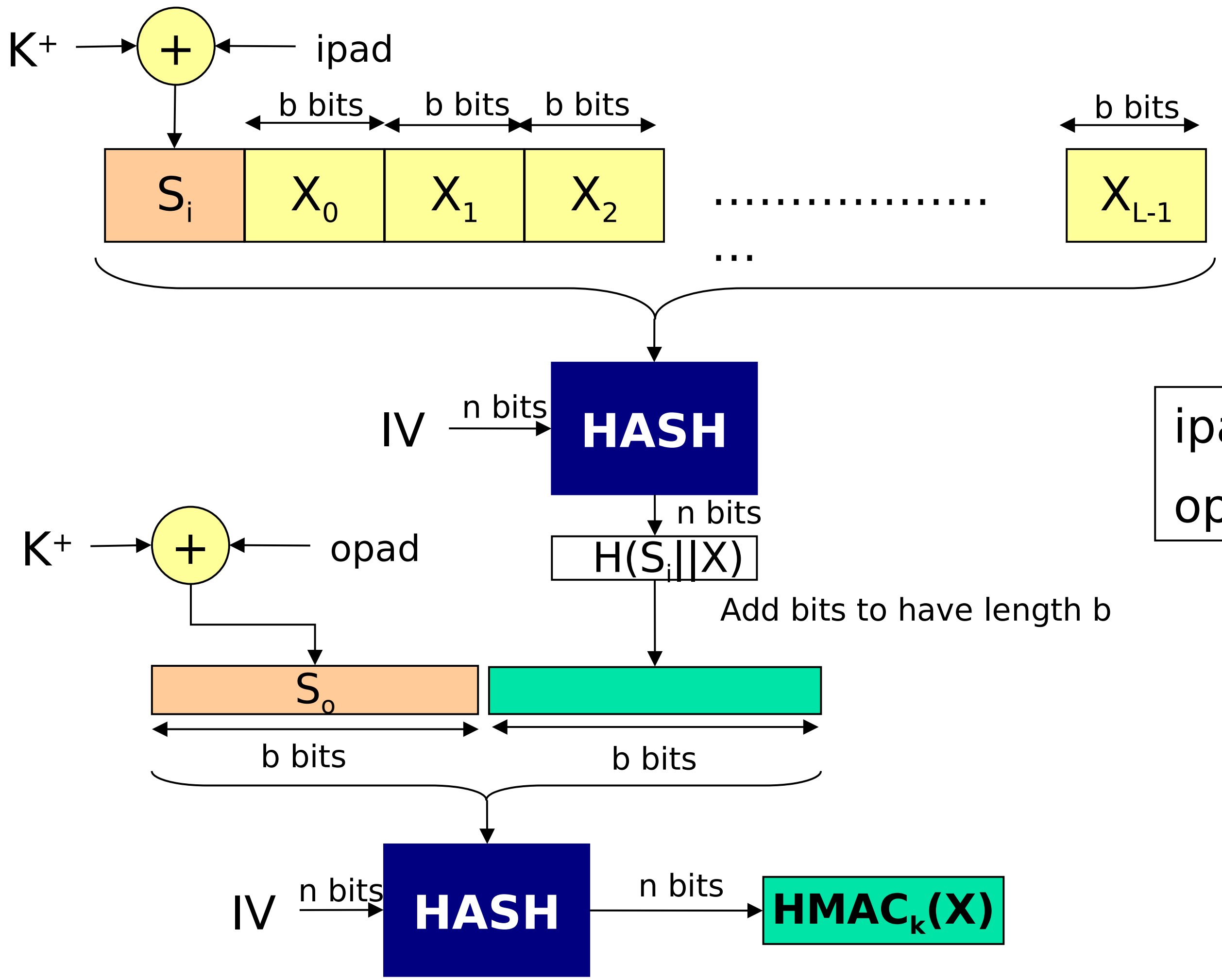
- SHA vs. MD5
  - Probability of finding two messages that generate the same *message digest* is of the order of 2^{80} operations
  - Probability of finding a known message the message digest is of the order of 2^{160}
  - There are no known cryptanalysis attacks on SHA
  - Both work fine on 32-bit architectures, SHA slower on same hardware
  - Simple

Contents

HMAC

- Develop a MAC from a hash code
- HMAC (RFC 2104)
- The hash function is considered a black box
- Algorithm:
  1. Add leading zeros of k to create bit string of length b $\rightarrow K^+$
  2. (K^+) XOR (ipad) $\rightarrow$ block S_i of b bits
  3. Concatenate M to S_i
  4. Apply hash function H to the stream generated in 3.
  5. (K^+) XOR (opad) $\rightarrow$ block S_{or} of b bits
  6. Concatenate hash code of 4 with S_{or}
  7. Apply hash function H to the stream generated in 6 $\rightarrow$ HMAC result

HMAC



ipad ≡ 00110110

opad ≡ 01011010

Contents

Kerberos

- Project Athena, Massachusetts Institute of Technology (MIT)
- Problem: open distributed environment where workstation users access network distributed server services
- Threats:
  - Impersonate identity
  - Alter network address
  - Listen and do *replay*

Kerberos

- Kerberos provides a centralized authentication server (versions 4 and 5)
- Use only symmetric encryption!
- Requirements
  - <u>insurance</u>, a listening user cannot get information to impersonate another
  - <u>Reliable</u>, an unavailability of Kerberos means an unavailability for all services that rely on it
  - <u>Transparent</u>, user is not aware of authentication beyond being prompted for a key
  - <u>Scalable</u>, capable of supporting a large number of clients and servers

Kerberos

ARCHITECTURE

- Realm- Management domain (up to 100,000)
- Model client/server
- Principals- These are the users, clients, and network services running on specific systems.
  - Principal identifier (40 characters max.): Principal name, realization name (system on which the service is provided, role of user, etc.) and realm name (Internet domain name in uppercase)
- Key distribution center (KDC): it consists of Authentication Server (AS) and Ticket Granting Servers (TGS)

Kerberos

ARCHITECTURE

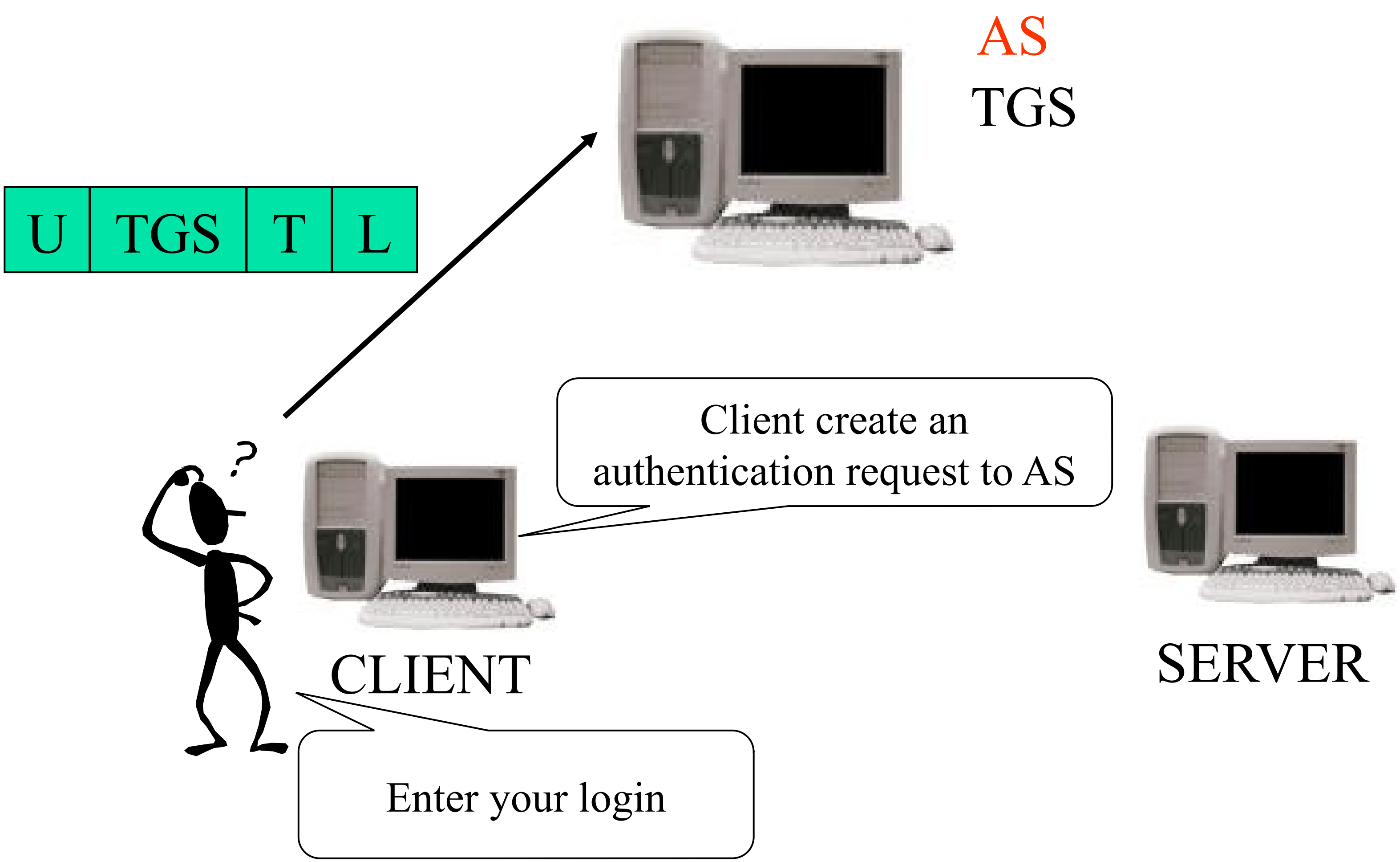
- KDC maintains a database with one entry for each principal registered in the realm.
- For each principal:
  - Principal identifier
  - Master key K for principal (or your key if you are a user)
  - Identity expiration date
  - Date of last modification of the record
  - Identity of the person who last modified the record
  - Maximum lifetime of the tickets supplied by the principal
  - Attributes
  - Implementation data

Encrypted with K_{KDC}

Kerberos

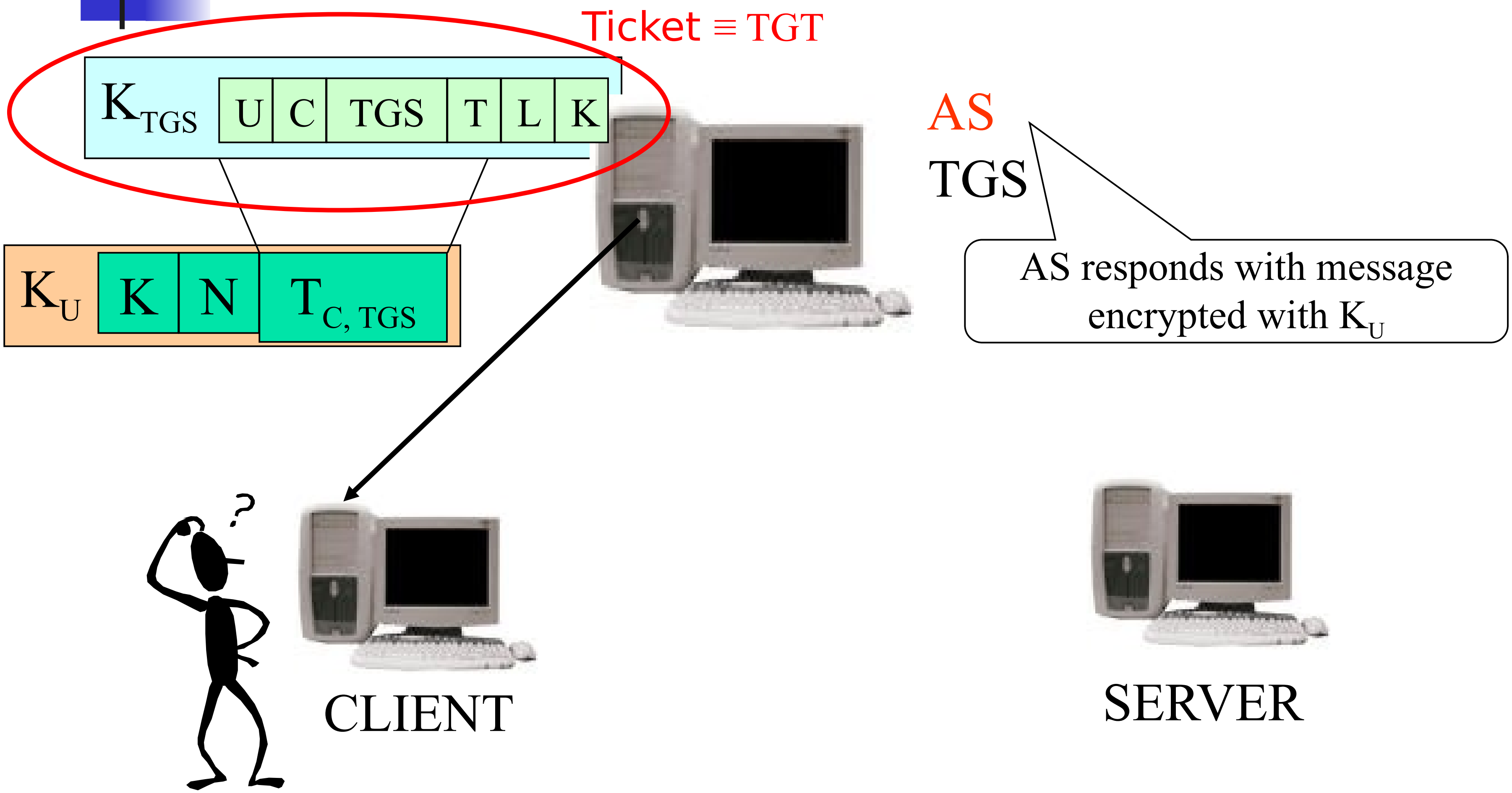

Authentication Server (AS)
Ticket-Granting Server (TGS)
CLIENT
USER
SERVER

Kerberos



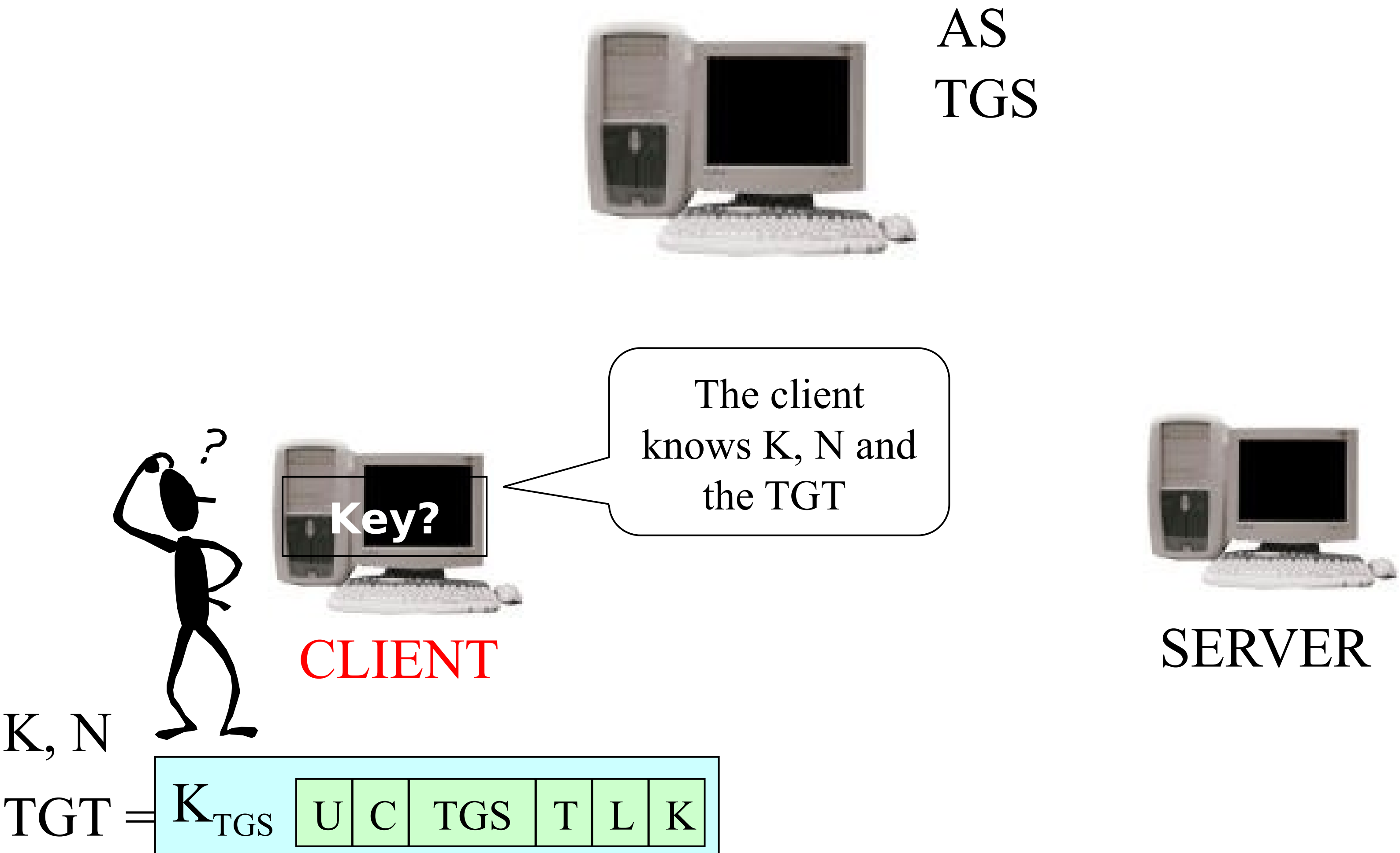
AS
TGS
U | TGS | T | L
Client create an authentication request to AS
CLIENT
SERVER
Enter your login

Kerberos



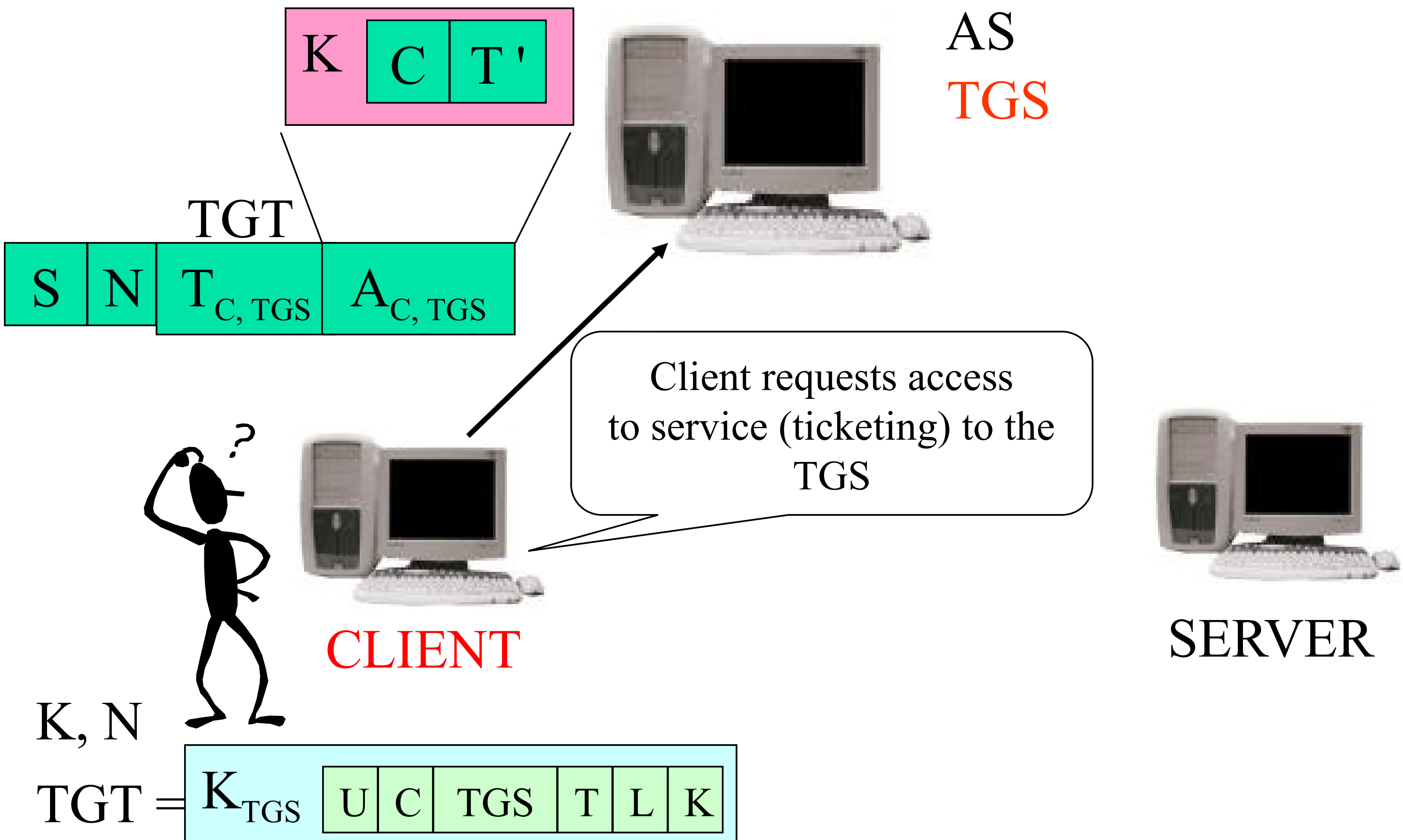
Ticket ≡ TGT
K_{TGS}
U
C
TGS
T
L
K
K_U
K
N
$T_{C, TGS}$
AS
TGS
AS responds with message encrypted with K_U
CLIENT
SERVER

Kerberos

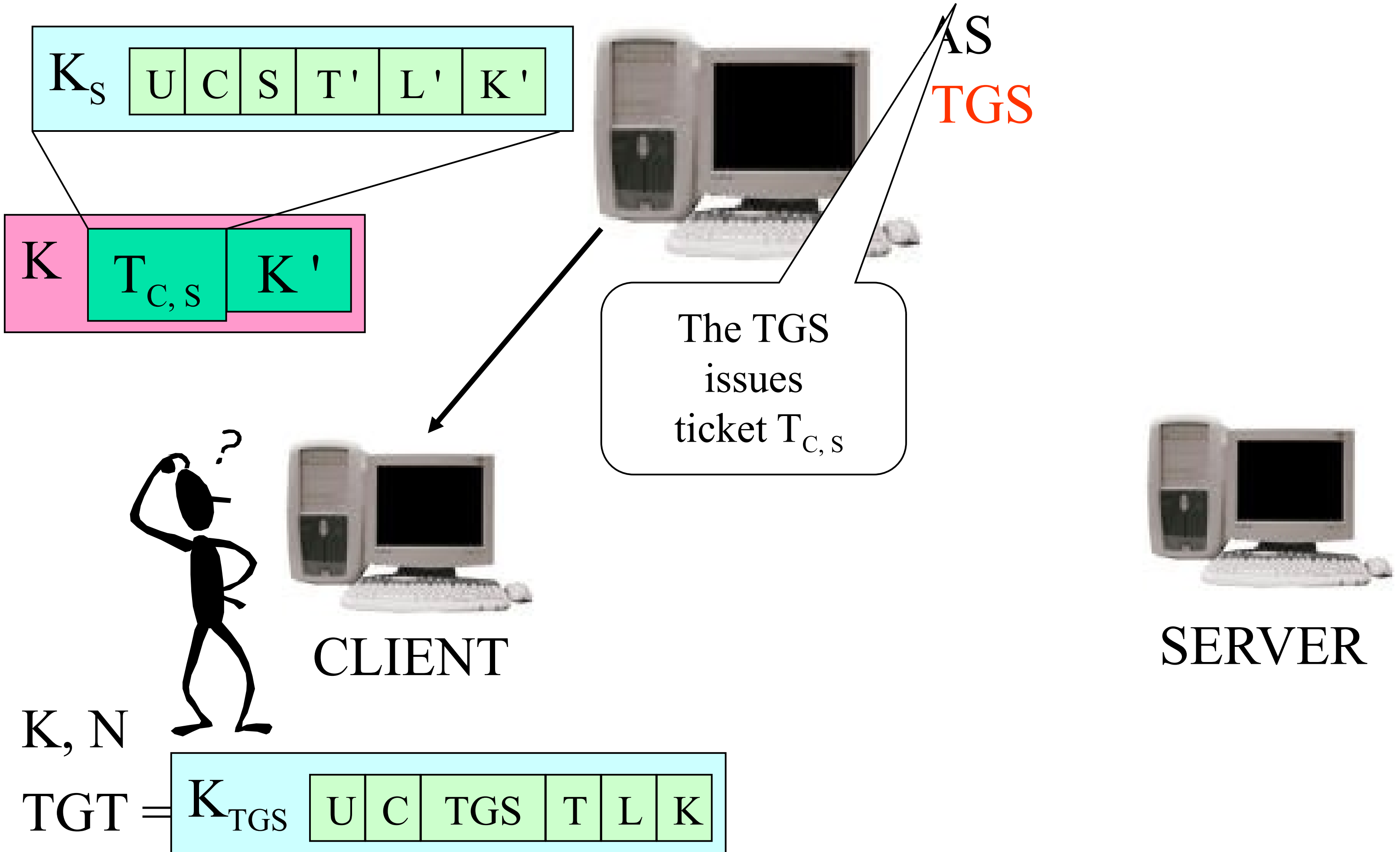


AS
TGS
Key?
The client knows K, N and the TGT
CLIENT
SERVER
K, N
TGT = K_{TGS} U C TGS T L K

Kerberos

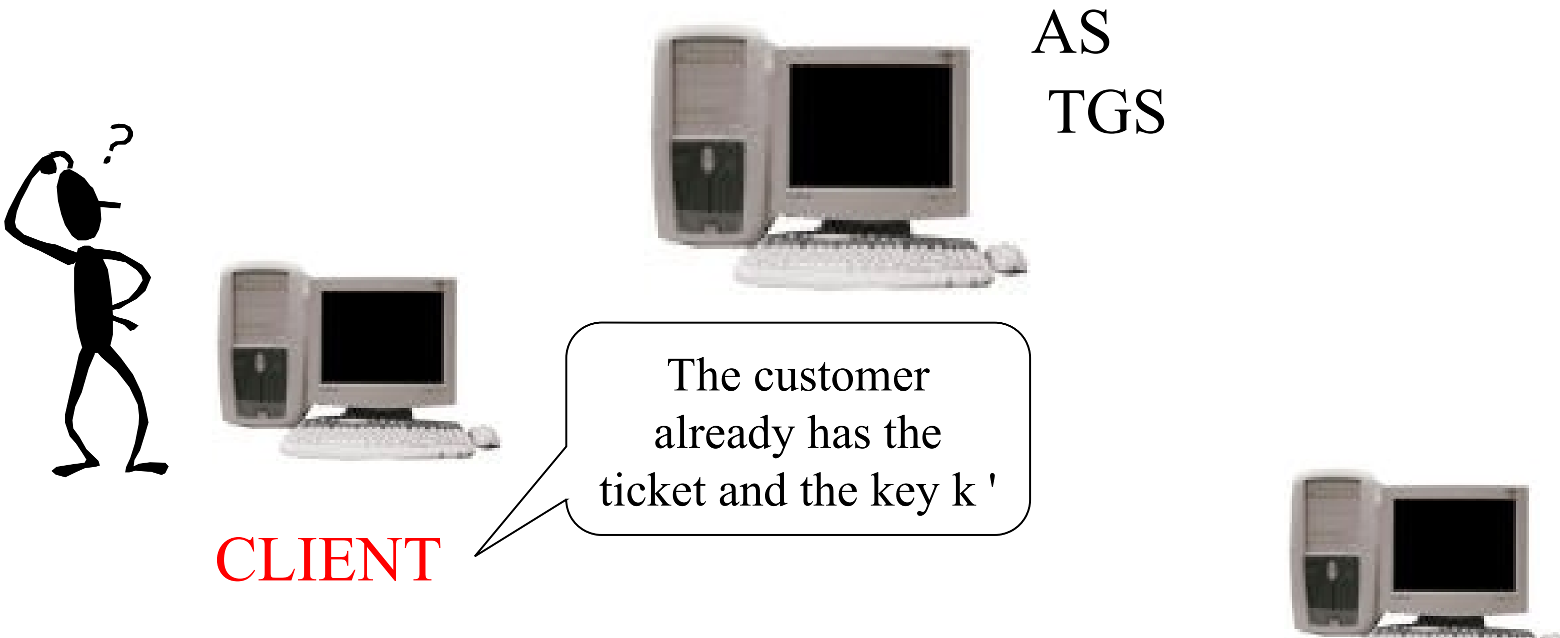

K
C
T '
AS
TGS
TGT
S
N
$T_{C, TGS}$
$A_{C, TGS}$
Client requests access to service (ticketing) to the TGS
CLIENT
SERVER
K, N
TGT = K_{TGS}
U
C
TGS
T
L
K

Kerberos



K_S | U | C | S | T ' | L ' | K '
K | $T_{C, S}$ | K '
AS
TGS
The TGS issues ticket $T_{C, S}$
CLIENT
SERVER
K, N
TGT = K_{TGS} | U | C | TGS | T | L | K

Kerberos

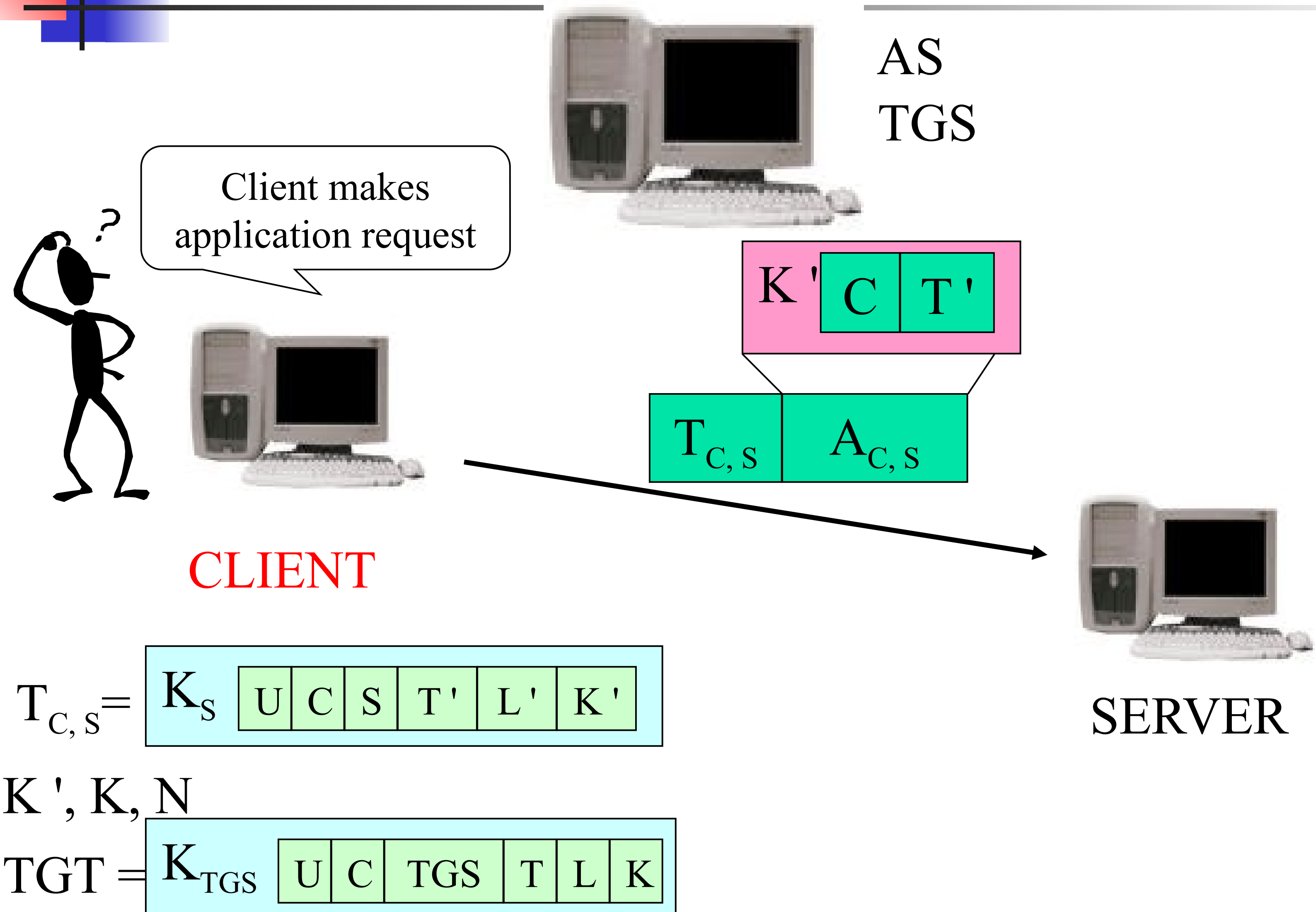


$$T_{C,S} = \boxed{K_S \; \boxed{U \mid C \mid S \mid T' \mid L' \mid K'}}$$

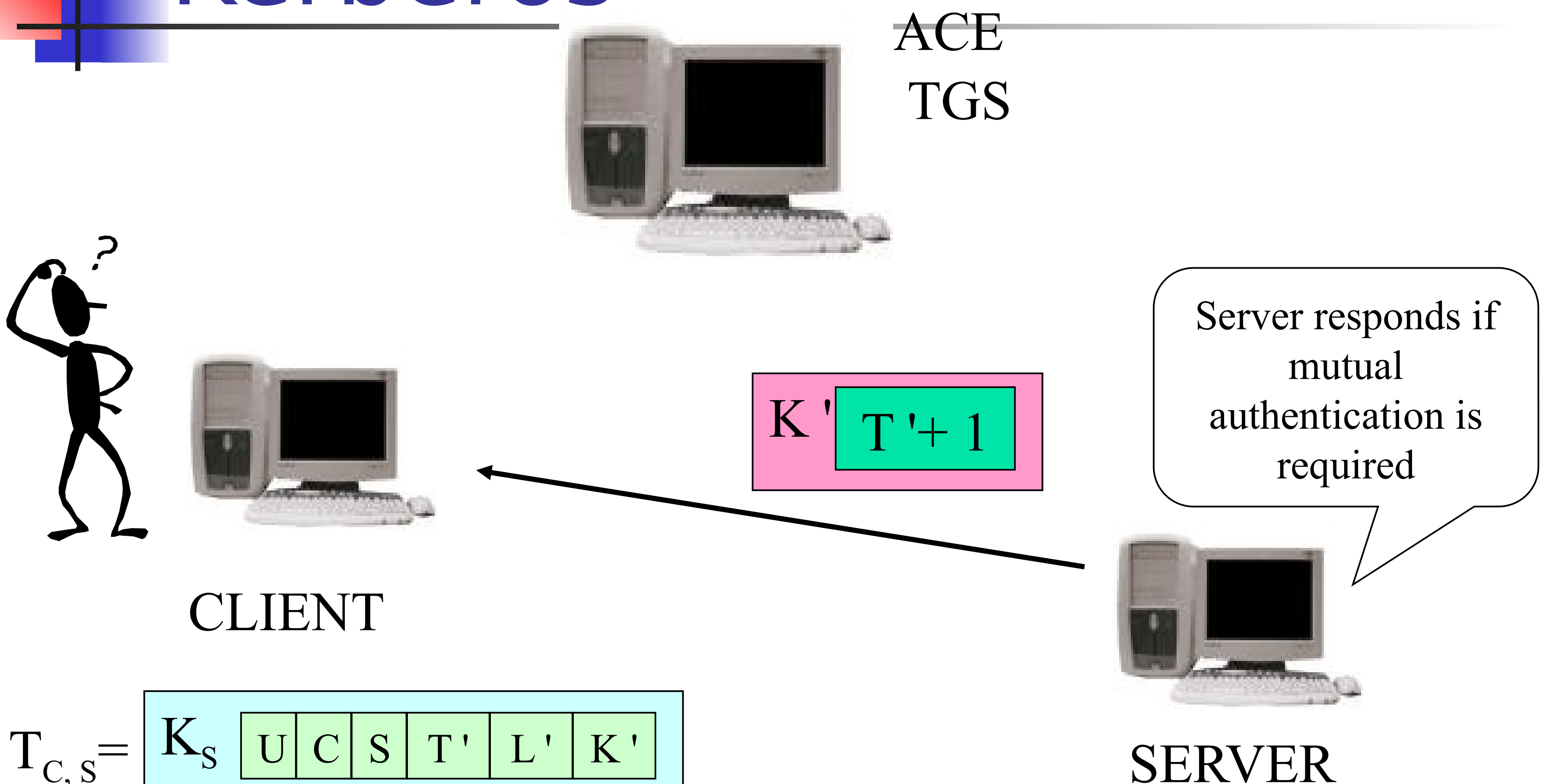
K ', K, N

$$TGT = \boxed{K_{TGS} \; \boxed{U \mid C \mid TGS \mid T \mid L \mid K}}$$

Kerberos


AS
TGS
Client makes application request
K ' C T '
$T_{C,S}$ $A_{C,S}$
CLIENT
SERVER
$T_{C,S}$ = K_S U C S T ' L ' K '
K ', K, N
TGT = K_{TGS} U C TGS T L K

Kerberos



$T_{C,S} = \boxed{K_S \; \boxed{U \mid C \mid S \mid T' \mid L' \mid K'}}$

K ', K, N

$TGT = \boxed{K_{TGS} \; \boxed{U \mid C \mid TGS \mid T \mid L \mid K}}$

Kerberos

- Kerberos v4 shortcomings:
  - Encryption system dependency (DES)
  - IP protocol dependency
  - Lifetime of the tickets (21 hours approx.)
  - Nomenclature of the principals
  - Cross-realm authentication
  - Authentication forwarding
  - Technical limitations: double encryption, PCBC encryption (DES non-standard mode),...

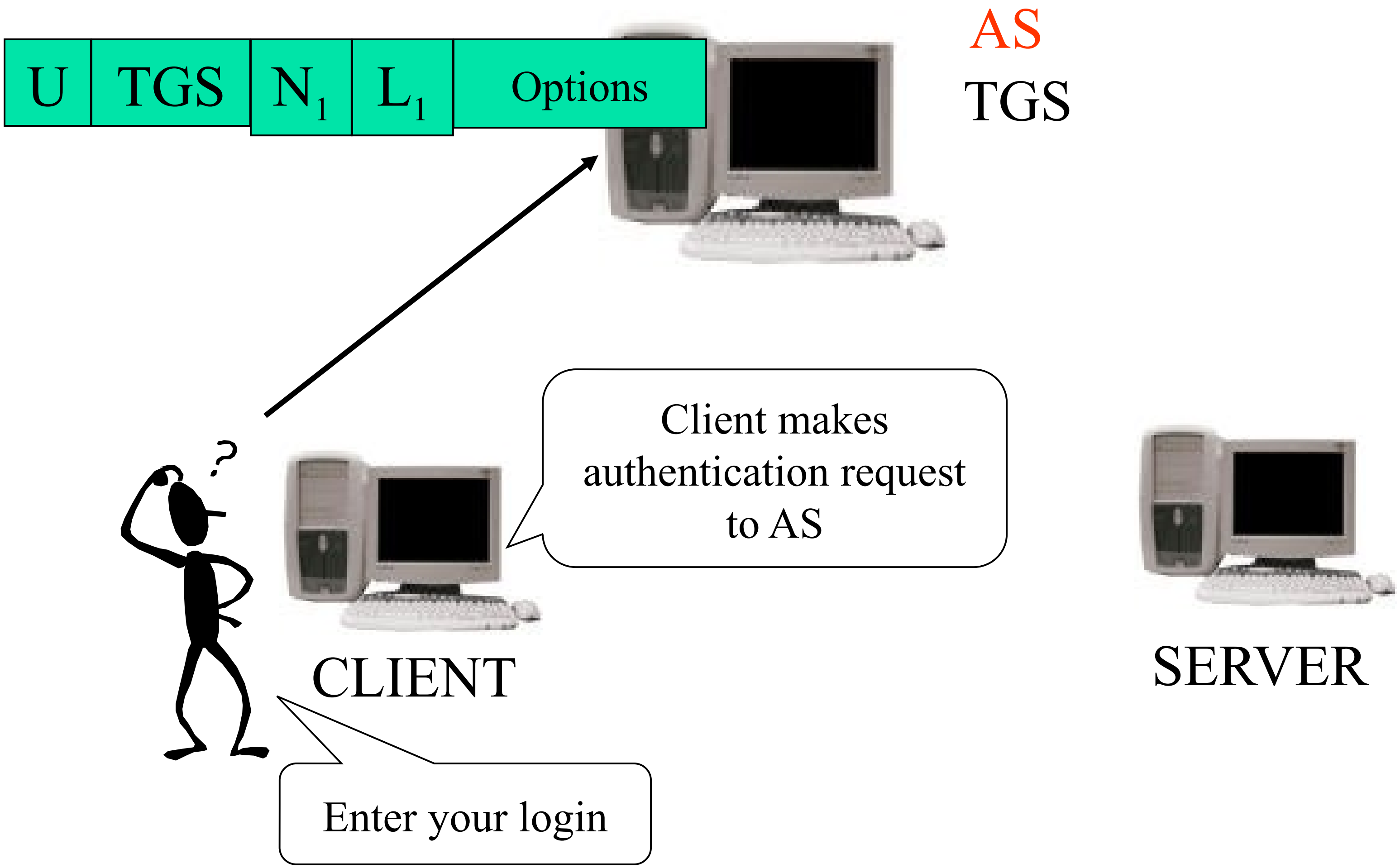
Kerberos

- Improvements introduced with Kerberos v5:
  - Principal identifiers
  - Use of encryption
  - Network addresses
  - Byte order
  - Operation Between Realms
  - Authentication forwarding

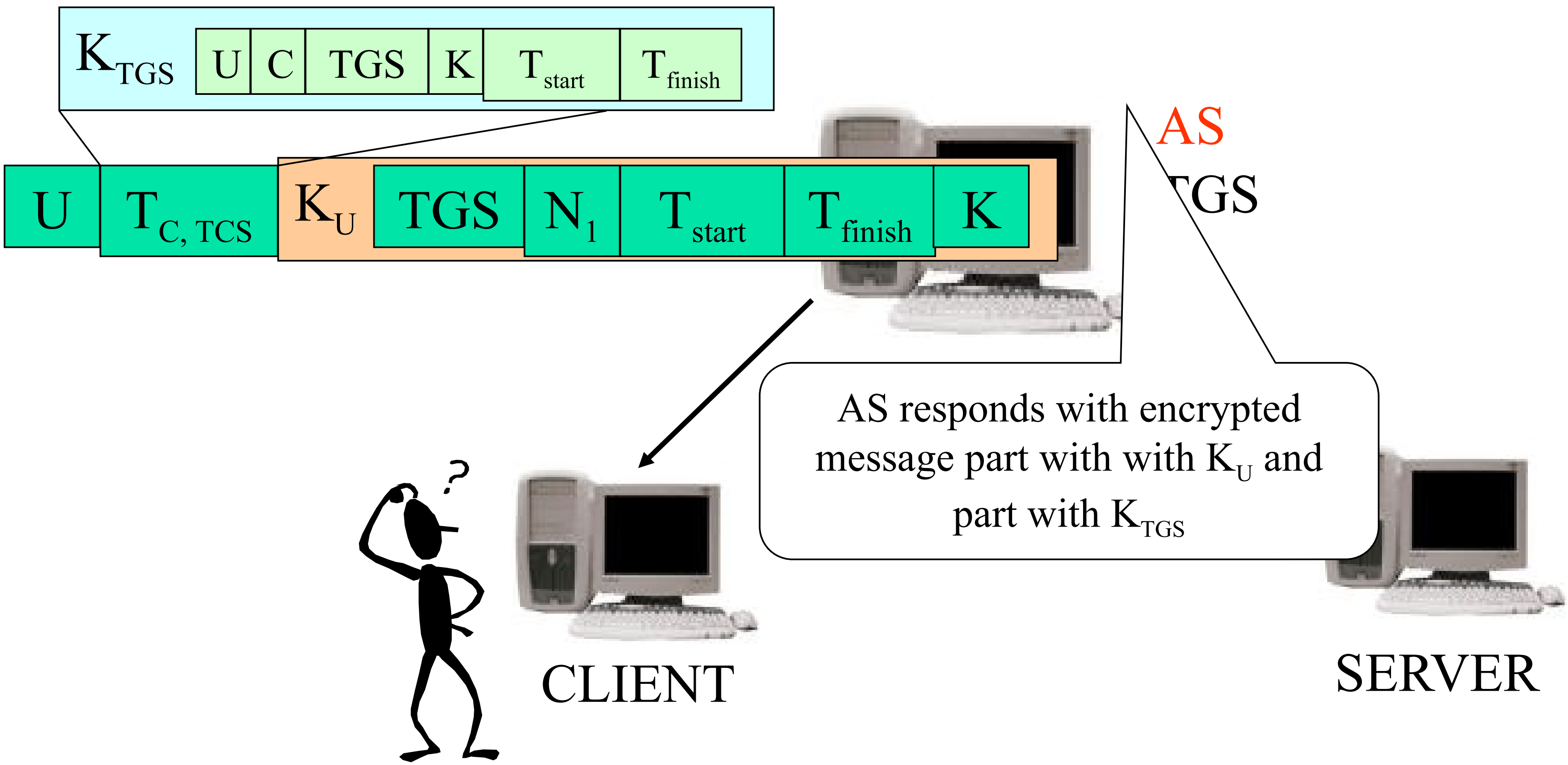
Kerberos


Authentication Server (AS)
Ticket-Granting Server (TGS)
USER
CLIENT
SERVER

Kerberos

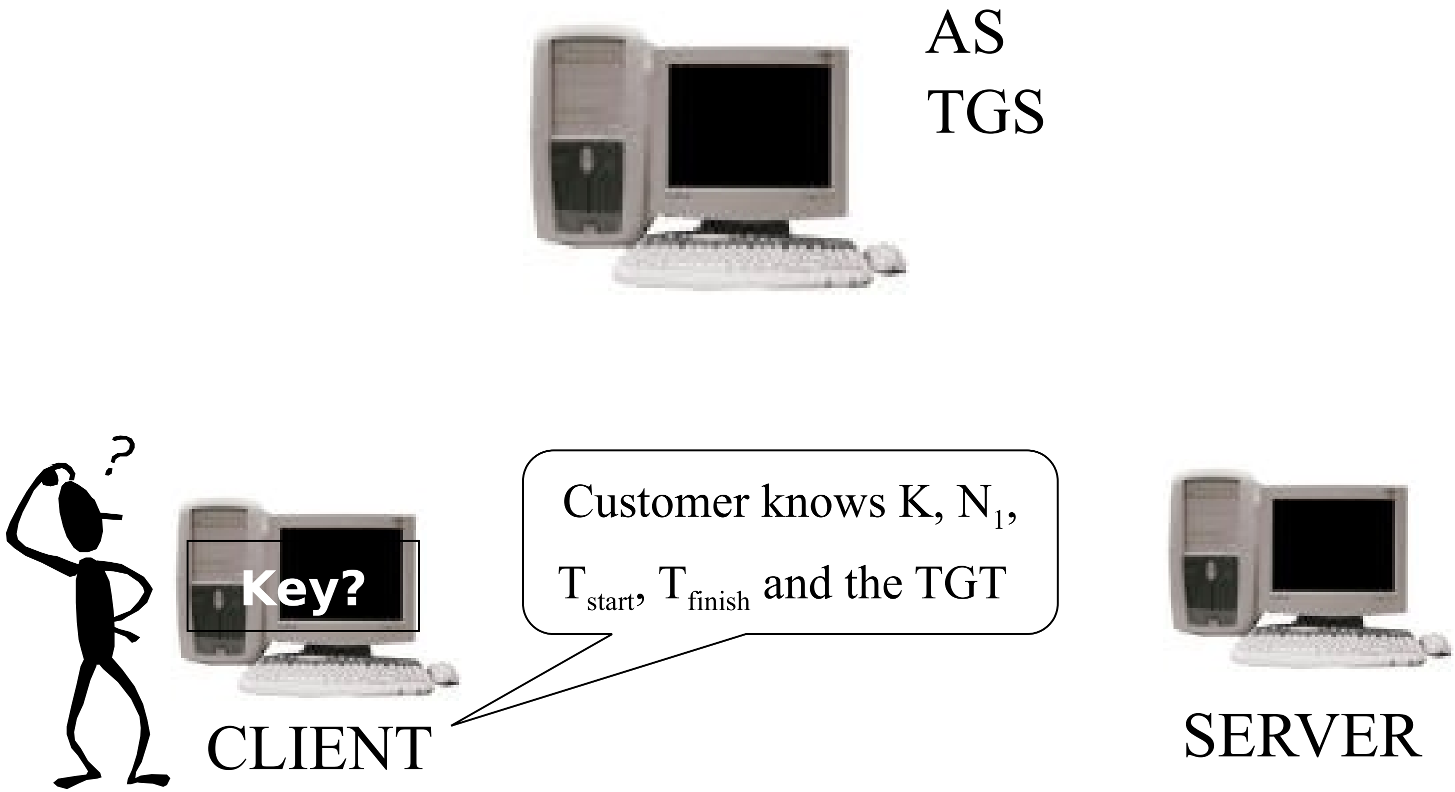

U
TGS
N_1
L_1
Options
AS
TGS
Client makes authentication request to AS
CLIENT
SERVER
Enter your login

Kerberos



K_{TGS} U C TGS K T_{start} T_{finish}
U $T_{C, TCS}$ K_U TGS N_1 T_{start} T_{finish} K
AS
TGS
AS responds with encrypted message part with with K_U and part with K_{TGS}
CLIENT
SERVER

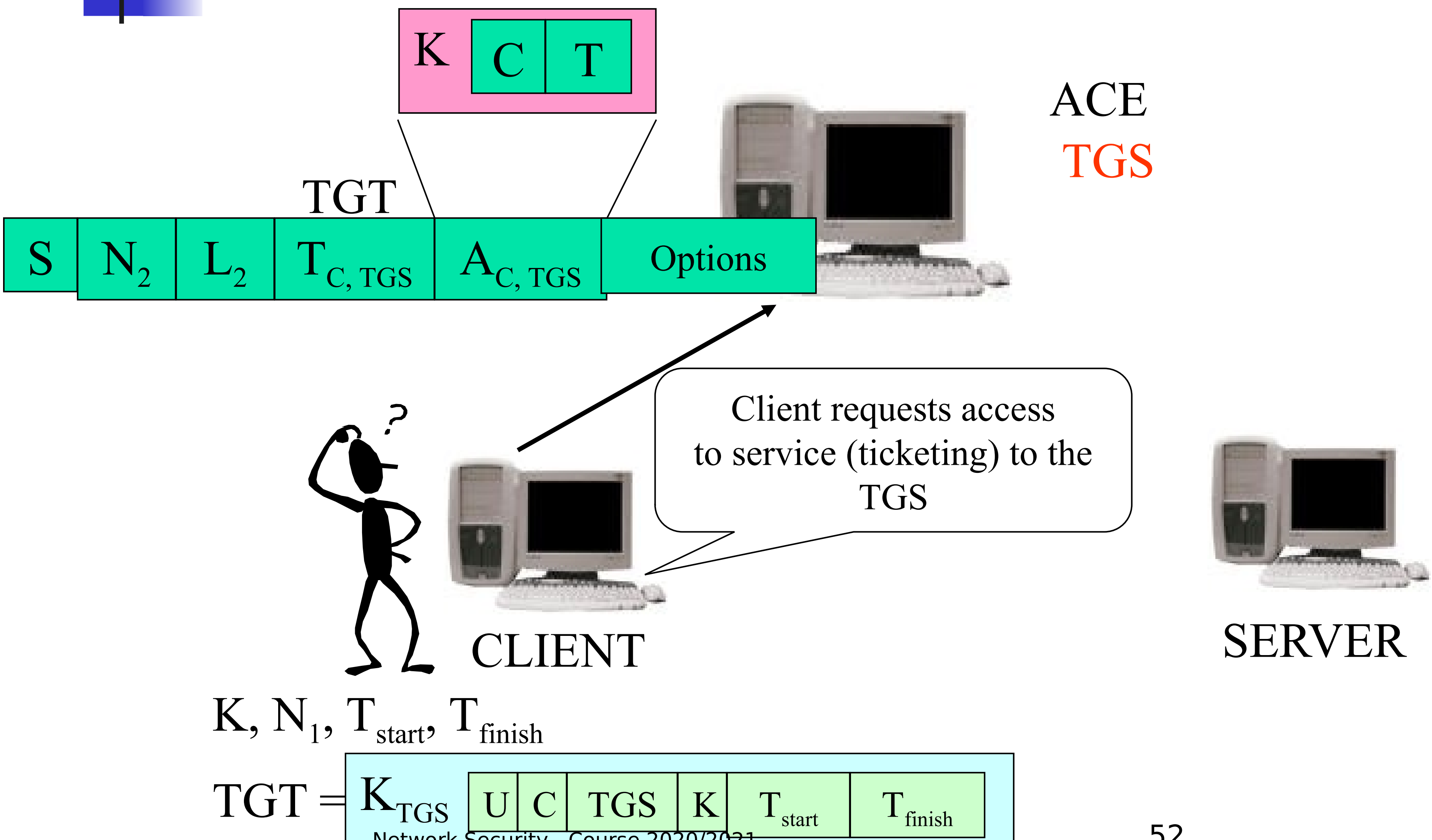
Kerberos



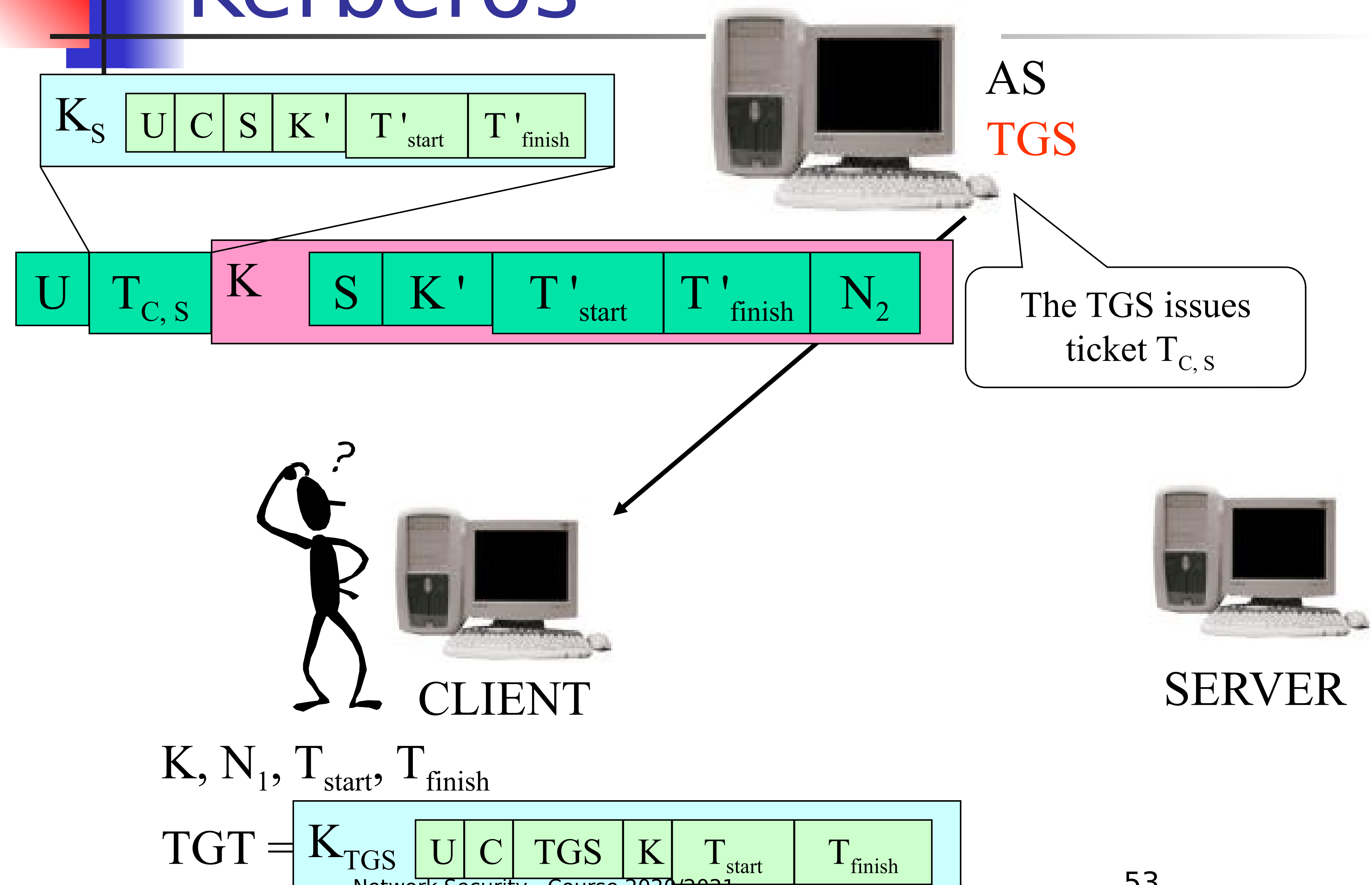
K, N_1, T_{start}, T_{finish}

$$TGT = \boxed{K_{TGS} \boxed{U \mid C \mid TGS \mid K \mid T_{start} \mid T_{finish}}}$$

Kerberos

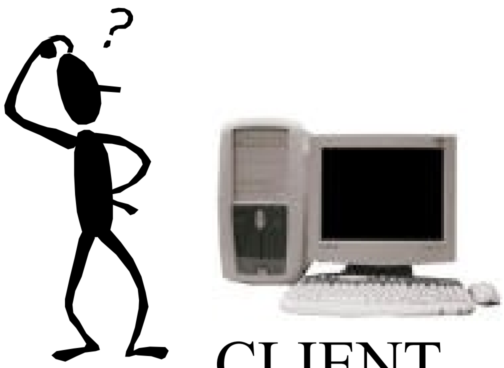

K C T
TGT
S
N2
L2
T C, TGS
A C, TGS
Options
ACE
TGS
Client requests access to service (ticketing) to the TGS
CLIENT
SERVER
K, N1, Tstart, Tfinish
TGT = KTGS U C TGS K Tstart Tfinish

Kerberos



K_S U C S K' T'_{start} T'_{finish}
AS
TGS
U $T_{C,S}$ K S K' T'_{start} T'_{finish} N_2
The TGS issues ticket $T_{C,S}$
CLIENT
SERVER
K, N_1, T_{start}, T_{finish}
TGT = K_{TGS} U C TGS K T_{start} T_{finish}

Kerberos



The customer already has the ticket and the key k '

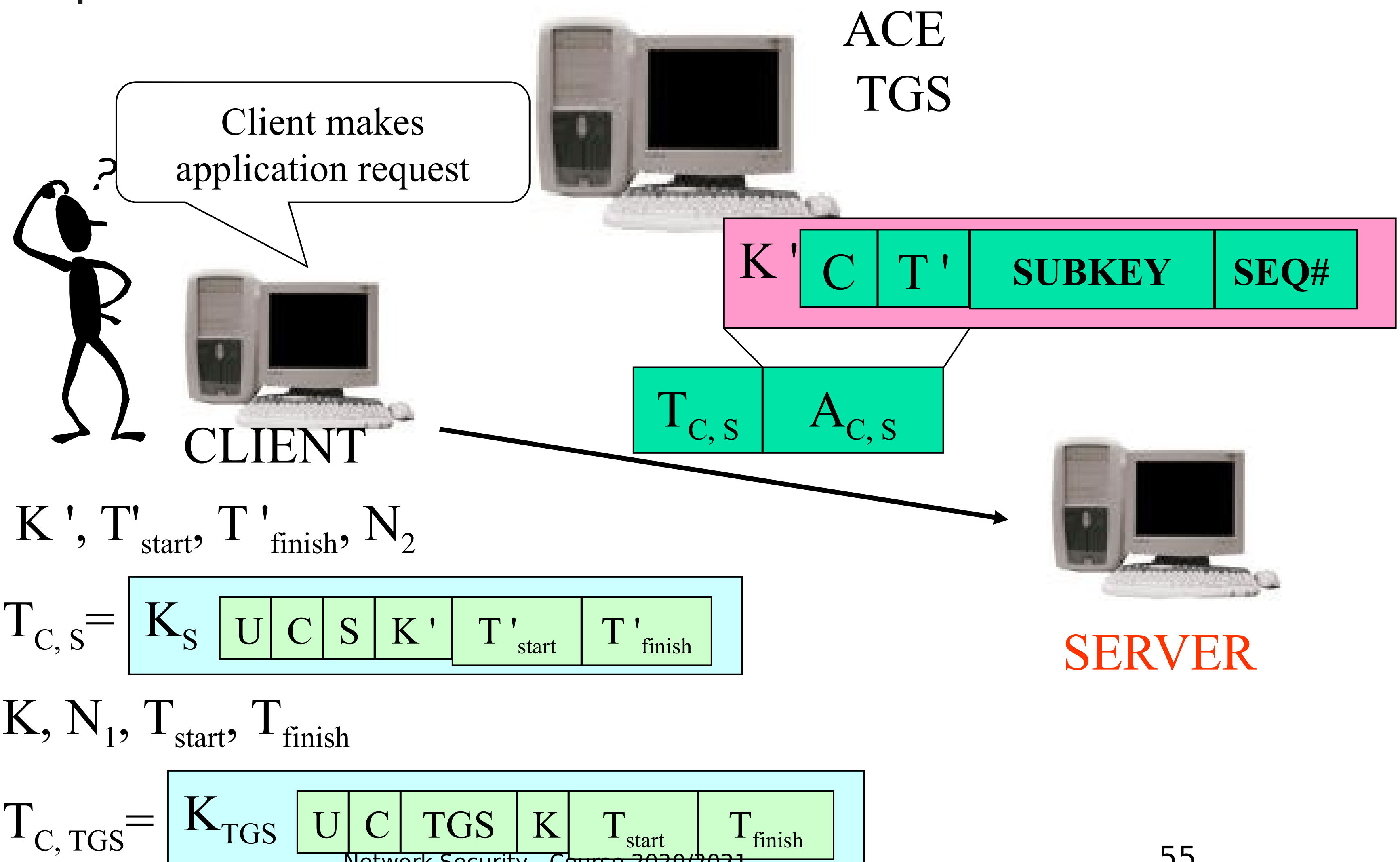
$K', T'_{start}, T'_{finish}, N_2$

$T_{C,S} = \boxed{K_S \; \boxed{U \mid C \mid S \mid K' \mid T'_{start} \mid T'_{finish}}}$

$K, N_1, T_{start}, T_{finish}$

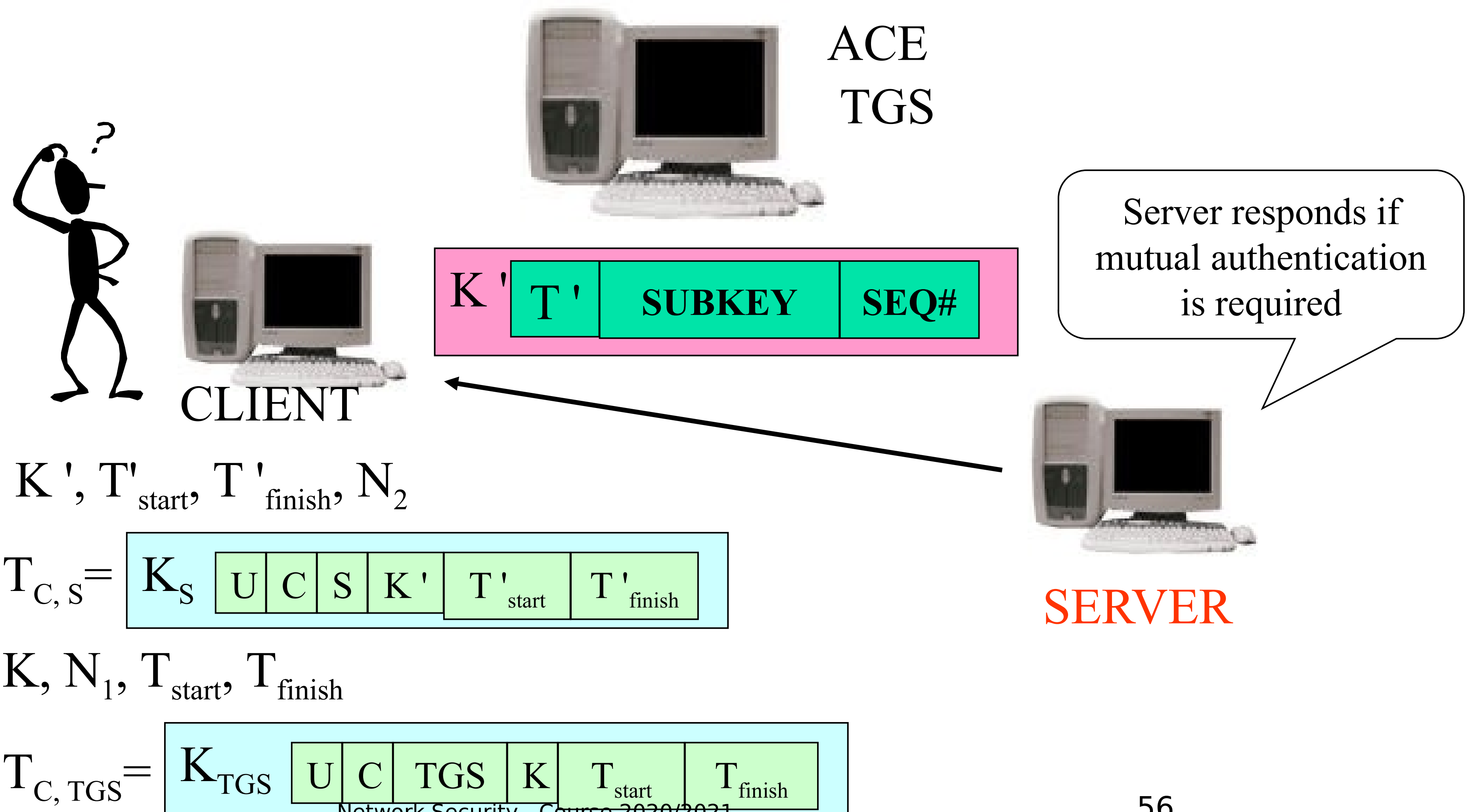
$T_{C,TGS} = \boxed{K_{TGS} \; \boxed{U \mid C \mid TGS \mid K \mid T_{start} \mid T_{finish}}}$

Kerberos



ACE
TGS
Client makes application request
CLIENT
K ', T'start, T 'finish, N2
TC, S=
KS
U
C
S
K '
T 'start
T 'finish
K, N1, Tstart, Tfinish
TC, TGS=
KTGS
U
C
TGS
K
Tstart
Tfinish
K '
C
T '
SUBKEY
SEQ#
TC, S
AC, S
SERVER

Kerberos


ACE
TGS
CLIENT
K' | T' | SUBKEY | SEQ#
Server responds if mutual authentication is required
SERVER
$K', T'_{start}, T'_{finish}, N_2$
$T_{C,S} =$ K_S | U | C | S | K' | T'_{start} | T'_{finish}
$K, N_1, T_{start}, T_{finish}$
$T_{C,TGS} =$ K_{TGS} | U | C | TGS | K | T_{start} | T_{finish}

Contents

EAP

- **EAP PPP** (Extensible Authentication Protocol) is a general authentication protocol in PPP that supports multiple authentication mechanisms
- General operation (RFC 2284):
  - After the link establishment phase, authenticator sends one or more requests (*Request*) to authenticate to the other end
  - The other end responds to each request (*Response*)
  - The authenticator ends the authentication phase with a success packet (*Success*) or failure (*Failure*)
- EAP PPP packet encapsulated in the PPP frame information field
  - (protocol field = 0xC227)

EAP

CODE	IDENTIFIER	LENGTH
DATA		

- CODE (1 byte): Identifies the type of packet
  - 1 -> Request
  - 2 -> Response
  - 3 -> Success
  - 4 -> Failure
- IDENTIFIER (1 byte): Match responses with requests
- LENGTH (2 byte): Length of the EAP packet including all fields
- DATA (0 or more bytes): The format of this field is determined by the code

EAP

- REQUEST PACKAGE:
  - Request packet is sent by authenticator to other end
  - Each Request has a TYPE field (1 byte) indicating what is requested
  - Variable data field content
- RESPONSE PACKAGE:
  - Response packet is only sent in response to a Request packet
  - Each Response has a TYPE field that normally matches that of the Request packet
  - The value of the IDENTIFIER field must be the same as that of the Request packet

CODE (1/2)	IDENTIFIER	LENGTH
KIND	DATA	

EAP

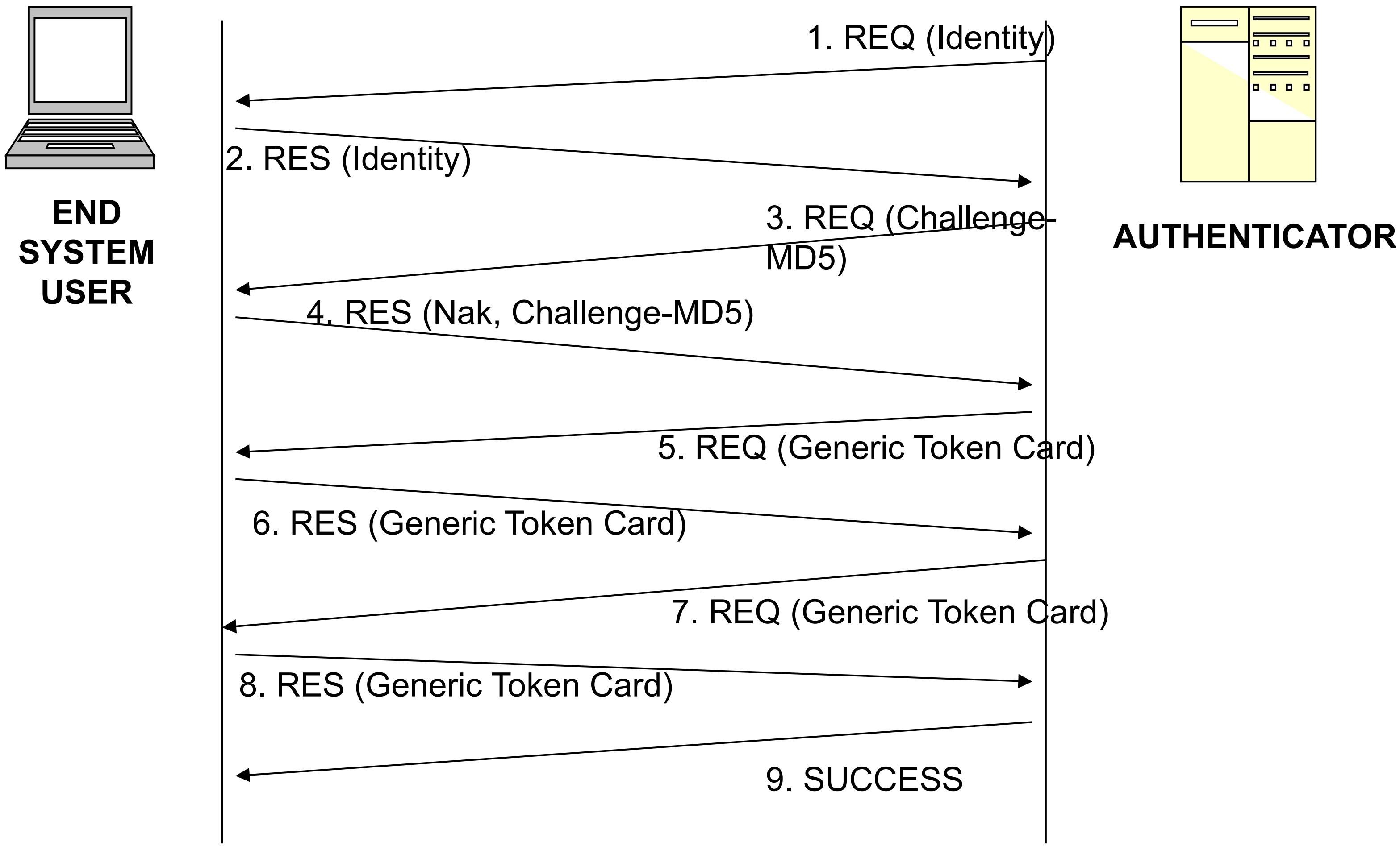
- RESQUEST / RESPONSE types:
  - 1 -> Identity, request identity from the other end
  - 2 -> Notification, message to show at the other end
  - 3 -> NAK, (only in Response) type of authentication desired is unacceptable
  - 4 -> MD5 Challenge
  - 5 -> One time password
  - 6 -> Generic Token Card
  - 13 -> Transport Layer Security

EAP

- SUCCESS / FAILURE PACKAGE:
  - Success pack acknowledges successful authentication
  - If the authenticator cannot authenticate to the other end it sends a Failure packet

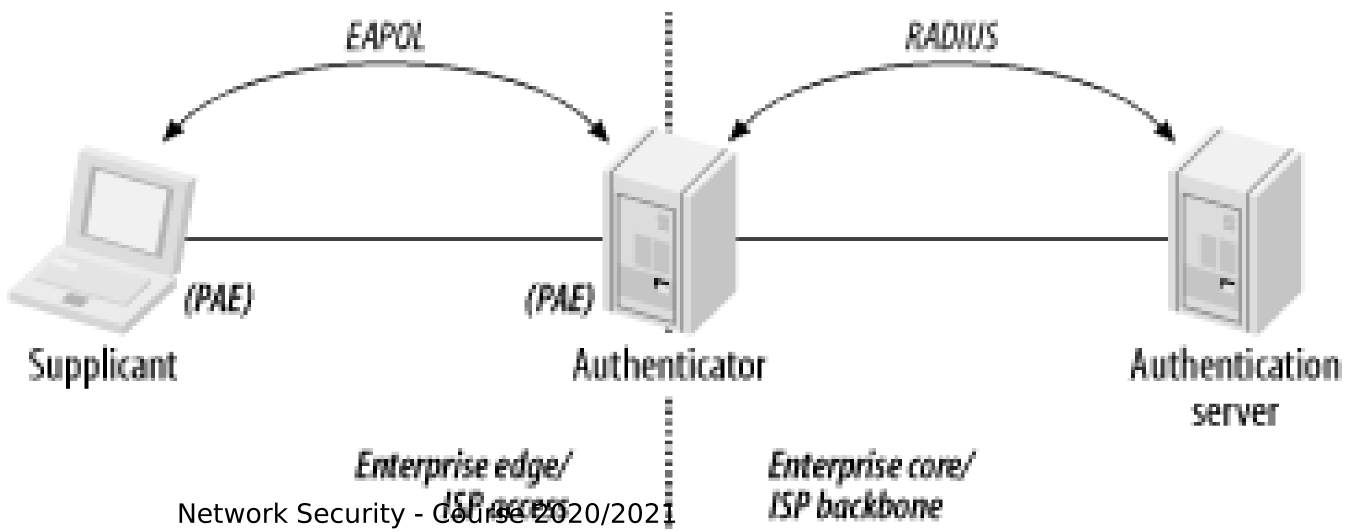
CODE (3/4)	IDENTIFIER	LENGTH

EAP


END SYSTEM USER
AUTHENTICATOR
1. REQ (Identity)
2. RES (Identity)
3. REQ (Challenge-MD5)
4. RES (Nak, Challenge-MD5)
5. REQ (Generic Token Card)
6. RES (Generic Token Card)
7. REQ (Generic Token Card)
8. RES (Generic Token Card)
9. SUCCESS

EAP: 802.1x

- Authentication standard where they are defined:
  - Supplicant
  - Authenticator
  - Authentication server
- Port Authentication Entities (PAEs)
  - Supplicant and Authenticator

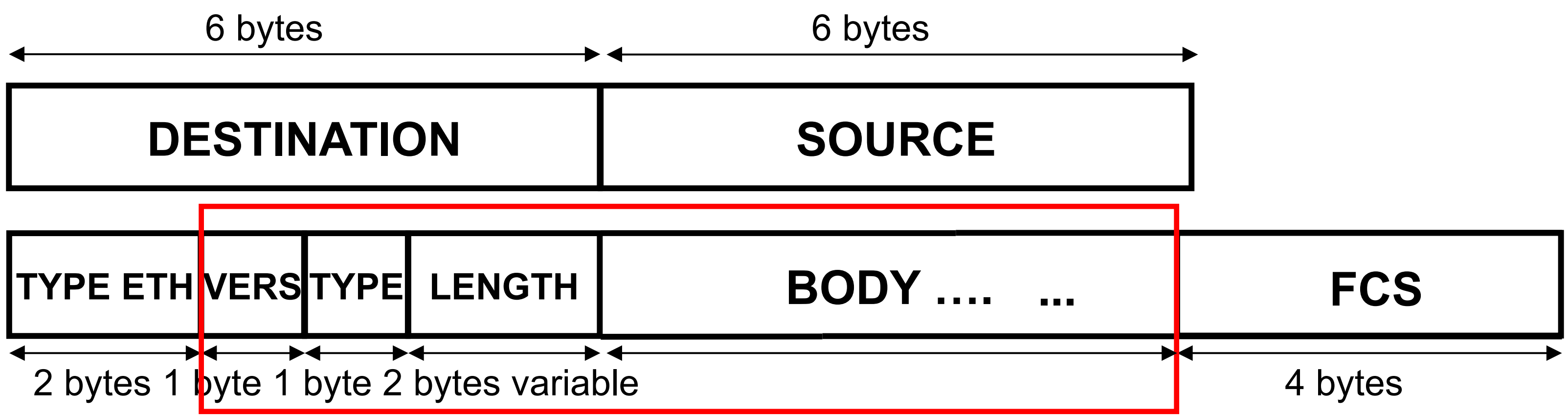


EAP: 802.1x

- Authentication exchange between supplicant and authentication server, authenticator acts as a bridge between both
  - EAPOL (EAP Over LAN) or EAPOW (EAP Over Wireless)
  - RADIUS (Remote Authentication Dial In User Service)
- Advantage: changes to the authentication method do not require complex changes to the end system or network infrastructure

EAP: 802.1x

- EAPOL frame format



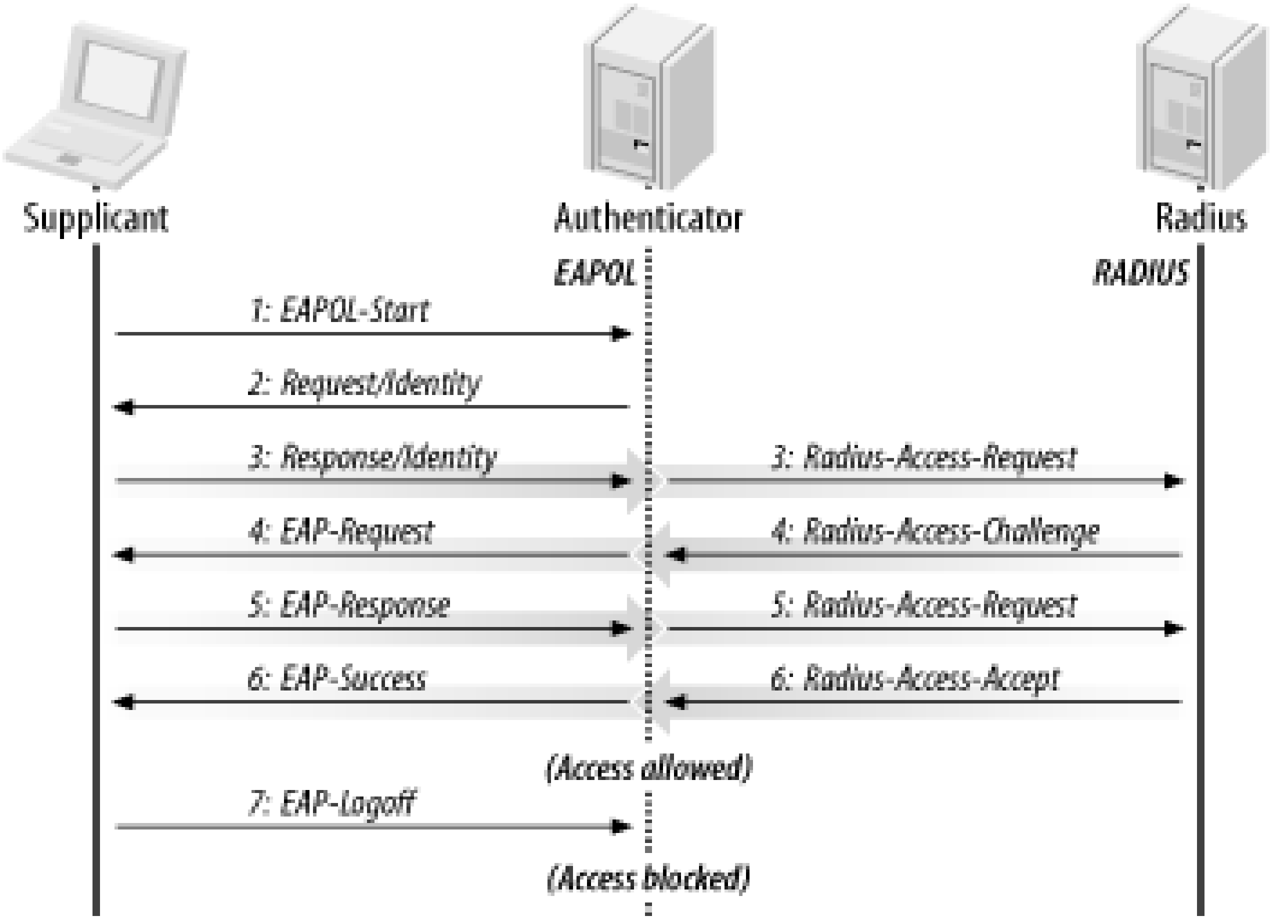
EAP: 802.1x

- DESTINATION / SOURCE (MAC Header)
  - Source and destination MAC addresses
  - In LAN of shared means the supplicants send the messages to the MAC 01: 80: C2: 00: 00: 03
  - In 802.11 networks the ports do not exist as such, EAPOW is executed after the association process between supplicant and authenticator
- ETHERNET TYPE
  - Code assigned to EAPOL 88: 8E
- VERSION
  - Currently only version 1 exists

EAP: 802.1x

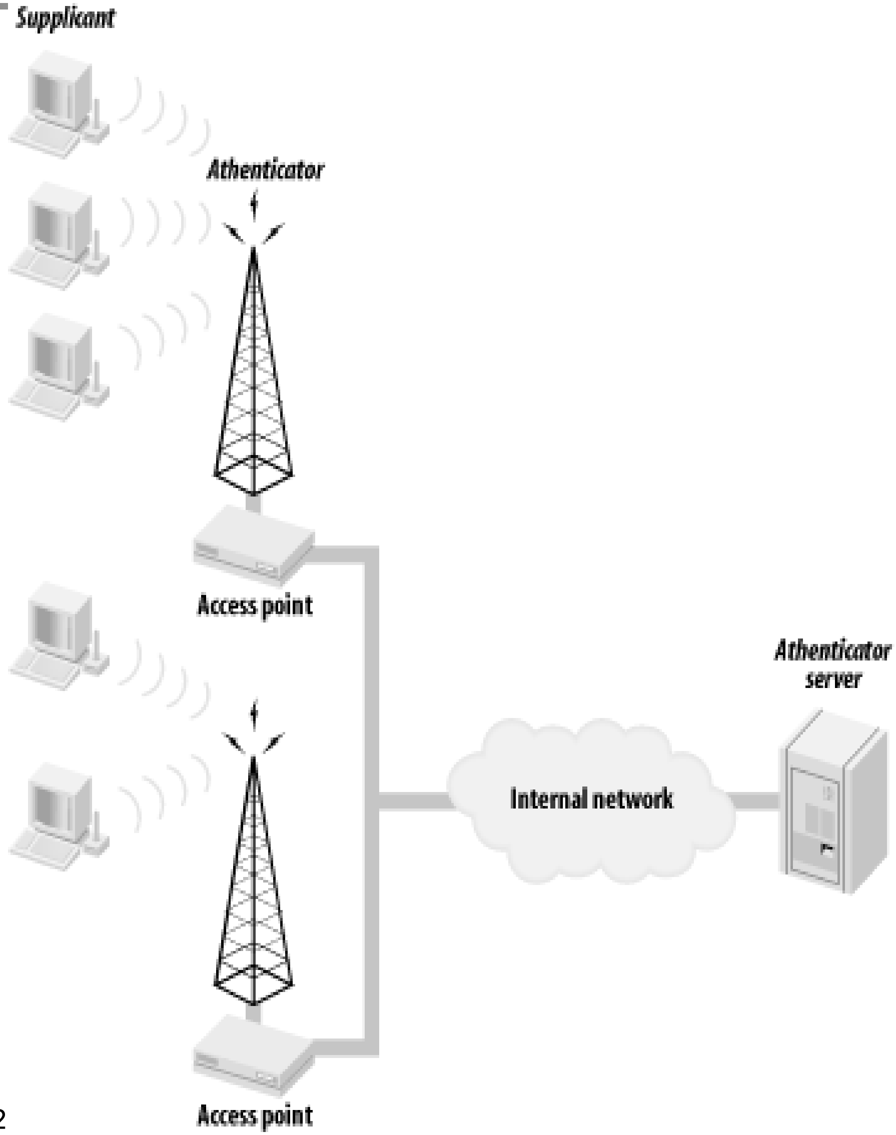
- PACKET TYPE
  - 0x00 EAP PACKAGE
  - 0x01 EAPOL START
  - 0x02 EAPOL LOGOFF
  - 0x03 EAPOL KEY
  - 0x04 EAPOL ENCAPSULATED ASF ALERT
- LENGTH
  - Length of the BODY field in bytes
- BODY
  - Field that encapsulates an EAP packet, an EAPOL KEY, or an EAPOL ENCAPSULATED ASF ALERT

EAP: 802.1x

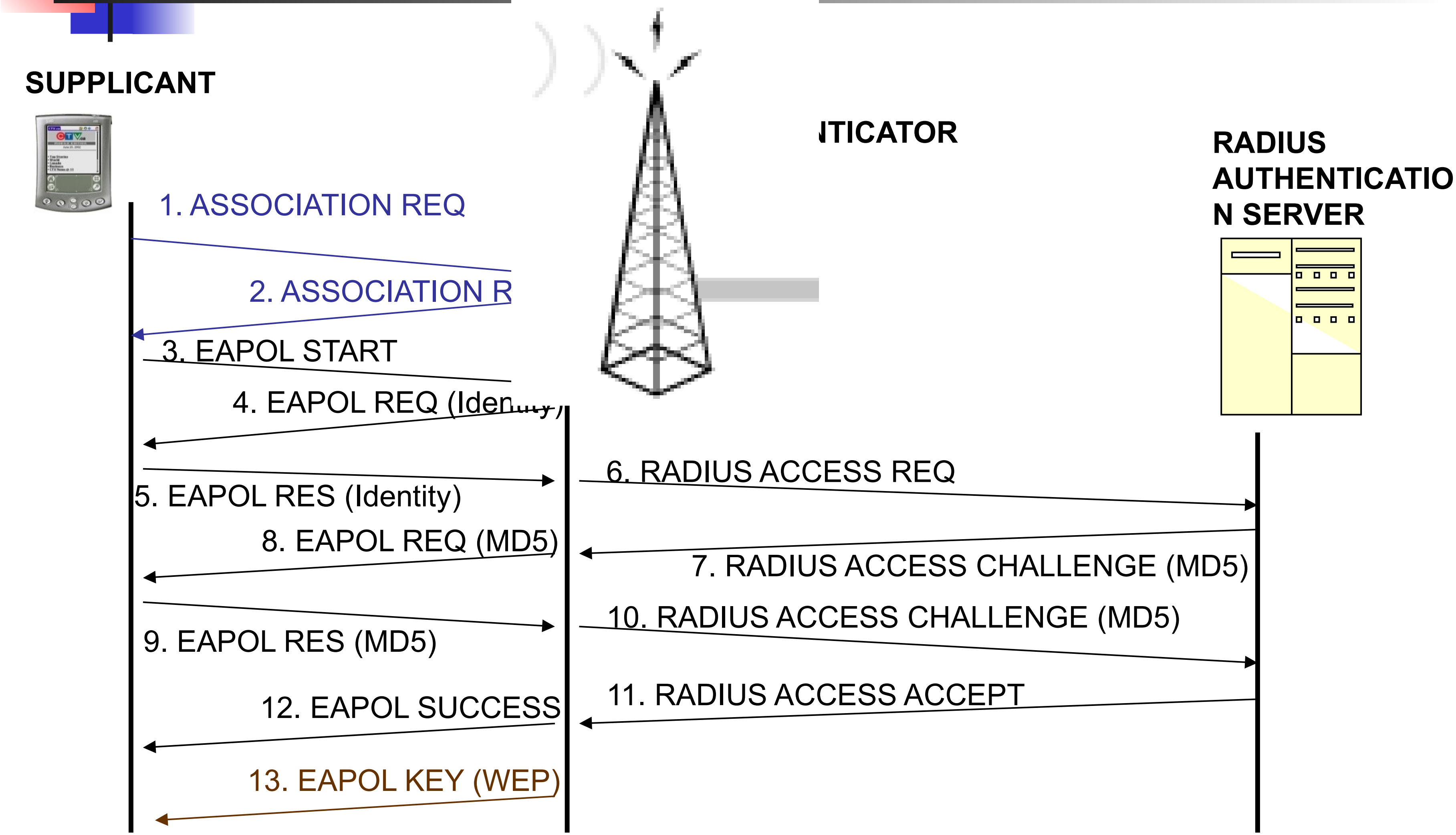

Supplicant
Authenticator
Radius
EAPOL
RADIUS
1: EAPOL-Start
2: Request/Identity
3: Response/Identity
3: Radius-Access-Request
4: EAP-Request
4: Radius-Access-Challenge
5: EAP-Response
5: Radius-Access-Request
6: EAP-Success
6: Radius-Access-Accept
(Access allowed)
7: EAP-Logoff
(Access blocked)

EAP: 802.1x

- In wireless networks "association between mobile station and access point" ≅ "Logical port"
- The access point drops all traffic until successful authentication
- EAPOL KEY frame can be used for dynamic key distribution in WEP (Wired Equivalent Privacy)



EAP: 802.1x


SUPPLICANT
TICATOR
RADIUS
AUTHENTICATIO
N SERVER
1. ASSOCIATION REQ
2. ASSOCIATION R
3. EAPOL START
4. EAPOL REQ (Iden
5. EAPOL RES (Identity)
6. RADIUS ACCESS REQ
7. RADIUS ACCESS CHALLENGE (MD5)
8. EAPOL REQ (MD5)
9. EAPOL RES (MD5)
10. RADIUS ACCESS CHALLENGE (MD5)
11. RADIUS ACCESS ACCEPT
12. EAPOL SUCCESS
13. EAPOL KEY (WEP)

Contents

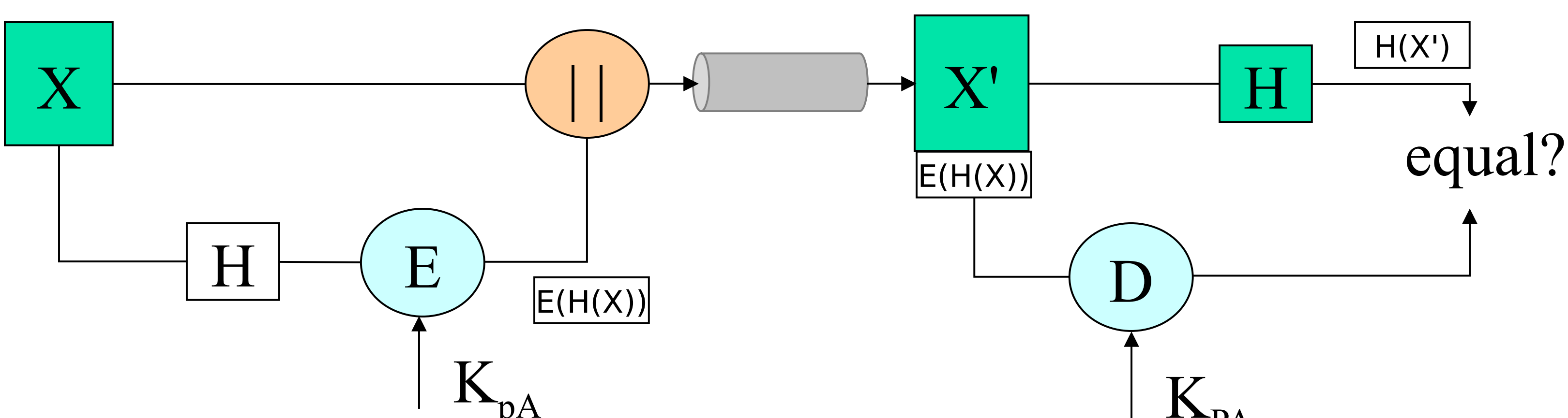
Digital signature

- Public key cryptography (asymmetric)

- The DIGITAL SIGNATURE must have the following properties:
  - Be able to verify author, date and time of signature
  - Being able to authenticate the content of the message at the time it was signed
  - It must be verified by a third party to avoid disputes

Digital signature

- Any DIGITAL SIGNATURE:
  - Firm ≡ bit pattern dependent on the signed message
  - Will use unique information from the issuer to avoid denial and falsification
  - Simple to create
  - Simple to recognize and verify
  - Falsifying it must be computationally not feasible



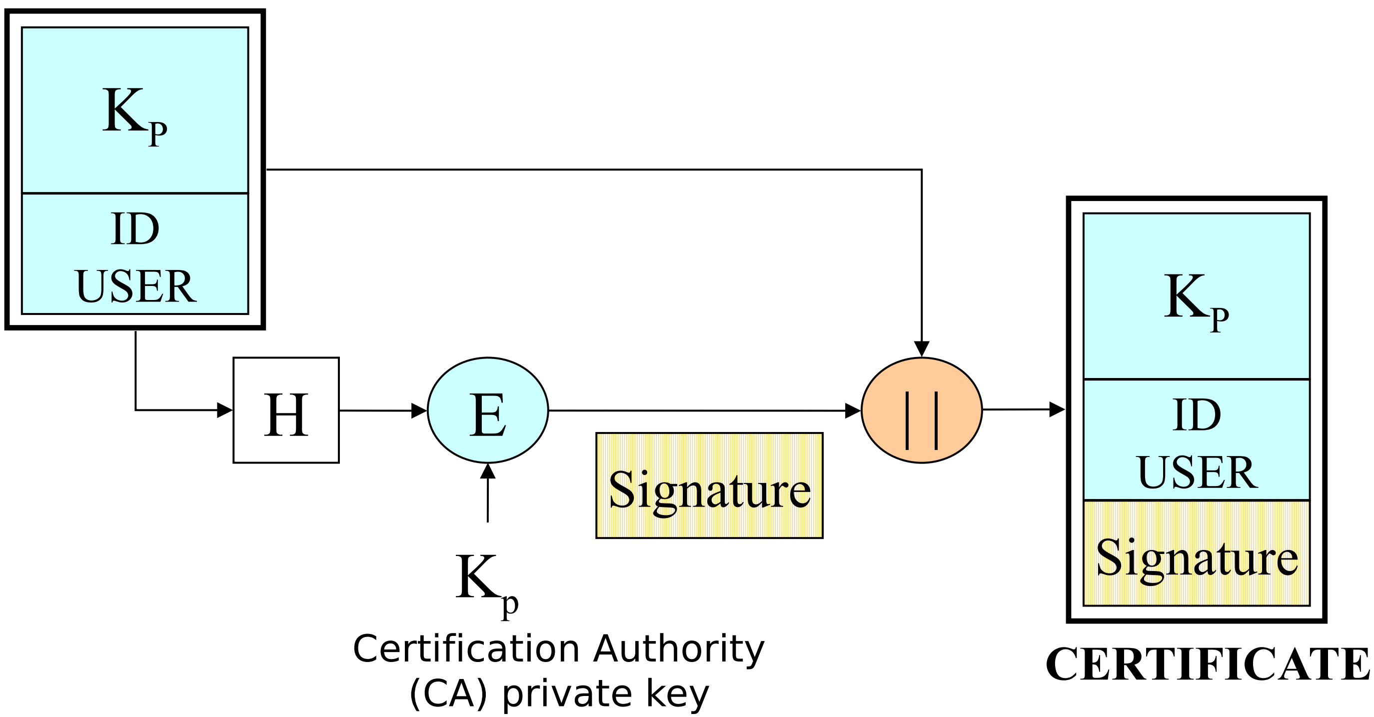
Digital signature

- Direct Digital Signature
  - Only the two communicators intervene
  - Destination knows issuer's public key
  - We sign the complete message or hash of the message with sender's private key K_p
  - Problem: secret key security
- Arbitrated Digital Signature
  - A third entity acts as an arbitrator
  - General operation: all messages go through the referee who checks the validity of origin and content
  - Total reliability in the referee

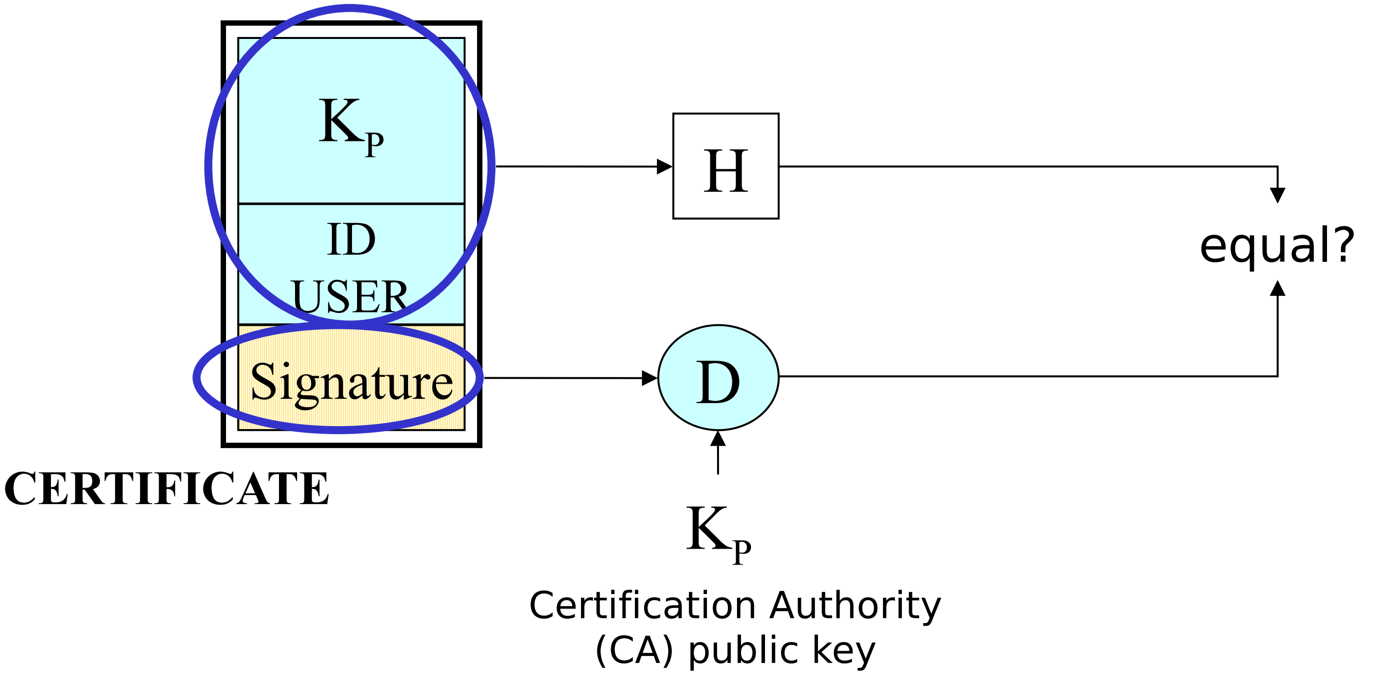
Contents

Certificates

- Public keys must be "public" → Impersonation problem?
  - Solution: public key certificates



Certificates



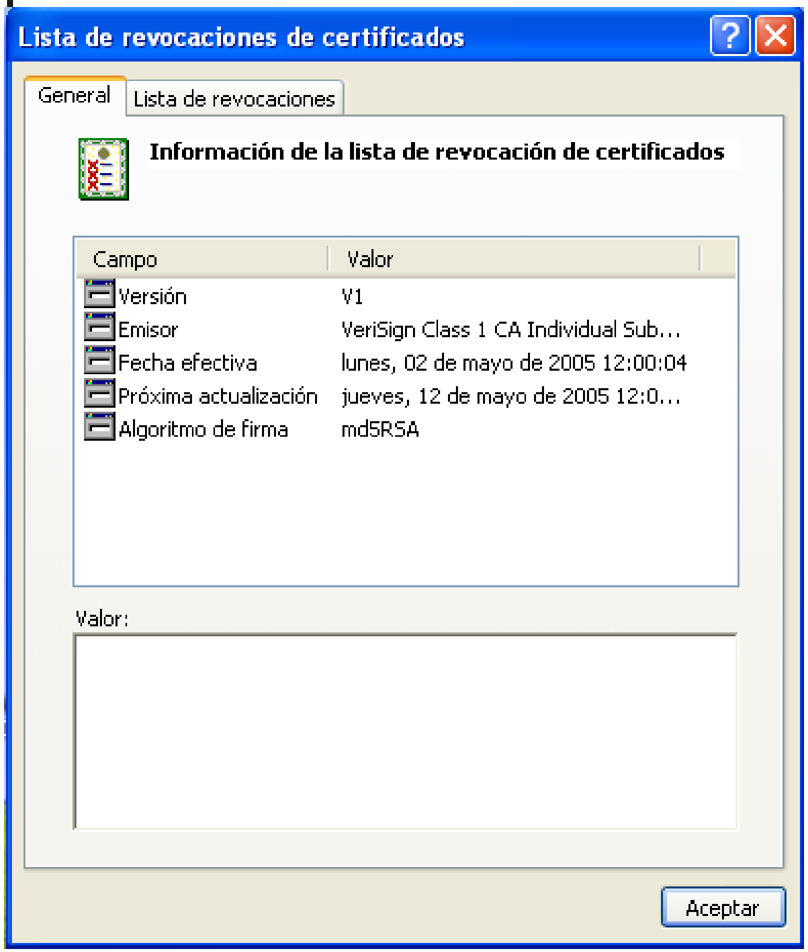
Certificates

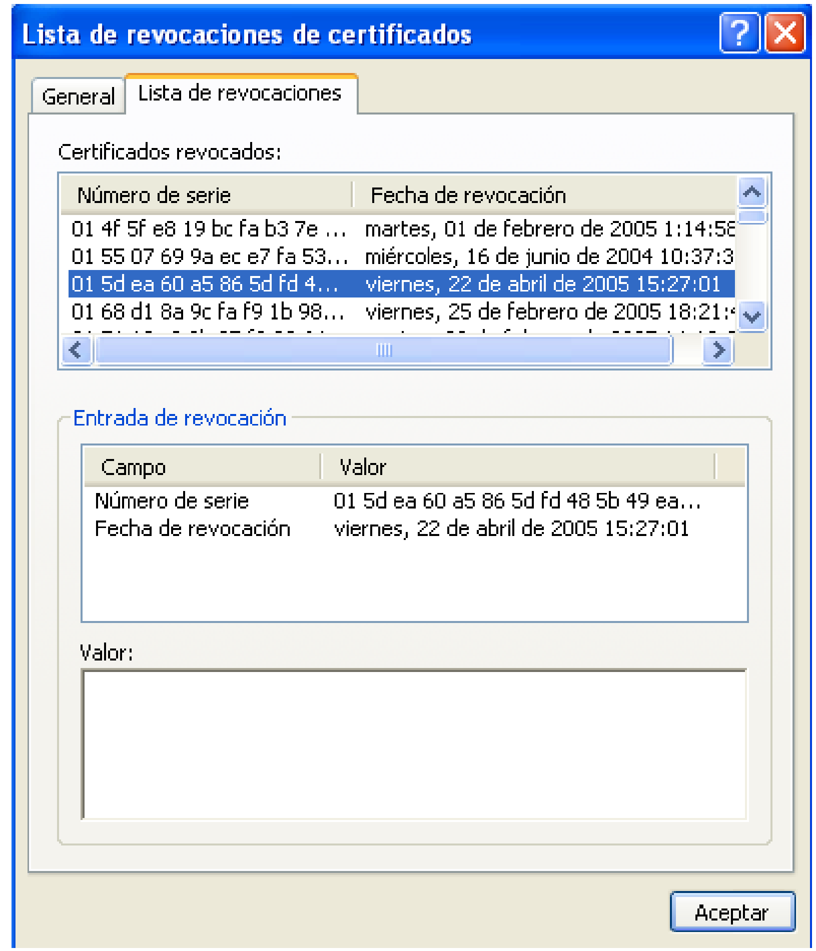
- Service offered by the CA
  - Internal CA, certify your own employees, positions and levels of authority
  - External employee CA, company hires another to certify its employees
  - External customer CA, company hires another to certify its customers
  - **Trusted third-party CA**, company or government operates a CA that relates public keys to legal names of individuals or companies

Certificates

- Certificate revocation:
  - Compromised user private key
  - CA issues certificate to wrong entity
  - User changes CA
  - AC security breach
- **Certificate revocation list** (CRL, Certification Revocation List)
  - Example: http://crl.verisign.com/

Certificates





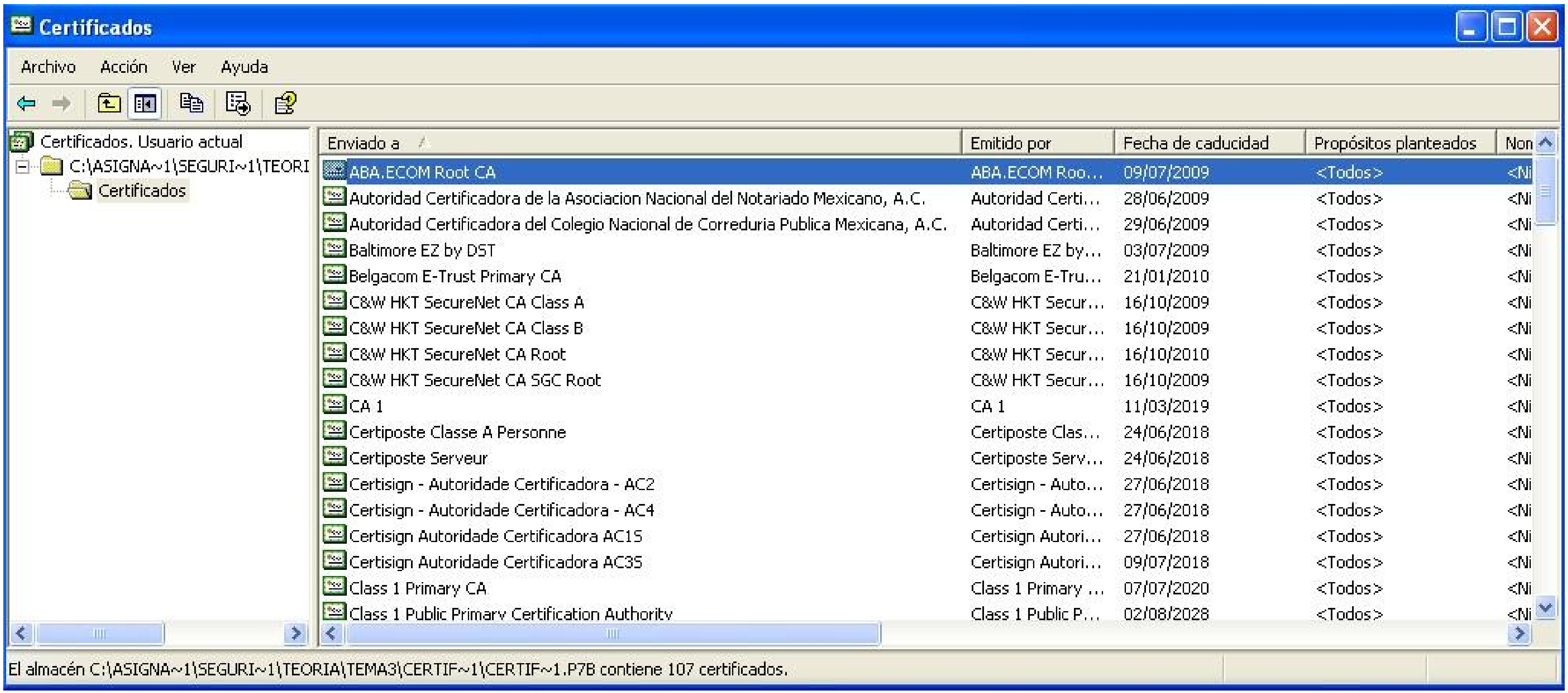
Certificates

- Certification authority certificates
- Server certificates
- Personal certificates
- Software publisher certificates

Certificates

CERTIFICATE OF CERTIFICATION AUTHORITY

- Name and public key of the CA
- They can be self-signed
- PKI (Public Key Infrastructure)

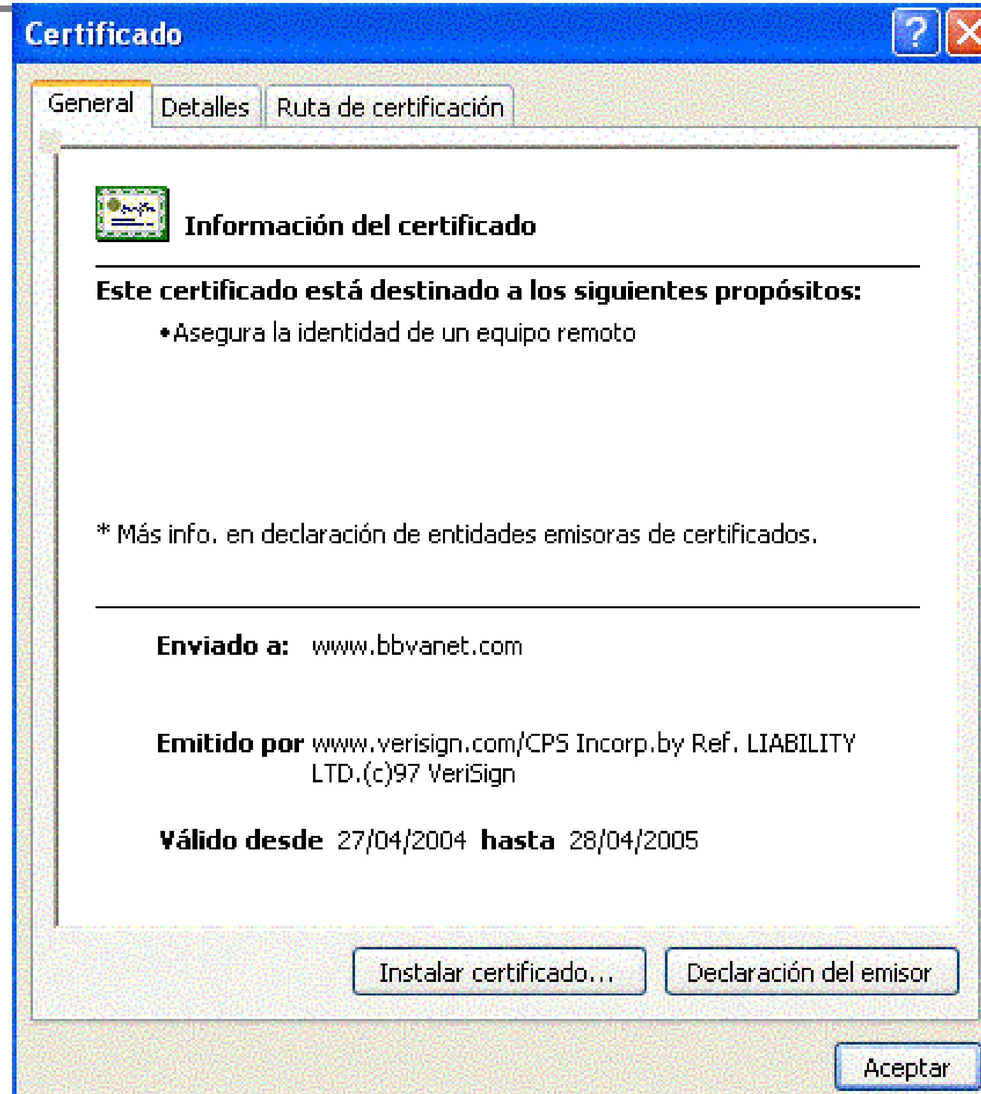


Enviado a	Emitido por	Fecha de caducidad	Propósitos planteados	Non
ABA.ECOM Root CA	ABA.ECOM Roo...	09/07/2009	<Todos>	<Ni
Autoridad Certificadora de la Asociacion Nacional del Notariado Mexicano, A.C.	Autoridad Certi...	28/06/2009	<Todos>	<Ni
Autoridad Certificadora del Colegio Nacional de Correduria Publica Mexicana, A.C.	Autoridad Certi...	29/06/2009	<Todos>	<Ni
Baltimore EZ by DST	Baltimore EZ by...	03/07/2009	<Todos>	<Ni
Belgacom E-Trust Primary CA	Belgacom E-Tru...	21/01/2010	<Todos>	<Ni
C&W HKT SecureNet CA Class A	C&W HKT Secur...	16/10/2009	<Todos>	<Ni
C&W HKT SecureNet CA Class B	C&W HKT Secur...	16/10/2009	<Todos>	<Ni
C&W HKT SecureNet CA Root	C&W HKT Secur...	16/10/2010	<Todos>	<Ni
C&W HKT SecureNet CA SGC Root	C&W HKT Secur...	16/10/2009	<Todos>	<Ni
CA 1	CA 1	11/03/2019	<Todos>	<Ni
Certiposte Classe A Personne	Certiposte Clas...	24/06/2018	<Todos>	<Ni
Certiposte Serveur	Certiposte Serv...	24/06/2018	<Todos>	<Ni
Certisign - Autoridade Certificadora - AC2	Certisign - Auto...	27/06/2018	<Todos>	<Ni
Certisign - Autoridade Certificadora - AC4	Certisign - Auto...	27/06/2018	<Todos>	<Ni
Certisign Autoridade Certificadora AC1S	Certisign Autori...	27/06/2018	<Todos>	<Ni
Certisign Autoridade Certificadora AC3S	Certisign Autori...	09/07/2018	<Todos>	<Ni
Class 1 Primary CA	Class 1 Primary ...	07/07/2020	<Todos>	<Ni
Class 1 Public Primary Certification Authority	Class 1 Public P...	02/08/2028	<Todos>	<Ni

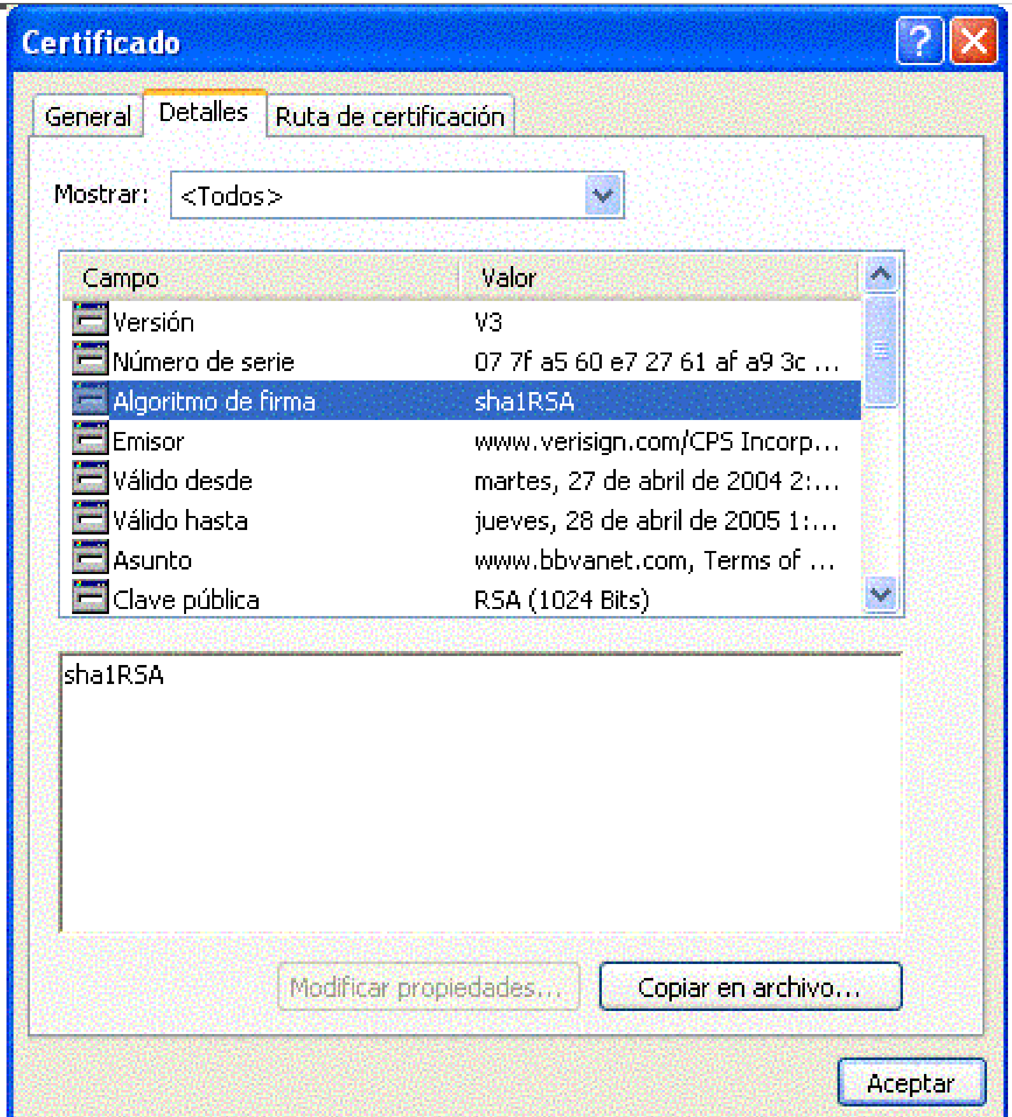
Certificates

SERVER CERTIFICATE

- Each SSL server -> one SSL server certificate
- Must contain:
  - signed key length
  - certificate serial number
  - signature algorithm
  - server name
- <u>Example</u>



Certificates



Certificates

PERSONAL CERTIFICATE

- Designed to verify the identity of an individual issued by a CA
- Benefits:
  - Eliminate the need to remember login and password
  - Proof of belonging to an organization
  - Encrypted communications
  - Restrict access to websites

Certificates

PERSONAL CERTIFICATE

- From v.3 of Navigator Netscape and Internet Explorer
  - Key creation
  - Obtaining certificates
  - Challenge / response
  - Safe storage
- In Spain:
  - National Currency and Stamp Factory (www.cert.fnmt.es)
  - ANF Certification Authority (www.anf.es)
  - AC Camerfirma (www.camerfirma.com)
  - Certification Authority of the Legal Profession (www.acabogacia.org)
  - Firma Profesional SA (www.firmaprofesional.com)

Certificates

- **Services that can be accessed with a user certificate in Spain**

Central administration
State Tax Administration Agency
Telecommunications Market Commission
Official Credit Institute
Statistics National Institute
Ministry of Economy
Presidency of Government
Social Security
General Directorate of Cadastre
Directorate General of Personnel Costs and Public Pensions
Ministry of labor and social affairs

Autonomous Administration
Madrid's community
Canary Islands Government
Government of Navarra
Government of La Rioja
Junta de Andalucía
Xunta of Galicia

Local Management
Alboraya City Council
Laredo City Council
Catarroja City Council
City of Madrid
Paterna Town Hall
Totana City Council
Valencia City Council
Barcelona Provincial Council

Others
Association of Internet Business Advisors
General Council of Notaries
Gestor de Infraestructuras SA
National Tourism Paradores
Saniline
Insurance Broker
Digital Society of Authors and Editors

Certificates

PERSONAL CERTIFICATE

- How to request it for free for two months:
  1) Go to https://digitalid.verising.com
  2) Select PersonalID -> Buy Now -> Enroll Now
  3) Complete the form *enrollment*
     - Name and surname
     - Email address
  4) Accept the agreement
  5) Check email, *verisign* will send an email with an identifier and the URL of a web page
  6) Go to the indicated web page and enter the identifier
  7) The browser will obtain the certificate
  8) To install it follow the instructions of the browser
  9) Checking in Internet Explorer: go to Tools -> Internet Options -> Content -> Certificates -> Personal

Certificates

SOFTWARE EDITOR CERTIFICATE

- Sign executable programs by electronic signature
- Improves the reliability of software distributed over the Internet
- Proposals:
  - Authenticode (Microsoft)
  - JAR, java file format that allows the use of digital signature

Certificates

- **X.509 certificate**
  - Part of the X.500 series of recommendations
  - X.509 allows authentication service
  - X.509 certificate structure used in many contexts (S / MIME, IP security, SSL / TLS, SET, ...)
  - Version 3 revised in 2000

Certificates

	VERSION	Version 1, Version 2, Version 3
	SERIAL NO.	
signature algorithm	ALGORITHM	
	PARAMETERS	
	CERTIFIED ISSUER NAME	
Period of validity	NOT BEFORE	
	NOT AFTER	
	SUBJECT NAME	
Subject's public key information	ALGORITHMS	
	PARAMETERS	
	KEY	
	ID. SINGLE CERTIFIED ISSUER	
	ID. SUBJECT'S ONLY	
	EXTENSIONS	
Firm	ALGORITHMS	All version
	PARAMETERS	
	Encryption	

CERTIFICATE X.509v3

example

Certificates

- Private keys are not people
- Distinguished names are not persons
- There are too many names of the same people
- Digital certificates don't say enough
- X.509 v.3 does not allow selective disclosure
- Digital certificates allow easy combination of data
- How many CAs does society need?
- How to lend a password?
- Are there better options to public key digital signatures?