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 using Asymmetric Cryptography

Topics Before …

- Key Distribution and Establishment Protocols using Symmetric Cryptography and KDCs (Key Distrib. Centers)
- Kerberos System (V4 and V5)

Now …

- Key Distribution and Establishment Protocols using Asymmetric Cryptography Methods
  - **Note: also use for the establishment of session symmetric keys or any security association parameters for secure communication protocols*

Outline

- **Key-Distribution Protocols using Asymmetric cryptography**
  - Base P2P Key-Establishment using Public-Key Cryptography
  - Needham-Schroeder Base Model (w/ a PKC)
  - P2P Key Establishment w/ Certification Chains
  - Examples
    - P2P Key Establishment using the Diffie Hellman Method and Certification Chains
    - PKINIT Kerberos

Outline

- **Key-Distribution Protocols using Asymmetric cryptography**
  - Base P2P Key-Establishment using Public-Key Cryptography
  - Needham-Schroeder Base Model (w/ a PKC)
  - P2P Key Establishment w/ Certification Chains
  - Examples
    - P2P Key Establishment using the Diffie Hellman Method and Certification Chains
    - PKINIT Kerberos

Key-Distribution Protocols using Asymmetric cryptography

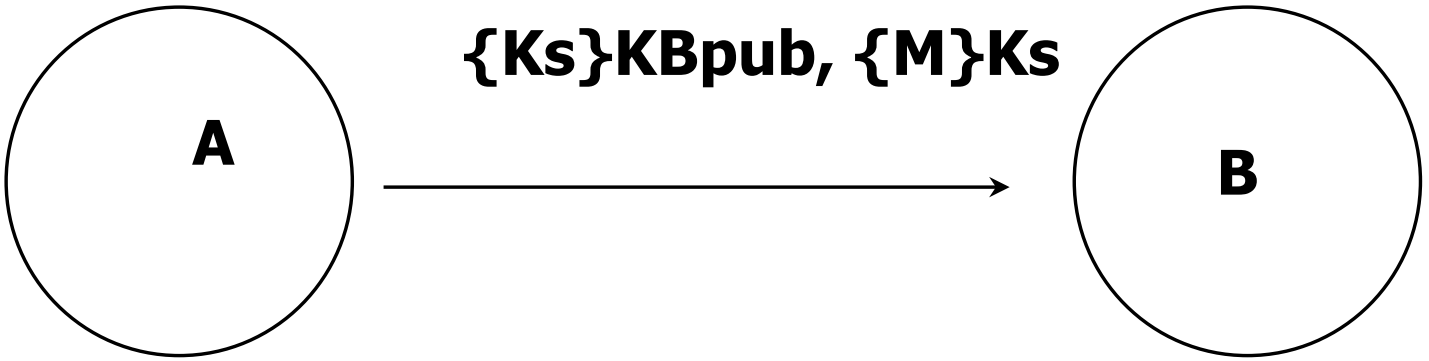
- Use of Asymmetric Cryptography Methods, Algorithms and Constructions to support Key Distribution Protocols
  - Based on the notion of PKCs (Public Key Centers)
- Diffie-Hellman Agreement Method is one of such solutions
  - Contributive Keys from DH Agreements
  - PFS and PBS guarantees
- But other Asymmetric Cryptographic – Public Key Methods can be used together with the use of PKCs

Outline

- **Key-Distribution Protocols using Asymmetric cryptography**
  - Base P2P Key-Establishment using Public-Key Cryptography
  - Needham-Schroeder Base Model (w/ a PKC)
  - P2P Key Establishment w/ Certification Chains
  - Examples
    - P2P Key Establishment using the Diffie Hellman Method and Certification Chains
    - PKINIT Kerberos

Base: use of Public Key Methods for KDPs (1)

Peer-to-Peer Confidential Key Distribution using **Public-Key Confidential Envelopes**

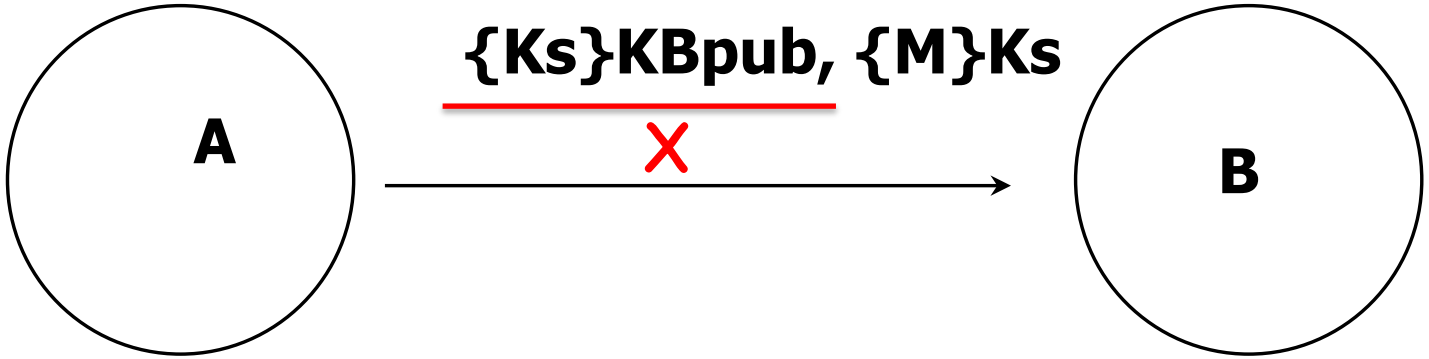


Can do it with Public Key Algorithms such as: RSA, ElGammal. ECC, etc…

If principals can trust on validity of public-keys as the correct public keys of other principals

Base: use of Public Key Methods for KDPs (2)

Peer-to-Peer Confidential Key Distribution using
Public-Key Confidential Envelopes
with Proof of Authentication



Can also include a Digital Signature of envelope X
for Peer-Authenticity Purposes:
$X, \{X\}_{KprivA}$

When we must be sure about the peer-authentication of principals !!!!

Key-Distribution with Secure Envelopes

- More complete example w/ Asymmetric Cryptography

A generates a Key Session (Ks)

A > B:

$\{Ks\}K_{pubB} \;||\; \{M\}_{Ks} \;||\; Sign_A(M) \;||\; H(MCC)$

Key-Distribution Envelope Part

Confidential Part

Peer-Authenticity Proof Part (Digital Signature)

Integrity Check (also possible with MACs)

MCC: Message Content Check

Key-Distribution with Secure Envelopes

- More complete example w/ Asymmetric Cryptography

  A generates a Key Session (Ks, Kmac or any secrecy parameters)

  A > B:

  $\{Ks, Kmac\}K_{pubB} \;||\; \{M\}_{Ks} \;||\; Sign_A(M) \;||\; MAC_{Kmac}(MCC)$

Key-Distribution Envelope Part

Confidential Part

Peer-Authenticity Proof Part (Digital Signature)

Fast Message Authentication and Integrity Check (possible with HMACs or CMACs)

MCC: Message Content Check

Principle of Contributive Peer-KDPs

A: generates a Key Session (Ks1 secret)

A > B:

$\{Ks_A, KmacA\}K_{pubB} \,||\, Sign_A(\ldots) \,||\, MAC_{KmacA}(MCC)$

B generates a Key Session (Ks1 secret)

B: > A:

$\{Ks_B, KmacB\}K_{pubB} \,||\, Sign_B(\ldots.) \,||\, MAC_{KmacB}(MCC)$

$$\text{Established Ks} = f\,(KS_A, KS_B\ , \ldots..)$$

Key-Distribution with Authenticated Diffie-Hellman Agreement

- **A** and **B** decide to share a Primitive Root and a Prime Number P

  A: generates a Private (Xa) and Public (Ya) Diffie Hellman Numbers

  A -> B:

  Ya || Sig_A(Ya) || H(MCC)

  Authenticity Proof Part (Digital Signature)

  Integrity Check (also possible with MACs)

  MCC: Message Content Check

Key-Distribution with Authenticated Diffie-Hellman Agreement

- **A** and **B** decide to share a Primitive Root and a Prime Number P

  B: generates a Private (Xb) and Public (Yb) Diffie Hellman Numbers

  B -> A:

  Yb || $Sig_B(Yb)$ || H(MCC)

Authenticity Proof Part (Digital Signature)

Integrity Check (also possible with MACs)

MCC: Message Content Check

Key-Distribution with Authenticated Diffie-Hellman Agreement

- **A** and **B** can compute independently a Session Key **K**
- Only using the Authenticated Diffie Helman Public Numbers !
  - (and at most the pre-shared DH Parameters)
- For each new Authenticated Diffie Hellman exchange with new (Public, Private) Diffie Hellman Numbers, the Session Key is established with:
  - Contributive Guarantees
  - Independent from any keys established in the past
  - Perfect Forward and Perfect Backward Secrecy Guarantees

Dynamically Negotiated Ciphersuites:

- A and B can negotiate dynamically all the ***ciphersuites*** they need
- Ex:, can use "special labeling-codes" to express the ***ciphersuites*** with flexibility, with standardized definitions

Code	Label
0x00,0xBD	DHE_DSS_WITH_AES_256_CBC_SHA
0x00 0x84	RSA_WITH_CAMELLIA_128_CBC_SHA
0x00,0xA1	RSA_WITH_AES_128_GCM_SHA384
0x00,0xBD	DHE_DSS_WITH_AES_256_GCM_SHA384
0x00,0xAE	RSA_PSS_WITH_AES_256_GCM_SHA384
0x00,0xC2	DH_RSA_WITH_CAMELLIA_256_CBC_SHA256
0xC0,0x28	ECDHE_RSA_WITH_AES_256_CBC_SHA384
Etc etc	

See, ex, for TLS:
https://www.iana.org/assignments/tls-parameters/tls-parameters.xhtml

Practical Discussion:
Revisitation of Work Assignmemt #1

- Discussion on the guarantees of the SHP Protocol (Phase 2) as a secure "handshake" protocol to establish keys and other secrecy parameters for SSP.

Outline

- **Key-Distribution Protocols using Asymmetric cryptography**
  - Base P2P Key-Establishment using Public-Key Cryptography
  - Needham-Schroeder Base Model (w/ a PKC)
  - P2P Key Establishment w/ Certification Chains
  - Examples
    - P2P Key Establishment using the Diffie Hellman Method and Certification Chains
    - PKINIT Kerberos

Needham-Schroeder with asymmetric cryptography

Components of the protocol and assumptions
PKC: has a pair (KprivPKC, KpubPKC)
PKC: resgistration service for (A, KpubA), (B, KpubB): TCB
A, has a key pair, KprivA, KpubA
B, has a key pair, KprivB, KpubB
Nonces: Na, Nb

A > PKC : A, B

PKC > A : { KpubB, B }KprivPKC

A > B : { Na, A } KpubB

B > PKC: B, A

PKC > B: { KpubA, A} KprivPKC

B > A: {Na+1, Nb, Ks}KpubA

A > B: { Nb+1 }Ks

Ks generation and distribution from A

Needham-Schroeder with asymmetric cryptography

1. A -> PKC : **A,B**

2. PKC -> A : **{ KBpub, B } KPKCpriv**

3. A -> B: **{ Na, A }KBpub**

---

4. B obtains (in a trust way) the KpubA from the PKC

---

5. B -> A: **{ Na+1, Nb, B, Ks } KApub**

In this case, B generates the session key

6. A -> B: **{ Nb+1 } Ks**

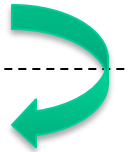
A replies with the response to the nonce Nb, which authenticates A. Question: is B authenticated from the A viewpoint ?

Needham-Schroeder with asymmetric cryptography

1. A -> PKC : **A,B**

2. PKC -> A : **{ KBpub, B } KPKCpriv**

3. A -> B: **{ Na, A }KBpub, { A, KApub } KPKCpriv**



B obtains (in a trust way) the KpubA from the PKC

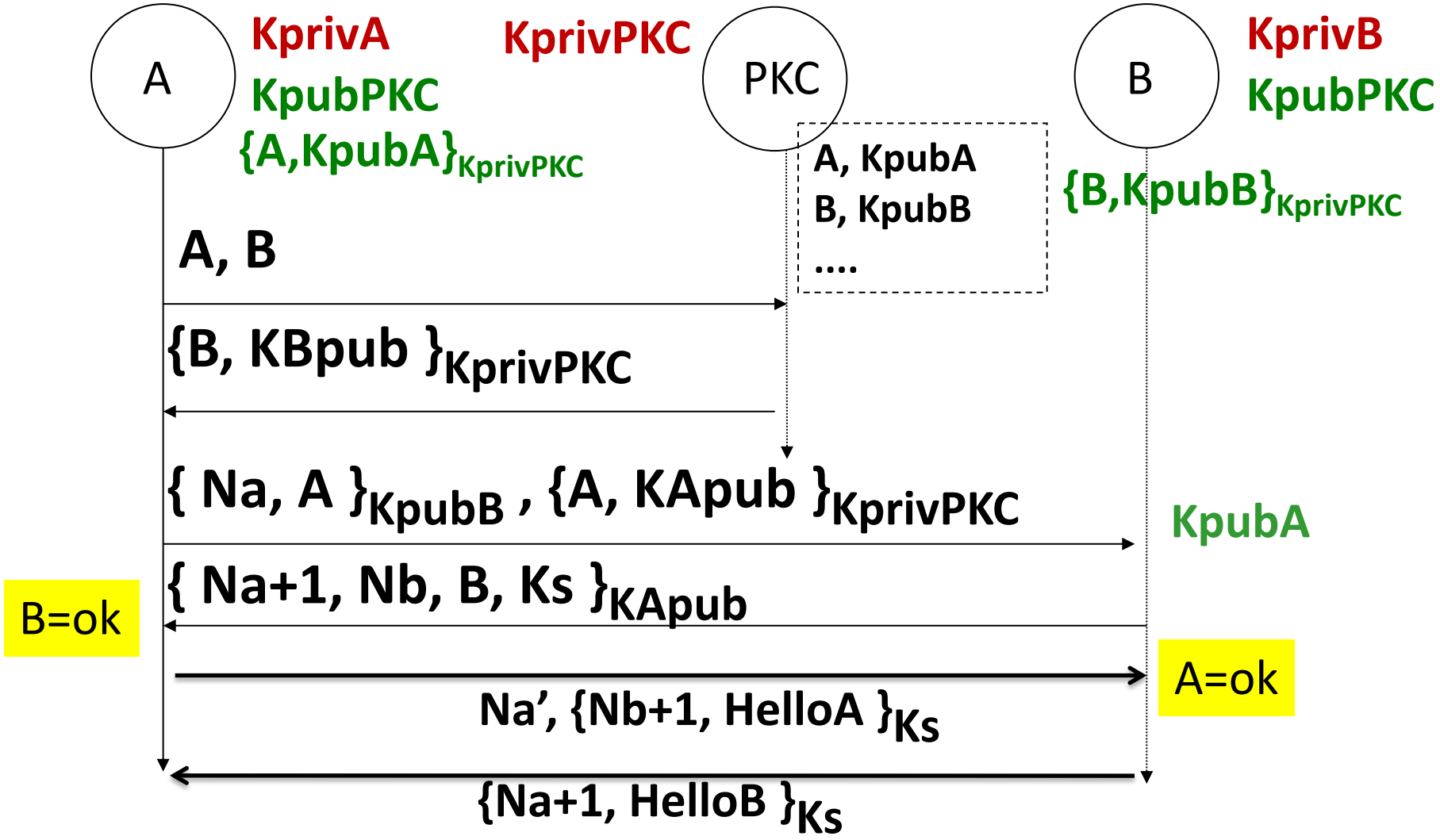
4. B -> A: **{ Na+1, Nb, B, Ks } KApub**

In this case, B generates the session key

5. A -> B: **{ Nb+1 } Ks**

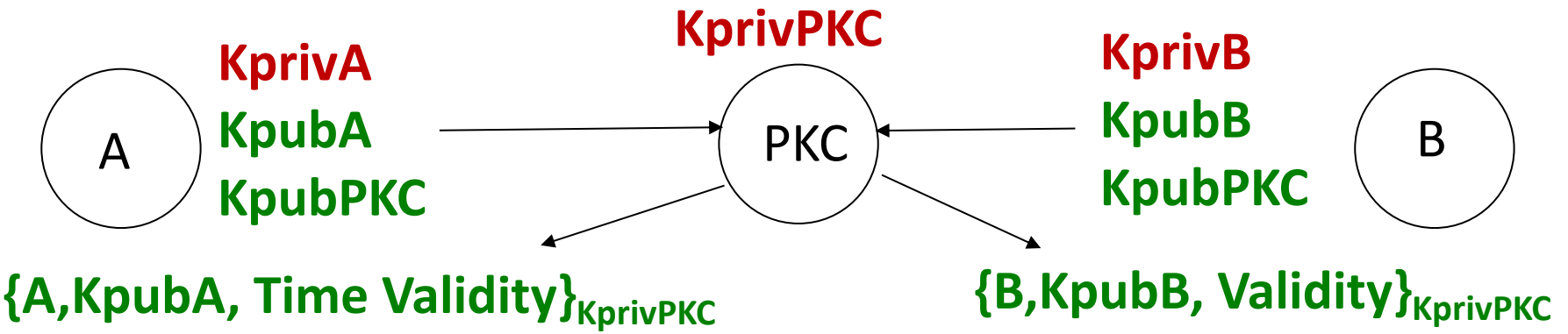
A replies with the response to the nonce Nb, which authenticates A. Question: is B authenticated from the A viewpoint ?

Needham Schroeder Base Method using Asymmetric Cryptography

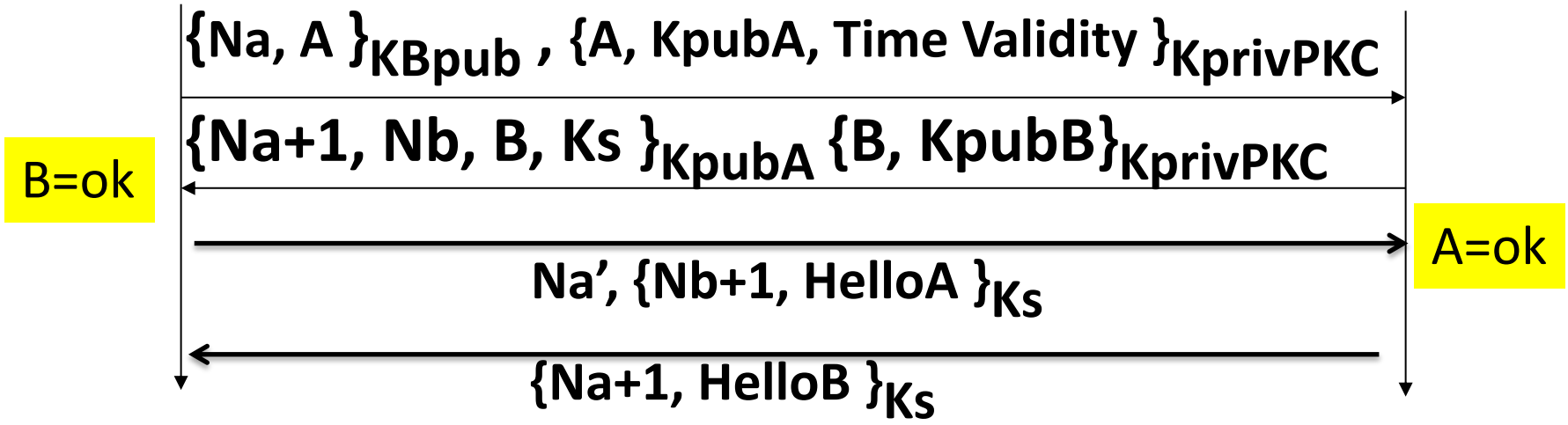


Confidentiality with symmetric crypto and session key Ks (in this case generated by B

What if wach pair has initially its <PubKey, ID> Association signted by the trusted PKC ?



Pre-Enrolment: (A, PKC) , (B, PKC)

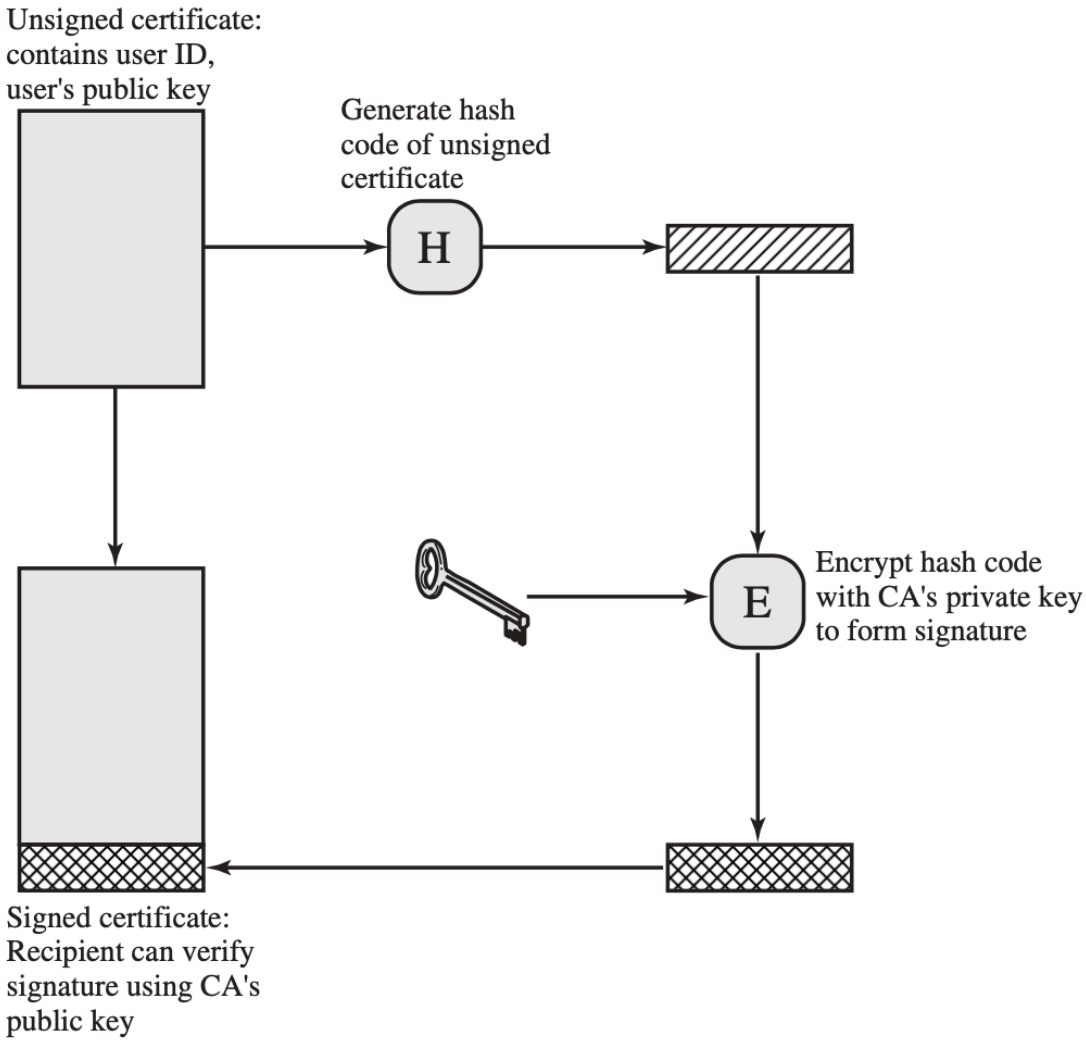


Confidentiality with symmetric crypto and session key Ks (in this case generated by B

Advantages …

- We only need a PKC (as a TCB)
  - The trust assumption is that the PKC gives (distributes) Public Keys of Principals in a correct way (correct verification of the association, signed by the PKC)
    - There is an enrolment (pre, asynchronous) process, verified by the PKC
  - This means that the PKC is trustable, acting as **a certifier of the Public Keys**
  - **So … We can think on PKC as an embrionary T-CA (Trusted Certification Authority)**

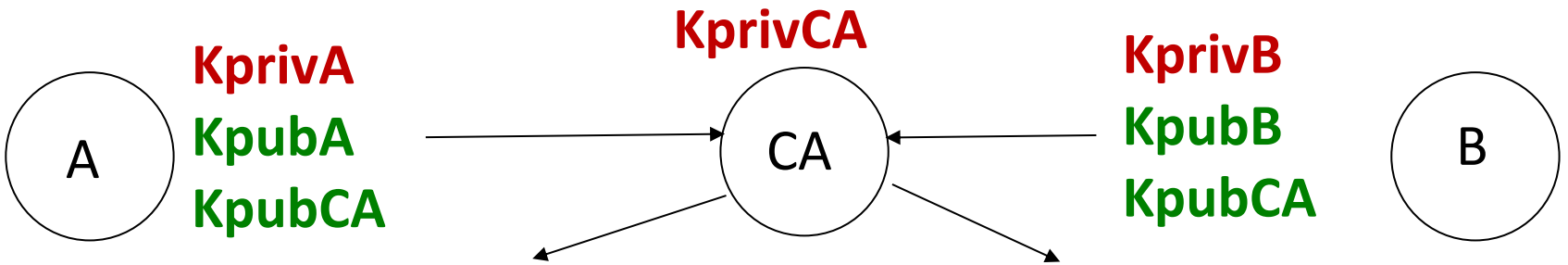
Base construction for an issued Public Key Certificate from a CA



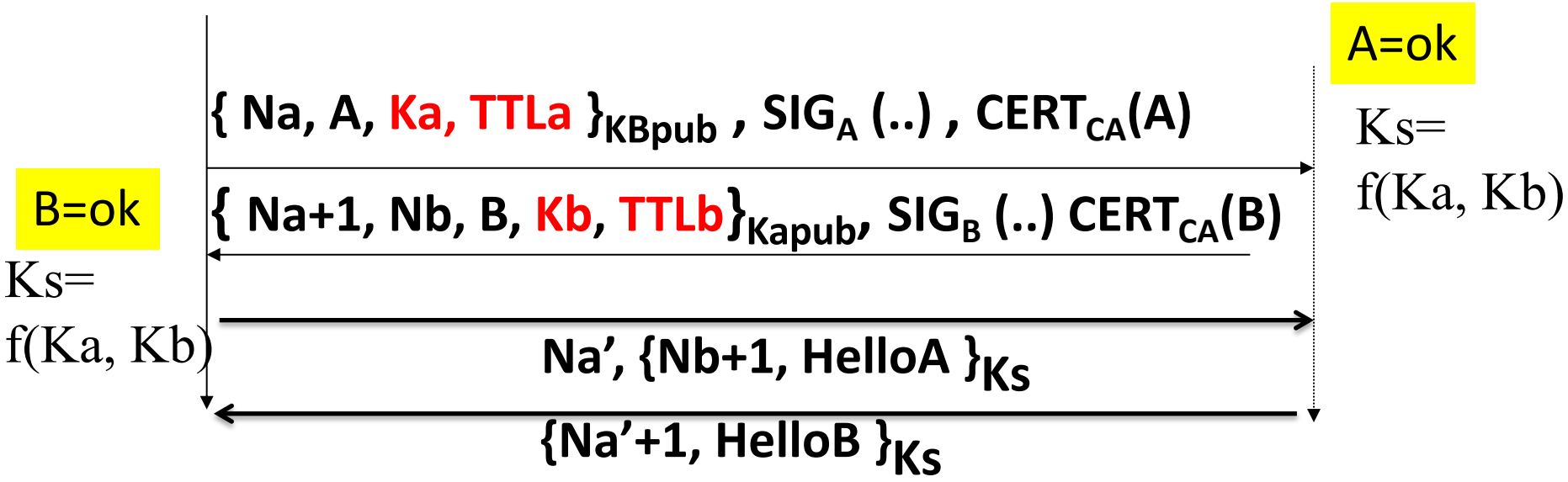
Outline

- **Key-Distribution Protocols using Asymmetric cryptography**
  - Base P2P Key-Establishment using Public-Key Cryptography
  - Needham-Schroeder Base Model (w/ a PKC)
  - P2P Key Establishment w/ Certification Chains
  - Examples
    - P2P Key Establishment using the Diffie Hellman Method and Certification Chains
    - PKINIT Kerberos

P2P Key-Establishment (w/ Contributive Keys)



Pre-Enrolment: (A, PKC) , (B, PKC)



TTL of Ks = min(TTLa, TTLb). When expired => REKeying !

P2P Key-Establishment (w/ Contributive Keys)

A

KprivA
CertCA

KprivCA
CertCA

B

KprivB
CertCA

CertA=
{A, ... Attributes, KpubA}$_{KprivCA}$

CertB =
{B, Attributes, KpubB}$_{KprivCA}$

Ks=
f(Ka, Kb)

B=ok

{ Na, A, Ka }$_{KBpub}$ ||CertA

{ Na+1, Nb, B, Kb }$_{Kbpub}$, CertB

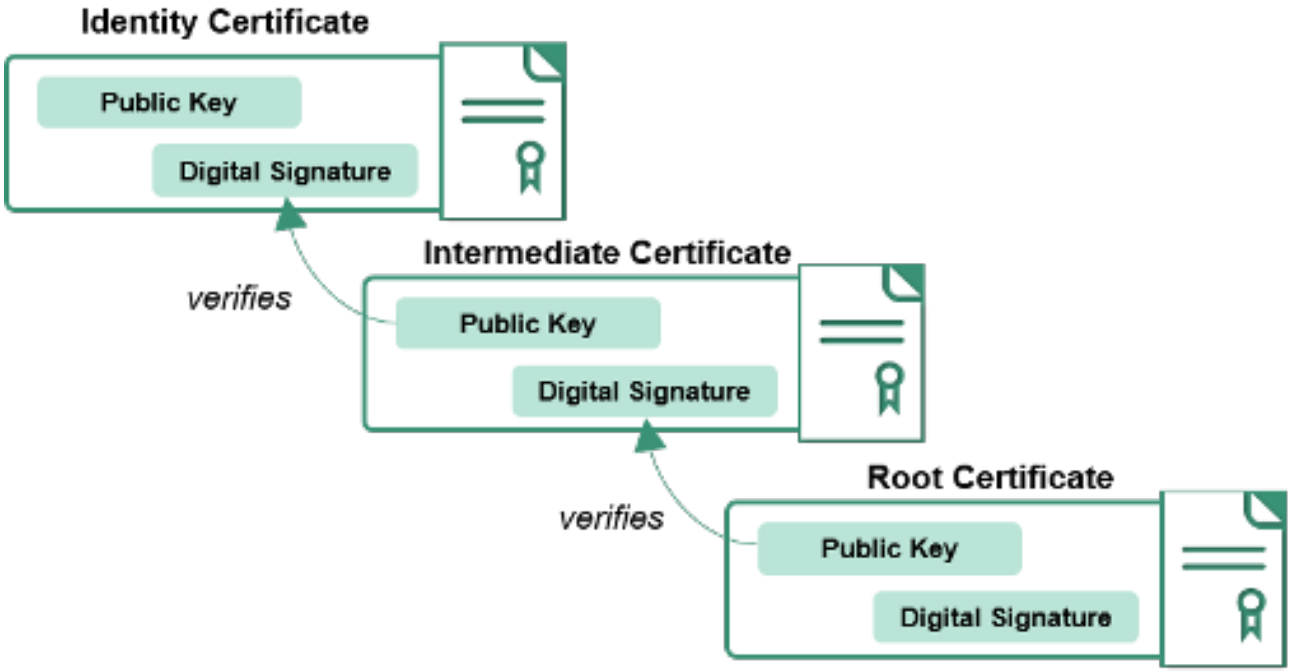
Na', {Nb+1, HelloA }$_{Ks}$

{Na+1, HelloB }$_{Ks}$

Ks=
f(Ka, Kb)

A=ok

Certification Chains



The Certification Chain:
CertCA-ROOT || CertCA-Intermediate || .... || CertCA-Indentity

What if there is a MiM ?

The ultimate TRUST: The Public Key of the CA

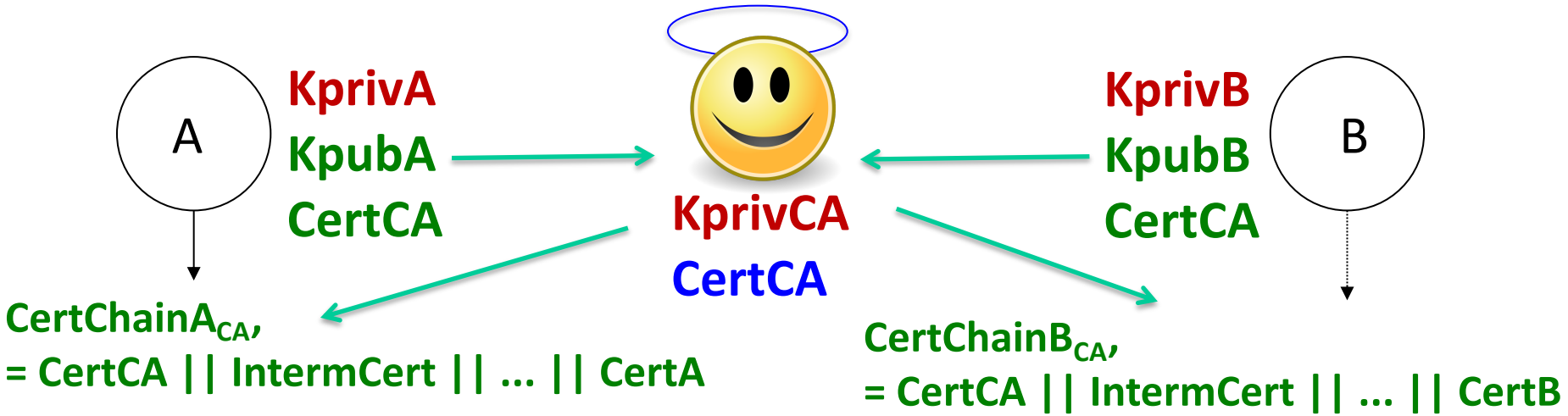
Meaning the root of TRUST:

- A TRUSTED CA ROOT Certificate

Can have chains with N certificates … Only need to have the ROOT of the Chain trusted, and intermediate certificates (with authority) to act as issuers of the successive certificates in the chain

OK we will discuss later these issues in the X509 Certification and Authentication Model and the Assumptions of this standard

P2P Key-Establishment (w/ Contributive Keys)



Ks= f(Ka, Kb)

$\{Na, A, Ka\}_{KBpub}$, SIG_A (..) , $CertChainA_{CA}$ →

B=ok

← $\{Na+1, Nb, B, Kb\}_{Kbpub}$, SIG_B (..) , $CertChainB_{CA}$

Na', $\{Nb+1, HelloA\}_{Ks}$ →

A=ok

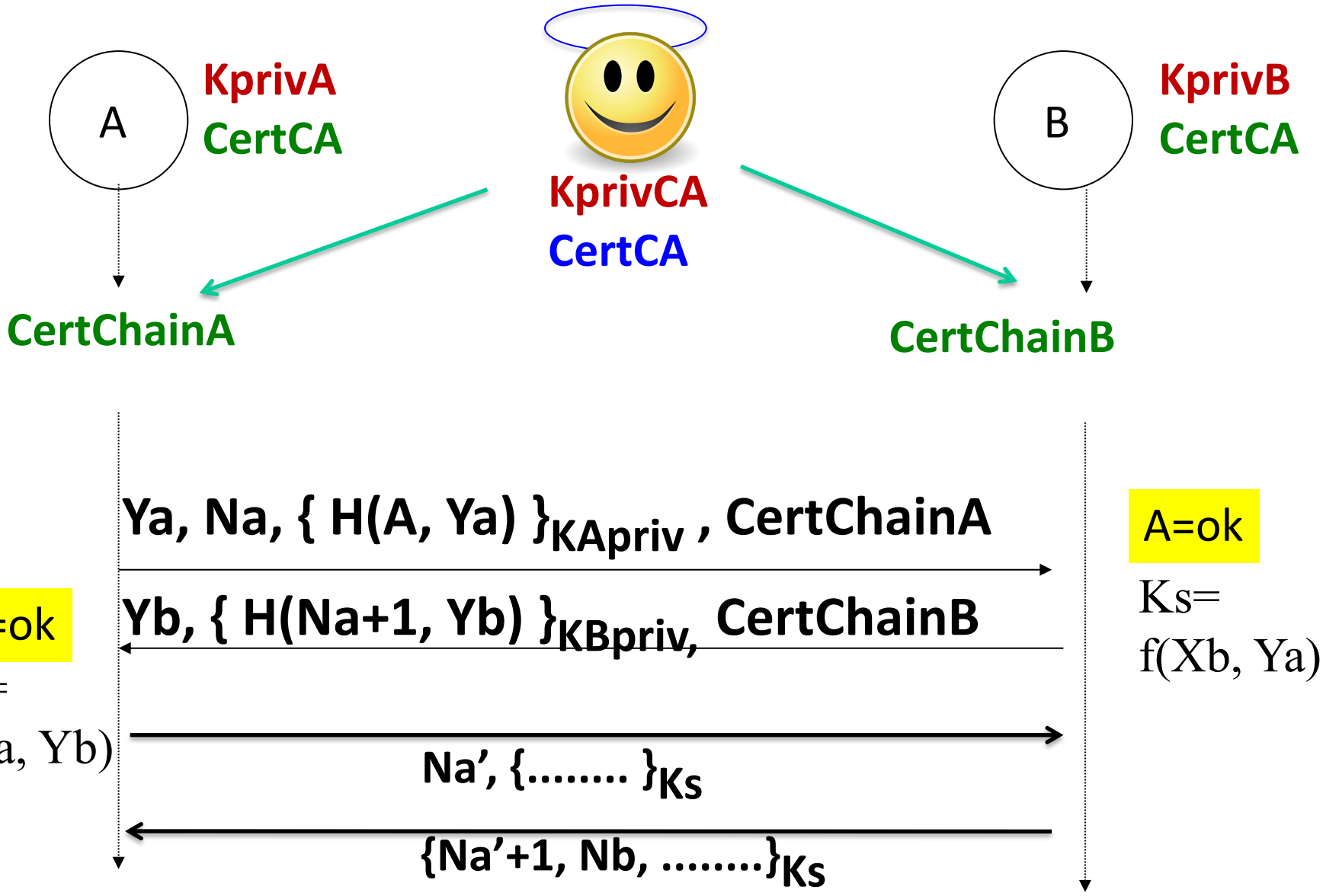
← $\{Na+1, HelloB\}_{Ks}$

Ks= f(Ka, Kb)

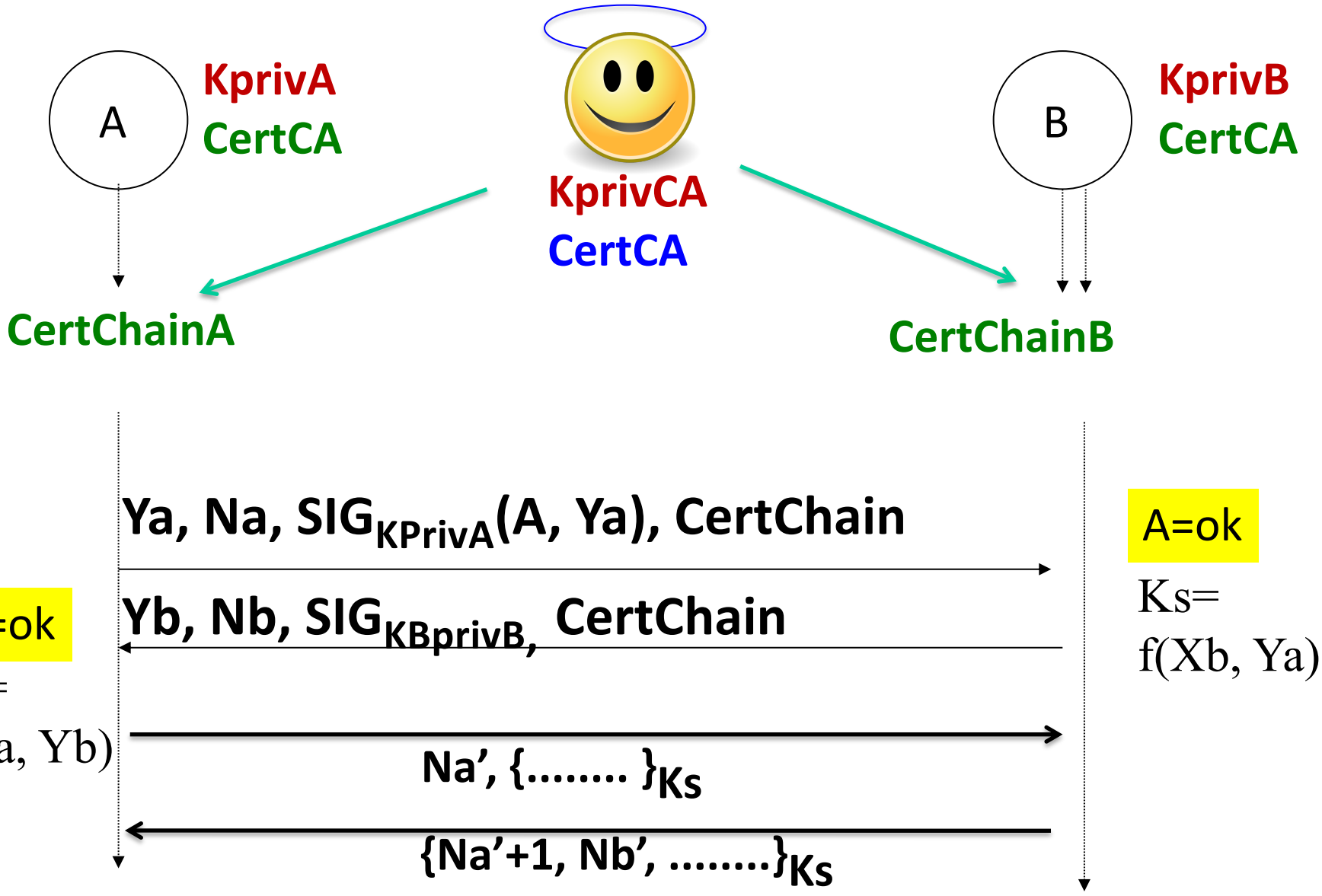
Outline

- **Key-Distribution Protocols using Asymmetric cryptography**
  - Base P2P Key-Establishment using Public-Key Cryptography
  - Needham-Schroeder Base Model (w/ a PKC)
  - P2P Key Establishment w/ Certification Chains
  - Examples
    - P2P Key Establishment using the Diffie Hellman Method and Certification Chains
    - PKINIT Kerberos

P2P Key-Establishment (w/ Authenticated DH)



P2P Key-Establishment (w/ Authenticated DH)



Analysis of the Distribution and Establishment of Key and Security Association Parameters in the Practical Work Assignment

SHP Protocol: Secure Handshake Protocol

Outline

- **Key-Distribution Protocols using Asymmetric cryptography**
  - Base P2P Key-Establishment using Public-Key Cryptography
  - Needham-Schroeder Base Model (w/ a PKC)
  - P2P Key Establishment w/ Certification Chains
  - Examples
    - P2P Key Establishment using the Diffie Hellman Method and Certification Chains
    - PKINIT Kerberos

Remembering the Kerberos Protocol (V5)

(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

PK-INIT Kerberos Model

Clients and AS use Public Key Certificates

Authentication Round:

C -> AS:

CertChain || Options || Id_C || $Realm_C$ || ID_{TGS} || TS || N_c || Sig_C (X) || H(M)

X

M

AS-> C:

X

CertChain || $Realm_C$ || $Ticket_{TGS}$ || E_{KpubC} ($K_{C,TGS}$ || TS || Nc+1 || ID_{TGS})
|| $_{SigAS}$ (X) || H(M)

$$M = X \,||\, _{SigAS}\,(X)$$

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,
4.3 – Key Distribution using Asymmetric Encryption