

# Announcements

## Homework 1

- Grade released
- Have ***1-week “rebuttal period”***
  - Submit re-grade request via GradeScope

# Lecture 10

## Protocols (Continued)

### Chapters 9 and 11 in KPS

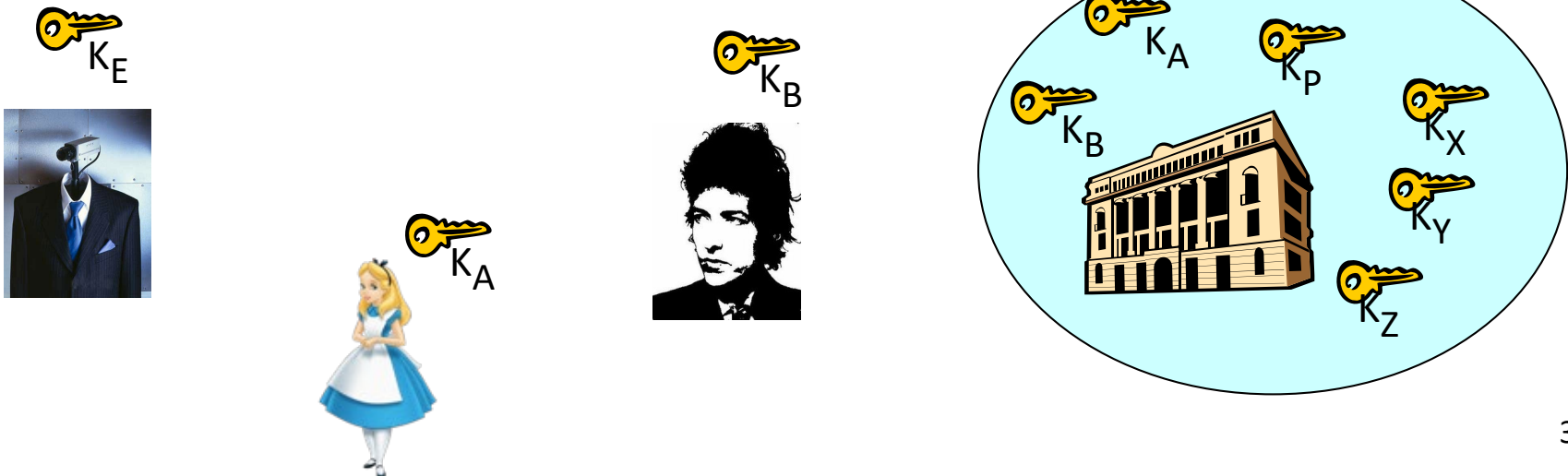
[lecture slides are adapted from previous slides by Prof. Gene Tsudik]

# Recap: Key Distribution Center (KDC) aka Trusted Third Part (TTP)

- Alice and Bob need to share a key
- KDC shares different master key with *each* registered user (many users)
- Alice and Bob know their own master keys:

$K_A$  and  $K_B$

for communicating with KDC



# Key Distribution Center (KDC) or Trusted Third Party (TTP)

$K(X)$  = Encryption of X with key K

KDC generates fresh K



Alice  
Obtains  
K

Msg1:  $K_A(A,B)$



Msg2:  $K_A(K, K_B(A,K))$



Bob obtains K and  
knows to use as a  
key for  
communicating with  
Alice

Msg3:  $K_B(A,K)$

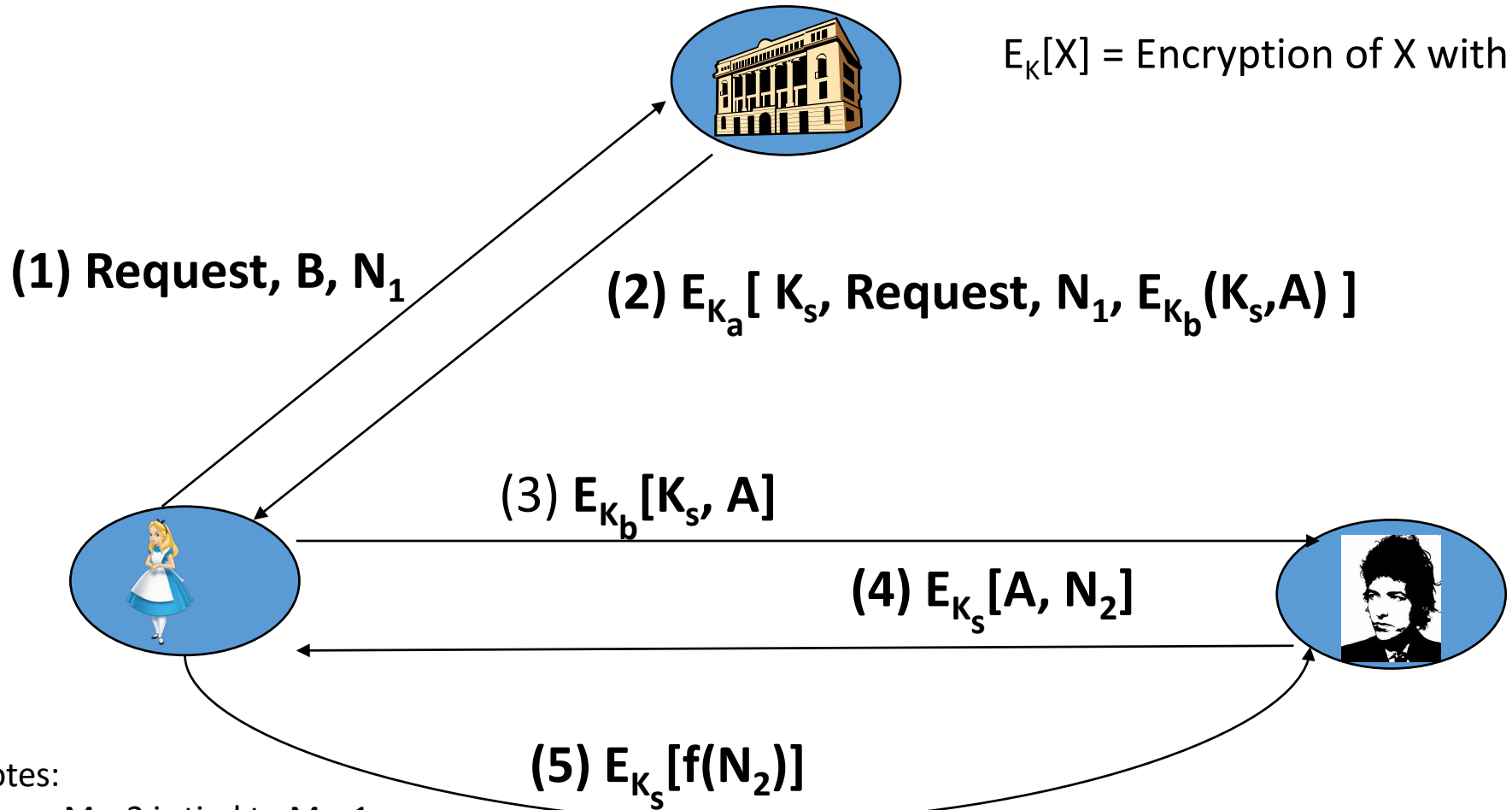
- Alice and Bob communicate using **K** as a short-term (*session*) key for encryption and/or data integrity

- Note:

- Msg2 is not tied to Msg1
- Msg1 is possibly old
- Msg2 is possibly old and so is Msg3
- Bob and Alice don't authenticate each other!

# A Typical Key Distribution Scenario

$E_K[X]$  = Encryption of X with K



Notes:

- Msg2 is tied to Msg1
- Msg2 is fresh/new
- Msg3 is possibly old \*
- Msg1 is possibly old (KDC doesn't authenticate Alice)
- Bob authenticates Alice
- Bob authenticates KDC
- Alice DOES NOT authenticate Bob

# Public Key Distribution

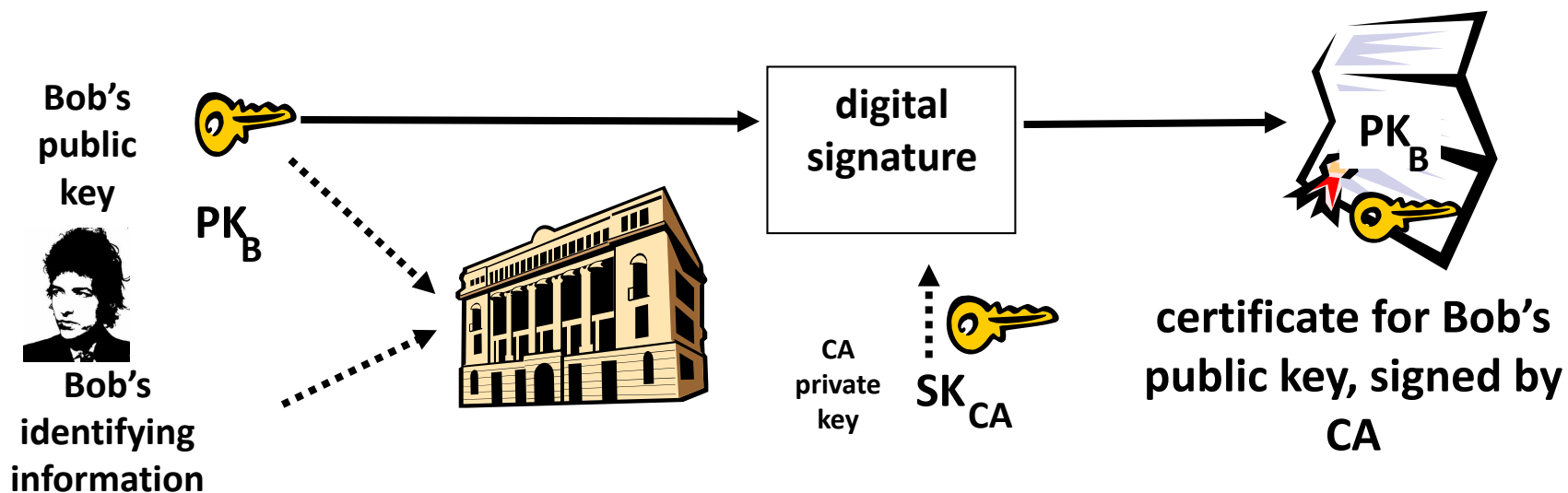
## General schemes:

- Public announcement (e.g., in a newsgroup or email message)
  - Can be **forged**
- Publicly available directory
  - Can be **tampered with**
- Public-key certificates (PKCs) issued by trusted off-line **Certification Authorities (CAs)**

# Certification Authorities

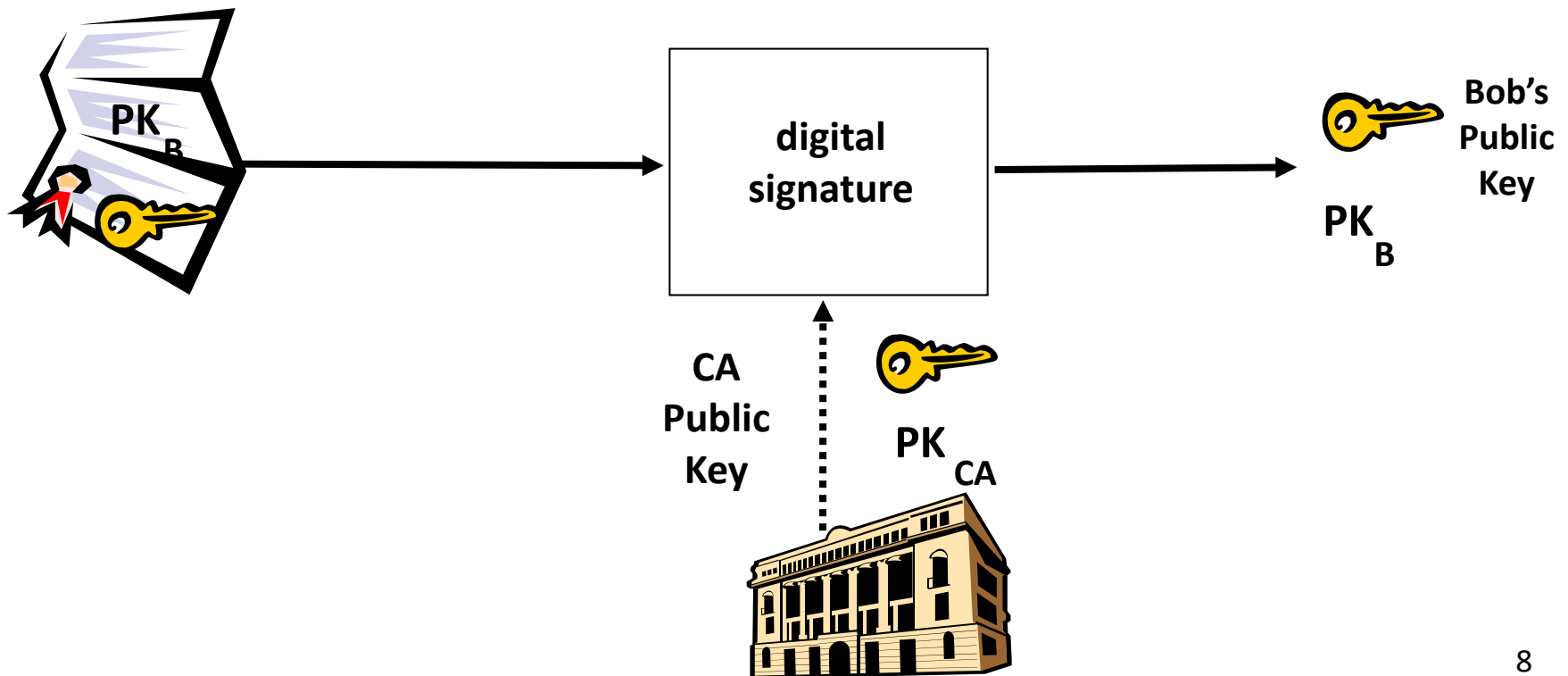
- Certification authority (CA): trusted, highly secure (physically and electronically) component
- Issues public key certificates; each binds a public key to a specific entity
- Each entity (user, host, etc.) registers its public key with CA.
  - Bob provides “proof of identity” to CA.
  - CA creates public key certificate binding Bob’s ID/name to this public key.
  - Certificate containing Bob’s public key is signed by CA:

**CA says: “this is Bob’s public key”**



# Certification Authority

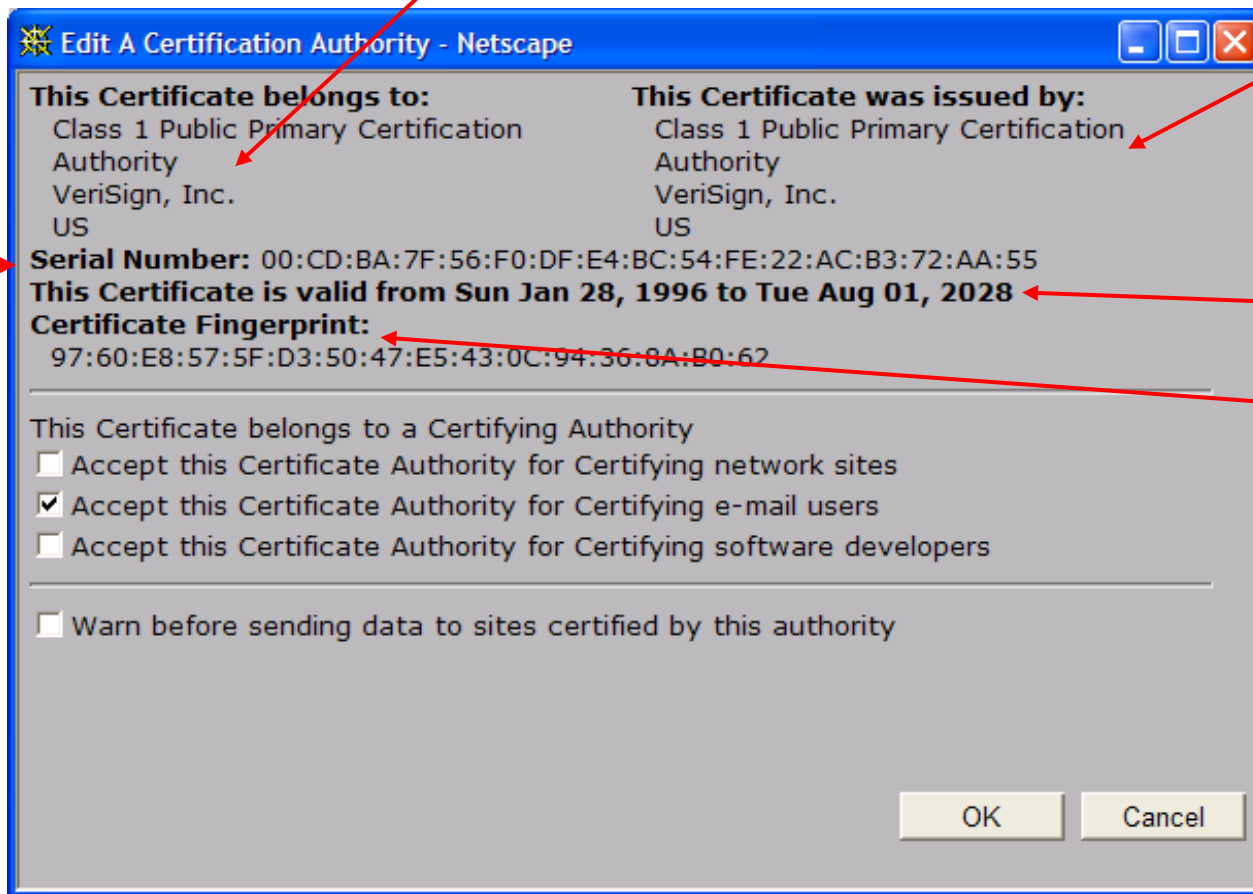
- When Alice wants to get Bob's public key:
  - Get Bob's certificate (from Bob or elsewhere)
  - Using CA's public key verify the signature on Bob's certificate
  - Check for expiration
  - Check for revocation (we'll talk about this later)
  - Extract Bob's public key





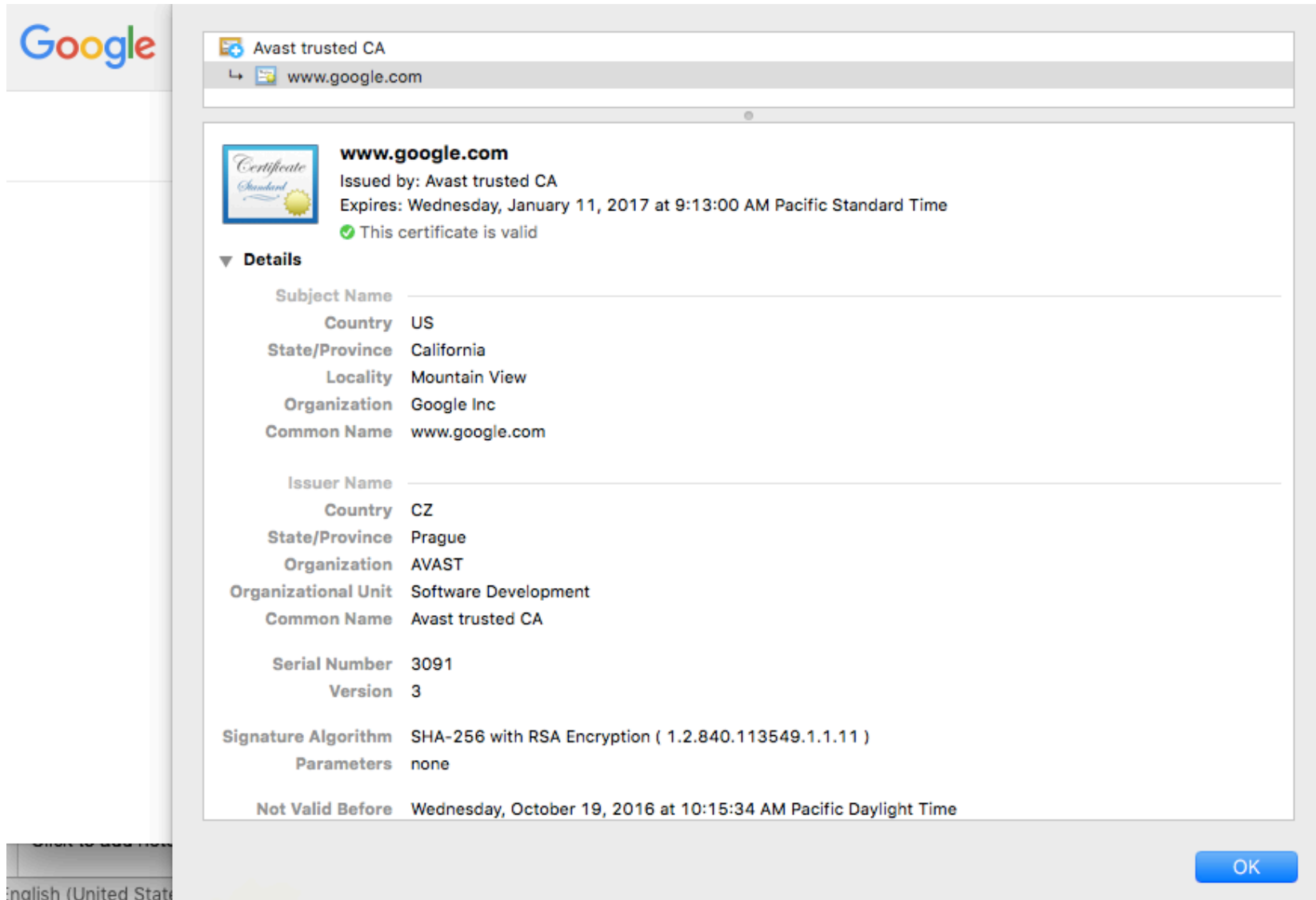
# A Certificate Contains

- Serial number (unique to issuer)
- Info about certificate owner, including algorithm and key value itself (not shown)



- info about certificate issuer
- valid dates
- digital signature by issuer

# A Sample Certificate (1/2)



The screenshot shows a web browser window with the Google logo in the top left corner. The address bar displays "Avast trusted CA" and "www.google.com". The main content area shows a certificate for "www.google.com" issued by "Avast trusted CA". The certificate is valid and expires on Wednesday, January 11, 2017 at 9:13:00 AM Pacific Standard Time. A "Details" section is expanded, showing the following information:

<b>Subject Name</b>	
<b>Country</b>	US
<b>State/Province</b>	California
<b>Locality</b>	Mountain View
<b>Organization</b>	Google Inc
<b>Common Name</b>	www.google.com
<b>Issuer Name</b>	
<b>Country</b>	CZ
<b>State/Province</b>	Prague
<b>Organization</b>	AVAST
<b>Organizational Unit</b>	Software Development
<b>Common Name</b>	Avast trusted CA
<b>Serial Number</b>	3091
<b>Version</b>	3
<b>Signature Algorithm</b>	SHA-256 with RSA Encryption ( 1.2.840.113549.1.1.11 )
<b>Parameters</b>	none
<b>Not Valid Before</b>	Wednesday, October 19, 2016 at 10:15:34 AM Pacific Daylight Time

An "OK" button is located in the bottom right corner of the certificate display area.

# A Sample Certificate (2/2)



Avast trusted CA

www.google.com

## Public Key Info

**Algorithm** RSA Encryption ( 1.2.840.113549.1.1.1 )  
**Parameters** none  
**Public Key** 256 bytes : D7 D3 86 4F 23 D4 E6 E4 ...  
**Exponent** 65537  
**Key Size** 2048 bits  
**Key Usage** Any  
**Signature** 256 bytes : 97 6B 72 86 AD 24 65 AD ...

**Extension** Subject Key Identifier ( 2.5.29.14 )

**Critical** NO

**Key ID** 84 61 D1 1A 2F B1 EF 8E 4F F4 6F F0 8D 26 FC 91 58 77 9C A3

**Extension** Authority Key Identifier ( 2.5.29.35 )

**Critical** NO

**Key ID** DB D4 F7 BB 15 76 6C 3B 01 A5 23 59 C2 37 26 97 46 5D DC 46

**Extension** Subject Alternative Name ( 2.5.29.17 )

**Critical** NO

**DNS Name** www.google.com

## Fingerprints

**SHA1** 30 69 24 F3 14 57 D4 84 73 7F B2 BE B8 F5 92 A2 46 8E 9D 2E

**MD5** 20 CD 07 D1 A3 F4 96 95 2F 33 43 4D E6 F3 D0 1E

OK

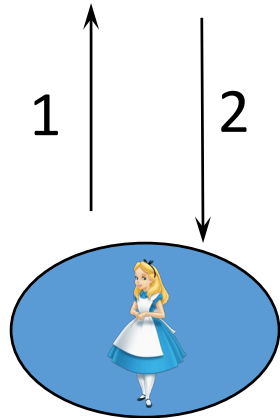
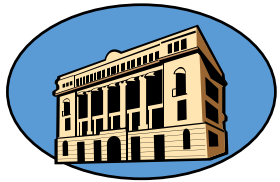
# Back to Protocols

# Needham-Schroeder Protocol (1978):

## First Distributed Security Protocol

$\{X\}_K$  = Encryption of X with key K

1.  $A \rightarrow T: A, B, N_A$
2.  $T \rightarrow A: \{N_A, B, K, \{K, A\}_{K_B}\}_{K_A}$
3.  $A \rightarrow B: \{K, A\}_{K_B}$
4.  $B \rightarrow A: \{N_B\}_K$
5.  $A \rightarrow B: \{N_B^{-1}\}_K$



1

2

3

4

5

# Security?

**Denning-Sacco Attack:** suppose Eve recorded **an old** protocol session for which she somehow knows the session key  $K'$ :

1.  $A \rightarrow T$ :  $A, B, N_A$
2.  $T \rightarrow A$ :  $\{N_A, B, K', \{K', A\}_{K_B}\}_{K_A}$
3.  $A \rightarrow B$ :  $\{K', A\}_{K_B}$

-----  
At a later time:

3.  $E \rightarrow B$ :  $\{K', A\}_{K_B}$
4.  $B \rightarrow E$ :  $\{N_B\}_{K'}$
5.  $E \rightarrow B$ :  $\{N_B - 1\}_{K'}$

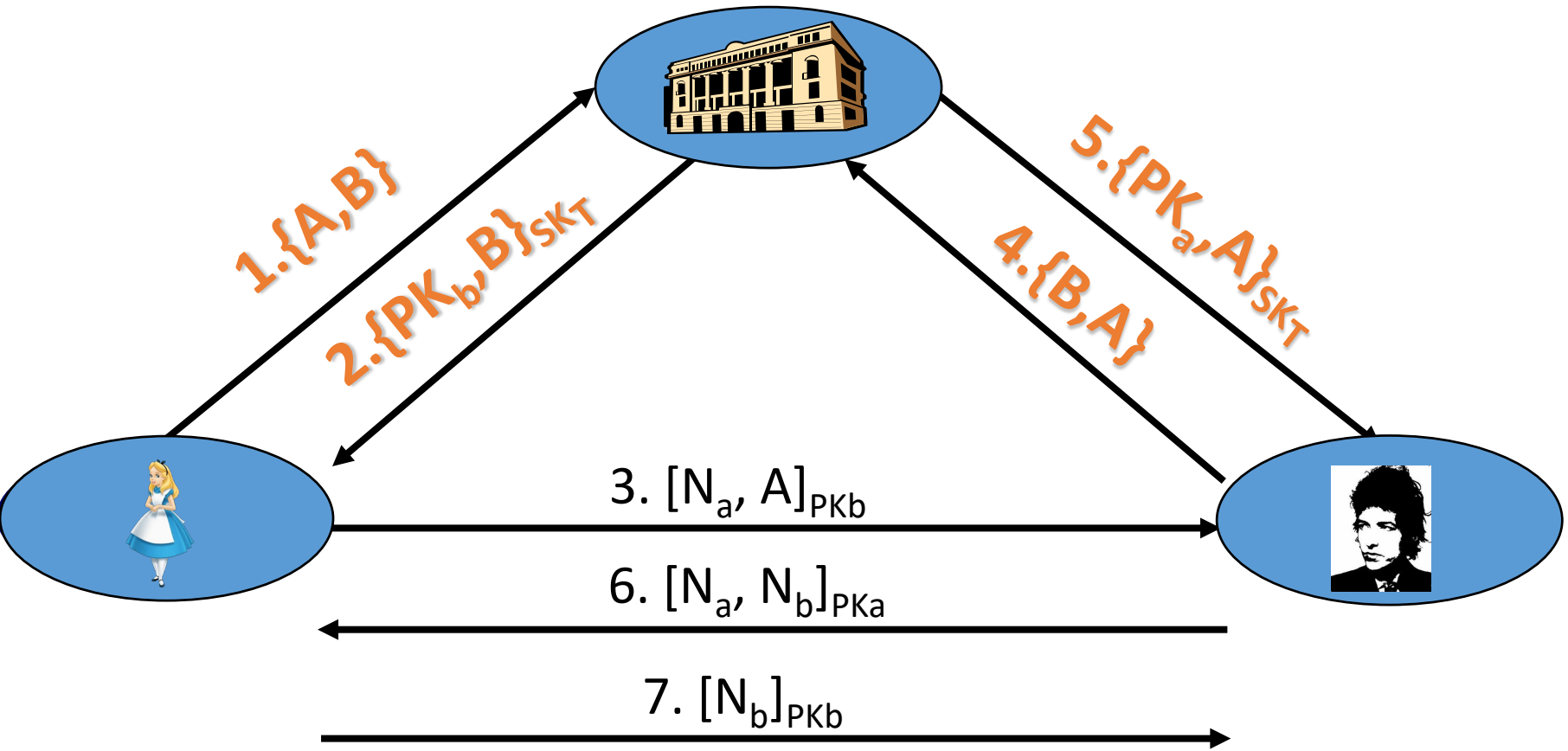
# Fixing the Attack

- Bob has no guarantees about freshness of the message in step 3.
- Eve exploits this to impersonate Alice to Bob - old session keys are useful.
- Can be fixed by adding timestamps:
  - Limits usefulness of old session keys
  - Eve's attack becomes:

3: E  $\rightarrow$  B:  $\{K', T', A\}_{K_B}$

attack is now thwarted because  $T'$  is stale

# PK-based Needham-Schroeder Protocol



- $CERT_B = \text{Message 2}$ ,  $CERT_A = \text{Message 5}$
  - $PK_A$ : Alice's public key,  $PK_B$ : Bob's public key
  - $SK_T$ : TTP's secret (private) key used for signing
  - Everyone knows TTP's public key  $PK_T$
- $[X]_K = \text{Encryption of } X \text{ with key } K$



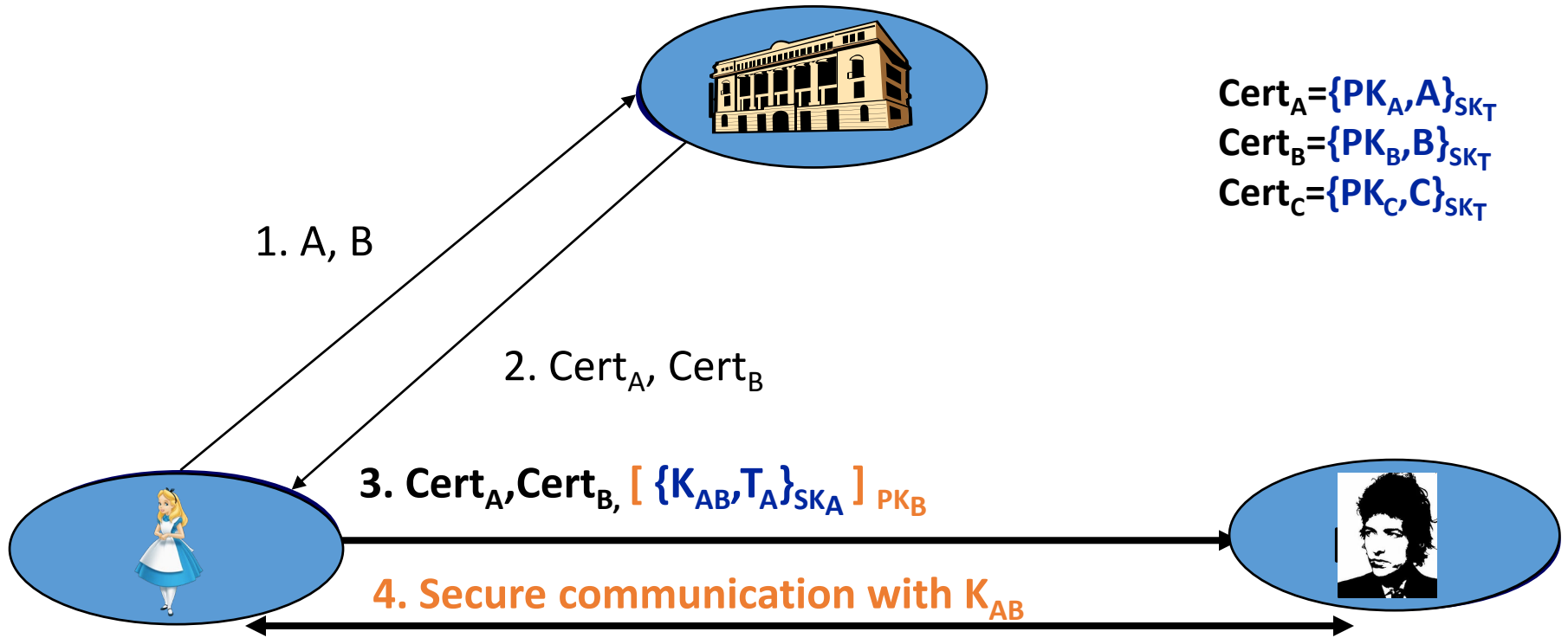
# Another Attack

- 1, 2, 4, 5: Delivery of public key
- Does not guarantee freshness of the public key

## How to solve it?

- Timestamp in messages 2 and 5 or challenges in messages 1&2 and 4&5
- Public Key Certificate: assign expiration time/data to each certificate (messages 2 and 5)

# PK-based Denning-Sacco Attack



Thinks she is talking to A

Bob impersonates Alice



# Lowé's Attack (Impersonation by Interleaving)

## Original

- 3.  $A \rightarrow B: [N_a, A]_{PK_b}$
- 6.  $B \rightarrow A: [N_a, N_b]_{PK_a}$
- 7.  $A \rightarrow B: [N_b]_{PK_b}$

## Fix

- 3.  $A \rightarrow B: [N_a, A]_{PK_b}$
- 6.  $B \rightarrow A: [B, N_a, N_b]_{PK_a}$
- 7.  $A \rightarrow B: [N_b]_{PK_b}$

## Attack: E impersonates A

- 3.  $A \rightarrow E: [N_a, A]_{PK_e}$
- 3.  $E \rightarrow B: [N_a, A]_{PK_b}$
- 6.  $B \rightarrow E: [N_a, N_b]_{PK_a}$
- 6.  $E \rightarrow A: [N_a, N_b]_{PK_a}$
- 7.  $A \rightarrow E: [N_b]_{PK_e}$
- 7.  $E \rightarrow B: [N_b]_{PK_b}$