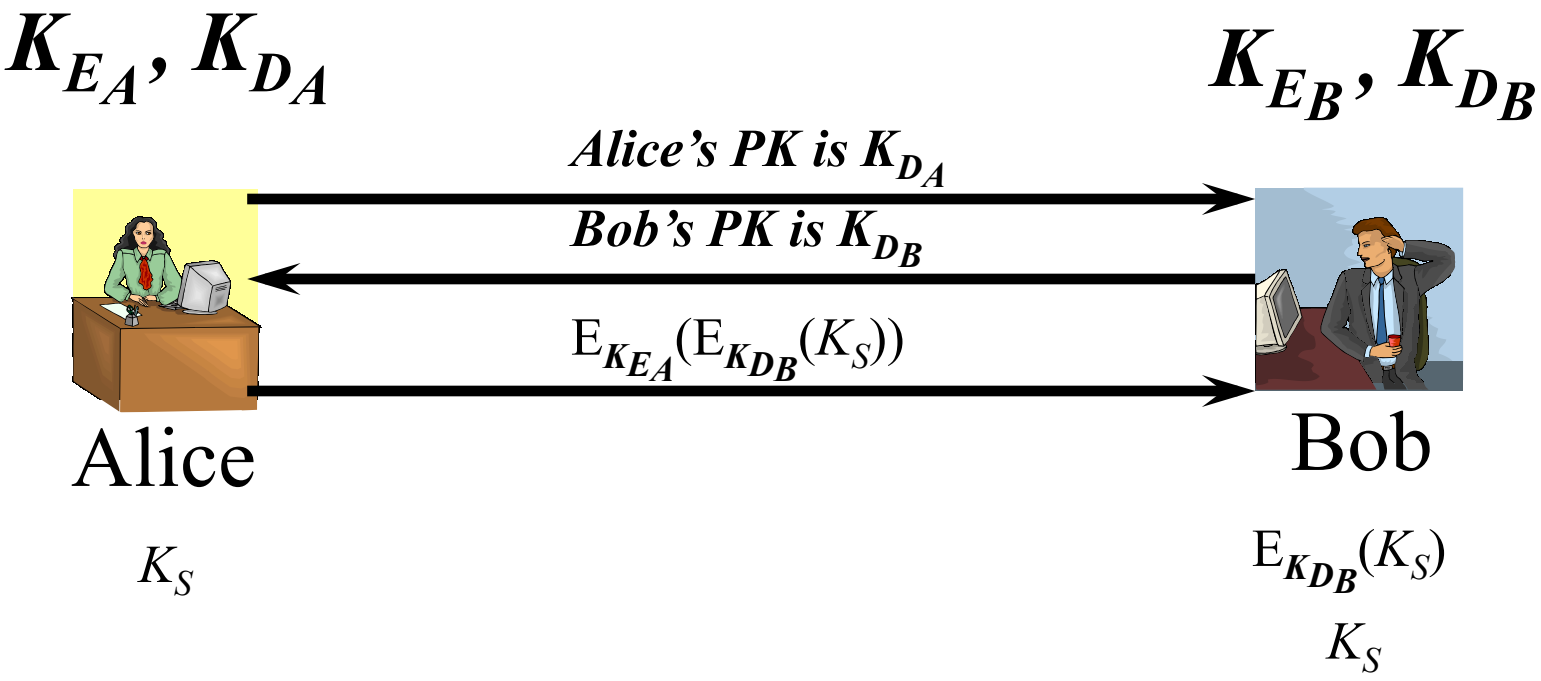
Key Exchange With Public Key Cryptography

- With no trusted arbitrator
- Alice sends Bob her public key
- Bob sends Alice his public key
- Alice generates a session key and sends it to Bob encrypted with his public key, signed with her private key
- Bob decrypts Alice's message with his private key
- Encrypt session with shared session key

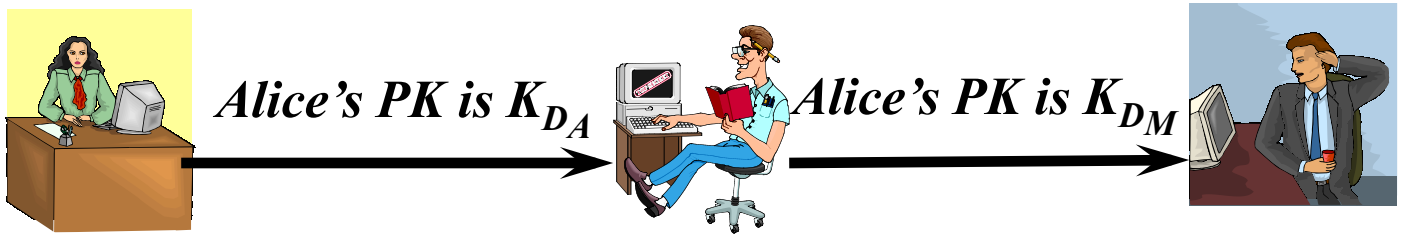
Basic Key Exchange Using PK



Bob verifies the message came from Alice
Bob extracts the key from the message

Man-in-the-Middle With Public Keys

K_{E_A}, K_{D_A} K_{E_M}, K_{D_M} K_{E_B}, K_{D_B}



Alice Mallory Bob

Now Mallory can pose as Alice to Bob

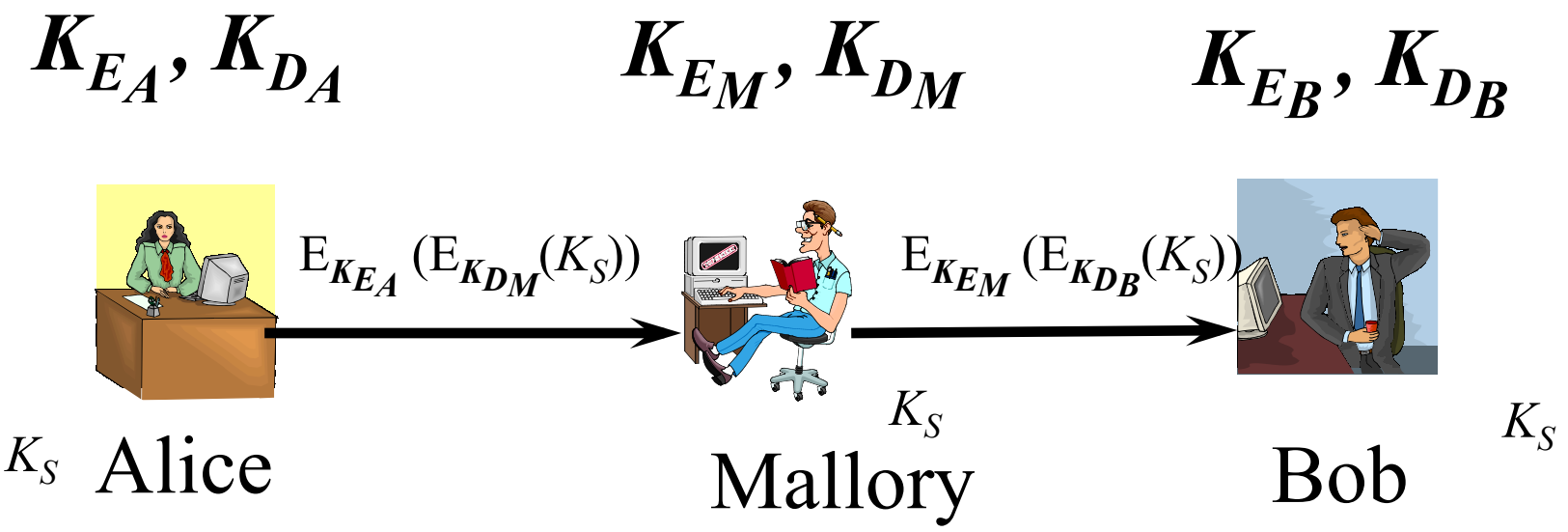
And Bob Sends His Public Key

K_{E_A}, K_{D_A} K_{E_M}, K_{D_M} K_{E_B}, K_{D_B}

Alice Mallory Bob

Now Mallory can pose as Bob to Alice

Alice Chooses a Session Key



Bob and Alice are sharing a session key

Unfortunately, they're also sharing it with Mallory

Combined Key Distribution and Authentication

- Usually the first requires the second
  - Not much good to be sure the key is a secret if you don't know who you're sharing it with
- How can we achieve both goals?
  - In a single protocol
  - With relatively few messages

Needham-Schroeder Key Exchange

- Uses symmetric cryptography
- Requires a trusted authority
  - Who takes care of generating the new key
- More complicated than some protocols we've seen

Needham-Schroeder, Step 1

K_A

R_A Alice

K_B

Bob

Alice,Bob,R_A

Trent

K_A K_B

What's the Point of R_A?

- R_A is random number chosen by Alice for this invocation of the protocol
  - Not used as a key, so quality of Alice's random number generator not too important
- Helps defend against replay attacks
- This kind of random number is sometimes called a *nonce*

Needham-Schroeder, Step 2

K_A

R_A Alice

Including R_A prevents replay

Including Bob prevents attacker from replacing Bob's identity

K_B

Bob

$E_{K_A}(R_A, Bob, K_S, E_{K_B}(K_S, Alice))$

What's all this stuff for?

Trent

R_A K_S K_A K_B

Including the encrypted message for Bob ensures Bob's message can't be replaced

Needham-Schroeder, Step 3

K_A

K_S

Alice

$E_{K_B}(K_S, \text{Alice})$

K_B

K_S

Bob

So we're done, right?

Wrong!

Trent

K_A K_B

Needham-Schroeder, Step 4

K_A

K_S

Alice

R_B

$E_{K_S}(R_B)$

K_B

K_S

Bob R_B

Trent

K_A K_B

Needham-Schroeder, Step 5

K_A

K_S

Alice

R_B

$E_{K_S}(R_B\text{-}1)$

K_B

K_S

Bob

R_B

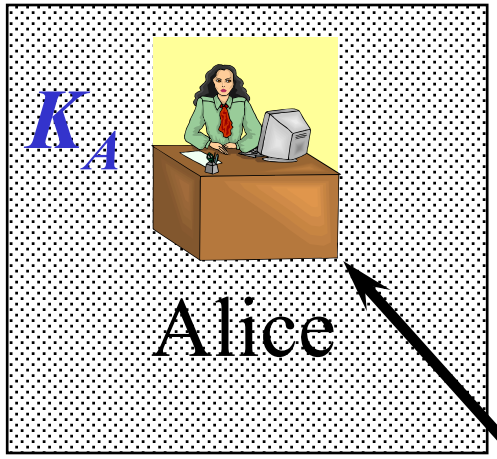
$R_B\text{-}1$

Now we're done!

Trent

K_A K_B

What's All This Extra Stuff For?



Alice knows she's talking to Bob

Trent said she was

Can Mallory jump in later?

$E_{K_A}(R_A, Bob, K_S,$

$E_{K_B}(K_S, Alice))$

K_S K_A K_B

No, only Bob could read the key package Trent created

What's All This Extra Stuff For?

K_A

K_S Alice

$E_{K_B}(K_S, \text{Alice})$

K_B

Bob knows he's talking to Alice

What about those random numbers?

Can Mallory T…

jump i… …

…, all later messages will use K_S, which Mallory doesn't know

Trent

K_A K_B

Mallory Causes Problems

- Alice and Bob do something Mallory likes
- Mallory watches the messages they send to do so
- Mallory wants to make them do it again
- Can Mallory replay the conversation?
  - Let's try it without the random numbers

Mallory Waits For His Chance

K_A

$E_{K_A}(Bob, K_S,$

$E_{K_B}(K_S, Alice))$

K_B

Alice

Mallory

Bob

Alice,Bob

Trent

K_A K_B

What Will Alice Do Now?

- The message could only have been created by Trent
- It properly indicates she wants to talk to Bob
- It contains a perfectly plausible key
- Alice will probably go ahead with the protocol

The Protocol Continues

K_A

K_S

Alice

$E_{K_B}(K_S, Alice)$

K_B

Bob

K_S

Mallory steps aside for a bit

Trent

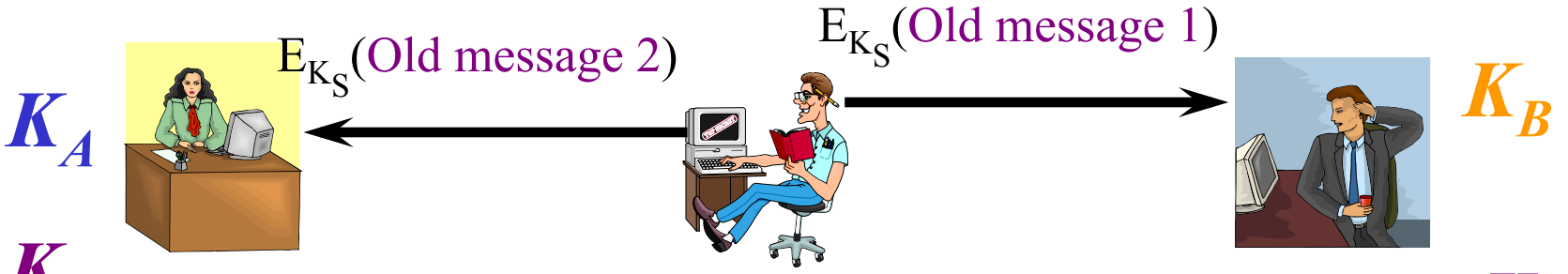
K_A K_B

With no nonces, we're done

So What's the Problem?

- Alice and Bob agree K_S is their key
  - They both know the key
  - Trent definitely created the key for them
  - Nobody else has the key
- But . . .

Mallory Steps Back Into the Picture



Mallory can replay Alice and Bob's old conversation

Trent

K_A K_B

It's using the current key, so Alice and Bob will accept it

How Do the Random Numbers Help?

- Alice's random number assures her that the reply from Trent is fresh
- But why does Bob need another random number?

Why Bob Also Needs a Random Number

$E_{K_B}(K_S, Alice)$

K_B

Mallory

Bob K_S

Let's say Alice <u>doesn't</u> want to talk to Bob

Trent

K_A K_B

But Mallory wants Bob to think Alice wants to talk

So What?

E_{K_S}(Old message 1)

K_B

K_S

Mallory

Bob

Mallory can now play back an old message from Alice to Bob

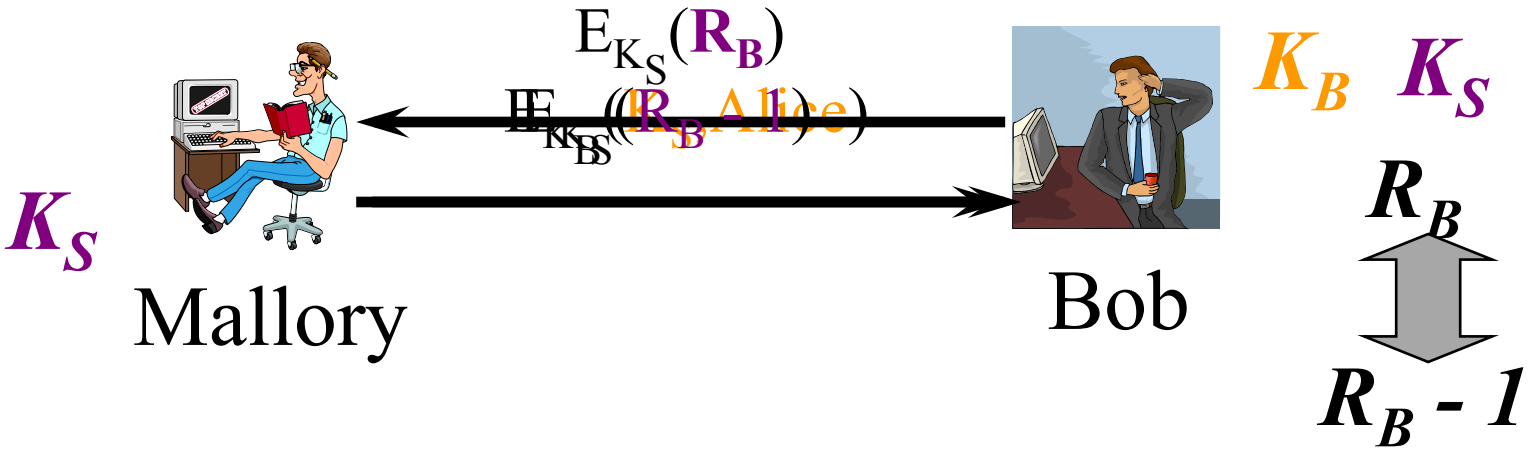
And Bob will have no reason to be suspicious

Bob's random number exchange assures him that Alice really wanted to talk

So, Everything's Fine, Right?

- Not if any key K_S ever gets divulged
- Once K_S is divulged, Mallory can forge Alice's response to Bob's challenge
- And convince Bob that he's talking to Alice when he's really talking to Mallory

Mallory Cracks an Old Key



Mallory compromises 10,000 computers belonging to 10,000 grandmothers to crack K_S

Unfortunately, Mallory knows K_S

So Mallory can answer Bob's challenge