Introduction to Cryptography and Security Mechanisms:

Unit 13

Cryptographic protocols

Learning Outcomes

- Explain the concept of a cryptographic protocol
- Analyse a simple cryptographic protocol
- Appreciate the difficulty of designing a secure cryptographic protocol
- Justify the typical properties of an authentication and key establishment protocol
- Compare the features of some different authentication and key establishment protocols, including Diffie-Hellman key agreement

Sections

1. Protocol basics
2. Analysing a simple protocol
3. AKE protocols

1. Protocol basics

What is a protocol?

1. **Can you think of examples of non-cryptographic protocols?**

2. **Why do we need protocols?**

What is a cryptographic protocol?

Cryptographic protocols are the methods by which the toolkit of cryptographic mechanisms are implemented together as a package in order to achieve precise and often sophisticated security goals.

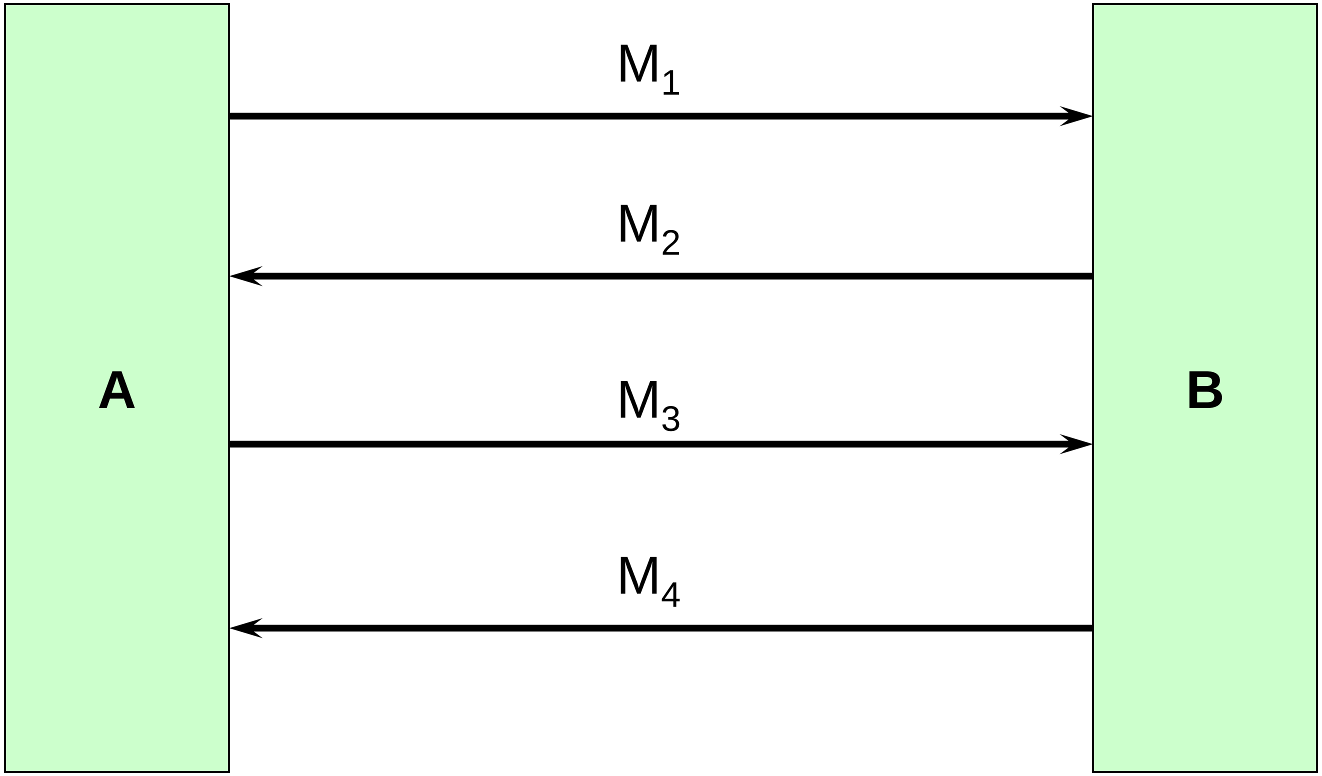
Protocols are normally defined by a sequence of steps that have to be followed in a specific order, most of which consist of a message that has to be passed from one entity to another.

We will only examine fairly simple protocols here – look out for examples of more complex protocols in other modules.

Components of a cryptographic protocol

- The protocol **assumptions**
  - *What needs to have happened* ***before*** *the protocol is run?*
- The protocol **flow**
  - *Who sends a message to whom (in what order)?*
- The protocol **messages**
  - *What information is exchanged at each step?*
- The protocol **actions**
  - *What needs to be done between each step?*

Model of a cryptographic protocol



After any step in the protocol, the protocol will only proceed if the last receiving party was happy with the received message. If they are not happy then the protocol terminates.

Stages of protocol design

Defining the security objectives
The problem statement

Determining the protocol goals
Translating the security objectives into a set of cryptographic requirements to be met by the end of the protocol

Specifying the protocol
Assumptions, flow, messages, actions

Simple example

Defining the security objectives

Bob wants to make sure that Alice was the source of a contract

Determining the protocol goals

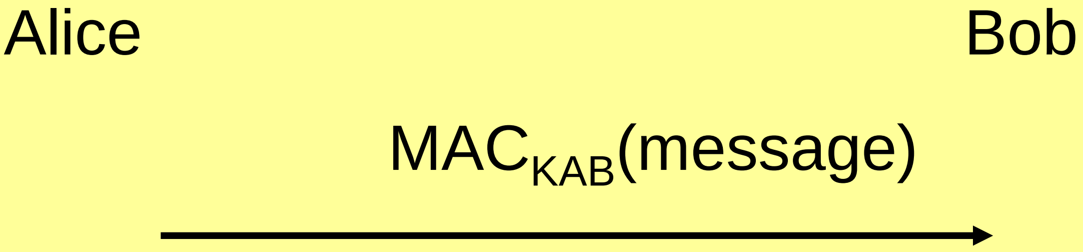
Bob requires data origin authentication of the contract received from Alice

Specifying the protocol

See next slide

Assumptions and actions

Protocol:



Alice

Bob

$\text{MAC}_{\text{KAB}}(\text{message})$

1. What assumptions are we making before we run this protocol?
2. What action is missing from the description of this simple protocol?

Words of caution

Cryptographic protocols are notoriously difficult to design properly.

The hardest stage is **specifying the protocol**.

A well-designed protocol can still fail if:

- it is not implemented correctly
- a weak primitive is deployed
- the supporting key management is inadequate

2. Analysing a simple protocol

Our security objectives

Alice and Bob are nodes in a multiuser network.

Alice and Bob trust one another, but the rest of the network is “hostile”.

Network delays are possible.

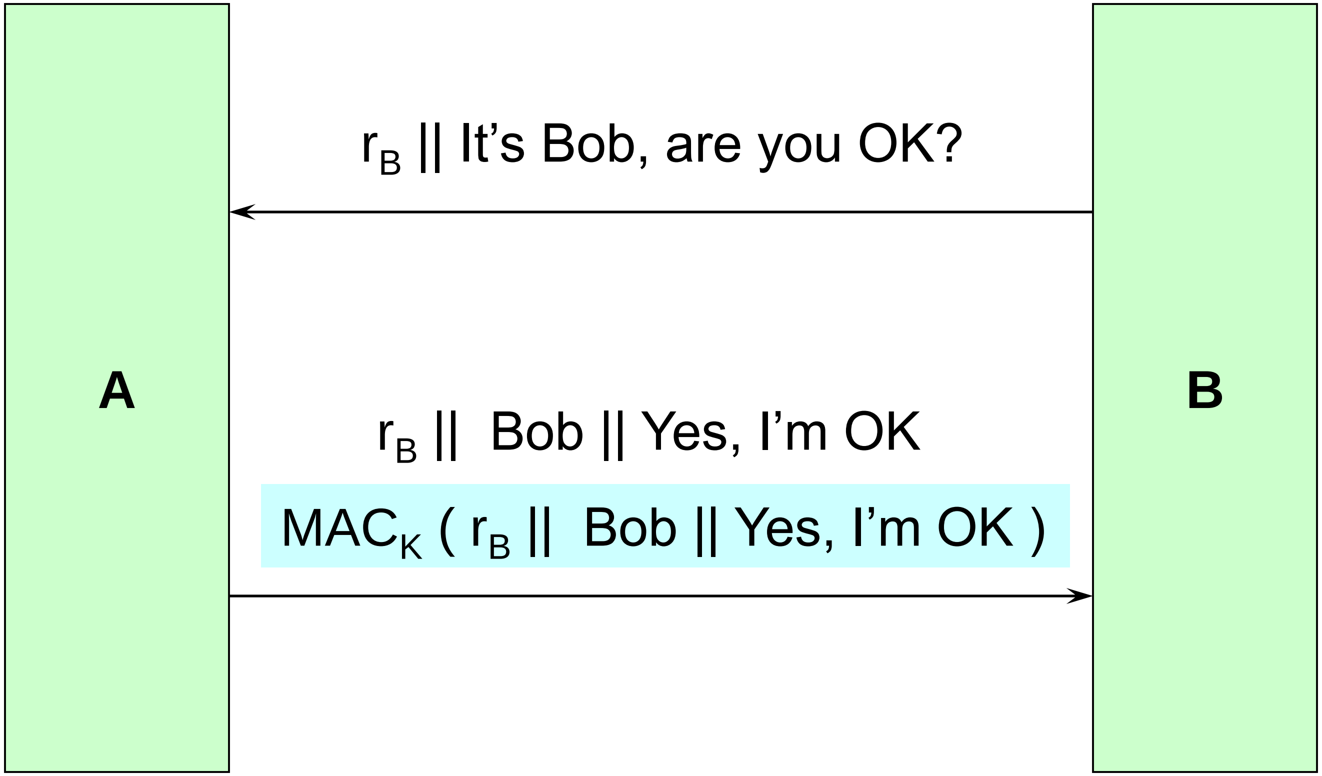
1. **Bob wants to regularly check that Alice is still alive (in other words, she is still connected to the network).**

2. **Bob wants to be able to identify which liveness query Alice is responding to.**

Protocol goals

What should the protocol goals be?

A simple protocol



A

B

r_B || It's Bob, are you OK?

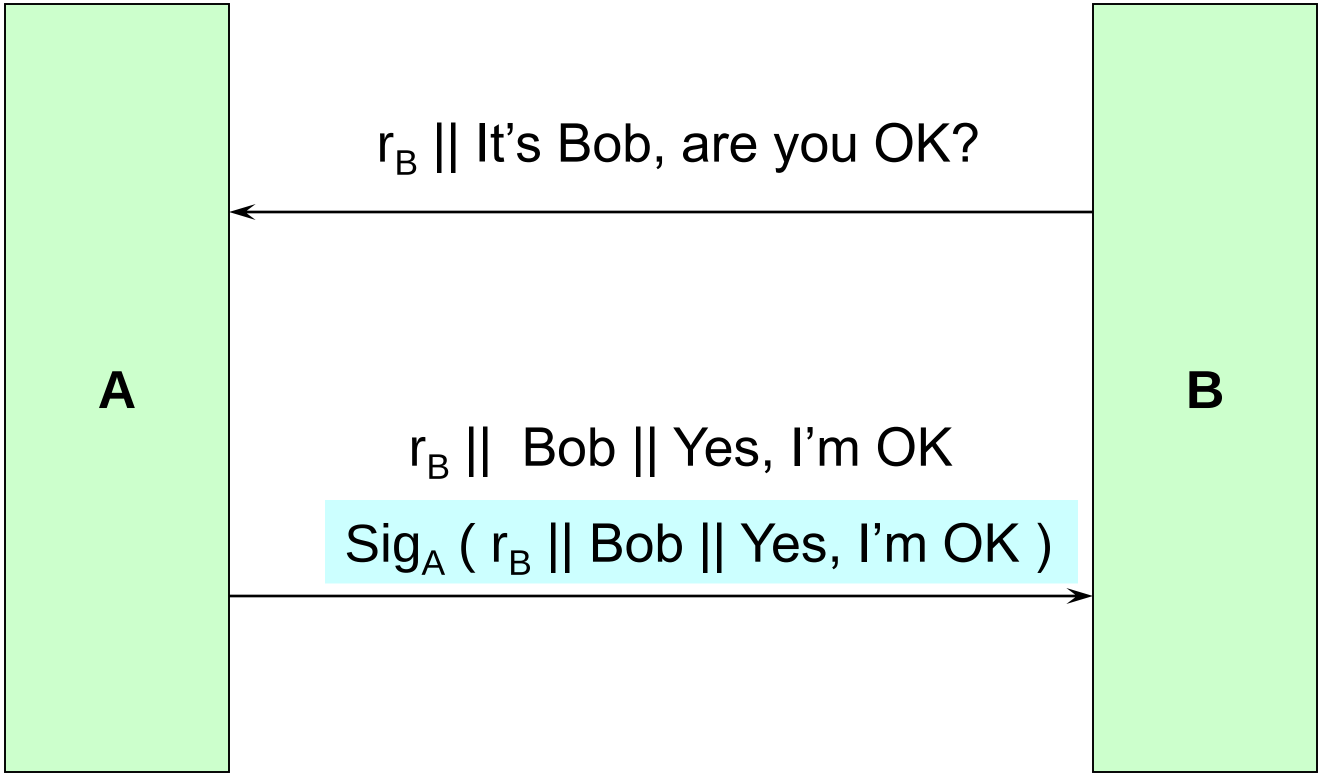
r_B || Bob || Yes, I'm OK

MAC_K (r_B || Bob || Yes, I'm OK)

The missing actions

What missing actions are “implicit” - but essential - in the simple protocol?

A simple protocol (2)

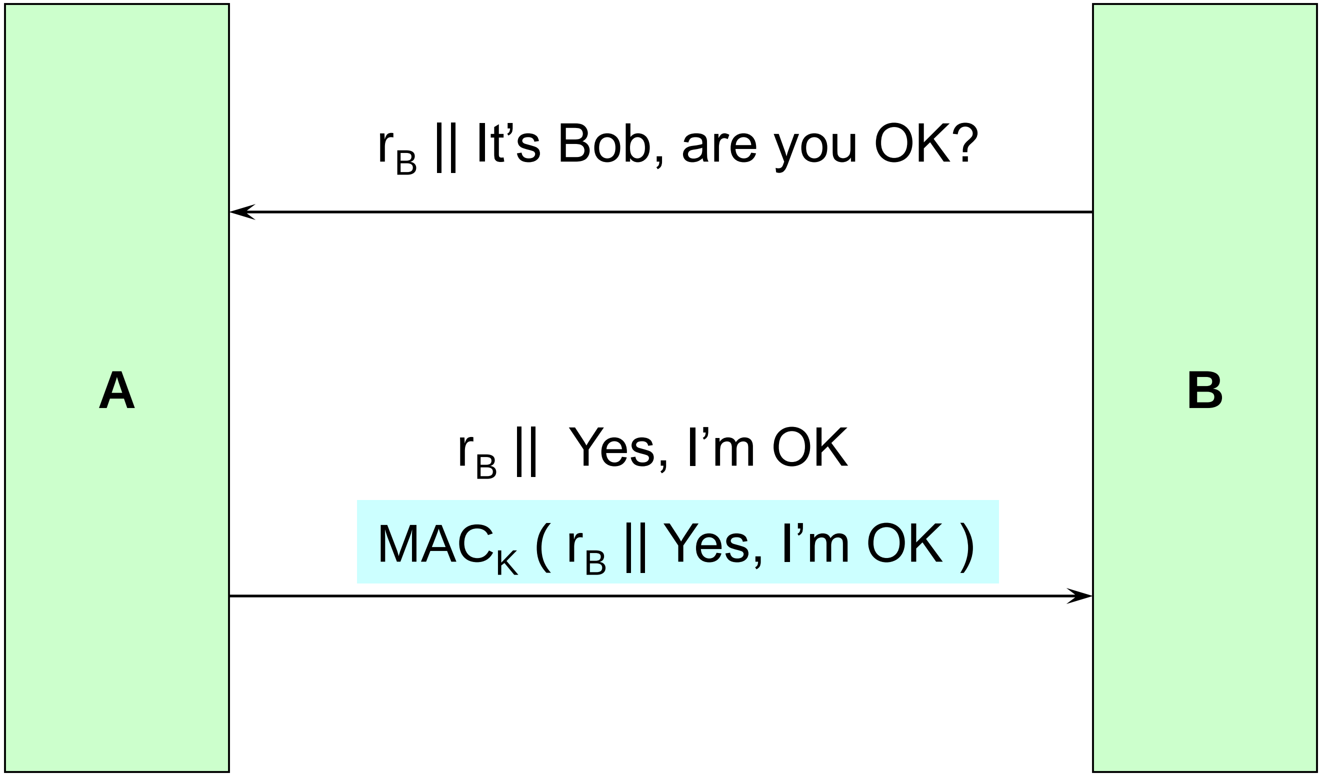


r_B || It's Bob, are you OK?

r_B || Bob || Yes, I'm OK

Sig_A (r_B || Bob || Yes, I'm OK)

A simple protocol (3)



A

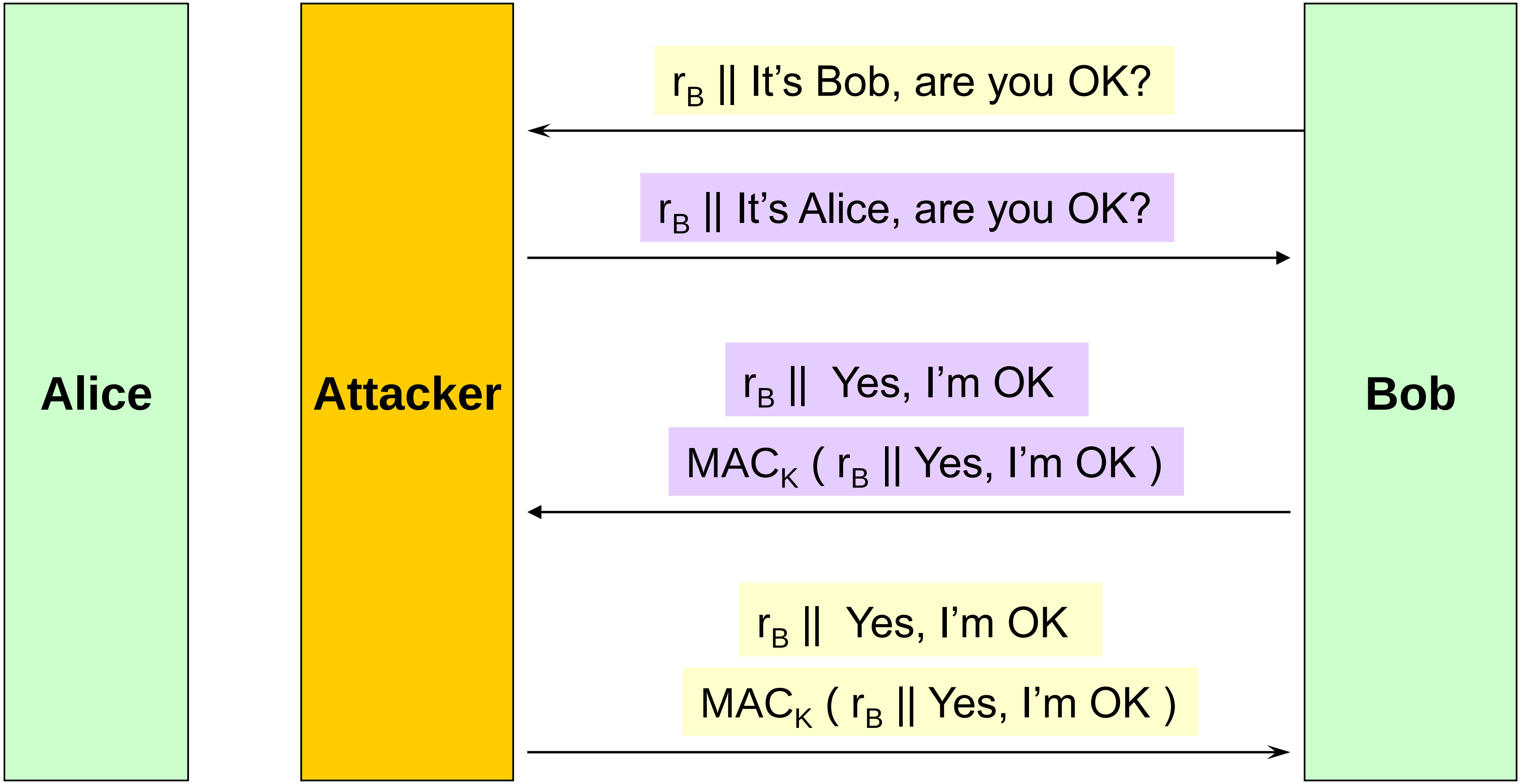
B

r_B || It’s Bob, are you OK?

r_B || Yes, I’m OK

MAC_K (r_B || Yes, I’m OK)

Reflection attack



Alice

Attacker

Bob

r_B || It’s Bob, are you OK?

r_B || It’s Alice, are you OK?

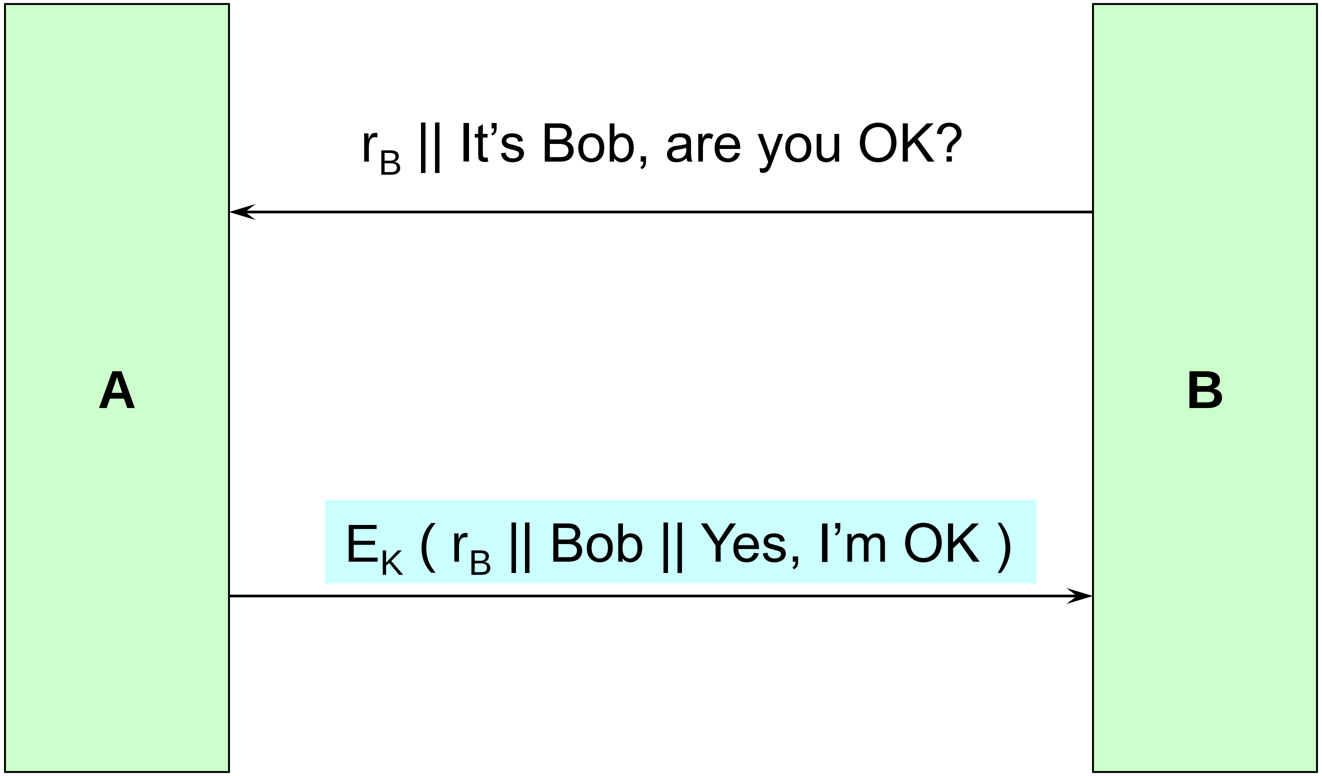
r_B || Yes, I’m OK

MAC_K (r_B || Yes, I’m OK)

r_B || Yes, I’m OK

MAC_K (r_B || Yes, I’m OK)

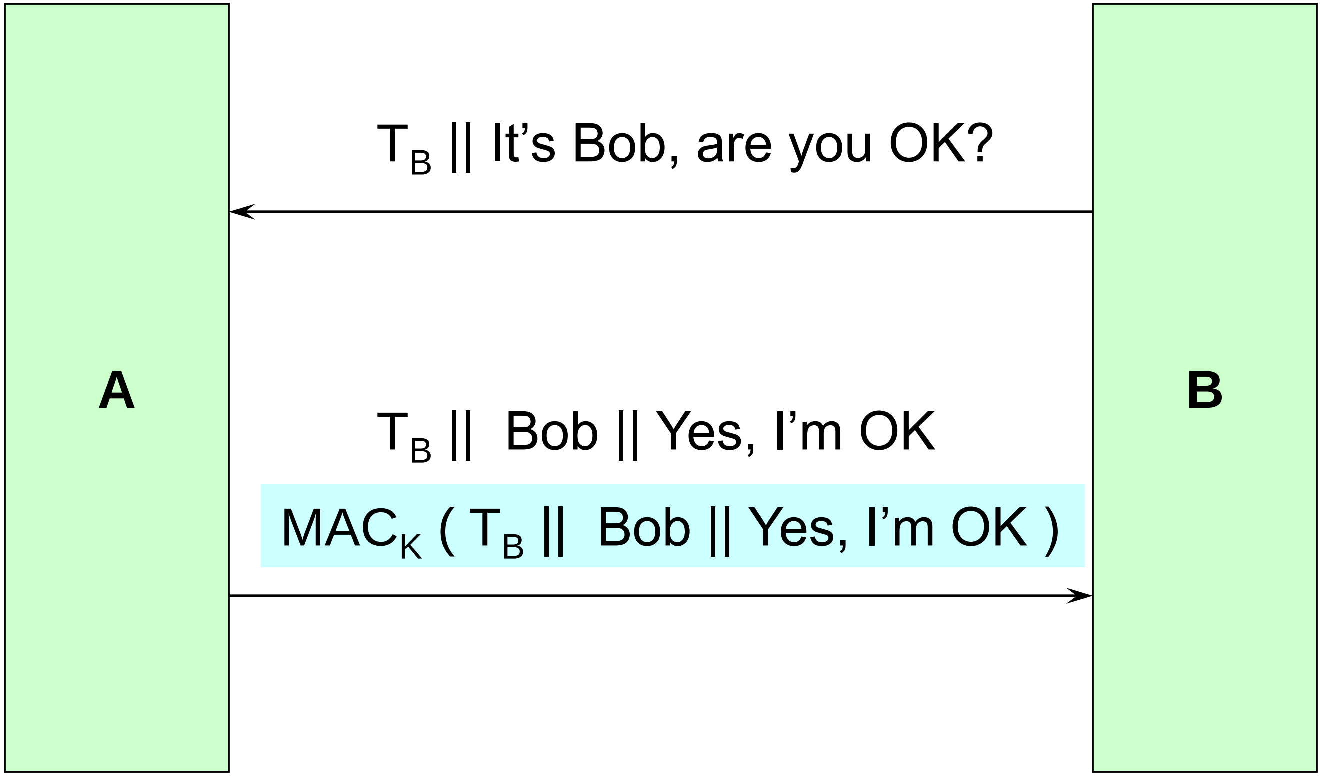
A simple protocol (4)



r_B || It’s Bob, are you OK?

E_K (r_B || Bob || Yes, I’m OK)

A simple protocol (5)

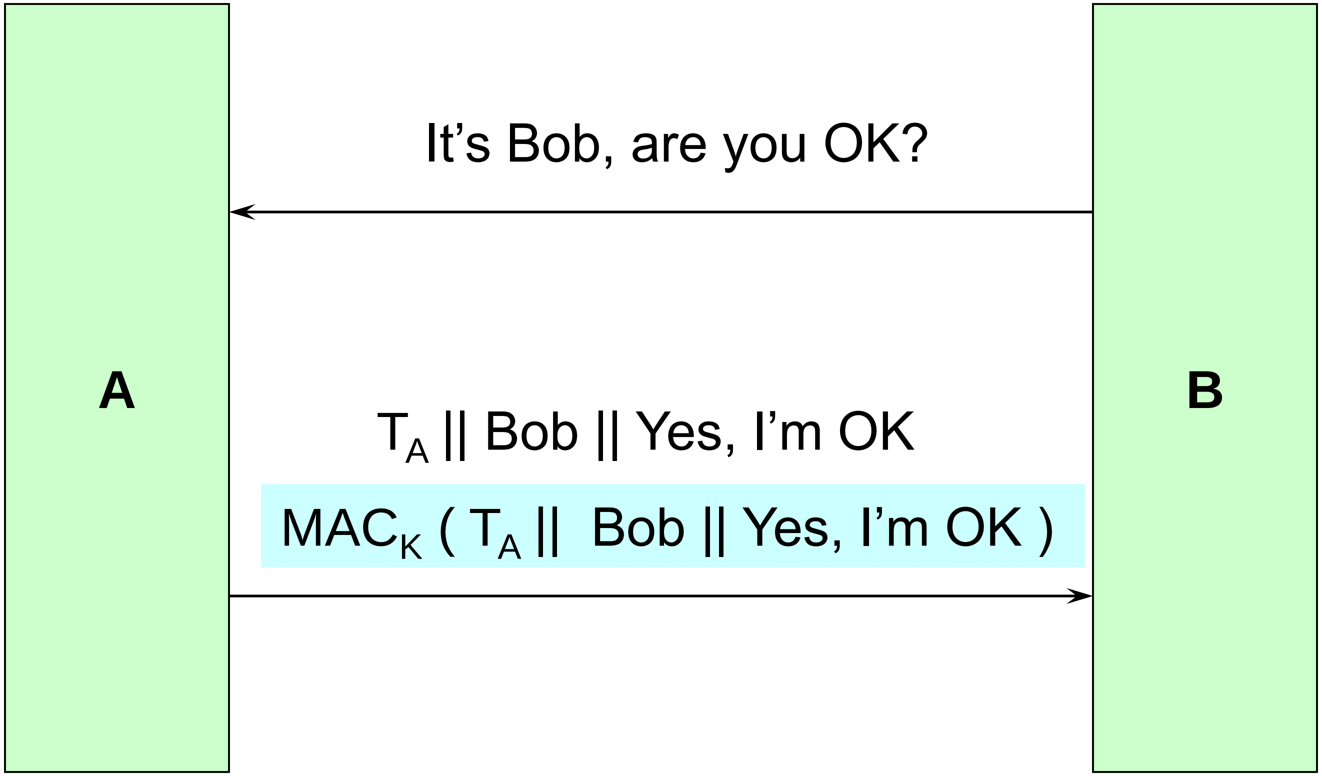


T_B || It’s Bob, are you OK?

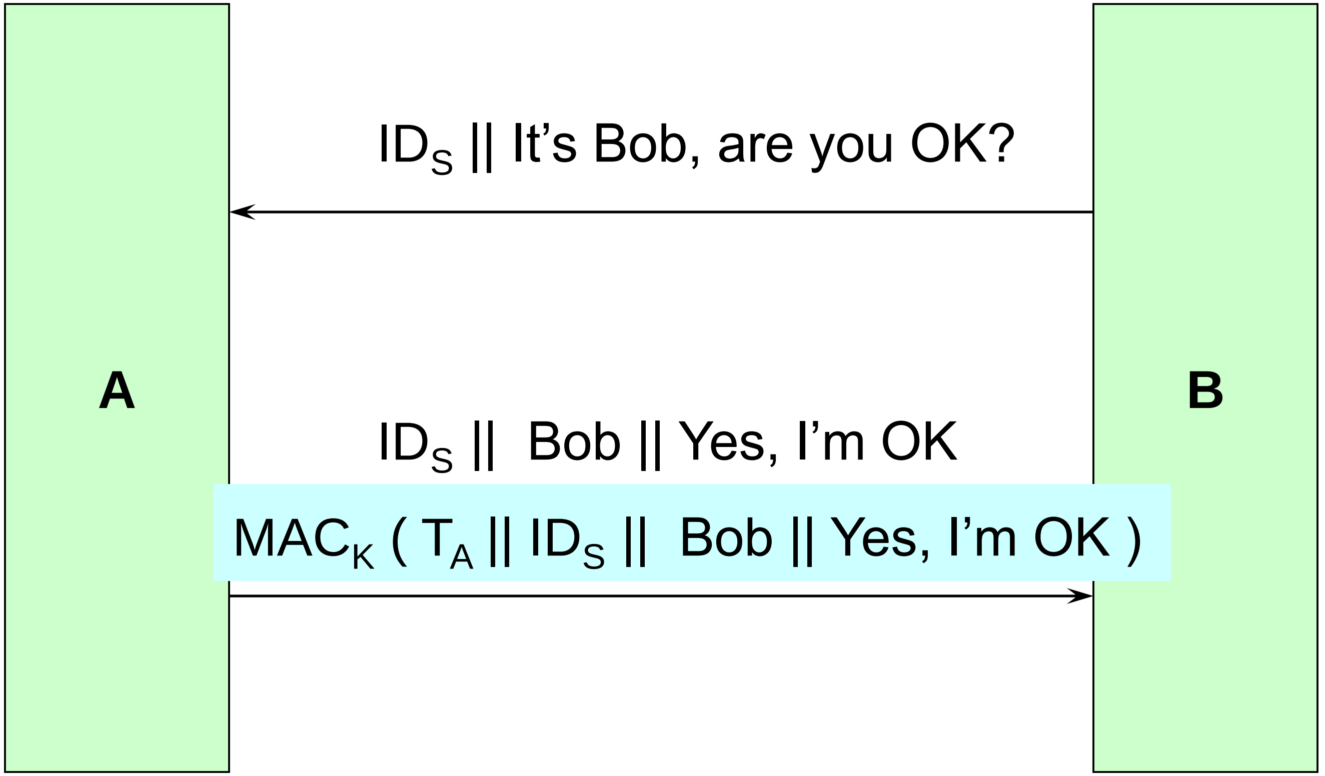
T_B || Bob || Yes, I’m OK

MAC_K (T_B || Bob || Yes, I’m OK)

A simple protocol (6)


It’s Bob, are you OK?
A
B
T_A || Bob || Yes, I’m OK
MAC_K (T_A || Bob || Yes, I’m OK)

A simple protocol (7)



ID_S || It’s Bob, are you OK?

ID_S || Bob || Yes, I’m OK

MAC_K (T_A || ID_S || Bob || Yes, I’m OK)

3. AKE protocols

AKE protocols

There is a lot of demand for cryptographic (authentication) protocols that offer a combination of:

1 – Mutual entity authentication

2 – Establishment of a common symmetric key

These are sometimes referred to as **authentication and key establishment** (**AKE**) protocols

Why do you think that:

1. Key establishment protocols often require mutual entity authentication?
2. Entity authentication protocols often require the establishment of a common symmetric key?

Typical AKE protocol requirements

Security Requirement	Explanation
Mutual entity authentication	**Alice and Bob are able to verify each other's identity to make sure that they know with whom they are currently establishing a key**
Mutual data origin authentication	**Alice and Bob are able to be sure that information being exchanged has originated with the other party and not an attacker**
Mutual key establishment	**At the end of the process Alice and Bob have established a common symmetric key**
Key confidentiality	**The established key should at no time have been accessible to any other party than Alice and Bob**
Key freshness	**Alice and Bob should be happy that the established key is not one that has been used before**

Optional AKE protocol requirements

Security Requirement	Explanation
Mutual key confirmation	**At the end of the process Alice and Bob should have some evidence that they have both ended up with the same key**
Unbiased key control	**At the end of the process Alice and Bob should be satisfied that neither party can unduly influence the generation of the established key**

Diffie-Hellman key agreement

DH key agreement is not an encryption algorithm.

- The **Diffie–Hellman (DH) key agreement protocol** was first defined in their seminal paper in 1976 (before the existence of public-key encryption algorithms).
- DH key agreement is a protocol for exchanging public information to obtain a shared secret.

DH key agreement has the following important properties:

1. The resulting shared secret cannot be computed by either of the parties without the cooperation of the other.
2. A third party observing all the messages transmitted during DH key agreement cannot deduce the resulting shared secret.

Principle behind DH

DH key agreement assumes first that there exists:

1. A public-key cryptosystem that has a special property (we come to this shortly).

2. A carefully chosen, publicly known function F that takes two numbers x and y as input, and outputs a third number F(x,y) (for example, multiplication is such a function).

Principle behind DH

Assume that Alice and Bob are the parties who wish to establish a shared secret, and let their public and private keys in the public key cryptosystem be denoted by (PA , SA) and (PB , SB) respectively.

The basic principle behind Diffie–Hellman key exchange is as follows:

1. Alice and Bob exchange their public keys PA and PB.
2. Alice computes F(SA , PB)
3. Bob computes F(SB, PA)
4. The special property of the public key cryptosystem, and the choice of the function F, are such that **F(SA , PB) = F(SB, PA)**. If this is the case then Alice and Bob now share a secret.
5. This shared secret can easily be converted by some public means into a bitstring suitable for use as a symmetric key.

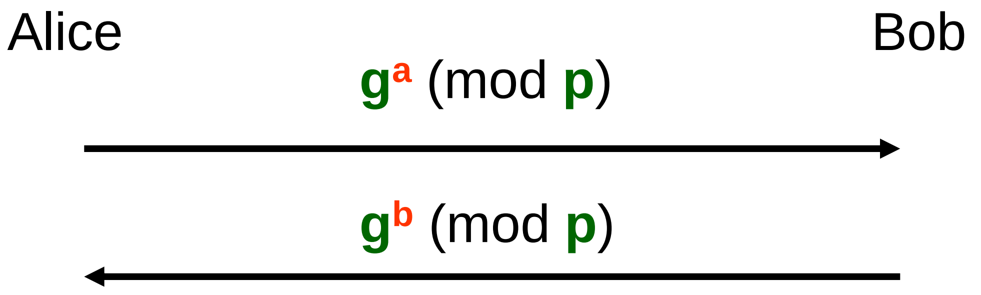
Diffie-Hellman system parameters

The most commonly described implementation of DH key exchange uses the keys of the ElGamal cryptosystem and a very simple function F.

The system parameters (which are public) are:

- a large prime number p – typically 1024 bits in length
- a **primitive element** g

Diffie-Hellman key agreement protocol



Alice

Bob

$g^a \pmod{p}$

$g^b \pmod{p}$

1. Alice generates a private random value a, calculates $g^a \pmod{p}$ and sends it to Bob. Meanwhile Bob generates a private random value b, calculates $g^b \pmod{p}$ and sends it to Alice.

2. Alice takes g^b and her private random value a to compute $(g^b)^a = g^{ab} \pmod{p}$.

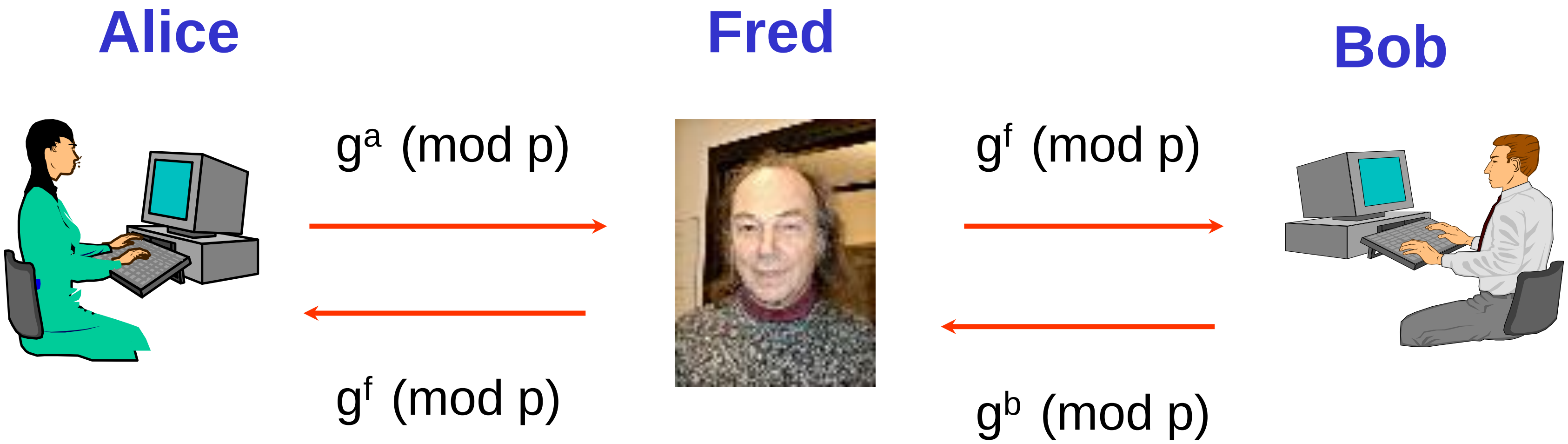
3. Bob takes g^a and his private random value b to compute $(g^a)^b = g^{ab} \pmod{p}$.

4. Alice and Bob adopt $g^{ab} \pmod{p}$ as the shared secret.

Diffie-Hellman questions

1. What is the hard problem on which the DH key agreement protocol is based?
2. Suppose that DH key agreement is used to generate a symmetric key. Why might that key be derived (but different from) the DH shared secret?
3. Does Diffie-Hellman meet the typical requirements of an AKE protocol?

Man-in-the-middle attack



Alice

Fred

Bob

g^a (mod p)

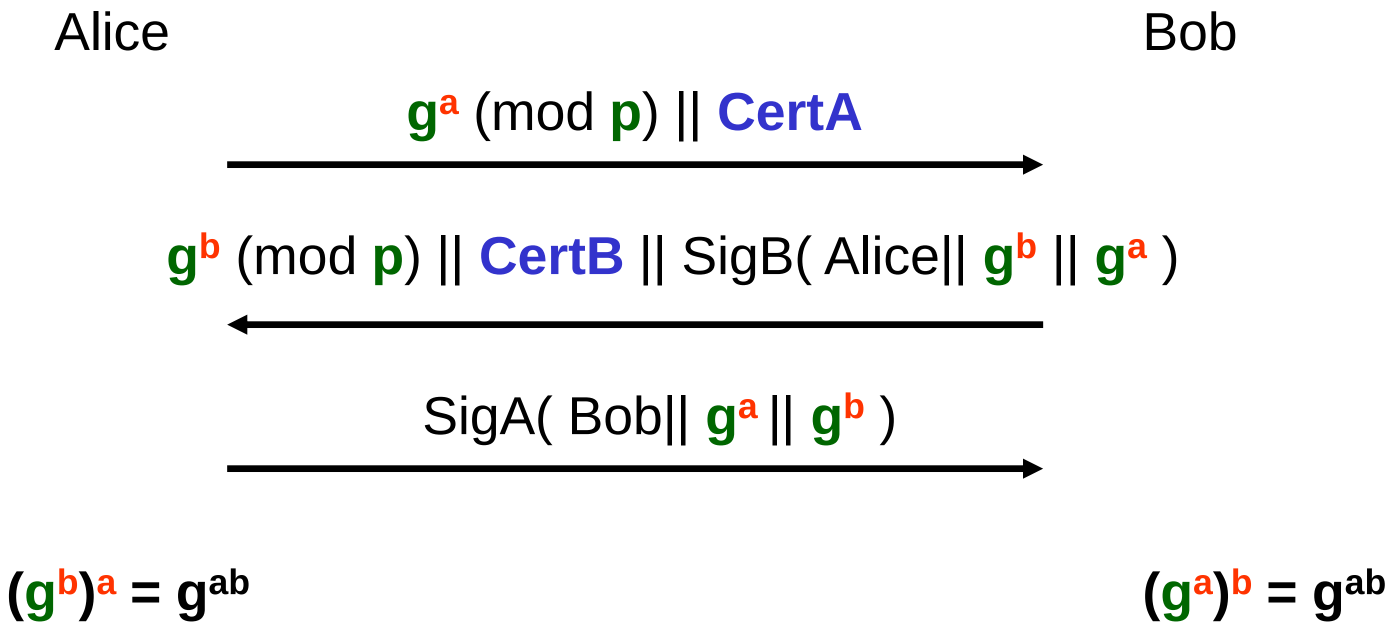
g^f (mod p)

g^f (mod p)

g^b (mod p)

1. What will happen when Alice tries to send a message to Bob, encrypted with a key based on her DH shared secret?
2. Can Fred obtain the correct DH shared secret that would have been established had he not interfered?

Station-to-station (STS) protocol



Alice

Bob

g^a (mod p) || CertA

g^b (mod p) || CertB || SigB(Alice|| g^b || g^a)

SigA(Bob|| g^a || g^b)

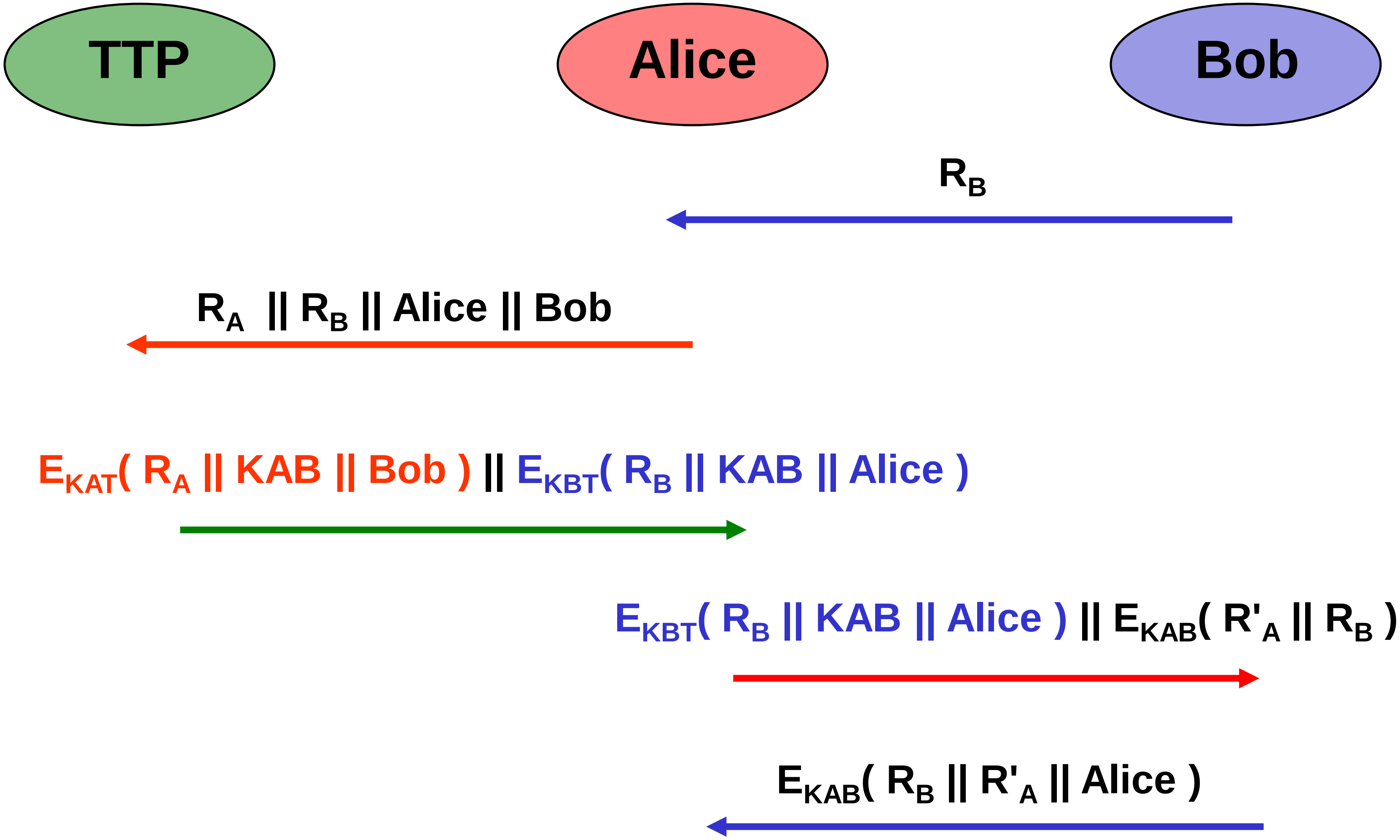
$(g^b)^a = g^{ab}$

$(g^a)^b = g^{ab}$

Does this version of the STS protocol meet:

1. **the typical requirements of an AKE protocol?**
2. **the optional requirements of an AKE protocol?**

ISO 9798-2 Example 8



TTP

Alice

Bob

R_B

R_A || R_B || Alice || Bob

E_{KAT}(R_A || KAB || Bob) || E_{KBT}(R_B || KAB || Alice)

E_{KBT}(R_B || KAB || Alice) || E_{KAB}(R'_A || R_B)

E_{KAB}(R_B || R'_A || Alice)

Summary

- Cryptographic protocols are carefully designed procedures that combine different cryptographic primitives to achieve combinations of security goals

- Cryptographic protocols are surprisingly difficult to design and, where possible, standard protocols should be adopted

- An important family of cryptographic protocols are those that provide mutual entity authentication and key establishment