CSCI471/971
Modern Cryptography

Commitment and Oblivious Transfer (fundamental for MPC)

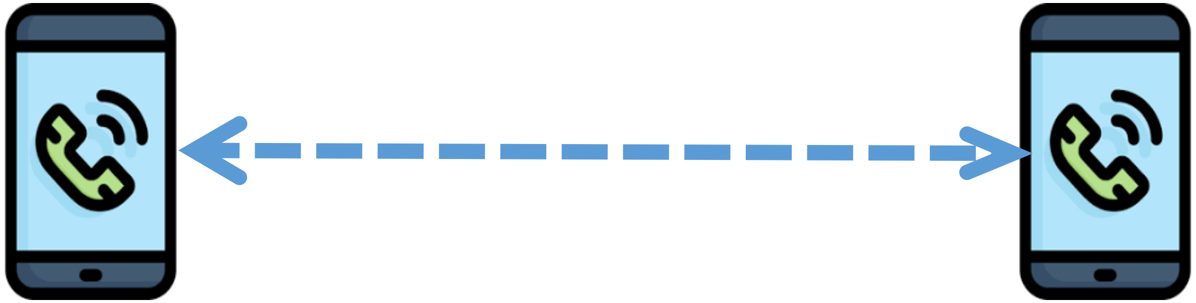
Fuchun Guo
SCIT UOW

Motivation

Commitment

Flip&Guess over a Phone

Alice



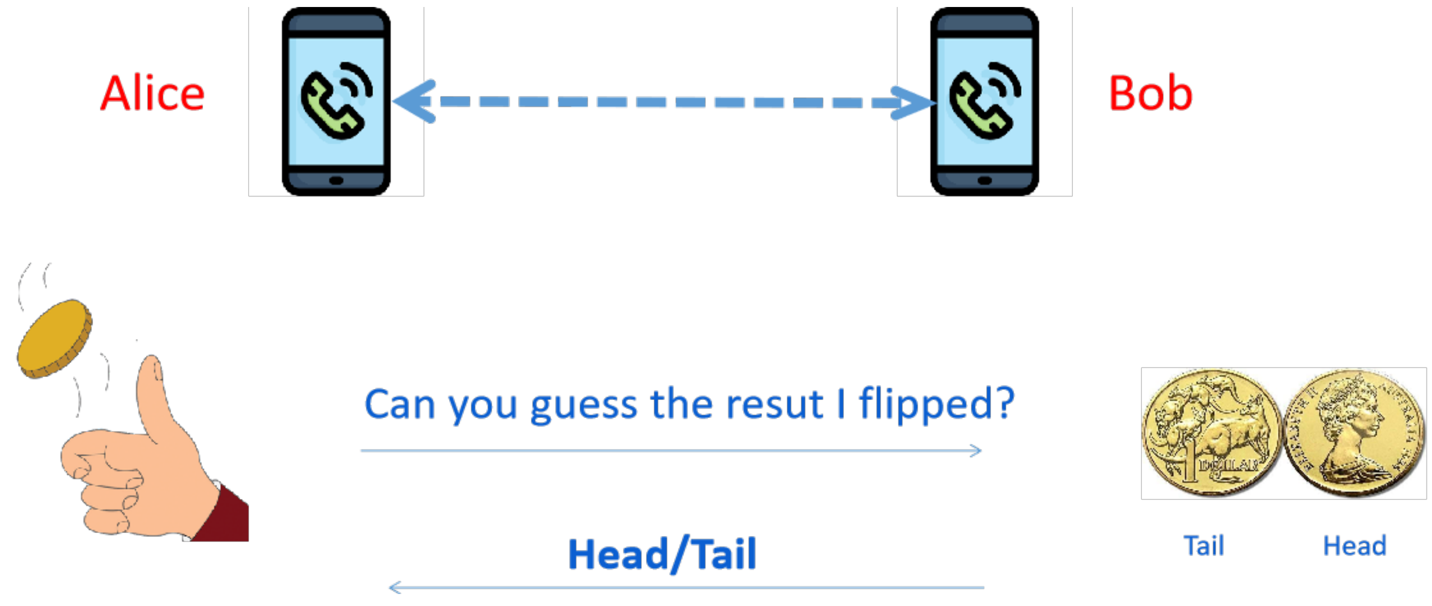
Bob

Can you guess the result I flipped?

Head/Tail

Tail Head

Flip&Guess over a Phone



Alice

Bob

Can you guess the resut I flipped?

Head/Tail

Tail

Head

Problem:

If Alice cannot be trusted, she can always cheat on Bob

How to play a fair game like this？

Definition
Commitment

Commitment

- A commitment scheme is a cryptographic primitive that allows one to commit to a chosen value (or chosen statement) while keeping it hidden to others, with the ability to reveal the committed value later.

- Commitment schemes are designed so that a party cannot change the value or statement after they have committed to it.

- There are two parties.

Commitment

A commitment scheme between sender and receiver is composed of two algorithms:

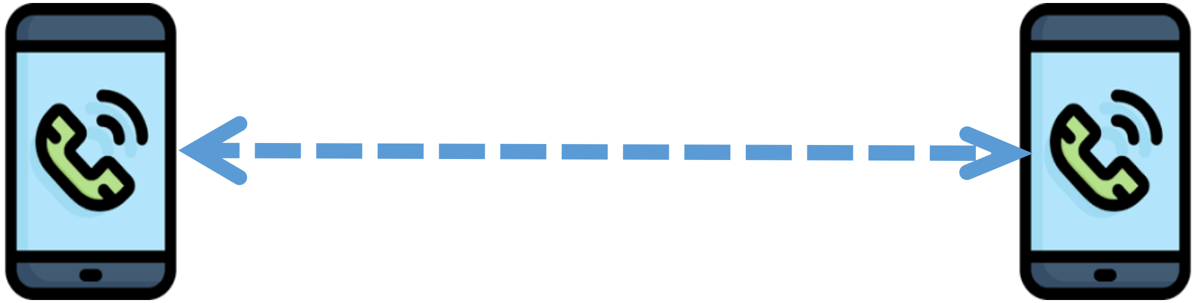
- Commit(m): Taking as input the value m to be committed, run the PPT algorithm which returns the commitment C and secrets r.

C is given to the receiver.

- Open(C,m,r): Taking as input (C,m,r), run the PPT algorithm to verify wherther the commitment is on m or not.

Flip&Guess over a Phone

Alice



Bob

b=1, Head

b=0, Tail

r is random number

Can you guess the result I flipped?

f(b,r) →

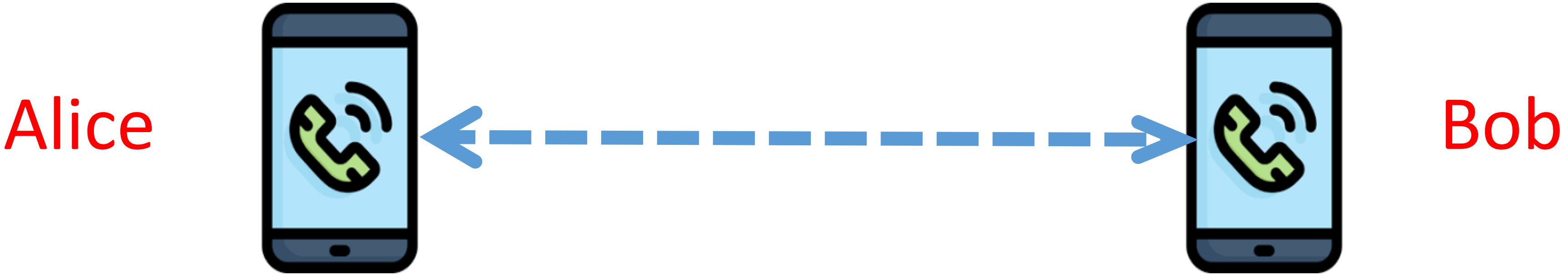
Head/Tail ←

b and r →

Tail Head

Let f() be a one-way function.

Flip&Guess over a Phone



Can you guess the resut I flipped?

f(b,r)

Head/Tail

b and r

b=1, Head
b=0, Tail
r is random number

Is this commitment secure if f() is a one-way function?

Commitment 1

- Commit(m): Taking as input the value b to be committed, choose a random r and compute

$$C=f(b,r)$$

- Open(C,b,r): Taking as input (C,m,r), run the PPT algorithm and accept if C is equal to f(b,r)

Here: f is a one-way function

Is this commitment secure if f() is a one-way function?

Commitment 1

- Commit(m): Taking as input the value b to be committed, choose a random r and compute

$$C=f(b,r)$$

- Open(C,b,r): Taking as input (C,m,r), run the PPT algorithm and accept if C is equal to f(b,r)

Here: f is a one-way function

No. Because it could be easy for the sender to find (b,r) and (b',r') such that

$$\mathbf{f(b,r)=f(b',r')}$$

Commitment 2: A collision-resistant hash H

- Commit(m): Taking as input the value b to be committed, choose a random r and compute

  C=H(b,r)

- Open(C,b,r): Taking as input (C,m,r), run the PPT algorithm and accept if C is equal to H(b,r)

Here: H is a one-way function

Is this commitment secure if H is collision resistant?

Commitment 2: A collision-resistant hash H

- Commit(m): Taking as input the value b to be committed, choose a random r and compute

C=H(b,r)

- Open(C,b,r): Taking as input (C,m,r), run the PPT algorithm and accept if C is equal to H(b,r)

Here: H is a one-way function

No.This is because it could be easy for the receiver to guess the first bit from C.

Commitment 2: A collision-resistant hash H

- Commit(m): Taking as input the value b to be committed, choose a random r and compute C=H(b,r)
- Open(C,b,r): Taking as input (C,m,r), run the PPT algorithm and accept if C is equal to H(b,r)

Here: H is a one-way function

No.This is because it could be easy for the receiver to guess the first bit from C.

For example: Let G be a secure collision-resistant hash function. We define

H(b,r)=b| G(r)

A secure commitment

Commit(m): Taking as input the value m to be committed, run the PPT algorithm which returns the commitment C and secrets r.

Open(C,m,r): Taking as input (C,m,r), run the PPT algorithm to verify wherther the commitment is on m or not.

Security requirements：

- Hinding: Given (C,m_0, m_1), it is computationally hard to guess the committed message in C is either m_0 or m_1.
- Binding: Given C, it is computationally hard to open with two different messages (m_1,r_1) and (m_2,r_2)

Cyclic Group

Let (G,*g*,p) be a cyclic group.

- G is the set of all group elements.
- The set has p number of group elements.
- *g* is the generator of the group G.

$$G = \{g^0, g^1, g^2, g^3, \ldots\ldots, g^{p-1}\}$$

Given x and g, we can compute g^x in a fast way. (See Lecture 6 exponentiation)

Given g and h, we can compute g·h (basic group operation)

Given g and x, we can compute $g^{-x} = g^{p-x}$*. In particuar, we can compute* g^{-1}

Given g and x, we can compute $g^{\frac{1}{x}}$ s.t. $x * \frac{1}{x} = 1\, mod\ p$

A secure commitment scheme

Let (g,h) be two group elements chosen by the receiver.

Commit(m): Taking as input the value m in Z_p to be committed, choose a random integer r from Z_p and compute

$$C=g^{m}h^{r}\text{ , secret}=r$$

Open(C,m',r'): Taking as input (C,m',r'), accept if

$$C=g^{m'}h^{r'}$$

A secure commitment scheme (proof of hinding)

Let (g,h) be two group elemenst chosen by the receiver.

Commit(m): Taking as input the value m to be committed, choose a random integer r and compute

$$C=g^{m}h^{r}\text{ , secret}=r$$

Proof.

Let g^a=h and C=g^A (A=m+ra)

C is a commitment on any two (m,r) and (m',r') if A=m+ra=m'+r'a

The receiver knows that it is a comit on m if r=(A-m)/a

The receiver knows that it is a comit on m' if r'=(A-m')/a

since the secret is randomly chosen, it is equal to r or r' with the same probability. That is, C is a commit on m and m' with the same probability. Therefore, it is unconditionally secure in hinding.

A secure commitment scheme (proof of binding)

Let (g,h) be two group elements chosen by the receiver.

Commit(m): Taking as input the value m to be committed, choose a random integer r and compute

$$C=g^m h^r \text{ , secret}=r$$

Proof.

Let g^a=h

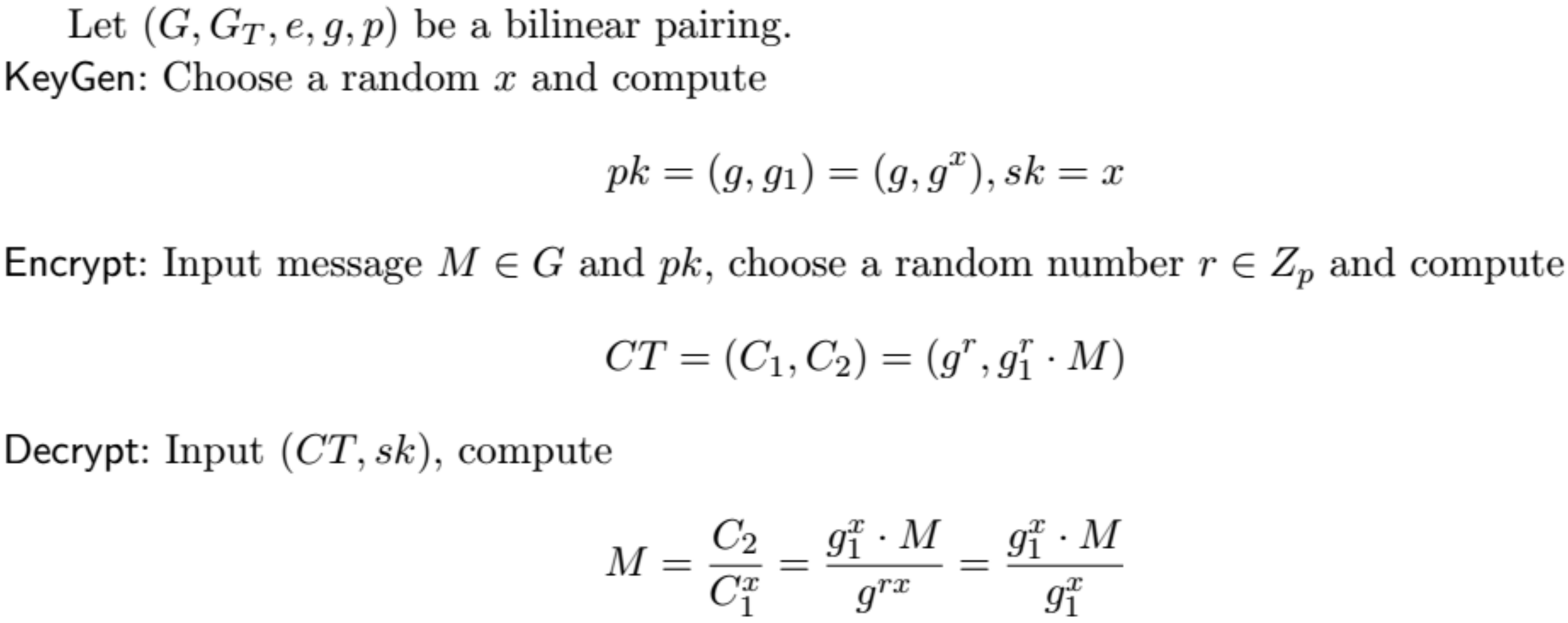
Suppose the sender can open C with (m_1, r_1) and (m_2, r_2).

We have m_1+r_1a=m_2+r_2a

Then, it impies that the sender can compute the DL $a=\frac{m\_1-m\_2}{r\_1-r\_2}$

which is computationally hard for the sender.

ElGamal Encryption (PKE)



Let (G, G_T, e, g, p) be a bilinear pairing.

KeyGen: Choose a random x and compute

$$pk = (g, g_1) = (g, g^x), sk = x$$

Encrypt: Input message $M \in G$ and pk, choose a random number $r \in Z_p$ and compute

$$CT = (C_1, C_2) = (g^r, g_1^r \cdot M)$$

Decrypt: Input (CT, sk), compute

$$M = \frac{C_2}{C_1^x} = \frac{g_1^x \cdot M}{g^{rx}} = \frac{g_1^x \cdot M}{g_1^x}$$

A secure commitment scheme (2)

Let (g,h) be two group elemenst chosen by the sender.

Commit(m): Taking as input the value m in G to be committed, choose a random integer r from Z_p and compute

$$C=(g^r, mh^r) , secret=r$$

Open(C,m',r'): Taking as input (C,m',r'), accept if C can be computed with (m',r')

A secure commitment scheme (2) hinding

Let (g,h) be two group elemenst chosen by the sender.

Commit(m): Taking as input the value m in G to be committed, choose a random integer r from Z_p and compute

$$C=(g^r, mh^r), \text{secret}=r$$

Proof.

The message is encrypted with IND-CPA secure ElGamal encryption so, it providing hiding and computationally secure

A secure commitment scheme (2) binding

Let (g,h) be two group elemenst chosen by the sender.

Commit(m): Taking as input the value m in G to be committed, choose a random integer r from Z_p and compute

$$C=(g^r, mh^r) , \text{secret}=r$$

Proof.

Suppose the sender can open C with (m_1, r_1) and (m_2, r_2).

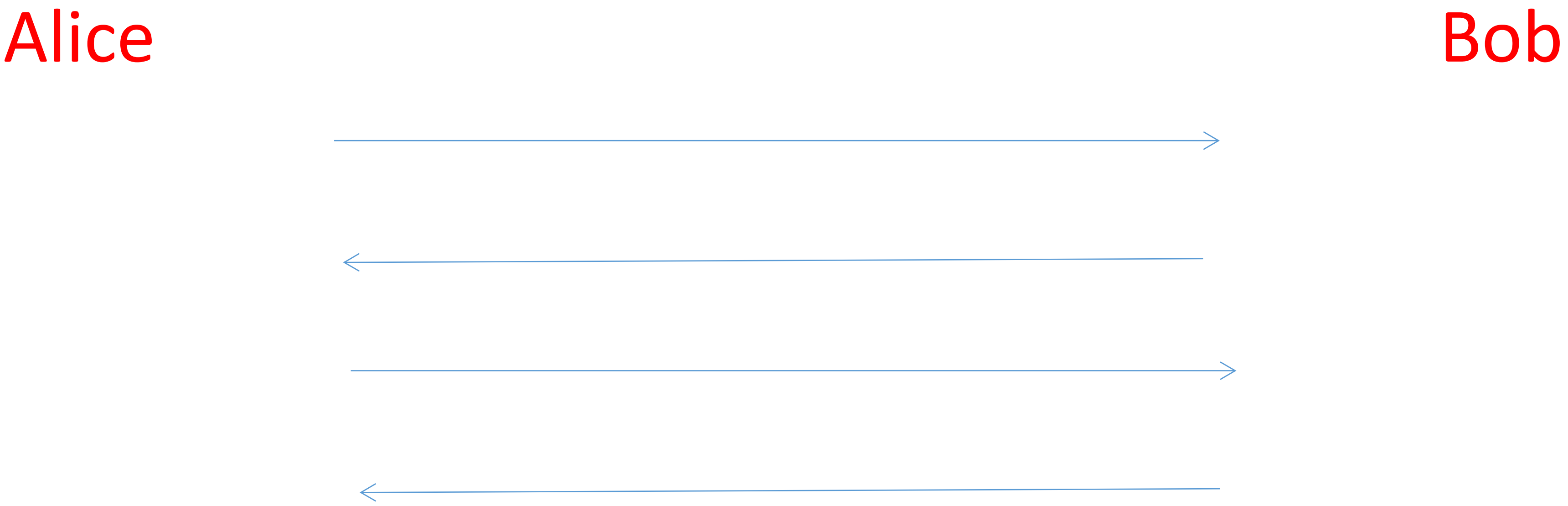
We have r_1 must be equal to r_2 from g^{r_1}=g^{r_2}

If m_1 is different from m_2, we have m*h^r must be different.

Therefore, it is unconditionally secure against the binding.

Motivation
Oblivious Transfer

Purchasing a movie from Alice

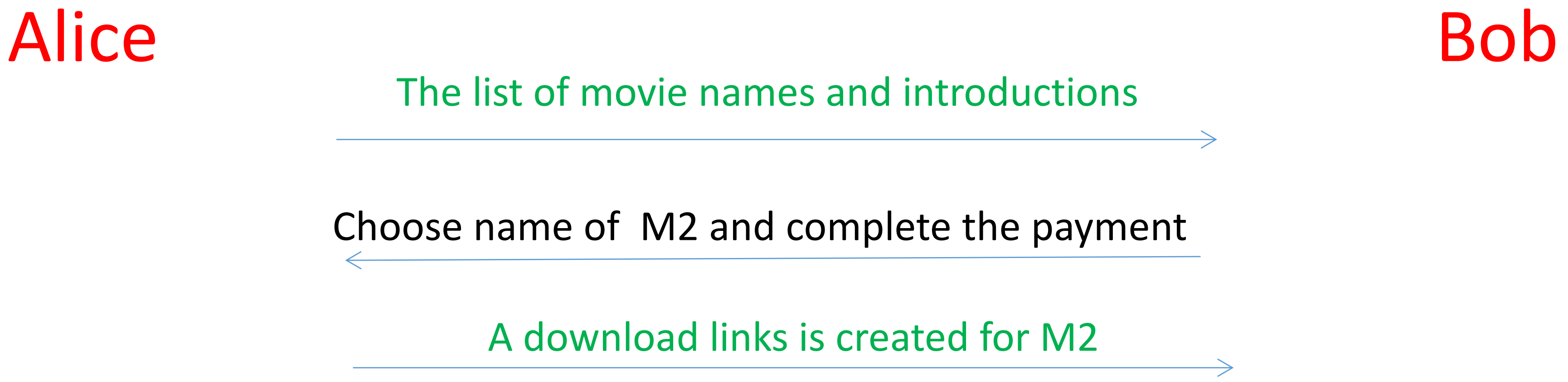


Alice

Bob

Alice is selling digital movies (M1, M2,M3,...,Mn)
Bob wants to purchase the digital movie M2 about the Alien.
How to complete the purchase?

Purchasing a movie from Alice



Alice

Bob

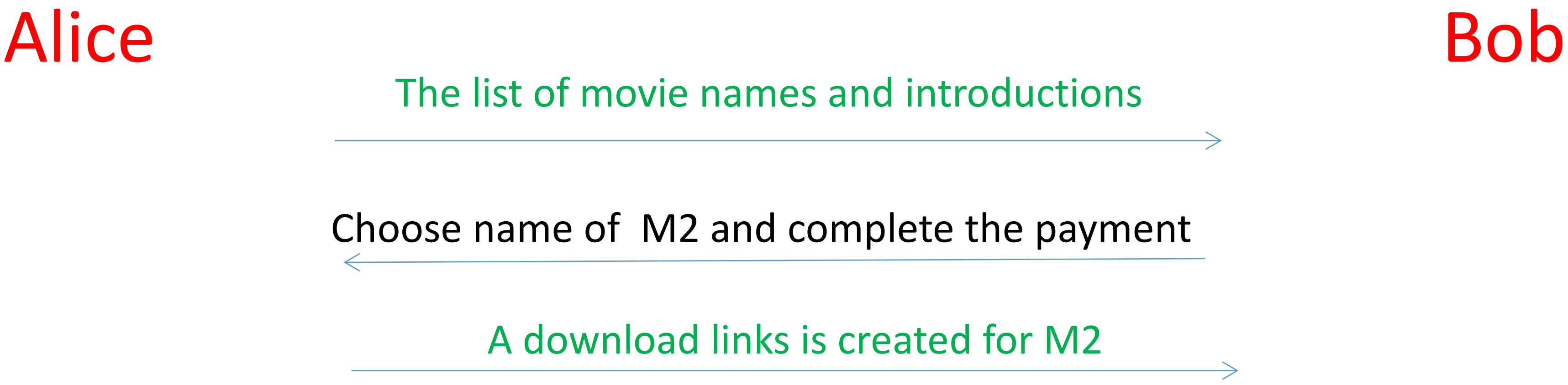
The list of movie names and introductions

Choose name of M2 and complete the payment

A download links is created for M2

What can we do to make the solution better with cryptography?

Purchasing a movie from Alice



Alice

Bob

The list of movie names and introductions

Choose name of M2 and complete the payment

A download links is created for M2

What can we do to make the solution better with cryptography?

Oblivious Transfer Protocol

Definition

OT

Oblivious Transfer

One sender and one receiver.

In cryptography, an oblivious transfer (OT) protocol is a type of protocol in which a sender transfers one of potentially many pieces of information to a receiver, but remains oblivious as to what piece (if any) has been transferred.

The sender doesn’t know what has been sent to the receiver.

↓

The sender doesn’t know what the receiver will get.

Oblivious Transfer

	Sender		Receiver
Input	M_1, M_2,.....,M_n		i
Output	Nothing		M_i

Definition: We say that an OT protocol is secure if

- The sender learns nothing about i, and
- The receiver learns nothing else except M_i.

1-out-of-2 Oblivious Transfer

	Sender		Receiver
Input	M_1, M_2		b
Output	Nothing		M_b

1-out-of-n Oblivious Transfer

	Sender		Receiver
Input	M_1, M_2,.....,M_n		i
Output	Nothing		M_i

k-out-of-n Oblivious Transfer

	Sender		Receiver
Input	M_1, M_2,.....,M_n		t_1,t_2,...,t_k
Output	Nothing		M_{t_i},i=1 to k

Cyclic Group

Let (G,*g*,p) be a cyclic group.

- G is the set of all group elements.
- The set has p number of group elements.
- *g* is the generator of the group G.

$$G = \{g^0, g^1, g^2, g^3, \ldots\ldots, g^{p-1}\}$$

Given x and g, we can compute g^x in a fast way. (See Lecture 6 exponentiation)

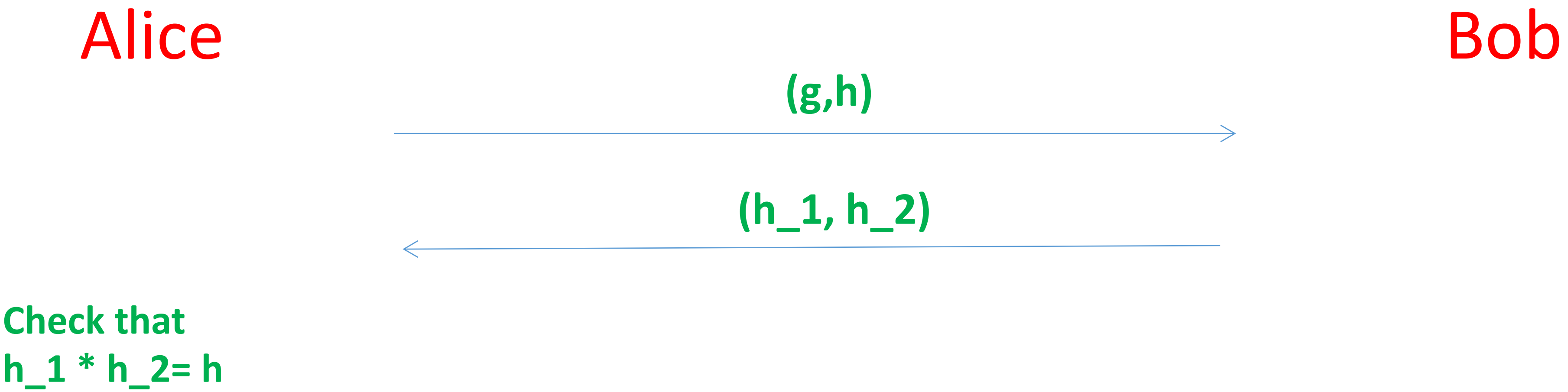
Given g and h, we can compute g·h (basic group operation)

Given g and x, we can compute $g^{-x} = g^{p-x}. \, In\ particuar,\ we\ can\ compute\ g^{-1}$

Given g and x, we can compute $g^{\frac{1}{x}}$ s.t. $x * \frac{1}{X} = 1\ mod\ p$

1-out-of-2 Oblivious Transfer

	Sender		Receiver
Input	M_1, M_2		b
Output	Nothing		M_b



Alice

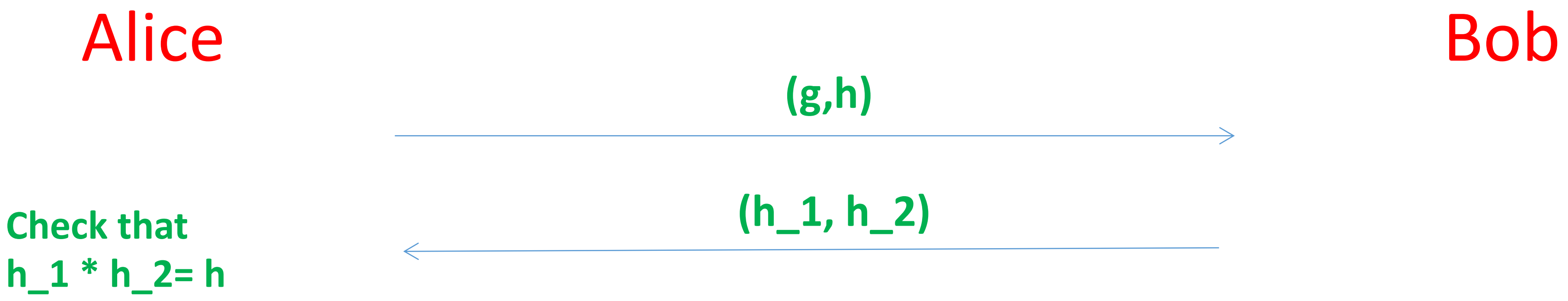
Bob

(g,h)

(h_1, h_2)

Check that
h_1 * h_2= h

1-out-of-2 Oblivious Transfer



Alice

Bob

(g,h)

(h_1, h_2)

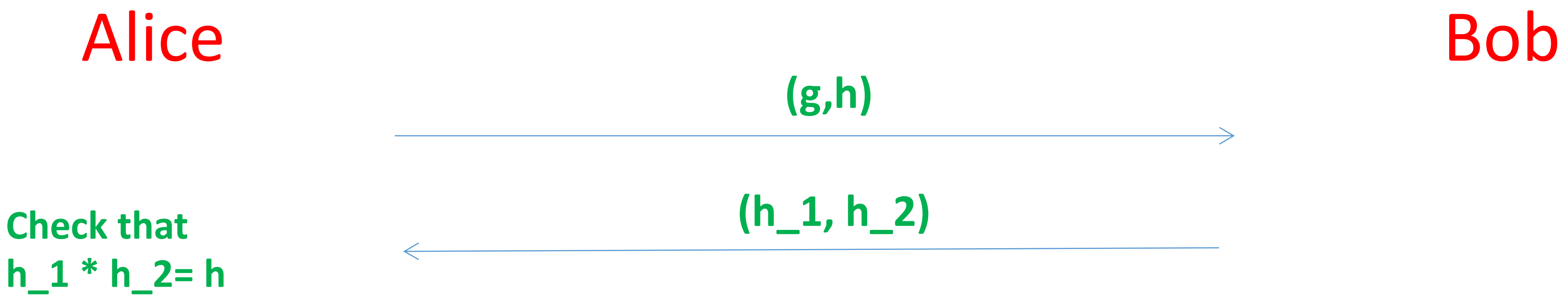
Check that
h_1 * h_2= h

Case 1: If h_1=g^x and h_2=h*g^{-x}, we have h_1*h_2= h.
(suppose x is randomly chosen by Bob)

- When M_1 is encrypted with ElGamal Encryption using pk_1=h_1, Bob can decrypt and get M_1 because sk_1=x.

- When M_2 is encrypted with ElGamal Encryption using pk_2=h_2, Bob cannot decrypt M_2 because sk_2 is unknown.

1-out-of-2 Oblivious Transfer

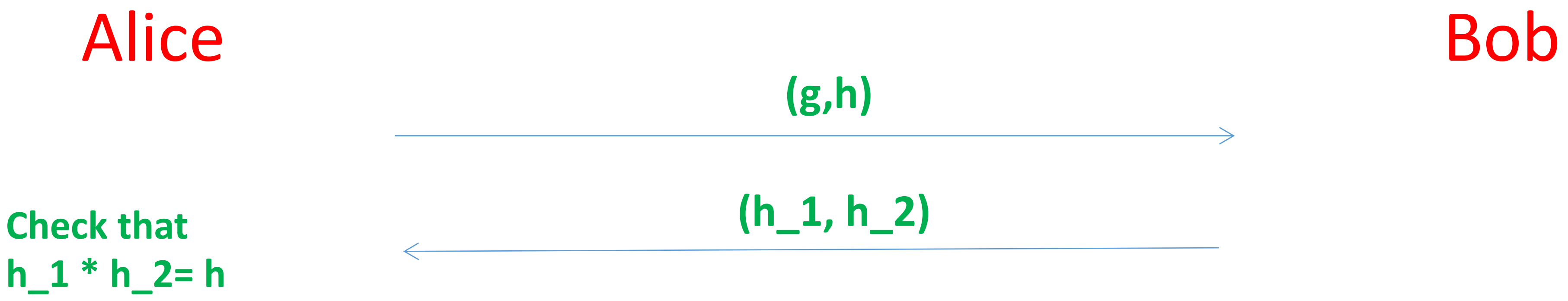


Case 2: If h_1=h*g^{-x} and h_2=g^x, we have h_1*h_2= h.
(suppose x is randomly chosen by Bob)

- When M_1 is encrypted with ElGamal Encryption using pk_1=h_1, Bob cannot decrypt M_1 because sk_1 is unknown.

- When M_2 is encrypted with ElGamal Encryption using pk_2=h_2, Bob can' decrypt M_2 because sk_2=x.

1-out-of-2 Oblivious Transfer



Alice

Bob

(g,h)

(h_1, h_2)

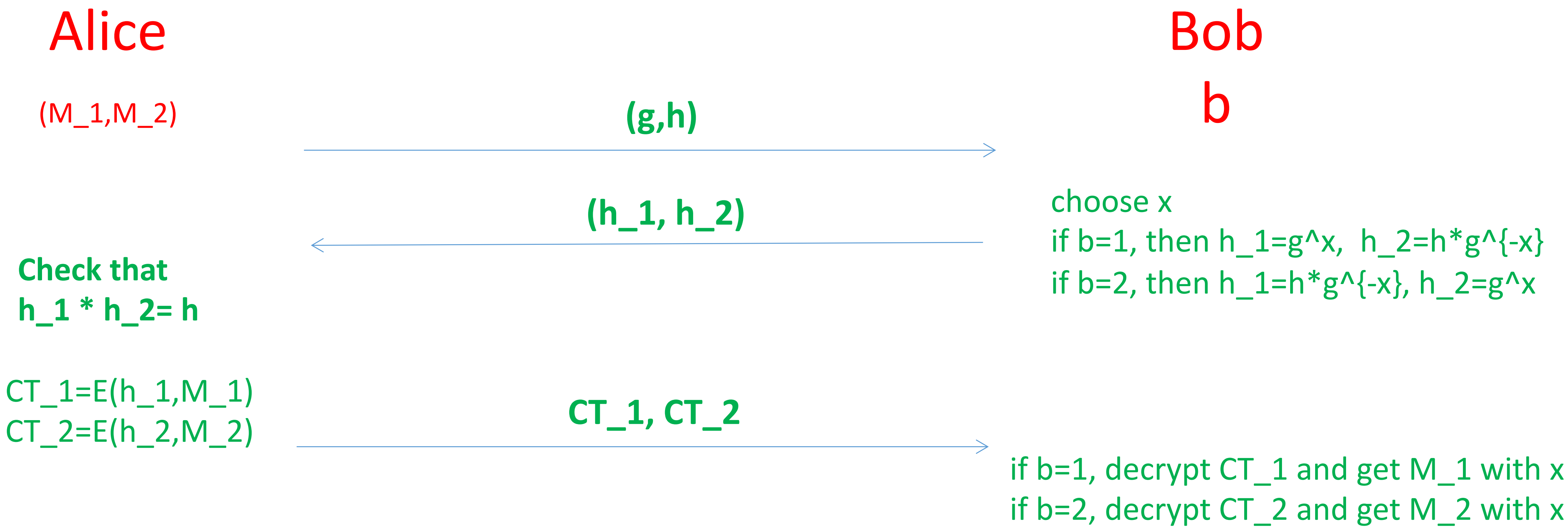
Check that
h_1 * h_2= h

Case 1: If h_1=g^x and h_2=h*g^{-x}, we have h_1*h_2= h.

Case 2: If h_1=h*g^{-x} and h_2=g^x, we have h_1*h_2= h.

If Alice doesn’t know x, it is impossible for Alice to know whether Bob follows case 1 or case 2.

1-out-of-2 Oblivious Transfer



Alice

(M_1,M_2)

Bob

b

(g,h)

choose x
if b=1, then h_1=g^x, h_2=h*g^{-x}
if b=2, then h_1=h*g^{-x}, h_2=g^x

(h_1, h_2)

Check that
h_1 * h_2= h

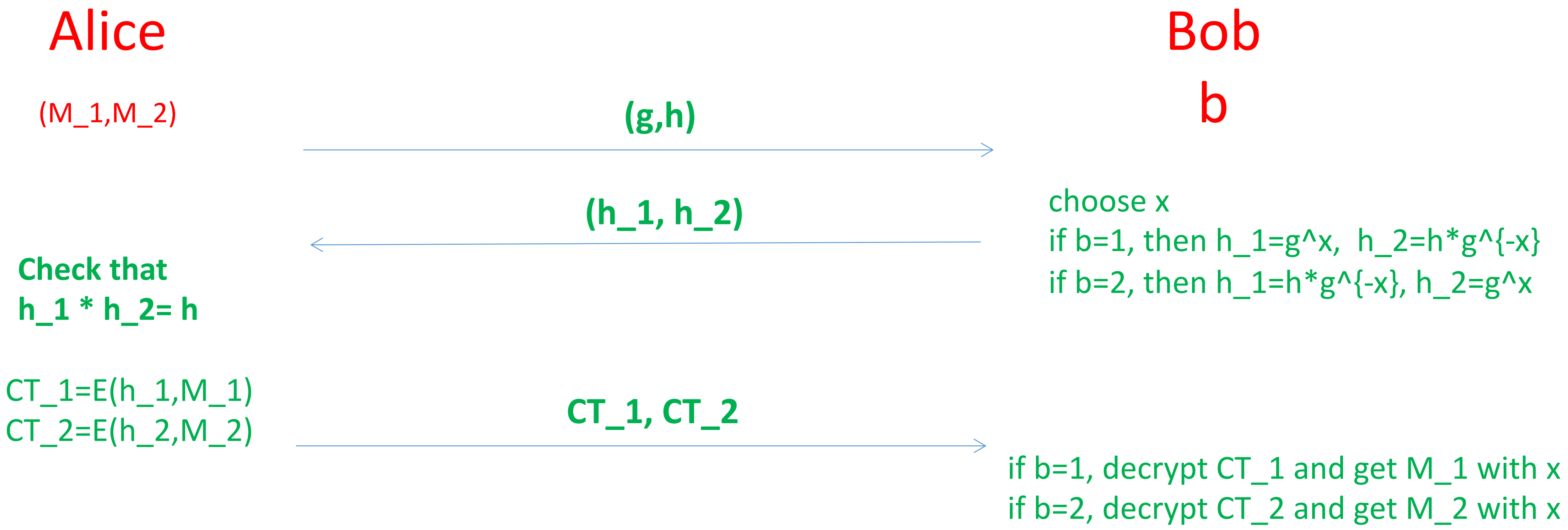
CT_1=E(h_1,M_1)
CT_2=E(h_2,M_2)

CT_1, CT_2

if b=1, decrypt CT_1 and get M_1 with x
if b=2, decrypt CT_2 and get M_2 with x

Alice doesn’t know b.
Bob knows nothing except M_b.

1-out-of-2 Oblivious Transfer



Alice

(M_1,M_2)

Bob

b

(g,h)

(h_1, h_2)

choose x
if b=1, then h_1=g^x, h_2=h*g^{-x}
if b=2, then h_1=h*g^{-x}, h_2=g^x

Check that
h_1 * h_2= h

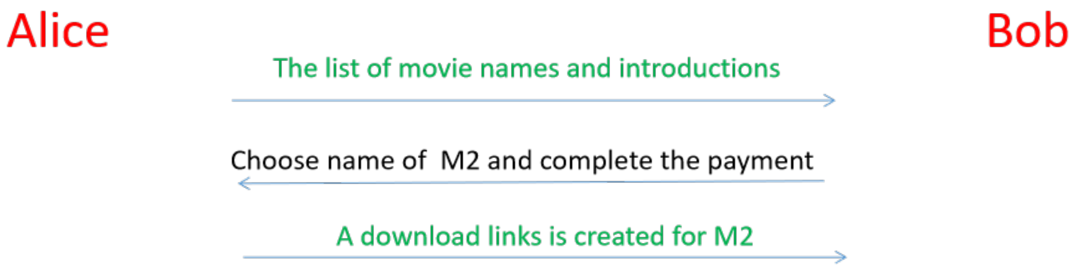
CT_1=E(h_1,M_1)
CT_2=E(h_2,M_2)

CT_1, CT_2

if b=1, decrypt CT_1 and get M_1 with x
if b=2, decrypt CT_2 and get M_2 with x

What are advantages and disadvantages compared to trivial solution?

1-out-of-2 Oblivious Transfer



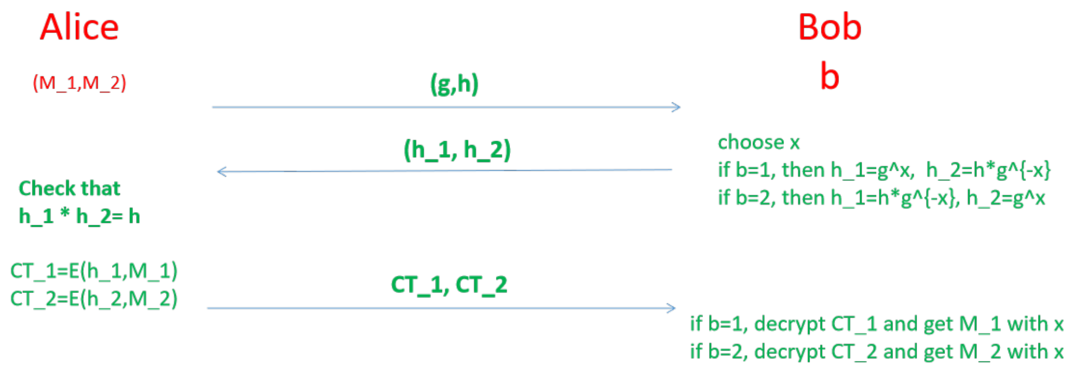
Alice

Bob

The list of movie names and introductions

Choose name of M2 and complete the payment

A download links is created for M2



Alice

(M_1,M_2)

Bob

b

(g,h)

(h_1, h_2)

choose x
if b=1, then h_1=g^x, h_2=h*g^{-x}
if b=2, then h_1=h*g^{-x}, h_2=g^x

Check that
h_1 * h_2= h

CT_1=E(h_1,M_1)
CT_2=E(h_2,M_2)

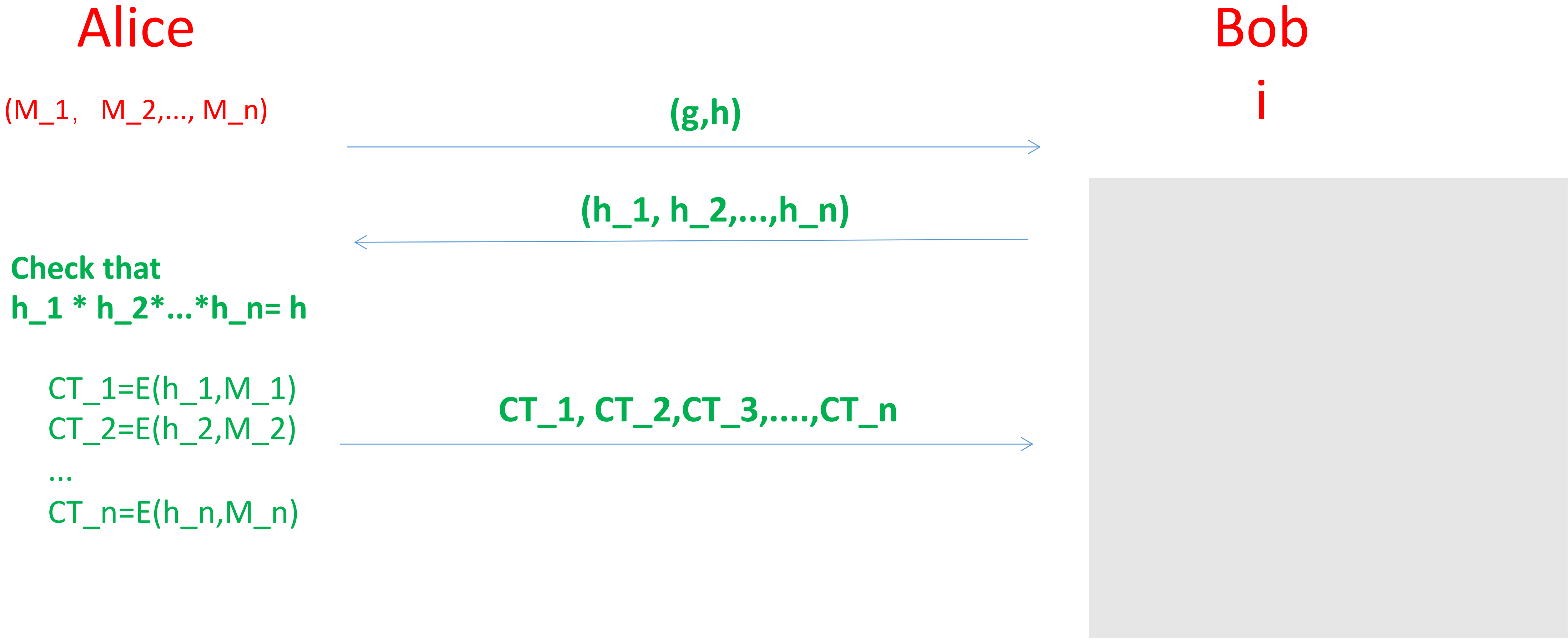
CT_1, CT_2

if b=1, decrypt CT_1 and get M_1 with x
if b=2, decrypt CT_2 and get M_2 with x

	Advantage	Disadvantage
Trivial Solution	easy and transmit M2 only	no privacy on receiver
OT	privacy on receiver	transmit both M1 and M2

1-out-of-n OT protocol

1-out-of-n Oblivious Transfer(try to attack this)



Alice

(M_1， M_2,…, M_n)

Bob

i

(g,h)

(h_1, h_2,…,h_n)

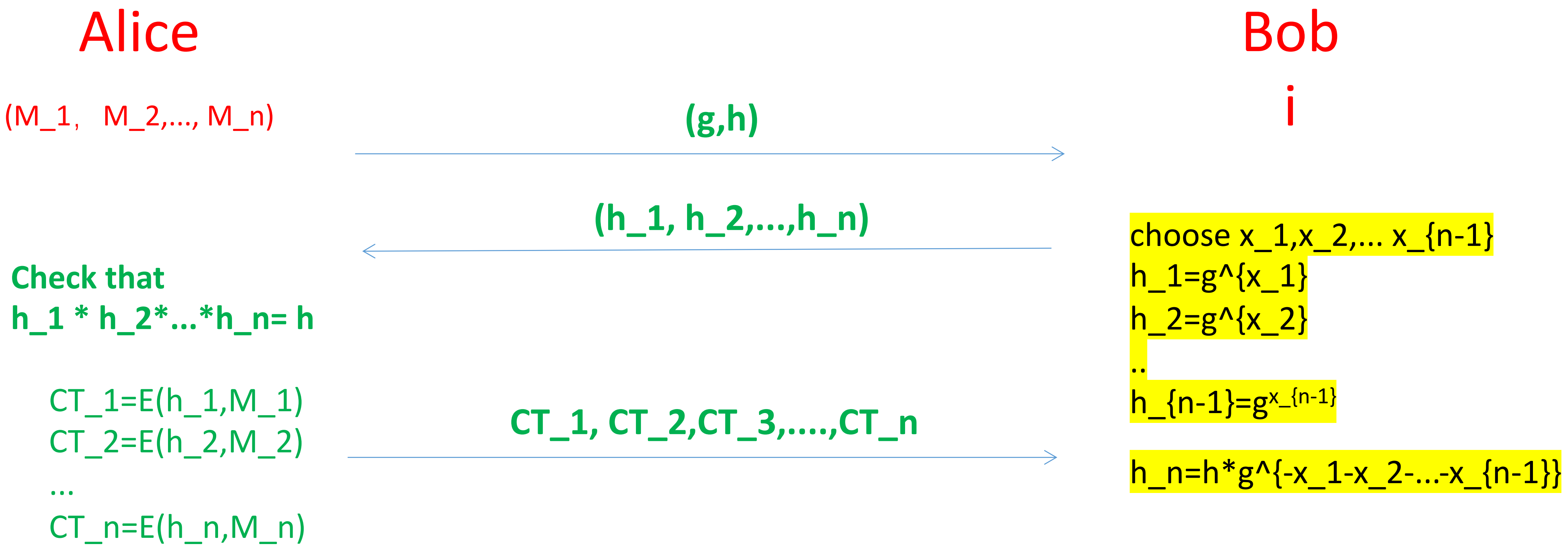
Check that
h_1 * h_2*…*h_n= h

CT_1=E(h_1,M_1)
CT_2=E(h_2,M_2)
…
CT_n=E(h_n,M_n)

CT_1, CT_2,CT_3,….,CT_n

This 1-out-of-n OT protocol is insecure. Can you find the reason？

1-out-of-n Oblivious Transfer(try to attack this)



Alice

(M_1, M_2,..., M_n)

Bob

i

(g,h)

(h_1, h_2,...,h_n)

choose x_1,x_2,... x_{n-1}
h_1=g^{x_1}
h_2=g^{x_2}
..
$h_{n-1}=g^{x_{n-1}}$

h_n=h*g^{-x_1-x_2-...-x_{n-1}}

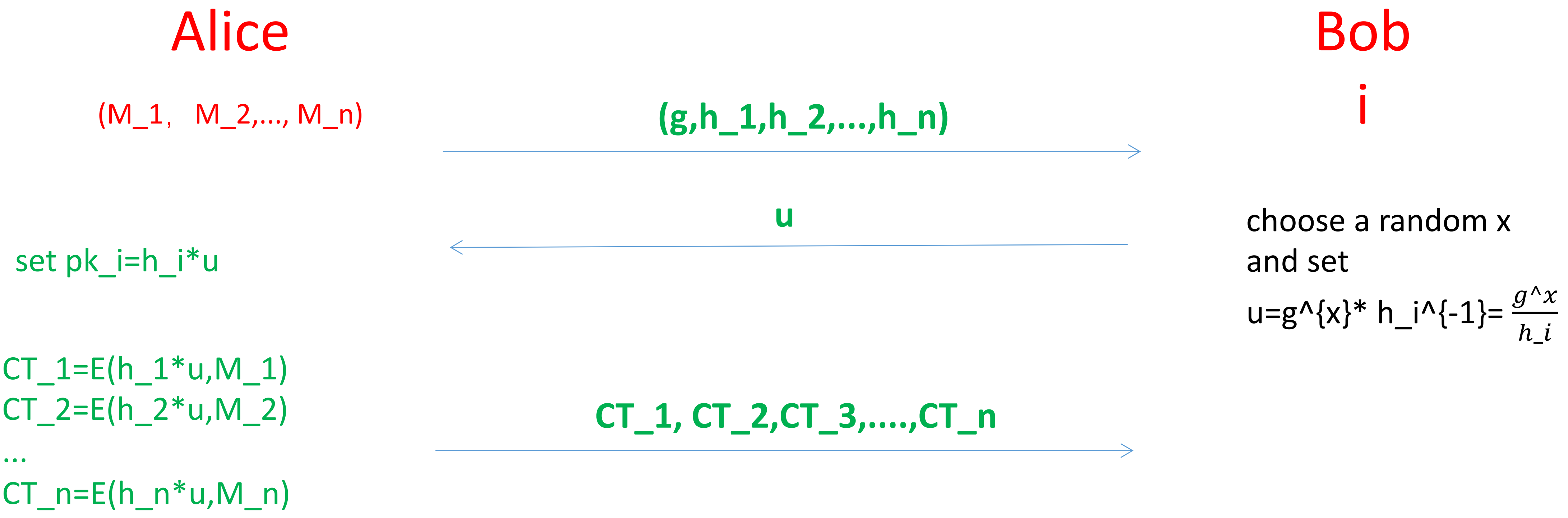
Check that
h_1 * h_2*...*h_n= h

CT_1=E(h_1,M_1)
CT_2=E(h_2,M_2)
...
CT_n=E(h_n,M_n)

CT_1, CT_2,CT_3,....,CT_n

This is actually an (n-1)-out-of-n OT protocol!

1-out-of-n Oblivious Transfer



Alice

(M_1, M_2,..., M_n)

(g,h_1,h_2,...,h_n)

Bob

i

u

set pk_i=h_i*u

choose a random x
and set
u=g^{x}* h_i^{-1}= $\frac{g^{\wedge}x}{h\_i}$

CT_1=E(h_1*u,M_1)
CT_2=E(h_2*u,M_2)
...
CT_n=E(h_n*u,M_n)

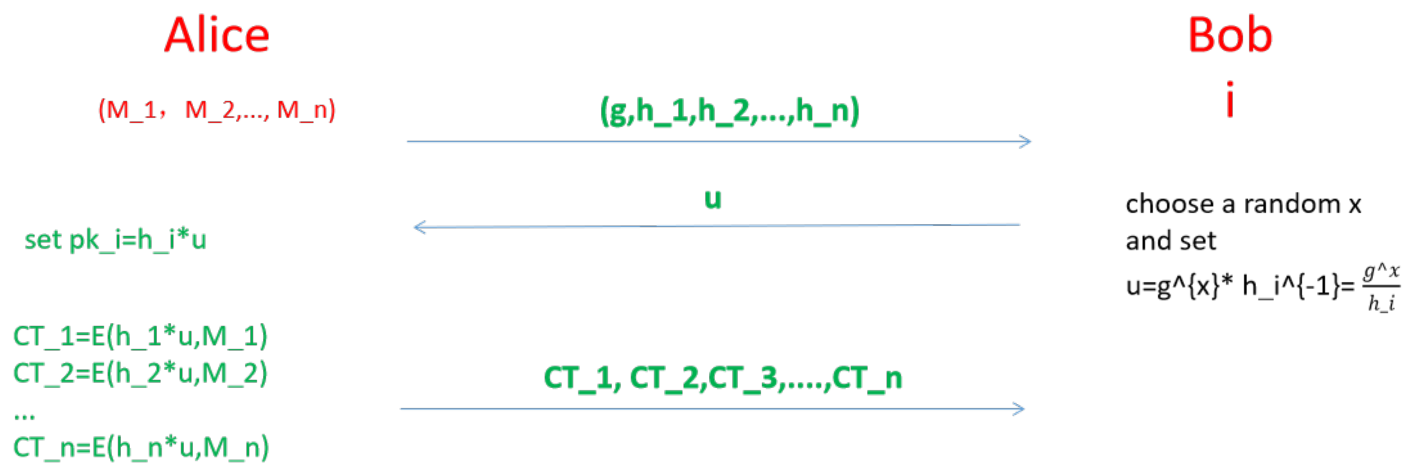
CT_1, CT_2,CT_3,....,CT_n

Then the receiver knows that the secret key of pk_i is sk_i=x.

k-out-of-n Oblivious Transfer

Bob runs the following 1-out-of-n protocol k times with Alice.

1-out-of-n Oblivious Transfer



Alice

(M_1，M_2,..., M_n)

Bob

i

(g,h_1,h_2,...,h_n)

u

choose a random x
and set
u=g^{x}* h_i^{-1}= $\frac{g^\wedge x}{h\_i}$

set pk_i=h_i*u

CT_1=E(h_1*u,M_1)
CT_2=E(h_2*u,M_2)
...
CT_n=E(h_n*u,M_n)

CT_1, CT_2,CT_3,....,CT_n

Workshop

Cyclic Group

Let (G,*g*,p) be a cyclic group.

- G is the set of all group elements.
- The set has p number of group elements.
- *g* is the generator of the group G.

$$G = \{g^0, g^1, g^2, g^3, \ldots\ldots, g^{p-1}\}$$

Let R: X × Y be defined as (x,y)=(h,*a*) such that g^*a*=h

L_R={any h: there exists a such that g^a=h}

I can prove x=h\in L_R <==> I know a such that h=g^a

Let (G,*g*,p) be a cyclic group.

Let R: X × Y be defined as

- x=(g_1, g_2, h_1, h_2)
- y=a

L_R={x: there exists a such that $h_1 = g_1^{a}$ & $h_2 = g_2^{a}$}

Let (G,*g*,p) be a cyclic group.

Let R: X × Y be defined as

- x=(g_1, g_2, h_1, h_2)
- y=a

L_R={x: there exists a such that $h_1 = g_1^{a}$ OR $h_2 = g_2^{a}$}

Let (G,*g*,p) be a cyclic group.

Let R: X × Y be defined as

x=(g, g_1, g_2, g_3)

y=a_1 or (a_2, a_3)

We define

R(x,y)=1 if

- y has one element a_1 and g_1=g^{a_1}, or
- y has two elements (a_2, a_3), and g_2=g^{a_2}, g_3=g^{a_3}

Alice wants to prove x\in L_R to Bob

Case 1 Prover

(g,g_1,g_2,g_3)

a_1, a_2, a_3

Verifier

(g,g_1,g_2,g_3)

1. Choose random r_1, r’_2,r’_3,c_2

2. R1=g^{r_1}

R2=g^{r’_2}*g_2^{-c_2}

R3=g^{r’_3}*g_3^{-c_2}

g_1 = g^a_1
g_2 = g^a_2
g_3 = g^a_3

R1,R2,R3 →

← c

3. Choose a random c\in Z_ p

4. c1+c_2=c mod p

Z1= r_1+c_1*a_1 mod p

Z2= (r’_2-c_2*a_2)+c_2*a_2 mod p

= r’_2

Z3= (r’_3-c_2*a_3)+c_2*a_3 mod p

= r’_3

c1,c2,Z1,Z2,Z3 →

5. Accept if c=c1+c2 mod p

$$g^{Z1} = \mathrm{R1} * g_1^{c_1}$$
$$g^{Z2} = \mathrm{R2} * g_2^{c_2}$$
$$g^{Z3} = \mathrm{R3} * g_3^{c_2}$$

Case 1 Prover

(g,g_1,g_2,g_3)

a_1

Step 1:

1. Choose random r_1, r’_2,r’_3,c_2

2. R1=g^{r_1}

   R2=g^{r’_2}*g_2^{-c_2}

   R3=g^{r’_3}*g_3^{-c_2}

3. Send R1,R2,R3 to the verifier

Step 2:

1. Choose random c from Z_p

2. Send c to the prover

Verifier

(g,g_1,g_2,g_3)

Step 3:

1. c1+c_2=c mod p
2. Z1= r_1+c_1*a_1 mod p
3. Z2= (r’_2-c_2*a_2)+c_2*a_2 mod p = r’_2
4. Z3= (r’_3-c_2*a_3)+c_2*a_3 mod p= r’_3
5. Send (c_1, c_2, Z1,Z2, Z3) to the verifier

Step 4:

Accept if

c=c1+c2 mod p

g^{Z_1}=R1* g_1^{c_1}

g^{Z_2}=R2* g_2^{c_2}

g^{Z_3}=R3* g_3^{c_2}

Case 2 Prover

(g,g_1,g_2,g_3)

a_2,a_3

Verifier

(g,g_1,g_2,g_3)

1. Choose random r’_1, r_2,r_3,c_1

2. R1=g^{r’_1}*g_1^{-c_1}

R2=g^{r_2}

R3=g^{r_3}

R1,R2,R3 →

← c

3. Choose a random c\in Z_ p

4. c1+c_2=c mod p

Z1= (r’_1-c_1*a_1)+c_1*a_1 mod p

= r’_1

Z2= r_2+c_2*a_2 mod p

Z3= r_3+c_2*a_3 mod p

c1,c2,Z1,Z2,Z3 →

5. Accept if c=c1+c2 mod p

$$g^{Z1} = \mathrm{R1} * g_1^{c_1}$$

$$g^{Z2} = \mathrm{R2} * g_2^{c_2}$$

$$g^{Z3} = \mathrm{R3} * g_3^{c_2}$$

Case 2 Prover

(g,g_1,g_2,g_3)

a_2,a_3

Step 1:

1.Choose random r’_1, r_2,r_3,c_1

2. R1=g^{r’_1}*g_1^{-c_1}

R2=g^{r_2}

R3=g^{r_3}

3. Send R1,R2,R3 to the verifier

Step 2:

1. Choose random c from Z_p

2. Send c to the prover

Verifier

(g,g_1,g_2,g_3)

Step 3:

1. c1+c_2=c mod p
2. Z1= (r’_1-c_1*a_1)+c_1*a_1 mod p = r’_2

   Z2= r_2+c_2*a_2 mod p

   Z3=r_3+c_2*a_3 mod p
3. Send (c_1, c_2, Z1,Z2, Z3) to the verifier

Step 4:

Accept if

c=c1+c2 mod p

g^{Z_1}=R1* g_1^{c_1}

g^{Z_2}=R2* g_2^{c_2}

g^{Z_3}=R3* g_3^{c_2}