EXPLORING CRYPTOGRAPHY PROTOCOLS

WITH LIMITED EMPHASIS ON MATHEMATICS ☺

Presented By
@7eRoM

ATTENTION

THESE SLIDES HAVE BEEN CRAFTED USING THE FOUNDATION OF MY MSC COURSE IN CRYPTOGRAPHY PROTOCOLS AT THE UNIVERSITY OF ISFAHAN.

I'VE MADE ADJUSTMENTS TO THE CONTENT TO ALIGN WITH THE SPECIFIC OBJECTIVES OF THIS PRESENTATION.

ALSO, MY INTENTION HAS BEEN TO MINIMIZE THE USE OF MATHEMATICAL CONCEPTS, WHICH MAY RESULT IN SOME CONCEPTS BEING SIMPLIFIED OR LESS PRECISE.

Presented By
@7eRoM

Agenda

1. Identification and Entity Authentications Protocols
2. Zero Knowledge Protocols
3. Key Establishment Protocols
4. Threshold Cryptography and Secret Sharing Protocols
5. **Special Purpose Protocols (like simultaneous contract signing, mental poker, fair exchange)**
6. Identity Based Cryptography
7. Types of Digital Signatures
8. Secure Multiparty Computations

Special Purpose Protocols

Content

- Commitment
- Yes/No Election
- Fair Coin Flipping by the Phone
- Oblivious Transfer
- Simultaneous Contract Signing
- Certified Email
- Fair Exchange
- Mental Poker
- Dining Cryptographers

Special Purpose Protocols

Commitment

- An electronic way to temporarily hide a value that cannot be changed

- **Stage 1 (Commit)**
  - Sender locks a message in a box and sends the locked box to another party called the Receiver

- **Stage 2 (Reveal)**
  - Sender proves to the Receiver that the message in the box is a certain message

Special Purpose Protocols

Commitment

Special Purpose Protocols

Commitment

- **Hiding**
  - at the end of Stage 1, no adversarial receiver learns any information about the committed value
- **Binding**
  - at the end of Stage 1, no adversarial sender can successfully reveal two different values in Stage 2

Special Purpose Protocols

Bit Commitment – Using Symmetric Cryptography

commits to b

Random bit string R

$E_K(R,b)$

K

- Bob decrypts the message to reveal the bit.
- He checks his random string to verify the bit's validity.

Special Purpose Protocols

Formalizing Security properties of commitment schemes

- **Two kinds of adversaries**
  - those with infinite computation power and those with limited computation power
- **Unconditional hiding**
  - the commitment phase does not leak any information about the committed message
- **Computational hiding**
  - an adversary with limited computation power cannot learn anything about the committed message

Special Purpose Protocols

Formalizing Security properties of commitment schemes

- **Unconditional binding**
  - after the commitment phase, an infinite powerful adversary sender cannot reveal two different values
- **Computational binding**
  - after the commitment phase, an adversary with limited computation power cannot reveal two different values

No commitment scheme can be both unconditional hiding and unconditional binding

Special Purpose Protocols

Bit Commitment – Using One-Way Functions

Random string R2

commits to b

Random string R1

h = H(R1|R2|b)

R2

Calculates

- h0 = H(R1|R2|0)
- h1 = H(R1|R2|1)

Check h against h0 and h1

Special Purpose Protocols

Commitment – Pederson Scheme

Large prime p
$2 \leq g \leq p-1$
Another element h
Random r

commits to x

$g^x h^r \bmod p$

x , r

Large prime p
$2 \leq g \leq p-1$

- **Unconditionally Hiding**
  - given x,r, and any x', there exists r' such that $g^x h^r = g^{x'} h^{r'}$
- **Computationally Binding**
  - the sender to open another value $x' \neq x$ is hard.

Special Purpose Protocols

Commitment – Pederson Scheme

Large prime p
$2 \leq g \leq p-1$
Another element h
Random y,s
$1 \leq y,s \leq q$

Public Commitment
$C=g^x h^r \bmod p$

Private Knowledge
x,r

$d = g^y h^s \bmod p$

$1 \leq$ random challenge e $\leq q$

$u=y+ex \bmod q$

$v=s+er \bmod q$

Large prime p
$2 \leq g \leq p-1$

Verification

$g^u h^v == dc^e \pmod p$

Remember?

Similar to Schnorr protocol

Special Purpose Protocols

Yes/No Election – Pederson Scheme

Vote $m \in \{0,1\}$

r_i is random from range $[1, q-1]$

Broadcast

$C_i = commit(m_i, r_i)$
$= h^{m_i} g^{r_i} \ mod\ p$

$e_T(g^{r_i})$

Public encryption function $e_T()$

Private decryption function $d_T()$

Broadcast

$$\boldsymbol{d_T\left(\prod_{i=1}^{n} e_{T(g^{r_i})}\right)}$$

$$= \boldsymbol{\prod_{i=1}^{n} (g^{r_i}) = g^{\Sigma r_i}}$$

$$= \boldsymbol{g^{r}}$$

Each user can calculate the result of election

$$\frac{\prod_{i=1(C_i)}^{n}}{g^r} = \frac{h^{\Sigma m_i}\, g^{\Sigma r_i}}{g^{\Sigma r_i}} = h^{\Sigma m_i}\ mod\ p$$

Special Purpose Protocols

Coin Flipping

Special Purpose Protocols

Coin Flipping – Using One-Way Function

Random number x

`y = H(x)`

Guesses x is even/odd

`Head/Tail , x`

Verification

`y == H(x)`

If Bob's guess is correct, the result of the coin flip is heads.

If Bob's guess is incorrect, the result of the coin flip is tails.

Special Purpose Protocols

Coin Flipping – Using Public-Key Cryptography

Commutative
$D_{k1}(E_{k2}(E_{k1}(m))) = E_{k2}(m)$

(pu_A , pr_A)

(m_1 , m_2) two messages, one indicating heads and the other indicating tails.

$E_A(M_1) , E_A(M_2)$

$E_B(E_A(M_X))$

$D_A(E_B(E_A(M_X))) = E_B(M_X)$

$D_B(E_B(M_X)) = M_X$

(pu_A , pr_A)

(pu_B , pr_B)

(pu_B , pr_B)

Special Purpose Protocols

Oblivious Transfer

Special Purpose Protocols

Oblivious Transfer

Joe Kilian's story:

> Suppose your netmail is being censored by Captain Yossarian. Whenever you send a message, he censors each bit of the message with probability ½, replacing each censored bit by some reversed character.

Special Purpose Protocols

Oblivious Transfer

- Before:
  - Alice knows secret *b*
  - Bob knows nothing
- After:
  - Either:
    - ½ probability: Bob knows *b*
    - ½ probability: Bob knows nothing
  - Alice doesn't know if Bob knows *b*

Special Purpose Protocols

Oblivious Transfer

- **1 out of 2 OT**
  - Alice has two secrets s_0 and s_1
  - At the end of the protocol
    - Bob gets exactly one of s_0 and s_1
    - Alice does not know which one Bob gets

- **1 out of n OT**
  - Alice has n messages
  - Bob gets exactly one message, Alice does not know which one Bob gets

Special Purpose Protocols

Oblivious Transfer – Coin Toss with Capt. Yossarian

Picks
$b=b_1 \oplus b_2 \oplus \ldots \oplus b_n$

$b_1, b_2, \ldots, b_n$

Yossarian Channel

Guesses b is even/odd

Win/Lose

Bob gets for example
$b_1, X, X, b_4, \ldots, b_{n-1}, X$

Bob wins if his guess is correct

What if Alice is a liar and tells Bob “You Lose”

Special Purpose Protocols

Oblivious Transfer – Coin Toss with Capt. Yossarian

Picks
$b=b_1 \oplus b_2 \oplus \ldots \oplus b_n$

$b_1, b_2, \ldots, b_n$ — Yossarian Channel →

← Guesses b is even/odd

$b_1, b_2, \ldots, b_n$ →

Bob gets for example
$B_1, X, X, b_4, \ldots, b_{n-1}, X$

Bob can verify through calculating

$b=b_1 \oplus b_2 \oplus \ldots \oplus b_n$

Bob wins if his guess is correct

Special Purpose Protocols

Oblivious Transfer – Coin Toss with Capt. Yossarian

Can we approximate oblivious transfer without Capt. Yossarian channel?

Generates 2 publc-private pair keys

(pu_1 , pr_1)

(pu_2 , pr_2)

pu_1 , pu_2

$E_{pux}(K)$

$E_{K1}(b_1) , E_{K2}(b_2)$

Generates symmetric key K

Picks either pu_1 or pu_2

$K_1 = E_{pr1}(E_{pux}(K)) = K$ or meaningless bits

$K_2 = E_{pr2}(E_{pux}(K)) = K$ or meaningless bits

Suppose Bob used pu_1

$D_K(E_{K1}(b_1)) = b_1$

$D_K(E_{K1}(b_2)) =$ meaningless bits

Special Purpose Protocols

Oblivious Transfer – Bellare–Micali 1–out–2–OT protocol

Secrets S_0 , S_1

group G of order q

generator g

c is random from range [1,q-1]

r_0, r_1 is random from range [1,q-1]

c

Z_0 , Z_1

$E_0=(g^{r_0}$, $H(Z_0^{r_0}) \oplus S_0)$

$E_1=(g^{r_1}$, $H(Z_1^{r_1}) \oplus S_1)$

group G of order q

generator g

k is random from range [1,q-1]

b is randomly 0 or 1

$Z_b = g^k$

$Z_{1-b} = c * g^{-k}$

Decrypts $E_b = (v_1, v_2)$ by computing

$H(v_1^k) \oplus v_2$

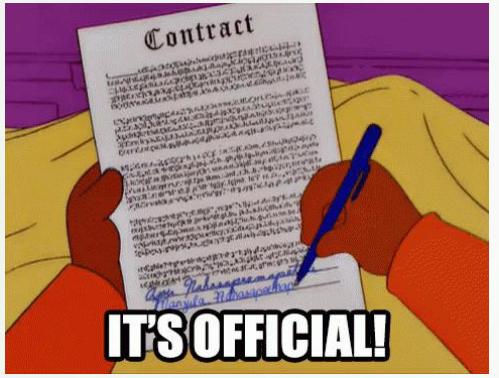
$E_b = H(g^{r_b K}) \oplus H(Z_b^{r_b}) \oplus S_b$

$= H(Z_b^{r_b}) \oplus H(Z_b^{r_b}) \oplus S_b$

$= S_b$

Special Purpose Protocols

Simultaneous Contract Signing



Alice and Bob wish to sign a contract under the condition that

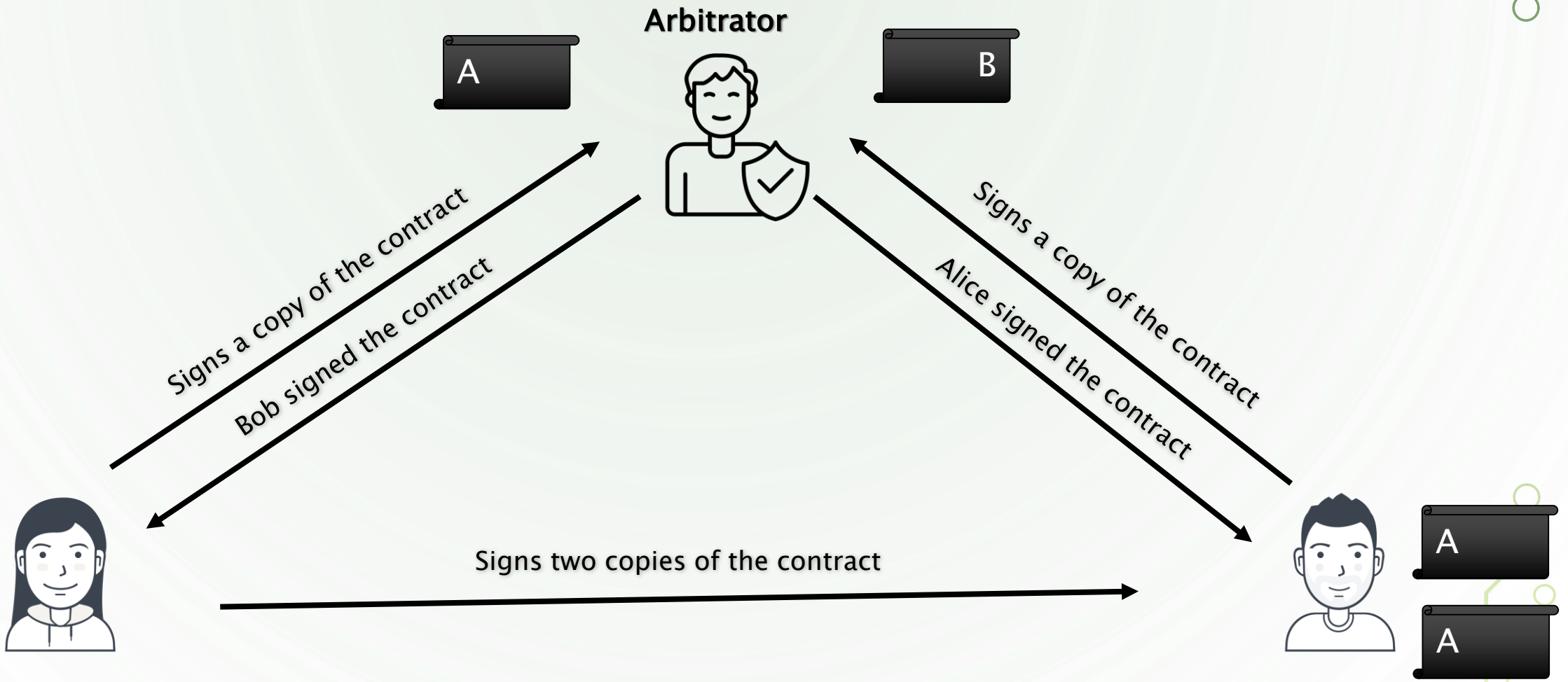
neither of them will sign it until they ensure that the other party has also signed.

Special Purpose Protocols

Simultaneous Contract Signing

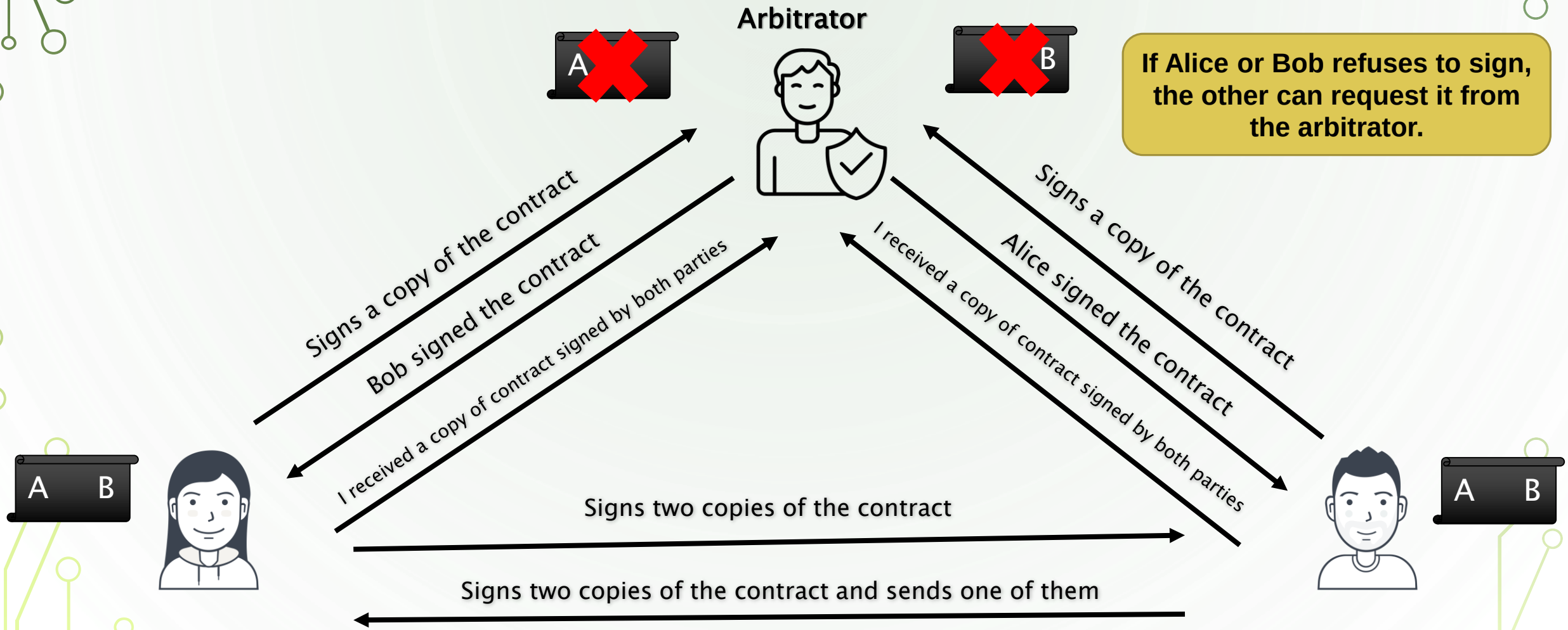
Special Purpose Protocols

Simultaneous Contract Signing – With Arbitrator



Special Purpose Protocols

Simultaneous Contract Signing – With Arbitrator



Special Purpose Protocols

Simultaneous Contract Signing – Face to Face & Without Arbitrator

Gradually Release

- How many letters are required to signify a commitment to the contract?
- There is no specific point for commitment.

Special Purpose Protocols

Simultaneous Contract Signing – Gradually Release

- Taking small steps alternately to reach the final signature
- Both parties send messages to each other one by one, alternatively, with each message stating *'I am committed to the contract with a probability of P.'*
- The judge will grant the signer's commitment with a probability of p.

Special Purpose Protocols

Simultaneous Contract Signing – Gradually Release

- Alice and Bob agree on the date by which the contract signature should be concluded
- Both parties send messages to each other one by one, alternatively, with each message stating *'I am committed to the contract with a probability of P.'*
- Alice decides that the difference between the probability of her commitment and Bob's commitment should not exceed 'a'.
- Bob decides that the difference between the probability of her commitment and Bob's commitment should not exceed 'b'.
- Steps:
  - Alice sends a signed message with a confidence probability of 'p = a' to Bob.
  - Bob sends a signed message with a confidence probability of 'p = a+b' to Bob.
  - Each time, Alice increases the probability by 'a' and sends the message to Bob.
  - Each time, Bob increases the probability by 'b' and sends the message to Alice.

Special Purpose Protocols

Simultaneous Contract Signing – Gradually Release

- This process continues until both receive messages with a confidence probability of '$p = 1$' or until the agreed-upon date passes.

- If both receive messages with a confidence probability of '1', the process is complete; otherwise, ...
  - Each party can present the contract and the opponent's last signed message, which includes the confidence probability 'p' to the judge.
  - The judge randomly selects 'r' between 0 and 1. If 'r' is less than 'p' both parties are bound by the contract; otherwise, the judge does not accept either party's commitment.

Special Purpose Protocols

Simultaneous Contract Signing – based on DES

$(K_{L1}, K_{R1})\ (M_{L1}, M_{R1})$
$(K_{L2}, K_{R2})\ (M_{L2}, M_{R2})$
$\ldots \quad \ldots$
$(K_{Ln}, K_{Rn})\ (M_{Ln}, M_{Rn})$

$E_{KLi}(M_{Li})=C_{Li}$ $E_{KRi}(M_{Ri})=C_{Ri}$

(C_{L1}, C_{R1})
(C_{L2}, C_{R2})
$\ldots$
(C_{Ln}, C_{Rn})

Decrypts each half part of the Bob's messages to verify

Now, decrypts the other half and obtains the signed contract

(C_{L1}, C_{R1})
(C_{L2}, C_{R2})
$\ldots$
(C_{Ln}, C_{Rn})

$K_{(L|R)1}$
$K_{(L|R)2}$
$\ldots$
$K_{(L|R)n}$

Oblivious Transfer

1st bit of 2n keys

2nd bit of 2n keys

$\vdots$

nth bit of 2n keys

Decrypts each half part of the Alice's messages to verify

Now, decrypts the other half and obtains the signed contract

$(K_{L1}, K_{R1})\ (M_{L1}, M_{R1})$
$(K_{L2}, K_{R2})\ (M_{L2}, M_{R2})$
$\ldots \quad \ldots$
$(K_{Ln}, K_{Rn})\ (M_{Ln}, M_{Rn})$

$E_{KLi}(M_{Li})=C_{Li}$ $E_{KRi}(M_{Ri})=C_{Ri}$

(C_{L1}, C_{R1})
(C_{L2}, C_{R2})
$\ldots$
(C_{Ln}, C_{Rn})

The probability of Alice sending incorrect bits while Bob does not understand is $\frac{1}{2^n}$

Special Purpose Protocols

Certified Email

Alice wants to send a letter to Bob in a way that he cannot read the letter until he signs an acknowledgment receipt.

Special Purpose Protocols

M = Alice's Mail
J = Junk Message
K = Key
K_{Li} = Random bits
$K_{Ri} = K \oplus K_{Li}$

(K_{L1}, K_{R1}) (J_{L1}, J_{R1})
(K_{L2}, K_{R2}) (J_{L2}, J_{R2})
...
(K_{Ln}, K_{Rn}) (J_{Ln}, J_{Rn})

$E_{KLi}(J_{Li})=C_{Li}$ $E_{KRi}(J_{Ri})=C_{Ri}$

(C_{L1}, C_{R1})
(C_{L2}, C_{R2})
...
(C_{Ln}, C_{Rn})

Certified Email

$E_K(M)$

(C_{L1}, C_{R1})
(C_{L2}, C_{R2})
...
(C_{Ln}, C_{Rn})

(C_{L1}, C_{R1})
(C_{L2}, C_{R2})
...
(C_{Ln}, C_{Rn})

$K_{(L|R)1}$
$K_{(L|R)2}$
...
$K_{(L|R)n}$

Oblivious Transfer

1st bit of 2n keys

2nd bit of 2n keys

⋮

nth bit of 2n keys

$R_i = R_{Li}+R_{Ri}$ = acknowledgment receipt

(K_{L1}, K_{R1}) (R_{L1}, R_{R1})
(K_{L2}, K_{R2}) (R_{L2}, R_{R2})
...
(K_{Ln}, K_{Rn}) (R_{Ln}, R_{Rn})

$E_{KLi}(R_{Li})=C_{Li}$ $E_{KRi}(R_{Ri})=C_{Ri}$

(C_{L1}, C_{R1})
(C_{L2}, C_{R2})
...
(C_{Ln}, C_{Rn})

Now, decrypts the other half and obtains the signed contract

Now, decrypts the other half and obtains the signed contract

Now, decrypts the other half and obtains the signed contract

Alice obtains Bob's secret

Bob regenerates K by doing XOR and then obtains Alice's secret

Special Purpose Protocols

Fair Exchange of Secrets

- Alice knows secret A and Bob knows secret B.
- Alice shares secret A with Bob and receives secret B from him.
- Also, Bob shares secret B with Alice and receives secret A from her.

How about the quality of the Secret!

Special Purpose Protocols

Fair Exchange of Secrets

Special Purpose Protocols

Fair Exchange of Secrets

M = Alice's Secret
J = Junk Message
K = Key
K_{Li} = Random bits
$K_{Ri} = K \oplus K_{Li}$

(K_{L1}, K_{R1}) (J_{L1}, J_{R1})
(K_{L2}, K_{R2}) (J_{L2}, J_{R2})
...
(K_{Ln}, K_{Rn}) (J_{Ln}, J_{Rn})

$E_{KLi}(J_{Li})=C_{Li}$ $E_{KRi}(J_{Ri})=C_{Ri}$

(C_{L1}, C_{R1})
(C_{L2}, C_{R2})
...
(C_{Ln}, C_{Rn})

$E_K(M)$

(C_{L1}, C_{R1})
(C_{L2}, C_{R2})
...
(C_{Ln}, C_{Rn})

(C_{L1}, C_{R1})
(C_{L2}, C_{R2})
...
(C_{Ln}, C_{Rn})

$K_{(L|R)1}$
$K_{(L|R)2}$
...
$K_{(L|R)n}$

Oblivious Transfer

1st bit of 2n keys

2nd bit of 2n keys

⋮

nth bit of 2n keys

Now, decrypts the other half and obtains the signed contract

Now, decrypts the other half and obtains the signed contract

Alice obtains the acknowledgment receipt

$R_i = R_{Li}+R_{Ri}$ = Bob's Secret

(K_{L1}, K_{R1}) (R_{L1}, R_{R1})
(K_{L2}, K_{R2}) (R_{L2}, R_{R2})
...
(K_{Ln}, K_{Rn}) (R_{Ln}, R_{Rn})

$E_{KLi}(R_{Li})=C_{Li}$ $E_{KRi}(R_{Ri})=C_{Ri}$

(C_{L1}, C_{R1})
(C_{L2}, C_{R2})
...
(C_{Ln}, C_{Rn})

Now, decrypts the other half and obtains the signed contract

Now, decrypts the other half and obtains the signed contract

Bob obtains K by doing XOR

Special Purpose Protocols

Mental Poker

- Alice and Bob want to play poker, we need a way to deal 5 cards to each of Alice and Bob so that
  - Alice's hand of 5 cards does not overlap with Bob's hand
  - Neither Alice nor Bob can control which cards they each get
  - Neither Alice nor Bob knows the other party's hand
  - Both hands should be random provided one party follows the protocol

- First solution due to Shamir, Rivest, and Adelman in 1980
  - uses commutative encryption schemes

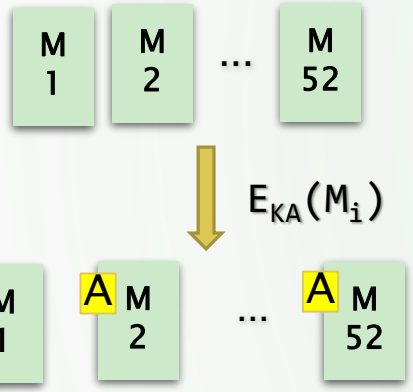
Special Purpose Protocols

Mental Poker

- Alice encrypts M1, M2, …, M52 and then randomly permute them and send the ciphertexts to Bob
- Bob picks 5 cards as Alice's hand and sends them to Alice
- Alice decrypts them to get his hand
- Bob picks 5 other cards as his hand, encrypts them using his key, and sends them to Alice
- Alice decrypts the 5 ciphertexts and sends to Bob
- Bob decrypts what Alice sends and gets his hand
- Both Alice and Bob reveals their key pairs to the other party and verify that the other party was not cheating.

Special Purpose Protocols

Mental Poker

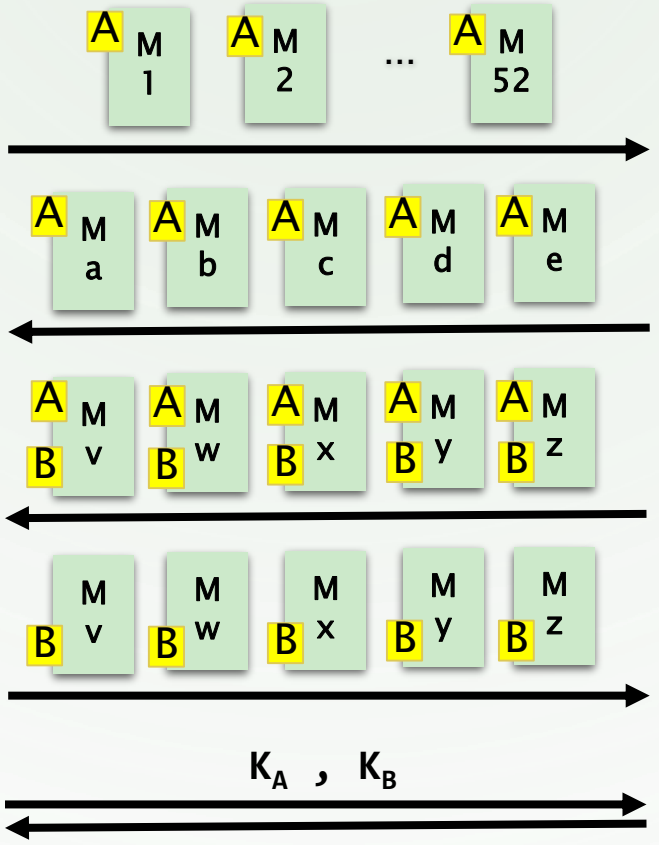


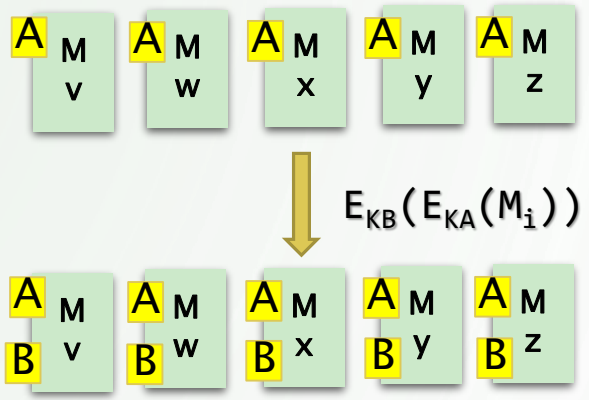
Here are Alice's hand.

She decrypts them.

Commutative

$D_{K1}(E_{k2}(E_{k1}(m))) = E_{k2}(m)$





Here are Bob's hand.

He decrypts them.

Special Purpose Protocols

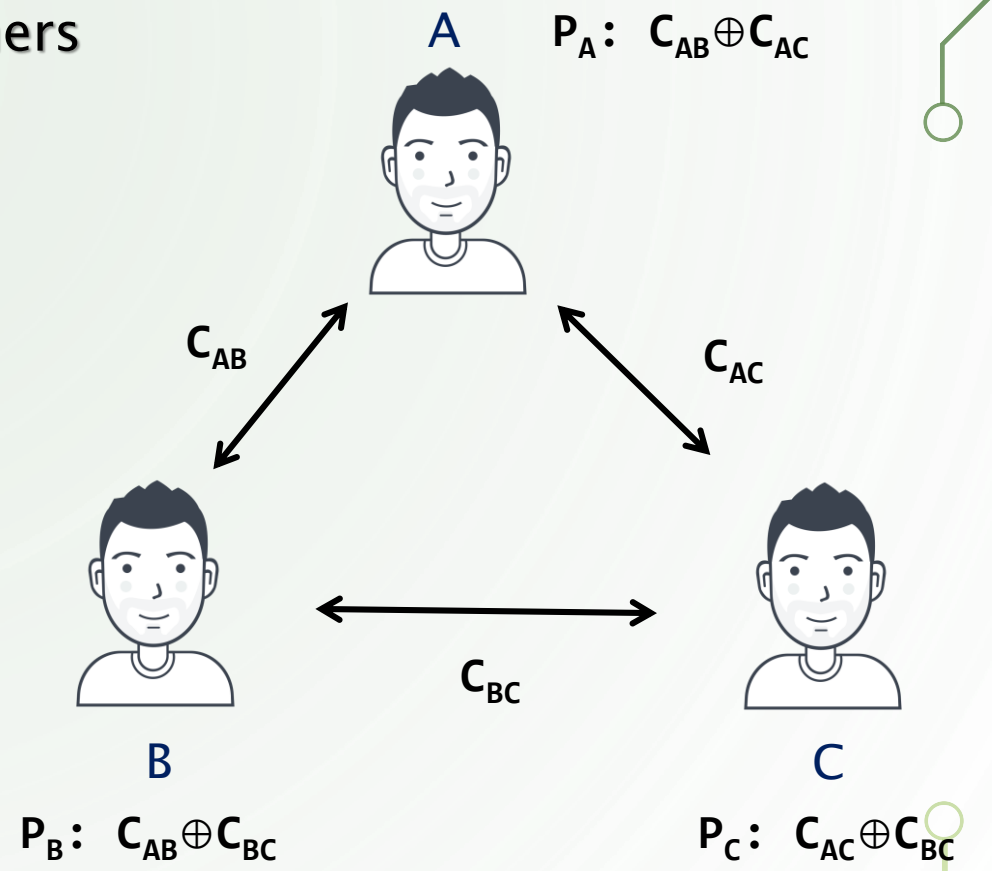
Dining Cryptographers

- Cryptographers gather around a table for dinner.
- The waiter informs them that the meal has been paid for by someone, who could be **one of the cryptographers** or **NSA**.
- The cryptographers respect each other's right to make an anonymous payment, but want to find out whether the NSA paid.

Special Purpose Protocols

Dining Cryptographers

- Suppose 3 cryptographers.
- Every two cryptographers establish a shared one-bit secret, say by tossing a coin.
- In the second stage, each cryptographer publicly announces a bit, which is:
  - if they didn't pay for the meal, the XOR of the two shared bits they hold with their two neighbours,
  - if they did pay for the meal, do an extra XOR with bit 1 (or negate his result)
- Computes the XOR of the three bits announced. If the result is 0, it implies that the NSA must have paid the bill. Otherwise, one of the cryptographers paid.



Result: $P_A \oplus P_B \oplus P_C$

Special Purpose Protocols

Dining Cryptographers

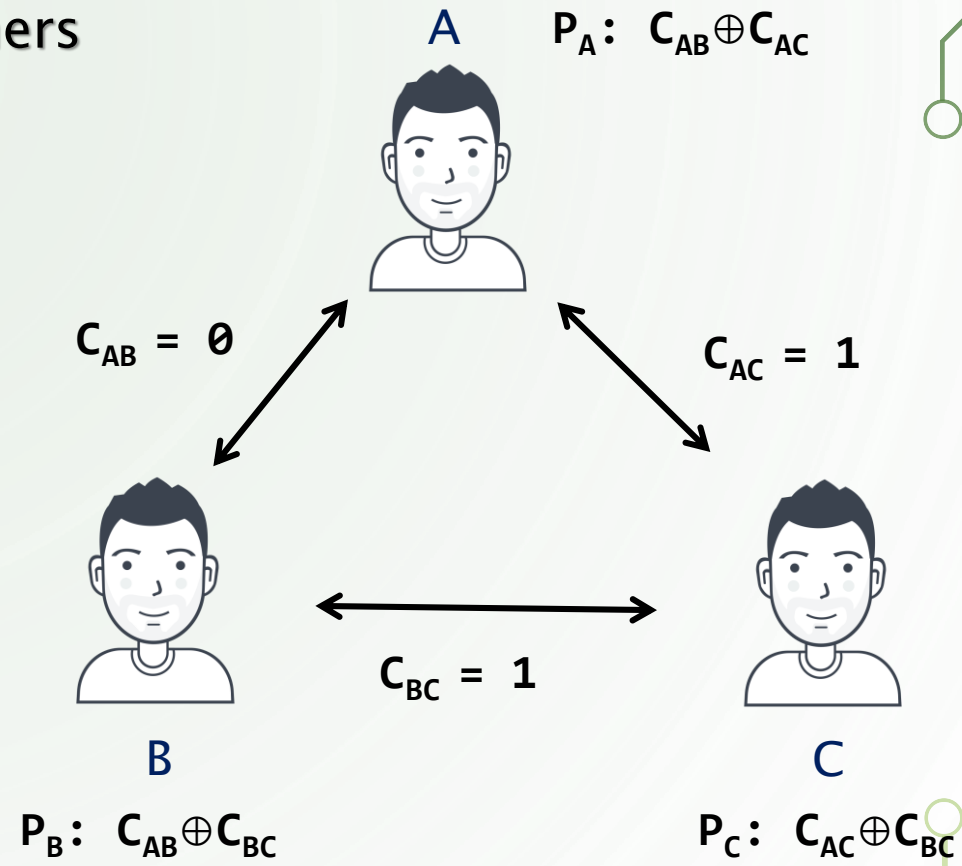
Scenario 1: NSA paid

P_A: $C_{AB} \oplus C_{AC} = 0 \oplus 1 = 1$

P_B: $C_{AB} \oplus C_{BC} = 0 \oplus 1 = 1$

P_C: $C_{AC} \oplus C_{BC} = 1 \oplus 1 = 0$

Result: $P_A \oplus P_B \oplus P_C = 1 \oplus 1 \oplus 0 = 0$



Result: $P_A \oplus P_B \oplus P_C$

Special Purpose Protocols

Dining Cryptographers

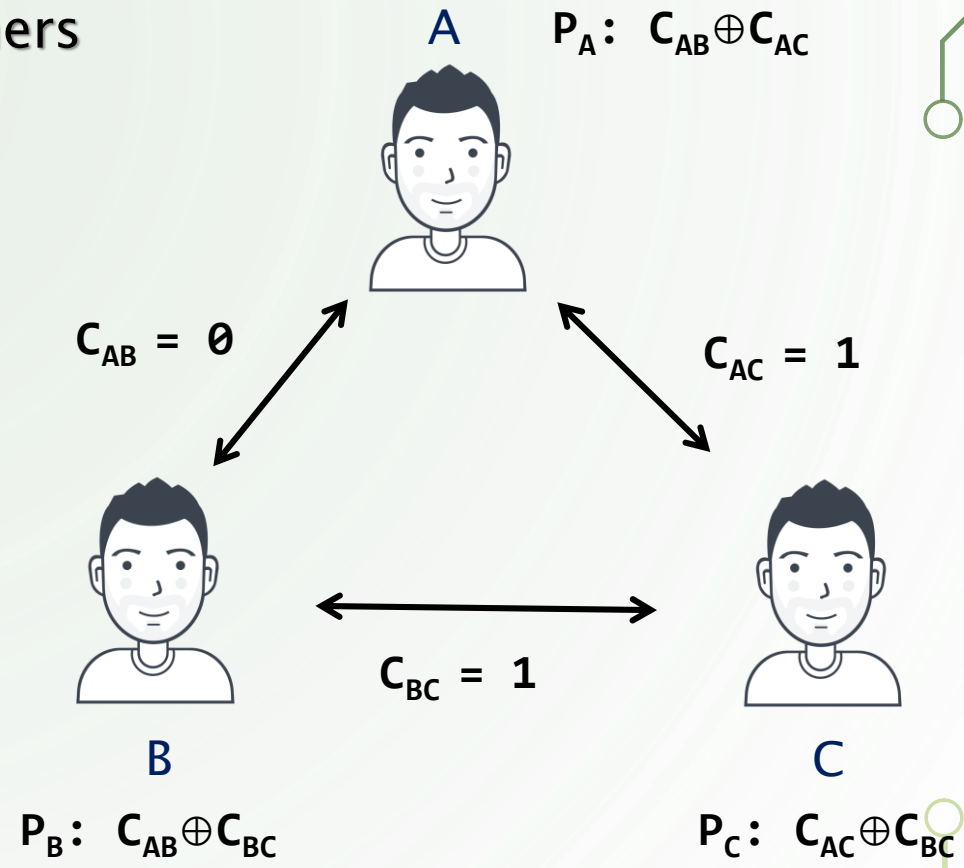
Scenario 2: One of the cryptographers paid (Suppose B)

P_A: $C_{AB} \oplus C_{AC}$ = 0⊕1 = 1

P_B: ~($C_{AB} \oplus C_{BC}$) = ~(0⊕1) = 0

P_C: $C_{AC} \oplus C_{BC}$ = 1⊕1 = 0

Result: $P_A \oplus P_B \oplus P_C$ = 1⊕0⊕0 = 1



Result: $P_A \oplus P_B \oplus P_C$

Ref

1. Cryptography Protocols Course, Dr. Hamid Mala, University of Isfahan
2. https://www.cs.purdue.edu/homes/ninghui/courses/Fall05/lectures/355_Fall05_lect25.pdf
3. https://en.wikipedia.org/wiki/Dining_cryptographers_problem
4. https:// freepik.com
5. https://www.iconfinder.com/UsersInsights
6. https://www.iconfinder.com/Chanut-is
7. https://www.iconfinder.com/iconsets/softwaredemo