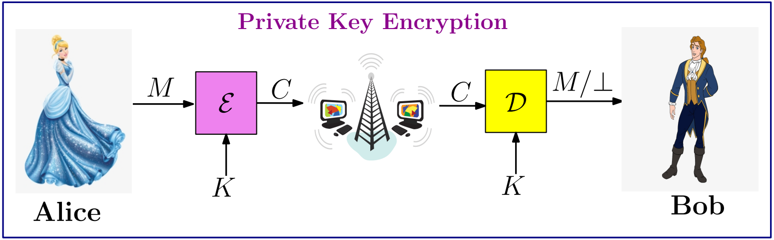
# Cryptology

**Sabyasachi Karati**

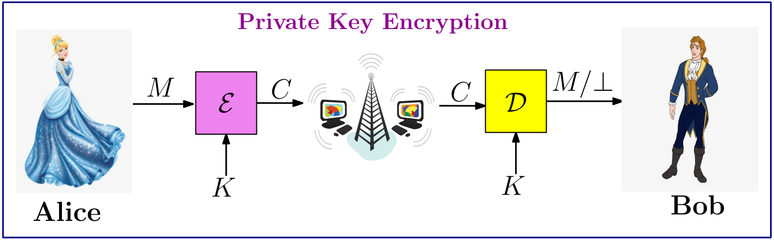
Assistant Professor
Cryptology and Security Research Unit (C.S.R.U)
R. C. Bose Centre for Cryptology and Security
Indian Statistical Institute (ISI)
Kolkata, India

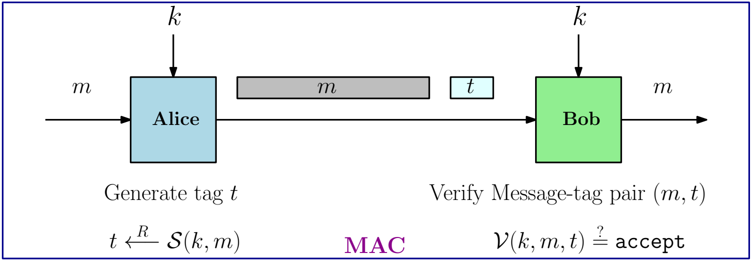
**Lecture 11**

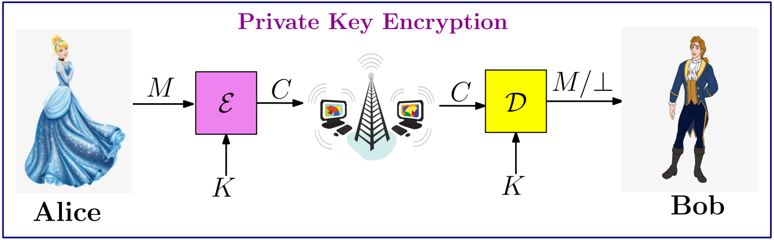
# Public Key Cryptography and Key Exchange Protocol


Private Key Encryption
$M$
$\mathcal{E}$
$C$
$K$
$C$
$\mathcal{D}$
$M/\perp$
$K$
Alice
Bob

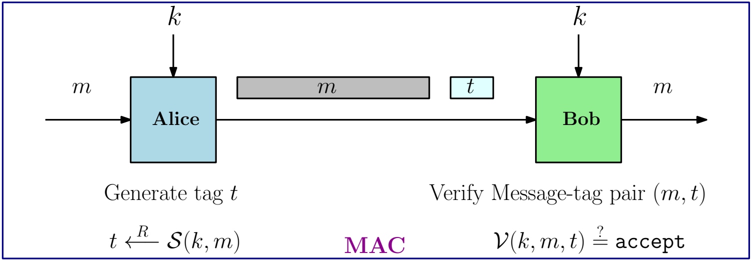
# Introduction






**How to setup the shared Key?**



## Introduction

- Initial sharing of a secret key can be done using a trusted messenger service.

- Initial sharing of a secret key can be done using a trusted messenger service.
  - Unavailable to the average person.

# Introduction

- Initial sharing of a secret key can be done using a trusted messenger service.
    - Unavailable to the average person.
    - Governments, the military, intelligence organizations, and other such entities do have the means to share keys in this way.

# Introduction

- Initial sharing of a secret key can be done using a trusted messenger service.
  - Unavailable to the average person.
  - Governments, the military, intelligence organizations, and other such entities do have the means to share keys in this way.
- A more pragmatic method.
  - Two parties physically meet.

# Introduction

- Initial sharing of a secret key can be done using a trusted messenger service.
  - Unavailable to the average person.
  - Governments, the military, intelligence organizations, and other such entities do have the means to share keys in this way.
- A more pragmatic method.
  - Two parties physically meet.
  - Creates a ramdom shared-key.

# Introduction

- Initial sharing of a secret key can be done using a trusted messenger service.
  - Unavailable to the average person.
  - Governments, the military, intelligence organizations, and other such entities do have the means to share keys in this way.
- A more pragmatic method.
  - Two parties physically meet.
  - Creates a ramdom shared-key.
  - For average persons, this may cause severe travel cost.

- Let there be $u$ members of an organization.

# Introduction

- Let there be $u$ members of an organization.
- Each user communicates with each other.

# Introduction

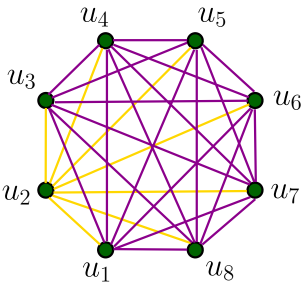
- Let there be $u$ members of an organization.
- Each user communicates with each other.
- Total $\binom{u}{2}$ keys are required.

# Introduction

- Let there be $u$ members of an organization.
- Each user communicates with each other.
- Total $\binom{u}{2}$ keys are required.
- Each user needs to store $(u-1)$ keys.

# Introduction

- Let there be $u$ members of an organization.
- Each user communicates with each other.
- Total $\binom{u}{2}$ keys are required.
- Each user needs to store $(u-1)$ keys.



- Let the organization in question is large.

# Introduction

- Let the organization in question is large.
    1. The proliferation of many secret keys is a significant logistical problem.

- Let the organization in question is large.
    1. The proliferation of many secret keys is a significant logistical problem.
    2. All these secret keys must be stored securely.

# Introduction

- Let the organization in question is large.
    1. The proliferation of many secret keys is a significant logistical problem.
    2. All these secret keys must be stored securely.
        - The more keys there are the harder it is to protect them.

- Let the organization in question is large.
  1. The proliferation of many secret keys is a significant logistical problem.
  2. All these secret keys must be stored securely.
     - The more keys there are the harder it is to protect them.
     - Higher the chance of some keys being stolen by an adversary.

- Let the organization in question is large.
  1. The proliferation of many secret keys is a significant logistical problem.
  2. All these secret keys must be stored securely.
     - The more keys there are the harder it is to protect them.
     - Higher the chance of some keys being stolen by an adversary.
     - Computer systems are often infected by viruses, worms, and other forms of malicious software.

**Problem**

*Solutions that are based on private-key cryptography are not sufficient to deal with the problem of secure communication in open systems where parties cannot physically meet, or where parties have transient interactions.*

**Problem**

*Solutions that are based on private-key cryptography are not sufficient to deal with the problem of secure communication in open systems where parties cannot physically meet, or where parties have transient interactions.*

1. Key distribution,

**Problem**

*Solutions that are based on private-key cryptography are not sufficient to deal with the problem of secure communication in open systems where parties cannot physically meet, or where parties have transient interactions.*

1. Key distribution,
2. Managing so many secret keys,

**Problem**

*Solutions that are based on private-key cryptography are not sufficient to deal with the problem of secure communication in open systems where parties cannot physically meet, or where parties have transient interactions.*

1. Key distribution,
2. Managing so many secret keys, and
3. Inapplicability of private-key cryptography in open systems

# Key distribution centers: A Partial Solution

## Key distribution centers

1. Requires a trusted third party, like IT manager.

# Key distribution centers: A Partial Solution

## Key distribution centers

1. Requires a trusted third party, like IT manager.
2. It set up a server called Key distribution center (KDC).

# Key distribution centers: A Partial Solution

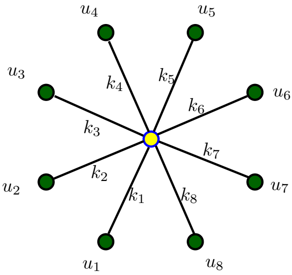
## Key distribution centers

1. Requires a trusted third party, like IT manager.
2. It set up a server called Key distribution center (KDC).
3. KDC acts as an intermediary between parties wish to communicate.

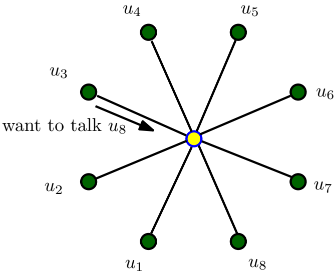
# Key distribution centers: A Partial Solution

## Key distribution centers

1. Requires a trusted third party, like IT manager.
2. It set up a server called Key distribution center (KDC).
3. KDC acts as an intermediary between parties wish to communicate.



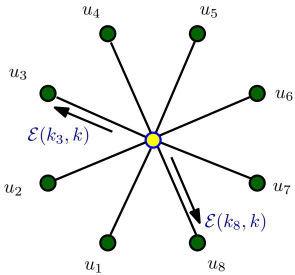
# Key distribution centers: A Partial Solution




## Key distribution centers

1. $u_3$ sends a request to KDC: want to talk to $u_8$

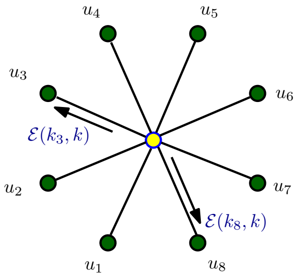
# Key distribution centers: A Partial Solution




## Key distribution centers

1. KDC computes a session key $k$.

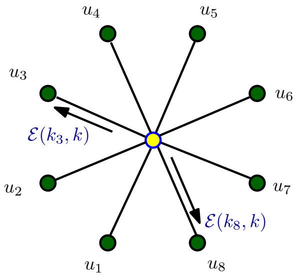
# Key distribution centers: A Partial Solution




## Key distribution centers

1. KDC computes a session key $k$.
2. Send $\mathcal{E}(k_3, k)$ to $u_3$.

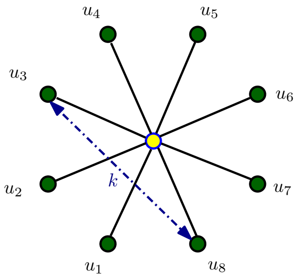
# Key distribution centers: A Partial Solution




## Key distribution centers

1. KDC computes a session key $k$.
2. Send $\mathcal{E}(k_3, k)$ to $u_3$.
3. Send $\mathcal{E}(k_8, k)$ to $u_8$.

# Key distribution centers: A Partial Solution




## Key distribution centers

1. $u_3$ and $u_8$ will use $k$ as shared key.
2. Must delete the key $k$ after use.

# Key distribution centers: A Partial Solution

## Key distribution centers

- Advantages:
    - Each employee needs to store only one secret key.

# Key distribution centers: A Partial Solution

## Key distribution centers

- Advantages:
  - Each employee needs to store only one secret key.
  - When an employee joins the organization, it only needs to sets up a key with the KDC.

# Key distribution centers: A Partial Solution

## Key distribution centers

- Advantages:
  - Each employee needs to store only one secret key.
  - When an employee joins the organization, it only needs to sets up a key with the KDC.
  - When an employee leaves the organization, KDC deletes the key.

# Key distribution centers: A Partial Solution

## Key distribution centers

- Advantages:
  - Each employee needs to store only one secret key.
  - When an employee joins the organization, it only needs to sets up a key with the KDC.
  - When an employee leaves the organization, KDC deletes the key.
- Disadvantages:
  - A successful attack on the KDC will result in a complete break of the system for all parties.

# Key distribution centers: A Partial Solution

## Key distribution centers

- Advantages:
  - Each employee needs to store only one secret key.
  - When an employee joins the organization, it only needs to sets up a key with the KDC.
  - When an employee leaves the organization, KDC deletes the key.
- Disadvantages:
  - A successful attack on the KDC will result in a complete break of the system for all parties.
  - The KDC is a single point of failure.

# New Directions in Cryptography (1976): Whitfield Diffie and Martin Hellman

## First two paragraphs of the paper

We stand today on the brink of a revolution in cryptography. The development of cheap digital hardware has freed it from the design limitations of mechanical computing and brought the cost of high grade cryptographic devices down to where they can be used in such commercial applications as remote cash dispensers and computer terminals.

In turn, such applications create a need for new types of cryptographic systems which minimize the necessity of secure key distribution channels and supply the equivalent of a written signature. At the same time, theoretical developments in information theory and computer science show promise of providing provably secure cryptosystems, changing this ancient art into a science.

# New Directions in Cryptography (1976): Whitfield Diffie and Martin Hellman

**First two paragraphs of the paper**

We stand today on the brink of a revolution in cryptography. The development of cheap digital hardware has freed it from the design limitations of mechanical computing and brought the cost of high grade cryptographic devices down to where they can be used in such commercial applications as remote cash dispensers and computer terminals.

In turn, such applications create a need for new types of cryptographic systems which minimize the necessity of secure key distribution channels and supply the equivalent of a written signature. At the same time, theoretical developments in information theory and computer science show promise of providing provably secure cryptosystems, changing this ancient art into a science.

- Introduces three distinct public-key primitives:
    - Interactive Key Exchange.
    - Public Key Encryption.
    - Digital Signatures.

## Key Exchange Protocol $P$

- A key exchange protocol $P$ is a pair of probabilistic machines $(A, B)$ that take turns in sending messages to each other.

# Key Exchange Protocol

## Key Exchange Protocol $P$

- A key exchange protocol $P$ is a pair of probabilistic machines $(A, B)$ that take turns in sending messages to each other.
- At the end of the protocol, when both machines terminate, they both obtain the same value $k$.

# Key Exchange Protocol

## Key Exchange Protocol $P$

- A key exchange protocol $P$ is a pair of probabilistic machines $(A, B)$ that take turns in sending messages to each other.
- At the end of the protocol, when both machines terminate, they both obtain the same value $k$.
- A protocol transcript $T_P$ is the sequence of messages exchanged between the parties in one execution of the protocol.

## Key Exchange Protocol $P$

- A key exchange protocol $P$ is a pair of probabilistic machines $(A, B)$ that take turns in sending messages to each other.
- At the end of the protocol, when both machines terminate, they both obtain the same value $k$.
- A protocol transcript $T_P$ is the sequence of messages exchanged between the parties in one execution of the protocol.
- Since $A$ and $B$ are probabilistic machines, we obtain a different transcript every time we run the protocol.

# Key Exchange Protocol

## Key Exchange Protocol $P$

- A key exchange protocol $P$ is a pair of probabilistic machines $(A, B)$ that take turns in sending messages to each other.
- At the end of the protocol, when both machines terminate, they both obtain the same value $k$.
- A protocol transcript $T_P$ is the sequence of messages exchanged between the parties in one execution of the protocol.
- Since $A$ and $B$ are probabilistic machines, we obtain a different transcript every time we run the protocol.
- Formally, the transcript $T_P$ of protocol $P$ is a random variable, which is a function of the random bits generated by $A$ and $B$.

# Key Exchange Protocol (Computational Version)

## Attack Game

- For a key exchange protocol $P = (A, B)$ and a given adversary $\mathcal{A}$, the attack game runs as follows.

# Key Exchange Protocol (Computational Version)

## Attack Game

- For a key exchange protocol $P = (A, B)$ and a given adversary $\mathcal{A}$, the attack game runs as follows.
  - The challenger runs the protocol between $A$ and $B$ to generate a shared key $k$ and transcript $T_P$.

### Attack Game

- For a key exchange protocol $P = (A, B)$ and a given adversary $\mathcal{A}$, the attack game runs as follows.
    - The challenger runs the protocol between $A$ and $B$ to generate a shared key $k$ and transcript $T_P$.
    - Then the challenger gives $T_P$ to $\mathcal{A}$.

## Attack Game

- For a key exchange protocol $P = (A, B)$ and a given adversary $\mathcal{A}$, the attack game runs as follows.
  - The challenger runs the protocol between $A$ and $B$ to generate a shared key $k$ and transcript $T_P$.
  - Then the challenger gives $T_P$ to $\mathcal{A}$.
  - $\mathcal{A}$ outputs a $\hat{k}$.

# Key Exchange Protocol (Computational Version)

## Attack Game

- For a key exchange protocol $P = (A, B)$ and a given adversary $\mathcal{A}$, the attack game runs as follows.
  - The challenger runs the protocol between $A$ and $B$ to generate a shared key $k$ and transcript $T_P$.
  - Then the challenger gives $T_P$ to $\mathcal{A}$.
  - $\mathcal{A}$ outputs a $\hat{k}$.

## Advantage

Let $W$ be the event that $\mathcal{A}$ outputs $\hat{k} = k$ in the Experiment. We define advantage of $\mathcal{A}$ in the attack game with respect to $P$ as

$$\mathsf{KEadv}_c[\mathcal{A}, P] = \Pr[W].$$

# Key Exchange Protocol (Indistinguishability Version)

## Attack Game

- This is much stronger notion than the computational version.

## Attack Game

- This is much stronger notion than the computational version.
- For a key exchange protocol $P = (A, B)$ and a given adversary $\mathcal{A}$, the attack game runs as follows.

# Key Exchange Protocol (Indistinguishability Version)

## Attack Game

- This is much stronger notion than the computational version.
- For a key exchange protocol $P = (A, B)$ and a given adversary $\mathcal{A}$, the attack game runs as follows.
  - The challenger runs the protocol between $A$ and $B$ to generate a shared key $k$ and transcript $T_P$.

# Key Exchange Protocol (Indistinguishability Version)

## Attack Game

- This is much stronger notion than the computational version.
- For a key exchange protocol $P = (A, B)$ and a given adversary $\mathcal{A}$, the attack game runs as follows.
  - The challenger runs the protocol between $A$ and $B$ to generate a shared key $k$ and transcript $T_P$.
  - The challenger chooses a random bit $b \xleftarrow{R} \{0, 1\}$.

### Attack Game

- This is much stronger notion than the computational version.
- For a key exchange protocol $P = (A, B)$ and a given adversary $\mathcal{A}$, the attack game runs as follows.
  - The challenger runs the protocol between $A$ and $B$ to generate a shared key $k$ and transcript $T_P$.
  - The challenger chooses a random bit $b \xleftarrow{R} \{0,1\}$.
    - If $b = 0$, $\hat{k} \longleftarrow \{0,1\}^n$, else $\hat{k} \longleftarrow k$.

# Key Exchange Protocol (Indistinguishability Version)

## Attack Game

- This is much stronger notion than the computational version.
- For a key exchange protocol $P = (A, B)$ and a given adversary $\mathcal{A}$, the attack game runs as follows.
  - The challenger runs the protocol between $A$ and $B$ to generate a shared key $k$ and transcript $T_P$.
  - The challenger chooses a random bit $b \xleftarrow{R} \{0, 1\}$.
    - If $b = 0$, $\hat{k} \longleftarrow \{0, 1\}^n$, else $\hat{k} \longleftarrow k$.
  - Then the challenger gives $(T_P, \hat{k})$ to $\mathcal{A}$.

# Key Exchange Protocol (Indistinguishability Version)

## Attack Game

- This is much stronger notion than the computational version.
- For a key exchange protocol $P = (A, B)$ and a given adversary $\mathcal{A}$, the attack game runs as follows.
  - The challenger runs the protocol between $A$ and $B$ to generate a shared key $k$ and transcript $T_P$.
  - The challenger chooses a random bit $b \xleftarrow{R} \{0,1\}$.
    - If $b = 0$, $\hat{k} \longleftarrow \{0,1\}^n$, else $\hat{k} \longleftarrow k$.
  - Then the challenger gives $(T_P, \hat{k})$ to $\mathcal{A}$.
  - $\mathcal{A}$ outputs a $\hat{b} \in \{0,1\}$.

# Key Exchange Protocol (Indistinguishability Version)

## Attack Game

- This is much stronger notion than the computational version.
- For a key exchange protocol $P = (A, B)$ and a given adversary $\mathcal{A}$, the attack game runs as follows.
  - The challenger runs the protocol between $A$ and $B$ to generate a shared key $k$ and transcript $T_P$.
  - The challenger chooses a random bit $b \xleftarrow{R} \{0,1\}$.
    - If $b = 0$, $\hat{k} \longleftarrow \{0,1\}^n$, else $\hat{k} \longleftarrow k$.
  - Then the challenger gives $(T_P, \hat{k})$ to $\mathcal{A}$.
  - $\mathcal{A}$ outputs a $\hat{b} \in \{0,1\}$.

## Advantage

Let $W$ be the event that $\mathcal{A}$ outputs $\hat{b} = b$ in the Experiment. We define advantage of $\mathcal{A}$ in the attack game with respect to $P$ as

$$\mathsf{KEadv}_i[\mathcal{A}, P] = \left| \Pr[W] - \frac{1}{2} \right|.$$

# Diffie-Hellman (DH) Key-Exchange Algorithm (DHKE)

- In public-key cryptography, each user has a pair of keys: $(d, e)$, where
  - $d$ : secret key of the user,
  - $e$ : public key of the user, and

- In public-key cryptography, each user has a pair of keys: $(d, e)$, where
  - $d$ : secret key of the user,
  - $e$ : public key of the user, and
  - it must be computationally hard to guess $d$ from $e$.

- Let $G$ be a finite abelian multiplicative group of order $n$,

- Let $G$ be a finite abelian multiplicative group of order $n$,
- Identity of $G$ as 1,

- Let $G$ be a finite abelian multiplicative group of order $n$,
- Identity of $G$ as 1,
- Let $g \in G$ having suitably large prime multiplicative order $m$,

# Diffie-Hellman (DH) Key-Exchange Algorithm (DHKE)

- Let $G$ be a finite abelian multiplicative group of order $n$,
- Identity of $G$ as 1,
- Let $g \in G$ having suitably large prime multiplicative order $m$,
- We work with the subgroup $H = \langle g \rangle$,

- Let $G$ be a finite abelian multiplicative group of order $n$,
- Identity of $G$ as 1,
- Let $g \in G$ having suitably large prime multiplicative order $m$,
- We work with the subgroup $H = \langle g \rangle$, and
- $\{G, g, n, m\}$ are public elements.

# Diffie-Hellman (DH) Key-Exchange Algorithm (DHKE)

| **Alice** | **Bob** |
| --- | --- |
| Generates a random $d_A \in \{2, \ldots, m-1\}$ | |
| Computes $e_A = g^{d_A}$ | |

# Diffie-Hellman (DH) Key-Exchange Algorithm (DHKE)

| **Alice** | **Bob** |
|---|---|
| Generates a random $d_A \in \{2, \ldots, m-1\}$ | Generates a random $d_B \in \{2, \ldots, m-1\}$ |
| Computes $e_A = g^{d_A}$ | Computes $e_B = g^{d_B}$ |

# Diffie-Hellman (DH) Key-Exchange Algorithm (DHKE)

| **Alice** | **Bob** |
|---|---|
| Generates a random $d_A \in \{2, \ldots, m-1\}$ | Generates a random $d_B \in \{2, \ldots, m-1\}$ |
| Computes $e_A = g^{d_A}$ | Computes $e_B = g^{d_B}$ |

$$\xrightarrow{\quad e_A \quad}$$

$$\xleftarrow{\quad e_B \quad}$$

# Diffie-Hellman (DH) Key-Exchange Algorithm (DHKE)

| **Alice** | **Bob** |
|---|---|
| Generates a random $d_A \in \{2, \ldots, m-1\}$ | Generates a random $d_B \in \{2, \ldots, m-1\}$ |
| Computes $e_A = g^{d_A}$ | Computes $e_B = g^{d_B}$ |

$$\xrightarrow{\quad e_A \quad}$$

$$\xleftarrow{\quad e_B \quad}$$

| | |
|---|---|
| Computes $k = (e_B)^{d_A}$ | Computes $k = (e_A)^{d_B}$ |

**Correctness**

$$(e_B)^{d_A} = \left(g^{d_B}\right)^{d_A} = (g)^{d_A d_B} = \left(g^{d_A}\right)^{d_B} = (e_A)^{d_B}.$$

# Diffie-Hellman (DH) Key-Exchange Algorithm (DHKE)

## Correctness

$$(e_B)^{d_A} = \left(g^{d_B}\right)^{d_A} = (g)^{d_A d_B} = \left(g^{d_A}\right)^{d_B} = (e_A)^{d_B}\ .$$

## Failure

If $k = 1$, then we have a Failure.

### Discrete Logarithm Problem (DLP)

Let $G$ be a finite cyclic group of order $n$ and $g$ be a generator. Given $(G, g, n)$ and an element $u = g^x \in G$ for some unknown integer $x$, compute $x$.

### Discrete Logarithm Problem (DLP)

Let $G$ be a finite cyclic group of order $n$ and $g$ be a generator. Given $(G, g, n)$ and an element $u = g^x \in G$ for some unknown integer $x$, compute $x$.

### DLP Attack Game

Let $G$ be a cyclic group of order $n$ generated by $g \in G$. For a given adversary $\mathcal{A}$, define the following attack game:

## Discrete Logarithm Problem (DLP)

Let $G$ be a finite cyclic group of order $n$ and $g$ be a generator. Given $(G, g, n)$ and an element $u = g^x \in G$ for some unknown integer $x$, compute $x$.

## DLP Attack Game

Let $G$ be a cyclic group of order $n$ generated by $g \in G$. For a given adversary $\mathcal{A}$, define the following attack game:

- The challenger and the adversary $\mathcal{A}$ take a description of $G$ as input. The description includes the order $n$ and a generator $g \in G$.

# Discrete Logarithm Problem (DLP)

## Discrete Logarithm Problem (DLP)

Let $G$ be a finite cyclic group of order $n$ and $g$ be a generator. Given $(G, g, n)$ and an element $u = g^x \in G$ for some unknown integer $x$, compute $x$.

## DLP Attack Game

Let $G$ be a cyclic group of order $n$ generated by $g \in G$. For a given adversary $\mathcal{A}$, define the following attack game:

- The challenger and the adversary $\mathcal{A}$ take a description of $G$ as input. The description includes the order $n$ and a generator $g \in G$.
- The challenger computes $x \xleftarrow{R} \mathbb{Z}_n$, $u \longleftarrow g^x$, and sends $u \in G$ to $\mathcal{A}$.

# Discrete Logarithm Problem (DLP)

## Discrete Logarithm Problem (DLP)

Let $G$ be a finite cyclic group of order $n$ and $g$ be a generator. Given $(G, g, n)$ and an element $u = g^x \in G$ for some unknown integer $x$, compute $x$.

## DLP Attack Game

Let $G$ be a cyclic group of order $n$ generated by $g \in G$. For a given adversary $\mathcal{A}$, define the following attack game:

- The challenger and the adversary $\mathcal{A}$ take a description of $G$ as input. The description includes the order $n$ and a generator $g \in G$.
- The challenger computes $x \xleftarrow{R} \mathbb{Z}_n$, $u \longleftarrow g^x$, and sends $u \in G$ to $\mathcal{A}$.
- The adversary outputs some $\hat{x} \longleftarrow \mathbb{Z}_n$.

# Discrete Logarithm Problem (DLP)

## Discrete Logarithm Problem (DLP)

Let $G$ be a finite cyclic group of order $n$ and $g$ be a generator. Given $(G, g, n)$ and an element $u = g^x \in G$ for some unknown integer $x$, compute $x$.

## DLP Attack Game

Let $G$ be a cyclic group of order $n$ generated by $g \in G$. For a given adversary $\mathcal{A}$, define the following attack game:

- The challenger and the adversary $\mathcal{A}$ take a description of $G$ as input. The description includes the order $n$ and a generator $g \in G$.
- The challenger computes $x \xleftarrow{R} \mathbb{Z}_n$, $u \longleftarrow g^x$, and sends $u \in G$ to $\mathcal{A}$.
- The adversary outputs some $\hat{x} \longleftarrow \mathbb{Z}_n$.

We define $\mathcal{A}$'s advantage in solving the discrete logarithm problem for $G$, denoted $\mathsf{DLadv}[\mathcal{A}, G]$, as the probability that $x = \hat{x}$.

# Discrete Logarithm Problem (DLP)

## Discrete Logarithm Problem (DLP)

Let $G$ be a finite cyclic group of order $n$ and $g$ be a generator. Given $(G, g, n)$ and an element $u = g^x \in G$ for some unknown integer $x$, compute $x$.

## DLP Attack Game

Let $G$ be a cyclic group of order $n$ generated by $g \in G$. For a given adversary $\mathcal{A}$, define the following attack game:

- The challenger and the adversary $\mathcal{A}$ take a description of $G$ as input. The description includes the order $n$ and a generator $g \in G$.
- The challenger computes $x \xleftarrow{R} \mathbb{Z}_n$, $u \longleftarrow g^x$, and sends $u \in G$ to $\mathcal{A}$.
- The adversary outputs some $\hat{x} \longleftarrow \mathbb{Z}_n$.

We define $\mathcal{A}$'s advantage in solving the discrete logarithm problem for $G$, denoted $\mathsf{DLadv}[\mathcal{A}, G]$, as the probability that $x = \hat{x}$.

## Discrete logarithm assumption

We say that the discrete logarithm (DL) assumption holds for $G$ if for all efficient adversaries $\mathcal{A}$ the quantity $\mathsf{DLadv}[\mathcal{A}, G]$ is negligible.

# Discrete Logarithm Problem (DLP)

## Some Properties of Discrete Logarithm

- $u = g^x \Leftrightarrow x = \log_g(u)$.

# Discrete Logarithm Problem (DLP)

## Some Properties of Discrete Logarithm

- $u = g^x \Leftrightarrow x = \log_g(u)$.
- Let $u_1 = g^{x_1}$ and $u_2 = g^{x_2}$,

$$u_1 \cdot u_2 = g^{x_1+x_2} \Longrightarrow \log_g(u_1 \cdot u_2) = x_1 + x_2 = \log_g(u_1) + \log_g(u_2).$$

# Discrete Logarithm Problem (DLP)

## Some Properties of Discrete Logarithm

- $u = g^x \Leftrightarrow x = \log_g(u)$.
- Let $u_1 = g^{x_1}$ and $u_2 = g^{x_2}$,

$$u_1 \cdot u_2 = g^{x_1+x_2} \Longrightarrow \log_g(u_1 \cdot u_2) = x_1 + x_2 = \log_g(u_1) + \log_g(u_2).$$

- Discrete in the sense that $x$ takes discrete values from $\{0, 1, \ldots, n\}$.

# Discrete Logarithm Problem (DLP)

## Some Properties of Discrete Logarithm

- $u = g^x \Leftrightarrow x = \log_g(u)$.
- Let $u_1 = g^{x_1}$ and $u_2 = g^{x_2}$,

$$u_1 \cdot u_2 = g^{x_1+x_2} \Longrightarrow \log_g(u_1 \cdot u_2) = x_1 + x_2 = \log_g(u_1) + \log_g(u_2).$$

- Discrete in the sense that $x$ takes discrete values from $\{0, 1, \ldots, n\}$.

## DLP and Diffie-Hellman Key Exchange Protocol

- If DLP is easy to solve in a group $G$, then
  - adversary can compute $d_A$ (or $d_B$) from $(G, g, n, e_A)$ (or $(G, g, n, e_B)$).

# Discrete Logarithm Problem (DLP)

## Some Properties of Discrete Logarithm

- $u = g^x \Leftrightarrow x = \log_g(u)$.
- Let $u_1 = g^{x_1}$ and $u_2 = g^{x_2}$,

$$u_1 \cdot u_2 = g^{x_1+x_2} \Rightarrow \log_g(u_1 \cdot u_2) = x_1 + x_2 = \log_g(u_1) + \log_g(u_2).$$

- Discrete in the sense that $x$ takes discrete values from $\{0, 1, \ldots, n\}$.

## DLP and Diffie-Hellman Key Exchange Protocol

- If DLP is easy to solve in a group $G$, then
  - adversary can compute $d_A$ (or $d_B$) from $(G, g, n, e_A)$ (or $(G, g, n, e_B)$).
  - Can compute $k = (e_B)^{d_A}$ (or $k = (e_A)^{d_B}$).

# Discrete Logarithm Problem (DLP)

## Some Properties of Discrete Logarithm

- $u = g^x \Leftrightarrow x = \log_g(u)$.
- Let $u_1 = g^{x_1}$ and $u_2 = g^{x_2}$,

$$u_1 \cdot u_2 = g^{x_1+x_2} \Rightarrow \log_g(u_1 \cdot u_2) = x_1 + x_2 = \log_g(u_1) + \log_g(u_2).$$

- Discrete in the sense that $x$ takes discrete values from $\{0, 1, \ldots, n\}$.

## DLP and Diffie-Hellman Key Exchange Protocol

- If DLP is easy to solve in a group $G$, then
  - adversary can compute $d_A$ (or $d_B$) from $(G, g, n, e_A)$ (or $(G, g, n, e_B)$).
  - Can compute $k = (e_B)^{d_A}$ (or $k = (e_A)^{d_B}$).
- Hardness of DLP is the necessary condition for DH key Exchange, but not sufficient.

# Computational Diffie-Hellman problem (CDHP)

**Computational Diffie-Hellman Problem (CDHP)**

Let $G$ be a finite cyclic group of order $n$ and $g$ be a generator. Given $(G, g, n)$, and $g^x$ and $g^y$ for some (unknown) integers $x$ and $y$, compute $g^{xy}$.

# Computational Diffie-Hellman problem (CDHP)

### Computational Diffie-Hellman Problem (CDHP)

Let $G$ be a finite cyclic group of order $n$ and $g$ be a generator. Given $(G, g, n)$, and $g^x$ and $g^y$ for some (unknown) integers $x$ and $y$, compute $g^{xy}$.

### Computational DHP Attack Game

Let $G$ be a cyclic group of order $n$ generated by $g \in G$. For a given adversary $\mathcal{A}$, define the following attack game:

# Computational Diffie-Hellman problem (CDHP)

**Computational Diffie-Hellman Problem (CDHP)**

Let $G$ be a finite cyclic group of order $n$ and $g$ be a generator. Given $(G, g, n)$, and $g^x$ and $g^y$ for some (unknown) integers $x$ and $y$, compute $g^{xy}$.

**Computational DHP Attack Game**

Let $G$ be a cyclic group of order $n$ generated by $g \in G$. For a given adversary $\mathcal{A}$, define the following attack game:

- The challenger and the adversary $\mathcal{A}$ take a description of $G$ as input.

# Computational Diffie-Hellman problem (CDHP)

## Computational Diffie-Hellman Problem (CDHP)

Let $G$ be a finite cyclic group of order $n$ and $g$ be a generator. Given $(G, g, n)$, and $g^x$ and $g^y$ for some (unknown) integers $x$ and $y$, compute $g^{xy}$.

## Computational DHP Attack Game

Let $G$ be a cyclic group of order $n$ generated by $g \in G$. For a given adversary $\mathcal{A}$, define the following attack game:

- The challenger and the adversary $\mathcal{A}$ take a description of $G$ as input.
- The challenger computes $x, y \xleftarrow{R} \mathbb{Z}_n$, $u \longleftarrow g^x$, $v \longleftarrow g^y$, $w \longleftarrow g^{xy}$, and sends $(u, v)$ to $\mathcal{A}$.

# Computational Diffie-Hellman problem (CDHP)

## Computational Diffie-Hellman Problem (CDHP)

Let $G$ be a finite cyclic group of order $n$ and $g$ be a generator. Given $(G, g, n)$, and $g^x$ and $g^y$ for some (unknown) integers $x$ and $y$, compute $g^{xy}$.

## Computational DHP Attack Game

Let $G$ be a cyclic group of order $n$ generated by $g \in G$. For a given adversary $\mathcal{A}$, define the following attack game:

- The challenger and the adversary $\mathcal{A}$ take a description of $G$ as input.
- The challenger computes $x, y \xleftarrow{R} \mathbb{Z}_n$, $u \longleftarrow g^x$, $v \longleftarrow g^y$, $w \longleftarrow g^{xy}$, and sends $(u, v)$ to $\mathcal{A}$.
- The adversary outputs some $\hat{w} \longleftarrow G$.

# Computational Diffie-Hellman problem (CDHP)

## Computational Diffie-Hellman Problem (CDHP)

Let $G$ be a finite cyclic group of order $n$ and $g$ be a generator. Given $(G, g, n)$, and $g^x$ and $g^y$ for some (unknown) integers $x$ and $y$, compute $g^{xy}$.

## Computational DHP Attack Game

Let $G$ be a cyclic group of order $n$ generated by $g \in G$. For a given adversary $\mathcal{A}$, define the following attack game:

- The challenger and the adversary $\mathcal{A}$ take a description of $G$ as input.
- The challenger computes $x, y \xleftarrow{R} \mathbb{Z}_n$, $u \longleftarrow g^x$, $v \longleftarrow g^y$, $w \longleftarrow g^{xy}$, and sends $(u, v)$ to $\mathcal{A}$.
- The adversary outputs some $\hat{w} \longleftarrow G$.

We define $\mathcal{A}$'s advantage in solving the computational Diffie-Hellman problem for $G$, denoted $\mathsf{CDHadv}[\mathcal{A}, G]$, as the probability that $w = \hat{w}$.

# Computational Diffie-Hellman problem (CDHP)

**Computational Diffie-Hellman assumption**

We say that the computational Diffie-Hellman (CDH) assumption holds for $G$ if for all efficient adversaries $\mathcal{A}$ the quantity $\mathsf{CDHadv}[\mathcal{A}, G]$ is negligible.

# Computational Diffie-Hellman problem (CDHP)

**Computational Diffie-Hellman assumption**

We say that the computational Diffie-Hellman (CDH) assumption holds for $G$ if for all efficient adversaries $\mathcal{A}$ the quantity $\mathsf{CDHadv}[\mathcal{A}, G]$ is negligible.

**DHKE and CDHP**

$\mathsf{KEadv}_c[\mathcal{A}, \mathrm{DHKE}] = \mathsf{CDHadv}[\mathcal{A}, G]$.

**Decisional Diffie-Hellman Problem (DDHP)**

Let $G$ be a finite cyclic group of order $n$ and $g$ be a generator. Distinguish between $(G, g, n, g^x, g^y, g^{xy})$ and $(G, g, n, g^x, g^y, g^z)$ for some (unknown) integers $x$, $y$ and $z$.

# Decisional Diffie-Hellman problem (DDHP)

### Decisional Diffie-Hellman Problem (DDHP)

Let $G$ be a finite cyclic group of order $n$ and $g$ be a generator. Distinguish between $(G, g, n, g^x, g^y, g^{xy})$ and $(G, g, n, g^x, g^y, g^z)$ for some (unknown) integers $x$, $y$ and $z$.

### Decisional DHP Attack Game

Let $G$ be a cyclic group of order $n$ generated by $g \in G$. For a given adversary $\mathcal{A}$, for $b \in \{0, 1\}$ define the Experiment $b$ of attack game as:

# Decisional Diffie-Hellman problem (DDHP)

## Decisional Diffie-Hellman Problem (DDHP)

Let $G$ be a finite cyclic group of order $n$ and $g$ be a generator. Distinguish between $(G, g, n, g^x, g^y, g^{xy})$ and $(G, g, n, g^x, g^y, g^z)$ for some (unknown) integers $x$, $y$ and $z$.

## Decisional DHP Attack Game

Let $G$ be a cyclic group of order $n$ generated by $g \in G$. For a given adversary $\mathcal{A}$, for $b \in \{0, 1\}$ define the Experiment $b$ of attack game as:

- The challenger and the adversary $\mathcal{A}$ take a description of $G$ as input.

# Decisional Diffie-Hellman problem (DDHP)

## Decisional Diffie-Hellman Problem (DDHP)

Let $G$ be a finite cyclic group of order $n$ and $g$ be a generator. Distinguish between $(G, g, n, g^x, g^y, g^{xy})$ and $(G, g, n, g^x, g^y, g^z)$ for some (unknown) integers $x$, $y$ and $z$.

## Decisional DHP Attack Game

Let $G$ be a cyclic group of order $n$ generated by $g \in G$. For a given adversary $\mathcal{A}$, for $b \in \{0, 1\}$ define the Experiment $b$ of attack game as:

- The challenger and the adversary $\mathcal{A}$ take a description of $G$ as input.
- The challenger computes $x, y, z \xleftarrow{R} \mathbb{Z}_n$, $u \longleftarrow g^x$, $v \longleftarrow g^y$, $w_0 \longleftarrow g^{xy}$, $w_1 \longleftarrow g^z$, and sends $(u, v, w_b)$ to $\mathcal{A}$.

# Decisional Diffie-Hellman problem (DDHP)

## Decisional Diffie-Hellman Problem (DDHP)

Let $G$ be a finite cyclic group of order $n$ and $g$ be a generator. Distinguish between $(G, g, n, g^x, g^y, g^{xy})$ and $(G, g, n, g^x, g^y, g^z)$ for some (unknown) integers $x$, $y$ and $z$.

## Decisional DHP Attack Game

Let $G$ be a cyclic group of order $n$ generated by $g \in G$. For a given adversary $\mathcal{A}$, for $b \in \{0, 1\}$ define the Experiment $b$ of attack game as:

- The challenger and the adversary $\mathcal{A}$ take a description of $G$ as input.
- The challenger computes $x, y, z \xleftarrow{R} \mathbb{Z}_n$, $u \longleftarrow g^x$, $v \longleftarrow g^y$, $w_0 \longleftarrow g^{xy}$, $w_1 \longleftarrow g^z$, and sends $(u, v, w_b)$ to $\mathcal{A}$.
- The adversary outputs some $\hat{b} \in \{0, 1\}$.

# Decisional Diffie-Hellman problem (DDHP)

**Advantage**

For $b = 0, 1$, let $W_b$ be the event that $\mathcal{A}$ outputs 1 in Experiment $b$. We define the advantage of $\mathcal{A}$ in the DDH attack with respect to $G$ as

$$\mathsf{DDHadv}[\mathcal{A}, G] = |\Pr[W_0] - \Pr[W_1]|.$$

# Decisional Diffie-Hellman problem (DDHP)

## Advantage

For $b = 0, 1$, let $W_b$ be the event that $\mathcal{A}$ outputs 1 in Experiment $b$. We define the advantage of $\mathcal{A}$ in the DDH attack with respect to $G$ as

$$\mathsf{DDHadv}[\mathcal{A}, G] = |\Pr[W_0] - \Pr[W_1]|.$$

## Decisional Diffie-Hellman assumption

We say that the decisional Diffie-Hellman (DDH) assumption holds for $G$ if for all efficient adversaries $\mathcal{A}$ the quantity $\mathsf{DDHadv}[\mathcal{A}, G]$ is negligible.

# Decisional Diffie-Hellman problem (DDHP)

## DHKE and DDHP

$$\Pr[W] \quad = \quad \Pr[\hat{b} = 0 \wedge b = 0] + \Pr[\hat{b} = 1 \wedge b = 1]$$

# Decisional Diffie-Hellman problem (DDHP)

## DHKE and DDHP

$$
\begin{aligned}
\Pr[W] &= \Pr[\hat{b} = 0 \wedge b = 0] + \Pr[\hat{b} = 1 \wedge b = 1] \\
&= \Pr[\hat{b} = 0 \mid b = 0] \cdot \Pr[b = 0] + \Pr[\hat{b} = 1 \mid b = 1] \cdot \Pr[b = 1]
\end{aligned}
$$

# Decisional Diffie-Hellman problem (DDHP)

## DHKE and DDHP

$$
\begin{aligned}
\Pr[W] &= \Pr[\hat{b} = 0 \wedge b = 0] + \Pr[\hat{b} = 1 \wedge b = 1] \\
&= \Pr[\hat{b} = 0 \mid b = 0] \cdot \Pr[b = 0] + \Pr[\hat{b} = 1 \mid b = 1] \cdot \Pr[b = 1] \\
&= \frac{1}{2}\left(\Pr[\hat{b} = 0 \mid b = 0] + \Pr[\hat{b} = 1 \mid b = 1]\right)
\end{aligned}
$$

# Decisional Diffie-Hellman problem (DDHP)

## DHKE and DDHP

$$
\begin{aligned}
\Pr[W] &= \Pr[\hat{b} = 0 \wedge b = 0] + \Pr[\hat{b} = 1 \wedge b = 1] \\
&= \Pr[\hat{b} = 0 \mid b = 0] \cdot \Pr[b = 0] + \Pr[\hat{b} = 1 \mid b = 1] \cdot \Pr[b = 1] \\
&= \frac{1}{2}\left(\Pr[\hat{b} = 0 \mid b = 0] + \Pr[\hat{b} = 1 \mid b = 1]\right) \\
&= \frac{1}{2}\left(\Pr[\mathcal{A}(G, g, n, g^x, g^y, g^z) = 0] + \Pr[\mathcal{A}(G, g, n, g^x, g^y, g^{xy}) = 1]\right)
\end{aligned}
$$

# Decisional Diffie-Hellman problem (DDHP)

## DHKE and DDHP

$$
\begin{aligned}
\Pr[W] &= \Pr[\hat{b} = 0 \wedge b = 0] + \Pr[\hat{b} = 1 \wedge b = 1] \\
&= \Pr[\hat{b} = 0 \mid b = 0] \cdot \Pr[b = 0] + \Pr[\hat{b} = 1 \mid b = 1] \cdot \Pr[b = 1] \\
&= \frac{1}{2}\left(\Pr[\hat{b} = 0 \mid b = 0] + \Pr[\hat{b} = 1 \mid b = 1]\right) \\
&= \frac{1}{2}\left(\Pr[\mathcal{A}(G, g, n, g^x, g^y, g^z) = 0] + \Pr[\mathcal{A}(G, g, n, g^x, g^y, g^{xy}) = 1]\right) \\
&= \frac{1}{2}\left(1 - \Pr[\mathcal{A}(G, g, n, g^x, g^y, g^z) = 1] + \Pr[\mathcal{A}(G, g, n, g^x, g^y, g^{xy}) = 1]\right)
\end{aligned}
$$

# Decisional Diffie-Hellman problem (DDHP)

## DHKE and DDHP

$$
\begin{aligned}
\Pr[W] &= \Pr[\hat{b}=0 \wedge b=0] + \Pr[\hat{b}=1 \wedge b=1] \\
&= \Pr[\hat{b}=0 \mid b=0]\cdot\Pr[b=0] + \Pr[\hat{b}=1 \mid b=1]\cdot\Pr[b=1] \\
&= \frac{1}{2}\Big(\Pr[\hat{b}=0 \mid b=0] + \Pr[\hat{b}=1 \mid b=1]\Big) \\
&= \frac{1}{2}\left(\Pr[\mathcal{A}(G,g,n,g^x,g^y,g^z)=0] + \Pr[\mathcal{A}(G,g,n,g^x,g^y,g^{xy})=1]\right) \\
&= \frac{1}{2}\left(1-\Pr[\mathcal{A}(G,g,n,g^x,g^y,g^z)=1] + \Pr[\mathcal{A}(G,g,n,g^x,g^y,g^{xy})=1]\right) \\
&= \frac{1}{2}+\frac{1}{2}\left(\Pr[\mathcal{A}(G,g,n,g^x,g^y,g^{xy})=1] - \Pr[\mathcal{A}(G,g,n,g^x,g^y,g^z)=1]\right)
\end{aligned}
$$

# Decisional Diffie-Hellman problem (DDHP)

## DHKE and DDHP

$$
\begin{aligned}
\Pr[W] &= \Pr[\hat{b} = 0 \wedge b = 0] + \Pr[\hat{b} = 1 \wedge b = 1] \\
&= \Pr[\hat{b} = 0 \mid b = 0] \cdot \Pr[b = 0] + \Pr[\hat{b} = 1 \mid b = 1] \cdot \Pr[b = 1] \\
&= \frac{1}{2}\Big(\Pr[\hat{b} = 0 \mid b = 0] + \Pr[\hat{b} = 1 \mid b = 1]\Big) \\
&= \frac{1}{2}\big(\Pr[\mathcal{A}(G, g, n, g^x, g^y, g^z) = 0] + \Pr[\mathcal{A}(G, g, n, g^x, g^y, g^{xy}) = 1]\big) \\
&= \frac{1}{2}\big(1 - \Pr[\mathcal{A}(G, g, n, g^x, g^y, g^z) = 1] + \Pr[\mathcal{A}(G, g, n, g^x, g^y, g^{xy}) = 1]\big) \\
&= \frac{1}{2} + \frac{1}{2}\big(\Pr[\mathcal{A}(G, g, n, g^x, g^y, g^{xy}) = 1] - \Pr[\mathcal{A}(G, g, n, g^x, g^y, g^z) = 1]\big) \\
&\leqslant \frac{1}{2} + \frac{1}{2}\big|\Pr[\mathcal{A}(G, g, n, g^x, g^y, g^{xy}) = 1] - \Pr[\mathcal{A}(G, g, n, g^x, g^y, g^z) = 1]\big|
\end{aligned}
$$

# Decisional Diffie-Hellman problem (DDHP)

## DHKE and DDHP

$$
\begin{aligned}
\Pr[W] &= \Pr[\hat{b} = 0 \wedge b = 0] + \Pr[\hat{b} = 1 \wedge b = 1] \\
&= \Pr[\hat{b} = 0 \mid b = 0] \cdot \Pr[b = 0] + \Pr[\hat{b} = 1 \mid b = 1] \cdot \Pr[b = 1] \\
&= \frac{1}{2}\Big(\Pr[\hat{b} = 0 \mid b = 0] + \Pr[\hat{b} = 1 \mid b = 1]\Big) \\
&= \frac{1}{2}\left(\Pr[\mathcal{A}(G, g, n, g^x, g^y, g^z) = 0] + \Pr[\mathcal{A}(G, g, n, g^x, g^y, g^{xy}) = 1]\right) \\
&= \frac{1}{2}\left(1 - \Pr[\mathcal{A}(G, g, n, g^x, g^y, g^z) = 1] + \Pr[\mathcal{A}(G, g, n, g^x, g^y, g^{xy}) = 1]\right) \\
&= \frac{1}{2} + \frac{1}{2}\left(\Pr[\mathcal{A}(G, g, n, g^x, g^y, g^{xy}) = 1] - \Pr[\mathcal{A}(G, g, n, g^x, g^y, g^z) = 1]\right) \\
&\leqslant \frac{1}{2} + \frac{1}{2}\left|\Pr[\mathcal{A}(G, g, n, g^x, g^y, g^{xy}) = 1] - \Pr[\mathcal{A}(G, g, n, g^x, g^y, g^z) = 1]\right| \\
&= \frac{1}{2} + \frac{1}{2}\mathsf{DDHadv}[\mathcal{A}, G]
\end{aligned}
$$

# Decisional Diffie-Hellman problem (DDHP)

## DHKE and DDHP

$$
\begin{aligned}
\Pr[W] &= \Pr[\hat{b}=0 \wedge b=0] + \Pr[\hat{b}=1 \wedge b=1] \\
&= \Pr[\hat{b}=0 \mid b=0]\cdot\Pr[b=0] + \Pr[\hat{b}=1 \mid b=1]\cdot\Pr[b=1] \\
&= \frac{1}{2}\Big(\Pr[\hat{b}=0 \mid b=0] + \Pr[\hat{b}=1 \mid b=1]\Big) \\
&= \frac{1}{2}\left(\Pr[\mathcal{A}(G,g,n,g^x,g^y,g^z)=0] + \Pr[\mathcal{A}(G,g,n,g^x,g^y,g^{xy})=1]\right) \\
&= \frac{1}{2}\left(1-\Pr[\mathcal{A}(G,g,n,g^x,g^y,g^z)=1] + \Pr[\mathcal{A}(G,g,n,g^x,g^y,g^{xy})=1]\right) \\
&= \frac{1}{2} + \frac{1}{2}\left(\Pr[\mathcal{A}(G,g,n,g^x,g^y,g^{xy})=1] - \Pr[\mathcal{A}(G,g,n,g^x,g^y,g^z)=1]\right) \\
&\leqslant \frac{1}{2} + \frac{1}{2}\left|\Pr[\mathcal{A}(G,g,n,g^x,g^y,g^{xy})=1] - \Pr[\mathcal{A}(G,g,n,g^x,g^y,g^z)=1]\right| \\
&= \frac{1}{2} + \frac{1}{2}\mathsf{DDHadv}[\mathcal{A},G]
\end{aligned}
$$

$$\mathsf{KEadv}_i[\mathcal{A},\mathrm{DHKE}] = \left|\Pr[W]-\frac{1}{2}\right| \leqslant \mathsf{DDHadv}[\mathcal{A},G].$$

# Choice of Underlying Group for DHKE

## Prime-Order Groups

- Let the order of $G$ be a composite $n = n_1 \cdot n_2$ and known.

### Prime-Order Groups

- Let the order of $G$ be a composite $n = n_1 \cdot n_2$ and known.
- Pohlig-Hellman algorithm reduces an instance of the DLP in to two instances of the DLP in groups of order $n_1$ and $n_2$.

# Choice of Underlying Group for DHKE

## Prime-Order Groups

- Let the order of $G$ be a composite $n = n_1 \cdot n_2$ and known.
- Pohlig-Hellman algorithm reduces an instance of the DLP in to two instances of the DLP in groups of order $n_1$ and $n_2$.
  - This does not mean that the DLP can be solved in polynomial time in non-prime order groups.
  - Merely means that the problem becomes easier.

# Choice of Underlying Group for DHKE

## Prime-Order Groups

- Let the order of $G$ be a composite $n = n_1 \cdot n_2$ and known.
- Pohlig-Hellman algorithm reduces an instance of the DLP in to two instances of the DLP in groups of order $n_1$ and $n_2$.
  - This does not mean that the DLP can be solved in polynomial time in non-prime order groups.
  - Merely means that the problem becomes easier.
- Every element is a generator.

# Choice of Underlying Group for DHKE

## Prime-Order Groups

- Let the order of $G$ be a composite $n = n_1 \cdot n_2$ and known.
- Pohlig-Hellman algorithm reduces an instance of the DLP in to two instances of the DLP in groups of order $n_1$ and $n_2$.
  - This does not mean that the DLP can be solved in polynomial time in non-prime order groups.
  - Merely means that the problem becomes easier.
- Every element is a generator.
- Multiplicative inverse exists.

# Choice of Underlying Group for DHKE

## Prime-Order Groups

- Let the order of $G$ be a composite $n = n_1 \cdot n_2$ and known.
- Pohlig-Hellman algorithm reduces an instance of the DLP in to two instances of the DLP in groups of order $n_1$ and $n_2$.
  - This does not mean that the DLP can be solved in polynomial time in non-prime order groups.
  - Merely means that the problem becomes easier.
- Every element is a generator.
- Multiplicative inverse exists.
- A final reason for working with prime-order groups applies in situations when the DDHP should be hard.
  - We stress that using a group of prime order is neither necessary nor sufficient for the DDH problem to be hard.

# Choice of Underlying Group for DHKE

**(Subgroup of) $\mathbb{Z}_p^*$**

- Conjecture: DLP is hard in $\mathbb{Z}_p^*$.

# Choice of Underlying Group for DHKE

**(Subgroup of) $\mathbb{Z}_p^*$**

- Conjecture: DLP is hard in $\mathbb{Z}_p^*$.
- If $p \geqslant 3$, Order of $\mathbb{Z}_p^*$ is $(p-1)$, not a prime.

# Choice of Underlying Group for DHKE

## (Subgroup of) $\mathbb{Z}_p^*$

- Conjecture: DLP is hard in $\mathbb{Z}_p^*$.
- If $p \geqslant 3$, Order of $\mathbb{Z}_p^*$ is $(p-1)$, not a prime.
  - DDHP is simply not hard in such groups.

# Choice of Underlying Group for DHKE

**(Subgroup of) $\mathbb{Z}_p^*$**

- Conjecture: DLP is hard in $\mathbb{Z}_p^*$.
- If $p \geqslant 3$, Order of $\mathbb{Z}_p^*$ is $(p-1)$, not a prime.
  - DDHP is simply not hard in such groups.
  - All the quadratic residues of $\mathbb{Z}_p^*$ forms a subgroup $H$ of order $q = (p-1)/2$.

# Choice of Underlying Group for DHKE

## (Subgroup of) $\mathbb{Z}_p^*$

- Conjecture: DLP is hard in $\mathbb{Z}_p^*$.
- If $p \geqslant 3$, Order of $\mathbb{Z}_p^*$ is $(p-1)$, not a prime.
  - DDHP is simply not hard in such groups.
  - All the quadratic residues of $\mathbb{Z}_p^*$ forms a subgroup $H$ of order $q = (p-1)/2$.
  - It is desirable to have $q$ as prime and we work with $H$.

# Choice of Underlying Group for DHKE

## (Subgroup of) $\mathbb{Z}_p^*$

- Conjecture: DLP is hard in $\mathbb{Z}_p^*$.
- If $p \geqslant 3$, Order of $\mathbb{Z}_p^*$ is $(p-1)$, not a prime.
  - DDHP is simply not hard in such groups.
  - All the quadratic residues of $\mathbb{Z}_p^*$ forms a subgroup $H$ of order $q = (p-1)/2$.
  - It is desirable to have $q$ as prime and we work with $H$.
  - If not, then it is desirable to have a subgroup of $\mathbb{Z}_p^*$ of large prime order.

# Small Subgroup Attack

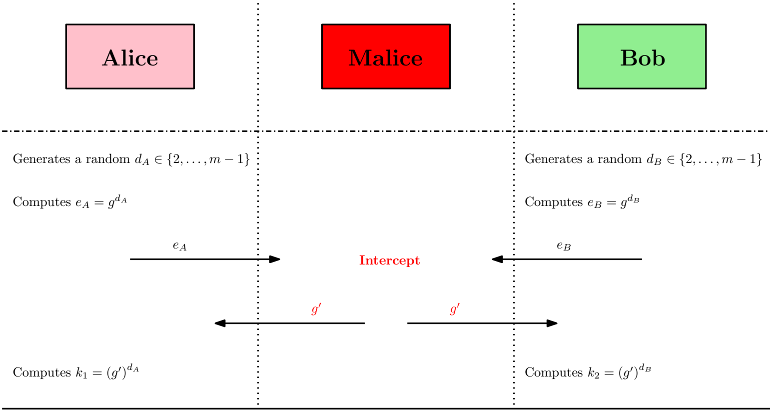
- Let $\{G, g, n, m\}$ are public elements,
- Let $h = \dfrac{n}{m}$ and
- $g' \in G$ has order $h$.

# Small Subgroup Attack

| Alice | Malice | Bob |
|---|---|---|
| Generates a random $d_A \in \{2, \ldots, m-1\}$ | | Generates a random $d_B \in \{2, \ldots, m-1\}$ |
| Computes $e_A = g^{d_A}$ | | Computes $e_B = g^{d_B}$ |

# Small Subgroup Attack

| Alice | Malice | Bob |
|---|---|---|
| Generates a random $d_A \in \{2, \ldots, m-1\}$ | | Generates a random $d_B \in \{2, \ldots, m-1\}$ |
| Computes $e_A = g^{d_A}$ | | Computes $e_B = g^{d_B}$ |
| $\xrightarrow{e_A}$ | **Intercept** | $\xleftarrow{e_B}$ |
| $\xleftarrow{g'}$ | | $\xrightarrow{g'}$ |


Alice
Malice
Bob
Generates a random $d_A \in \{2, \ldots, m-1\}$
Generates a random $d_B \in \{2, \ldots, m-1\}$
Computes $e_A = g^{d_A}$
Computes $e_B = g^{d_B}$
$e_A$
Intercept
$e_B$
$g'$
$g'$
Computes $k_1 = (g')^{d_A}$
Computes $k_2 = (g')^{d_B}$

# Small Subgroup Attack

**If $d_A \equiv d_B \equiv d \pmod{h}$**

- Let $d_A = q_A h + d$ and $d_B = q_B h + d$

$$(g')^{d_A} = (g')^{q_A h + d} = \left(g'^h\right)^{q_A} (g')^d = 1 \cdot (g')^d = (g')^d$$

$$(g')^{d_B} = (g')^{q_B h + d} = \left(g'^h\right)^{q_d} (g')^d = 1 \cdot (g')^d = (g')^d$$

- As $h$ is small, Malice can find $d$ by exhaustive search.

# Small Subgroup Attack

## If $d_A \equiv d_B \equiv d \pmod{h}$

- Let $d_A = q_A h + d$ and $d_B = q_B h + d$

$$(g')^{d_A} = (g')^{q_A h + d} = \left(g'^h\right)^{q_A} (g')^d = 1 \cdot (g')^d = (g')^d$$

$$(g')^{d_B} = (g')^{q_B h + d} = \left(g'^h\right)^{q_d} (g')^d = 1 \cdot (g')^d = (g')^d$$

- As $h$ is small, Malice can find $d$ by exhaustive search.

## If $d_A \not\equiv d_B \pmod{h}$

- Let $d_A = q_A h + d_1$ and $d_B = q_B h + d_2$

$$(g')^{d_A} = (g')^{q_A h + d_1} = \left(g'^h\right)^{q_A} (g')^{d_1} = 1 \cdot (g')^{d_1} = (g')^{d_1}$$

$$(g')^{d_B} = (g')^{q_B h + d_2} = \left(g'^h\right)^{q_B} (g')^{d_2} = 1 \cdot (g')^{d_2} = (g')^{d_2}$$

- As $h$ is small, Malice can find $d_1$ and $d_2$ by exhaustive search.

# Small Subgroup Attack resistant DH Protocol

| **Alice** | **Bob** |
| --- | --- |
| Generates a random $d_A \in \{2, \ldots, m-1\}$ | Generates a random $d_B \in \{2, \ldots, m-1\}$ |
| Computes $e_A = g^{d_A}$ | Computes $e_B = g^{d_B}$ |

# Small Subgroup Attack resistant DH Protocol

| Alice | Bob |
|---|---|
| Generates a random $d_A \in \{2, \ldots, m-1\}$ | Generates a random $d_B \in \{2, \ldots, m-1\}$ |
| Computes $e_A = g^{d_A}$ | Computes $e_B = g^{d_B}$ |

$$\text{Alice} \xrightarrow{e_A} \text{Bob}$$

$$\text{Alice} \xleftarrow{e_B} \text{Bob}$$

# Small Subgroup Attack resistant DH Protocol

| Alice | Bob |
| --- | --- |
| Generates a random $d_A \in \{2, \ldots, m-1\}$ | Generates a random $d_B \in \{2, \ldots, m-1\}$ |
| Computes $e_A = g^{d_A}$ | Computes $e_B = g^{d_B}$ |

$$\text{Alice} \xrightarrow{e_A} \text{Bob}$$

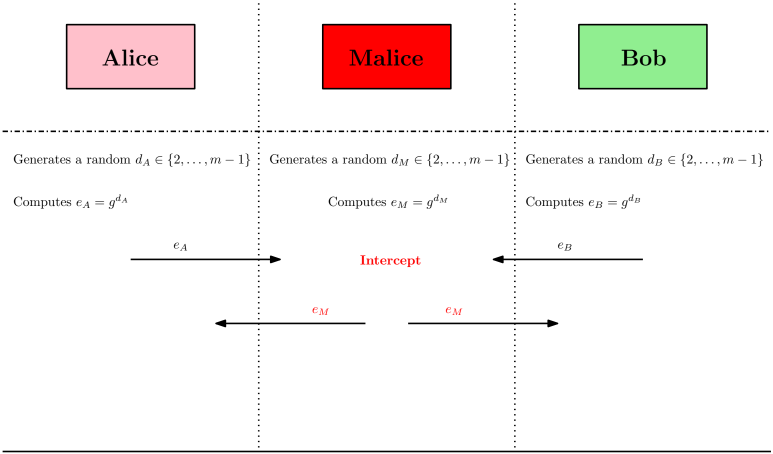
$$\text{Alice} \xleftarrow{e_B} \text{Bob}$$

| Alice | Bob |
| --- | --- |
| Computes $k = (e_B)^{hd_A}$ | Computes $k = (e_A)^{hd_B}$ |

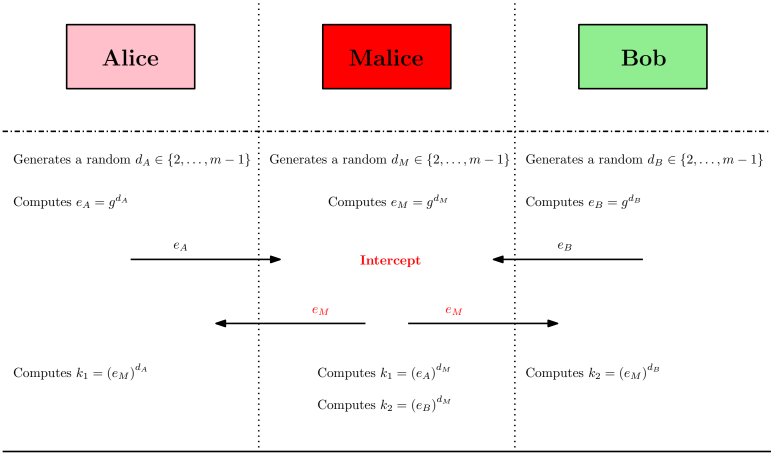
# Unknown Key Share

| Alice | Malice | Bob |
|---|---|---|
| Generates a random $d_A \in \{2, \ldots, m-1\}$ | Generates a random $d_M \in \{2, \ldots, m-1\}$ | Generates a random $d_B \in \{2, \ldots, m-1\}$ |
| Computes $e_A = g^{d_A}$ | Computes $e_M = g^{d_M}$ | Computes $e_B = g^{d_B}$ |
| $e_A \longrightarrow$ | | $\longleftarrow e_B$ |

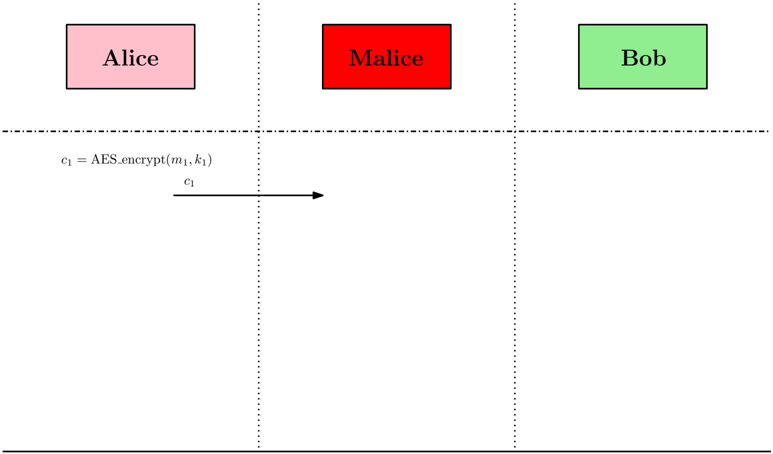
# Unknown Key Share



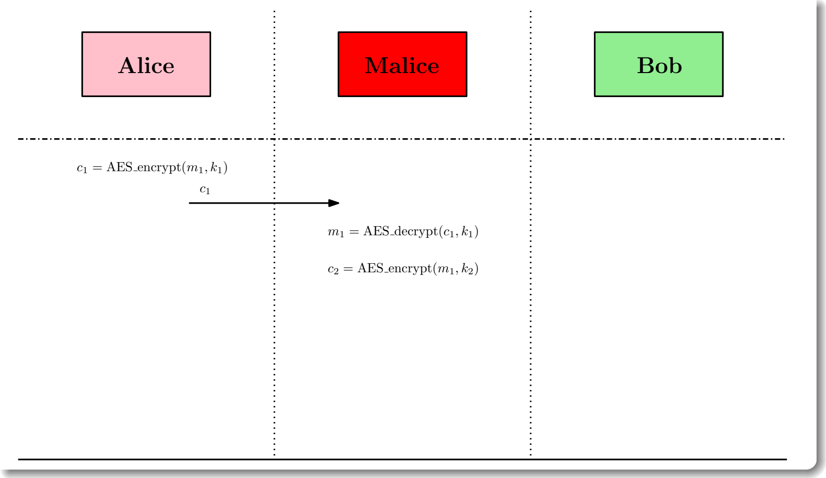
# Unknown Key Share



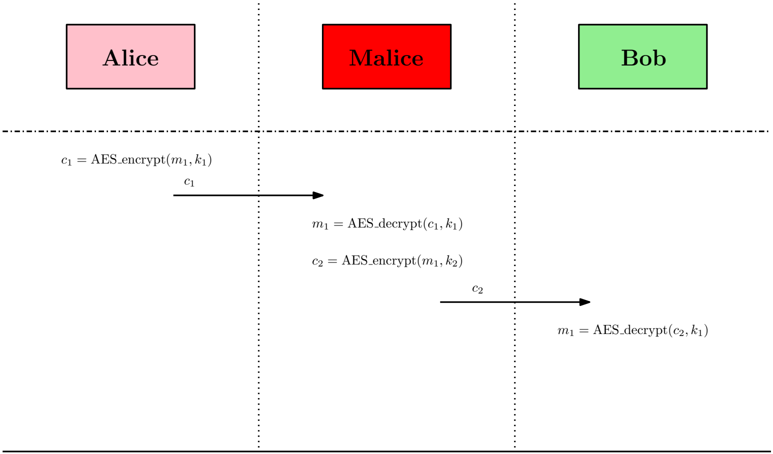
# Unknown Key Share



# Unknown Key Share



# Unknown Key Share



- Let $\{G, g, n, m, h\}$ are public elements,

# The Menezes-Qu-Vanstone (MQV) key-exchange protocol

- Let $\{G, g, n, m, h\}$ are public elements,
- Each entity uses two key pairs:

# The Menezes-Qu-Vanstone (MQV) key-exchange protocol

- Let $\{G, g, n, m, h\}$ are public elements,
- Each entity uses two key pairs:
  - $(D, E)$ : static or the long-term key pair where $E = g^D$,

# The Menezes-Qu-Vanstone (MQV) key-exchange protocol

- Let $\{G, g, n, m, h\}$ are public elements,
- Each entity uses two key pairs:
  - $(D, E)$ : static or the long-term key pair where $E = g^D$, and
  - $(d, e)$ : ephemeral or the short-term key pair where $e = g^d$.

# The Menezes-Qu-Vanstone (MQV) key-exchange protocol

- Let $\{G, g, n, m, h\}$ are public elements,
- Each entity uses two key pairs:
  - $(D, E)$ : static or the long-term key pair where $E = g^D$, and
  - $(d, e)$ : ephemeral or the short-term key pair where $e = g^d$.
- Static key of an entity is assumed to be authentic, say, certified by a trusted authority.

# The Menezes-Qu-Vanstone (MQV) key-exchange protocol

- Let $\{G, g, n, m, h\}$ are public elements,
- Each entity uses two key pairs:
  - $(D, E)$ : static or the long-term key pair where $E = g^D$, and
  - $(d, e)$ : ephemeral or the short-term key pair where $e = g^d$.
- Static key of an entity is assumed to be authentic, say, certified by a trusted authority.
- A new ephemeral key pair during each invocation of the protocol.

# The Menezes-Qu-Vanstone (MQV) key-exchange protocol

- Let $\{G, g, n, m, h\}$ are public elements,
- Each entity uses two key pairs:
  - $(D, E)$ : static or the long-term key pair where $E = g^D$, and
  - $(d, e)$ : ephemeral or the short-term key pair where $e = g^d$.
- Static key of an entity is assumed to be authentic, say, certified by a trusted authority.
- A new ephemeral key pair during each invocation of the protocol.
- The ephemeral key is validated using the static private key.

# The Menezes-Qu-Vanstone (MQV) key-exchange protocol

- Let $\{G, g, n, m, h\}$ are public elements,
- Each entity uses two key pairs:
  - $(D, E)$ : static or the long-term key pair where $E = g^D$, and
  - $(d, e)$ : ephemeral or the short-term key pair where $e = g^d$.
- Static key of an entity is assumed to be authentic, say, certified by a trusted authority.
- A new ephemeral key pair during each invocation of the protocol.
- The ephemeral key is validated using the static private key.
- $l$ : bit-length of $m$.
- Publicly known function: $f : G \longrightarrow \mathbb{N}_0$ such that for $a \in G$

$$\hat{a} = 2^{\lceil l/2 \rceil} + f(a) \operatorname{rem} 2^{\lceil l/2 \rceil}.$$

  In particular, $\hat{a} \not\equiv 0 \pmod{m}$.

# The Menezes-Qu-Vanstone (MQV) key-exchange protocol

| **Alice** | **Bob** |
|---|---|
| Generates a random $d_A \in \{2, \ldots, m-1\}$ | Generates a random $d_B \in \{2, \ldots, m-1\}$ |
| Computes $e_A = g^{d_A}$ | Computes $e_B = g^{d_B}$ |

$$\text{Alice} \xrightarrow{e_A} \text{Bob}$$

$$\text{Alice} \xleftarrow{e_B} \text{Bob}$$

# The Menezes-Qu-Vanstone (MQV) key-exchange protocol

| **Alice** | **Bob** |
|---|---|
| Generates a random $d_A \in \{2, \ldots, m-1\}$ | Generates a random $d_B \in \{2, \ldots, m-1\}$ |
| Computes $e_A = g^{d_A}$ | Computes $e_B = g^{d_B}$ |

$$\text{Alice} \xrightarrow{e_A} \text{Bob}$$

$$\text{Alice} \xleftarrow{e_B} \text{Bob}$$

| **Alice** | **Bob** |
|---|---|
| Computes $\sigma_A = d_A + \hat{e}_A D_A$ | Computes $\sigma_B = d_B + \hat{e}_B D_B$ |
| Computes $k = \left(e_B E_B^{\hat{e}_B}\right)^{h\sigma_A}$ | Computes $k = \left(e_A E_A^{\hat{e}_A}\right)^{h\sigma_B}$ |

## Correctness of MQV

$$\left(e_B E_B^{\hat{e}_B}\right)^{h\sigma_A} = \left(g^{d_B}\left(g^{D_B}\right)^{\hat{e}_B}\right)^{h\sigma_A}$$

## Correctness of MQV

$$
\begin{aligned}
\left(e_B E_B^{\hat{e}_B}\right)^{h\sigma_A} &= \left(g^{d_B}\left(g^{D_B}\right)^{\hat{e}_B}\right)^{h\sigma_A} \\
&= \left(g^{d_B+\hat{e}_B D_B}\right)^{h\sigma_A}
\end{aligned}
$$

## Correctness of MQV

$$
\begin{aligned}
\left(e_B E_B^{\hat{e}_B}\right)^{h\sigma_A} &= \left(g^{d_B}\left(g^{D_B}\right)^{\hat{e}_B}\right)^{h\sigma_A} \\
&= \left(g^{d_B+\hat{e}_B D_B}\right)^{h\sigma_A} \\
&= g^{h\sigma_A\sigma_B}
\end{aligned}
$$

## Correctness of MQV

$$
\begin{aligned}
\left(e_B E_B^{\hat{e}_B}\right)^{h\sigma_A} &= \left(g^{d_B}\left(g^{D_B}\right)^{\hat{e}_B}\right)^{h\sigma_A} \\
&= \left(g^{d_B+\hat{e}_B D_B}\right)^{h\sigma_A} \\
&= g^{h\sigma_A\sigma_B} \\
&= \left(g^{d_A+\hat{e}_A D_B}\right)^{h\sigma_B}
\end{aligned}
$$

## Correctness of MQV

$$
\begin{aligned}
\left(e_B E_B^{\hat{e}_B}\right)^{h\sigma_A} &= \left(g^{d_B}\left(g^{D_B}\right)^{\hat{e}_B}\right)^{h\sigma_A} \\
&= \left(g^{d_B+\hat{e}_B D_B}\right)^{h\sigma_A} \\
&= g^{h\sigma_A\sigma_B} \\
&= \left(g^{d_A+\hat{e}_A D_B}\right)^{h\sigma_B} \\
&= \left(g^{d_A}\left(g^{D_A}\right)^{\hat{e}_A}\right)^{h\sigma_B}
\end{aligned}
$$

# The Menezes-Qu-Vanstone (MQV) key-exchange protocol

## Correctness of MQV

$$
\begin{aligned}
\left(e_B E_B^{\hat{e}_B}\right)^{h\sigma_A} &= \left(g^{d_B}\left(g^{D_B}\right)^{\hat{e}_B}\right)^{h\sigma_A} \\
&= \left(g^{d_B+\hat{e}_B D_B}\right)^{h\sigma_A} \\
&= g^{h\sigma_A\sigma_B} \\
&= \left(g^{d_A+\hat{e}_A D_B}\right)^{h\sigma_B} \\
&= \left(g^{d_A}\left(g^{D_A}\right)^{\hat{e}_A}\right)^{h\sigma_B} \\
&= \left(e_A E_A^{\hat{e}_A}\right)^{h\sigma_B}
\end{aligned}
$$

# End