Modern Alice's Adventures in Cryptoland

Francisco Rodríguez-Henríquez
Cinvestav, México

Co-Crypto2019

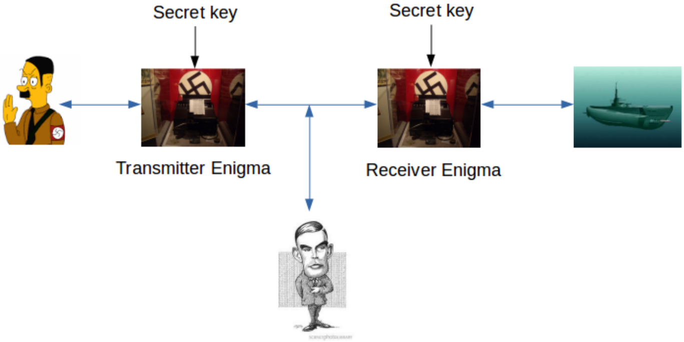
Medellin, Colombia

June, 10 2019

Terminology

- Cryptology: All-inclusive term used for the study of secure communication over non-secure channels and related problems.
- Cryptography: The process of designing systems to realize secure communications over non-secure channels.
- Cryptanalysis: The discipline of breaking cryptographic systems
- Plaintext: Message that we want to transmit in a secure way.
- ciphertext: Resulting document after performing encryption.
- key: Secret information utilized for encrypting/decrypting documents.

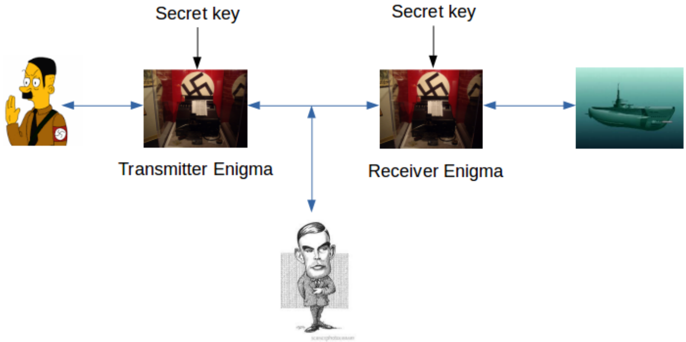
Enigma security model



Alan Turing potential goals:

1. To figure out who are Hitler's receiver partners [traffic analysis]

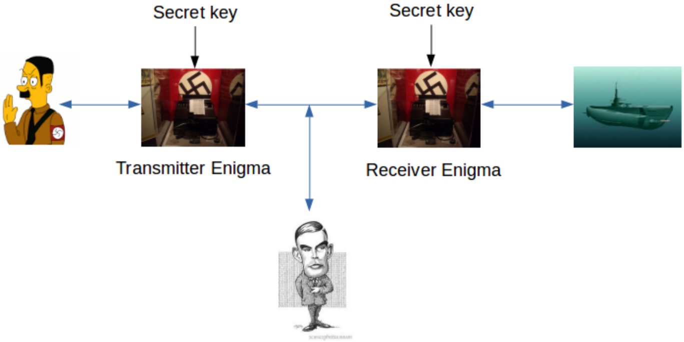
Enigma security model



Alan Turing potential goals:

1. To figure out who are Hitler's receiver partners [traffic analysis]
2. To read [decrypt] the original message

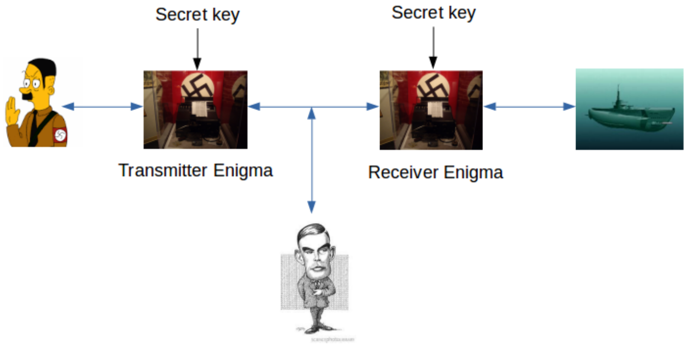
Enigma security model



Alan Turing potential goals:

1. To figure out who are Hitler's receiver partners [traffic analysis]
2. To read [decrypt] the original message
3. To get Hitler's secret key.

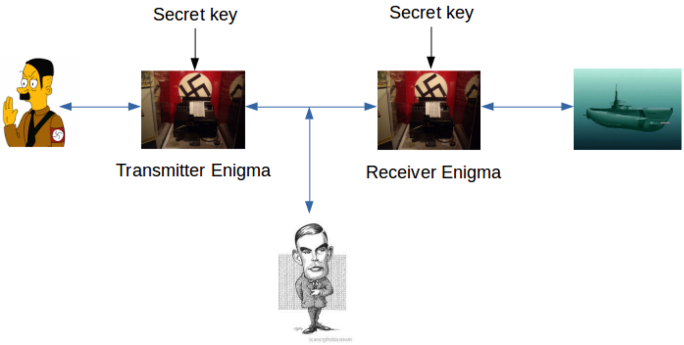
Enigma security model



Alan Turing potential goals:

1. To figure out who are Hitler's receiver partners [traffic analysis]
2. To read [decrypt] the original message
3. To get Hitler's secret key.
4. To modify the contents of the original message.

Enigma security model



Alan Turing potential goals:

1. To figure out who are Hitler's receiver partners [traffic analysis]
2. To read [decrypt] the original message
3. To get Hitler's secret key.
4. To modify the contents of the original message.
5. To impersonate Hitler

Kerckhkoff Principle

"*La sécurité repose sur le secret de la clé, et non sur le secret de l'algorithme.*"
It is assumed that the adversary **knows** the cryptographic algorithm being used.
Therefore, the security of the algorithm must be based on:

Kerckhkoff Principle

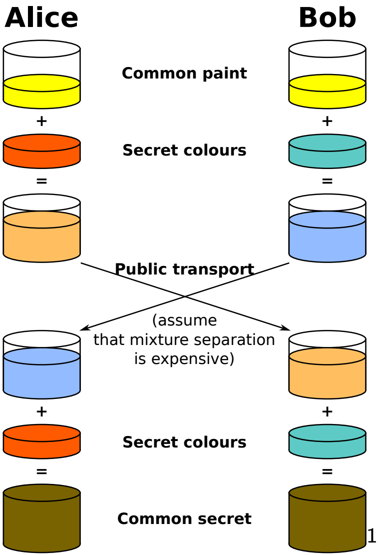
"*La sécurité repose sur le secret de la clé, et non sur le secret de l'algorithme.*"
It is assumed that the adversary **knows** the cryptographic algorithm being used. Therefore, the security of the algorithm must be based on:

- The quality (cryptographic strength) of the algorithm
- Secret key search space (size in bits of the secret key)

Secret Sharing - Diffie Hellman

Problem:

- Alice and Bob wants to paint his houses of the same color.
- They will not allow Eve to know the color.

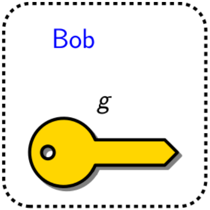


[1]

[1] https://commons.wikimedia.org/wiki/File:Diffie-Hellman_Key_Exchange.svg

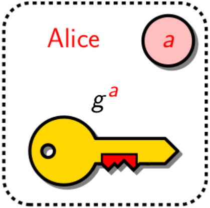
Design problem: How to share a secret?

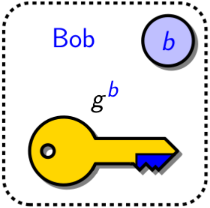
Design problem: How to share a secret?. Solution: Diffie-Hellman Protocol 1976



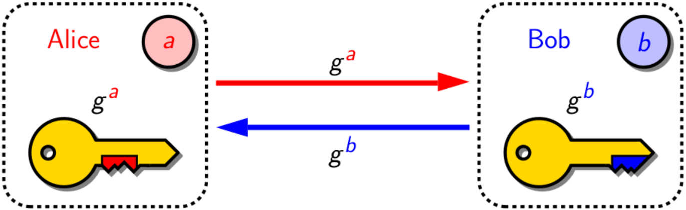
Design problem: How to share a secret?. Solution: Diffie-Hellman Protocol 1976

Design problem: How to share a secret?. Solution: Diffie-Hellman Protocol 1976

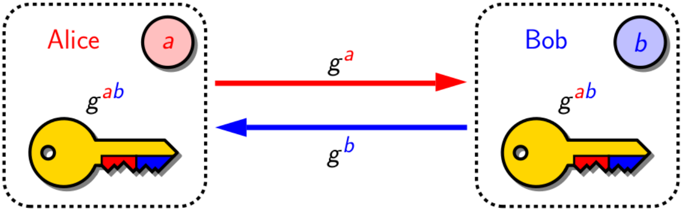




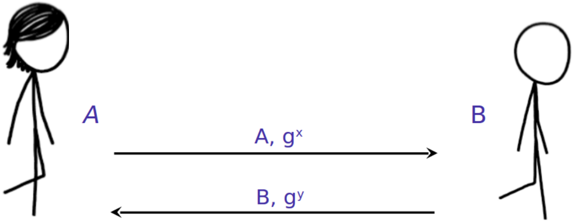
Design problem: How to share a secret?. Solution: Diffie-Hellman Protocol 1976



Design problem: How to share a secret?. Solution: Diffie-Hellman Protocol 1976



Design problem: How to share a secret?. Solution: Diffie-Hellman Protocol 1976

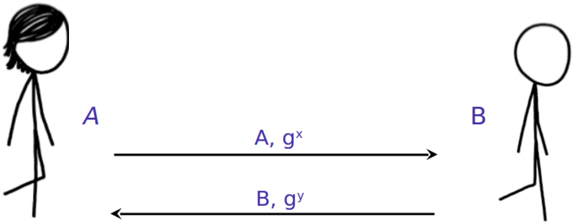


- Alice and Bob adecide to work in the $\mathbb{Z}_p$ group, with p a large odd prime. They also chose a generator $g \in \mathbb{Z}_p$ (i.e., $Ord(g) = p - 1$).
- Alice and Bob Choose $x, y \in \mathbb{Z}_p$, respectively
- Alice and Bob compute a shared secret as,

$$K = (g^x)^y = (g^y)^x$$

Note: This protocol can only be secure against passive attackers

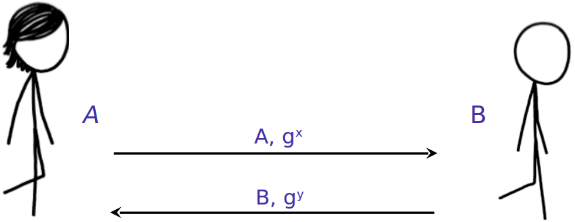
Design problem: How to share a secret?. Solution: Diffie-Hellman Protocol 1976



Protocol's security lies in the computational intractability of solving the Discrete Logarithm Problem (DLP), namely,

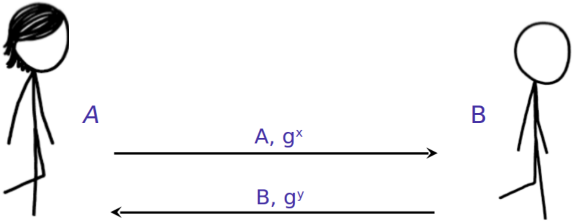
Given a prime p and a generator $g, h \in [1, p-1]$, find an integer x (if it exists) such that, $g^x \equiv h \bmod p$.

Design problem: How to share a secret?. Solution: Diffie-Hellman Protocol 1976



Protocol's security lies in the computational intractability of solving the Discrete Logarithm Problem (DLP), namely,
Given a prime p and a generator $g, h \in [1, p-1]$, find an integer x (if it exists) such that, $g^x \equiv h \bmod p$.

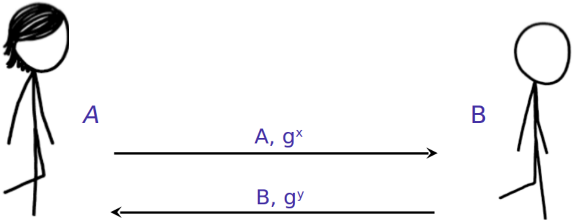
Design problem: How to share a secret?. Solution: Diffie-Hellman Protocol 1976



Protocol's security lies in the computational intractability of solving the Discrete Logarithm Problem (DLP), namely,

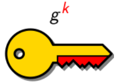
Given a prime p and a generator $g, h \in [1, p-1]$, find an integer x (if it exists) such that, $g^x \equiv h \bmod p$.

Design problem: How to share a secret?. Solution: Diffie-Hellman Protocol 1976

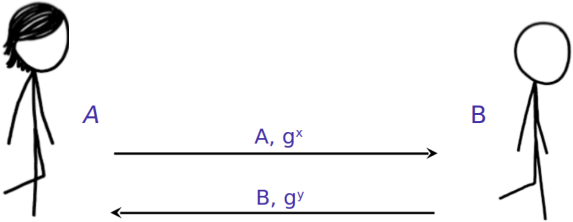


Protocol's security lies in the computational intractability of solving the Discrete Logarithm Problem (DLP), namely,

Given a prime p and a generator $g, h \in [1, p-1]$, find an integer x (if it exists) such that, $g^x \equiv h \bmod p$.

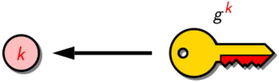


Design problem: How to share a secret?. Solution: Diffie-Hellman Protocol 1976

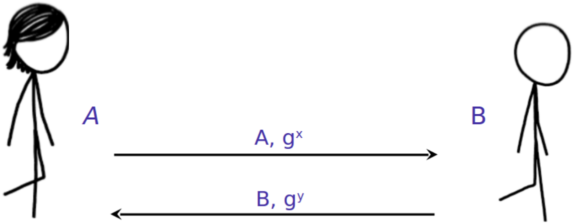


Protocol's security lies in the computational intractability of solving the Discrete Logarithm Problem (DLP), namely,

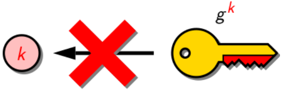
Given a prime p and a generator $g, h \in [1, p-1]$, find an integer x (if it exists) such that, $g^x \equiv h \bmod p$.



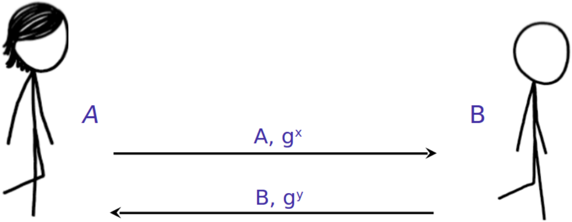
Design problem: How to share a secret?. Solution: Diffie-Hellman Protocol 1976



Protocol's security lies in the computational intractability of solving the Discrete Logarithm Problem (DLP), namely,
Given a prime p and a generator $g, h \in [1, p-1]$, find an integer x (if it exists) such that, $g^x \equiv h \bmod p$.



Design problem: How to share a secret?. Solution: Diffie-Hellman Protocol 1976



- Diffie and Hellman published their protocol in their breakthrough paper, Diffie, W.; Hellman, M. (1976). "New directions in cryptography". IEEE Transactions on Information Theory. 22 (6): 644–654."
- Diffie and Hellman won the 2015 Turing award
- Since its publication in 1976, "New directions in cryptography" has inspired many new ideas in the discipline. In this talk we will review four different versions of this protocol [!]]

Main primitives and building blocks in modern cryptography

Main primitives and building blocks in modern cryptography

- Primitives:
  - Encryption/decryption of digital documents [this task is typically solved using symmetric cryptography]

Main primitives and building blocks in modern cryptography

- Primitives:
  - Encryption/decryption of digital documents [this task is typically solved using symmetric cryptography]
  - Sharing a secret among two or more parties [this task is usually solved using the Diffie-Hellman protocol or its variants]

Main primitives and building blocks in modern cryptography

- Primitives:
  - Encryption/decryption of digital documents [this task is typically solved using symmetric cryptography]
  - Sharing a secret among two or more parties [this task is usually solved using the Diffie-Hellman protocol or its variants]
  - Signature/verification of digital documents [This task is usually solved using public key cryptography]

Main primitives and building blocks in modern cryptography

- Primitives:
  - Encryption/decryption of digital documents [this task is typically solved using symmetric cryptography]
  - Sharing a secret among two or more parties [this task is usually solved using the Diffie-Hellman protocol or its variants]
  - Signature/verification of digital documents [This task is usually solved using public key cryptography]
- Building blocks:
  - Block ciphers and stream ciphers
  - Hash functions
  - Public key crypto-schemes
  - ...

Hard computational problems

1. Integer factorization problem: Given a positive integer N find its prime factors.

Hard computational problems

1. Integer factorization problem: Given a positive integer N find its prime factors.
   [$2019 = ?$]

Hard computational problems

1. Integer factorization problem: Given a positive integer N find its prime factors.
   [$2019 = ?$]
   [answer: $2019 = 3 \cdot 673$]

Hard computational problems

1. Integer factorization problem: Given a positive integer N find its prime factors.
   [$2019 = ?$]
   [answer: $2019 = 3 \cdot 673$]
2. Discrete logarithm problem: Given a prime p and $g, h \in [1, p-1]$, find an integer x (if one exists) such that, $g^x \equiv h \bmod p$.
   [find x such that $2^x \equiv 304 \bmod 419$]

Hard computational problems

1. Integer factorization problem: Given a positive integer N find its prime factors.
   [$2019 = ?$]
   [answer: $2019 = 3 \cdot 673$]
2. Discrete logarithm problem: Given a prime p and $g, h \in [1, p-1]$, find an integer x (if one exists) such that, $g^x \equiv h \bmod p$.
   [find x such that $2^x \equiv 304 \bmod 419$]
   answer: $2^{343} \equiv 304 \bmod 419$.

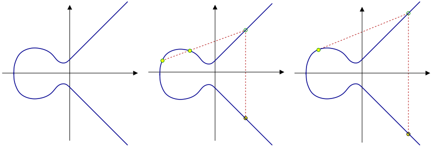
Hard computational problems

1. Integer factorization problem: Given a positive integer N find its prime factors.
   [$2019 = ?$]
   [answer: $2019 = 3 \cdot 673$]
2. Discrete logarithm problem: Given a prime p and $g, h \in [1, p-1]$, find an integer x (if one exists) such that, $g^x \equiv h \bmod p$.
   [find x such that $2^x \equiv 304 \bmod 419$]
   answer: $2^{343} \equiv 304 \bmod 419$.
   More generally: Given $g, h \in \mathbb{F}_q^*$, find an integer x (if one exists) such that, $g^x \equiv h$, where $q = p^l$ is the power of a prime

Elliptic-curve-based cryptography

- Elliptic-curve-based cryptography (ECC) was independently proposed by Victor Miller and Neal Koblitz in 1985.
- It took more than two decades for ECC to be widely accepted and become the most popular public-key cryptographic scheme (above its archrival RSA)
- Nowadays ECC is massively used in everyday applications

Elliptic-curve-based cryptography



An elliptic curve is defined by the set of affine points $(x, y) \in \mathbb{F}_p \times \mathbb{F}_p$, with $p > 3$ an odd large prime, which satisfies the short Weierstrass equation given as,

$$E : y^2 = x^3 + ax + b,$$

along with a point at infinity denoted as $\mathcal{O}$.
Let $E(\mathbb{F}_p)$ be the set of points that satisfy the elliptic curve equation above. This set forms an Abelian group with order (size) given as, $\#E(\mathbb{F}_p) = h \cdot r$, where r is a large prime and the cofactor is a small integer.

Elliptic curves

- E defined by a Weierstraß equation of the form

$$y^2 = x^3 + Ax + B$$

Elliptic curves

- E defined by a Weierstraß equation of the form

$$y^2 = x^3 + Ax + B$$

- $E(K)$ set of rational points over a field K

Elliptic curves

- E defined by a Weierstraß equation of the form

$$y^2 = x^3 + Ax + B$$

- $E(K)$ set of rational points over a field K
- Additive group law over $E(K)$

Elliptic curves

- E defined by a Weierstraß equation of the form

$$y^2 = x^3 + Ax + B$$

- $E(K)$ set of rational points over a field K
- Additive group law over $E(K)$
- Many applications in cryptography since 1985
  - EC-based Diffie-Hellman key exchange
  - EC-based Digital Signature Algorithm
  -

Elliptic curves

- E defined by a Weierstraß equation of the form

$$y^2 = x^3 + Ax + B$$

- $E(K)$ set of rational points over a field K
- Additive group law over $E(K)$
- Many applications in cryptography since 1985
  - EC-based Diffie-Hellman key exchange
  - EC-based Digital Signature Algorithm
  -
- Interest: smaller keys than usual cryptosystems (RSA, ElGamal, ...)

Elliptic curves

- E defined by a Weierstraß equation of the form

$$y^2 = x^3 + Ax + B$$

- $E(K)$ set of rational points over a field K
- Additive group law over $E(K)$
- Many applications in cryptography since 1985
  - EC-based Diffie-Hellman key exchange
  - EC-based Digital Signature Algorithm
  -
- Interest: smaller keys than usual cryptosystems (RSA, ElGamal, ...)
- But there's more:
  - Bilinear pairings
  - Isogenous elliptic curves

Group cryptography

- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$

Group cryptography

- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$
- P, a generator of the group: $\mathbb{G}_1 = \langle P \rangle$

Group cryptography

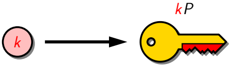
- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$
- P, a generator of the group: $\mathbb{G}_1 = \langle P \rangle$
- Scalar multiplication: for any integer k, we have
$kP = \underbrace{P + P + \cdots + P}_{k \text{ times}}$

Group cryptography

- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$
- P, a generator of the group: $\mathbb{G}_1 = \langle P \rangle$
- Scalar multiplication: for any integer k, we have
  $kP = \underbrace{P + P + \cdots + P}_{k \text{ times}}$

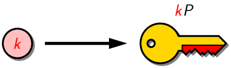
Group cryptography

- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$
- P, a generator of the group: $\mathbb{G}_1 = \langle P \rangle$
- Scalar multiplication: for any integer k, we have $kP = \underbrace{P + P + \cdots + P}_{k \text{ times}}$



Group cryptography

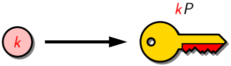
- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$
- P, a generator of the group: $\mathbb{G}_1 = \langle P \rangle$
- Scalar multiplication: for any integer k, we have
  $kP = \underbrace{P + P + \cdots + P}_{k \text{ times}}$



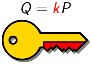
- Discrete logarithm: given $Q \in \mathbb{G}_1$, compute k such that $Q = kP$

Group cryptography

- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$
- P, a generator of the group: $\mathbb{G}_1 = \langle P \rangle$
- Scalar multiplication: for any integer k, we have
  $kP = \underbrace{P + P + \cdots + P}_{k \text{ times}}$

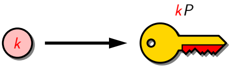


- Discrete logarithm: given $Q \in \mathbb{G}_1$, compute k such that $Q = kP$

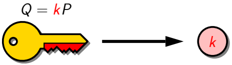


Group cryptography

- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$
- P, a generator of the group: $\mathbb{G}_1 = \langle P \rangle$
- Scalar multiplication: for any integer k, we have
  $kP = \underbrace{P + P + \cdots + P}_{k \text{ times}}$

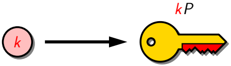


- Discrete logarithm: given $Q \in \mathbb{G}_1$, compute k such that $Q = kP$

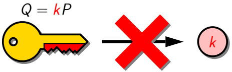


Group cryptography

- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$
- P, a generator of the group: $\mathbb{G}_1 = \langle P \rangle$
- Scalar multiplication: for any integer k, we have
  $kP = \underbrace{P + P + \cdots + P}_{k \text{ times}}$

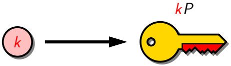


- Discrete logarithm: given $Q \in \mathbb{G}_1$, compute k such that $Q = kP$

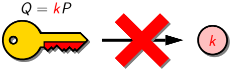


Group cryptography

- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$
- P, a generator of the group: $\mathbb{G}_1 = \langle P \rangle$
- Scalar multiplication: for any integer k, we have
  $kP = \underbrace{P + P + \cdots + P}_{k \text{ times}}$

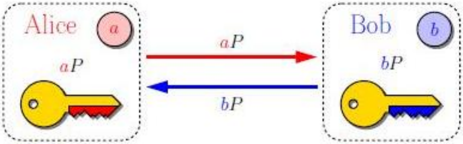


- Discrete logarithm: given $Q \in \mathbb{G}_1$, compute k such that $Q = kP$



- We assume that the discrete logarithm problem (DLP) in $\mathbb{G}_1$ is hard

The Elliptic Curve Diffie-Hellman (ECDH) Protocol



The Elliptic Curve Diffie-Hellman (ECDH) Protocol

Algorithm 1 The elliptic curve Diffie-Hellman protocol

Public parameters: Prime p, curve $E/\mathbb{F}_p$, point $P = (x, y) \in E(\mathbb{F}_p)$ of order r

Phase 1: Key pair generation

Alice	**Bob**
1: Select the private key $d_A \xleftarrow{\$} [1, r-1]$	1: Select the private key $d_B \xleftarrow{\$} [1, r-1]$
2: Compute the public key $Q_A \leftarrow d_A P$	2: Compute the public key $Q_B \leftarrow d_B P$

Phase 2: Shared secret computation

Alice	**Bob**
3: Send Q_A to Bob	3: Send Q_B to Alice
4: Compute $R \leftarrow d_A Q_B$	4: Compute $R \leftarrow d_B Q_A$

Final phase: The shared secret is x-coordinate of the point R

How to efficiently compute the Elliptic Curve Diffie-Hellman (ECDH) Protocol?

The Montgomery ladder

A famous elliptic curve: Curve25519

- Curve25519 satisfies the Montgomery elliptic curve,

$$E : y^2 = x^3 + 48666 \cdot x^2 + x,$$

- Curve25519 is used for generating shared-secrets on applications such as TLS 1.3 and WhatsApp, among others.
- Proposed by Daniel J. Bernstein en 2006, it became massively popular around 2013

Daniel J. Bernstein: "Curve25519: New Diffie-Hellman Speed Records". Public Key Cryptography 2006: 207-228

Algorithm 2 Left-to-right Montgomery ladder [Montgomery'87]

Require: $P = (u_P, v_P) \in E_A(\mathbb{F}_p)$, $k = (k_{n-1} = 1, k_{n-2}, \ldots, k_1, k_0)_2$
Ensure: $u_{Q = k \cdot P}$

$R_0 \leftarrow \mathcal{O}$; $R_1 \leftarrow u_P$;
for $i = n-1$ **downto** 0 **do**
  if $k_i = 1$ **then**
    $R_0 \leftarrow R_0 +_{(P)} R_1$; $R_1 \leftarrow 2R_1$
  else
    $R_1 \leftarrow R_0 +_{(P)} R_1$; $R_0 \leftarrow 2R_0$
  end if
end for
return $u_Q \leftarrow R_0$

Algorithm 2 Left-to-right Montgomery ladder [Montgomery'87]

Require: $P = (u_P, v_P) \in E_A(\mathbb{F}_p)$, $k = (k_{n-1} = 1, k_{n-2}, \ldots, k_1, k_0)_2$
Ensure: $u_{Q=k \cdot P}$

$R_0 \leftarrow \mathcal{O}$; $R_1 \leftarrow u_P$;
for $i = n - 1$ **downto** 0 **do**
  if $k_i = 1$ **then**
    $R_0 \leftarrow R_0 +_{(P)} R_1$; $\quad R_1 \leftarrow 2R_1$
  else
    $R_1 \leftarrow R_0 +_{(P)} R_1$; $\quad R_0 \leftarrow 2R_0$
  end if
end for
return $u_Q \leftarrow R_0$

Peter L. Montgomery.: "Speeding the Pollard and elliptic curve methods of factorization". Math. Comput. 48(177), 243–264 (1987)

Algorithm 2 Left-to-right Montgomery ladder [Montgomery'87]

Require: $P = (u_P, v_P) \in E_A(\mathbb{F}_p),\ k = (k_{n-1} = 1, k_{n-2}, \ldots, k_1, k_0)_2$
Ensure: $u_{Q=k\cdot P}$

```
R_0 ← O; R_1 ← u_P;
for i = n − 1 downto 0 do
    if k_i = 1 then
        R_0 ← R_0 +_(P) R_1;    R_1 ← 2R_1
    else
        R_1 ← R_0 +_(P) R_1;    R_0 ← 2R_0
    end if
end for
return u_Q ← R_0
```

Remark 1: The Montgomery ladder maintains the invariant $R_1 - R_0 = P$ by computing at each iteration

$$(R_0, R_1) \leftarrow \begin{cases} (2R_0, 2R_0 + P), & \text{if } k_i = 0 \\ (2R_0 + P, 2R_0 + 2P), & \text{if } k_i = 1. \end{cases}$$

Algorithm 2 Left-to-right Montgomery ladder [Montgomery'87]

Require: $P = (u_P, v_P) \in E_A(\mathbb{F}_p),\ k = (k_{n-1} = 1, k_{n-2}, \dots, k_1, k_0)_2$
Ensure: $u_{Q=k\cdot P}$

$R_0 \leftarrow \mathcal{O};\ R_1 \leftarrow u_P;$
for $i = n-1$ **downto** 0 **do**
    if $k_i = 1$ **then**
        $R_0 \leftarrow R_0 +_{(P)} R_1;\quad R_1 \leftarrow 2R_1$
    else
        $R_1 \leftarrow R_0 +_{(P)} R_1;\quad R_0 \leftarrow 2R_0$
    end if
end for
return $u_Q \leftarrow R_0$

Remark 2: If the difference between the points R_1 and R_0 is known, it is possible to derive efficient differential addition formulas, namely,

$$U_{R_1} \leftarrow Z_P \cdot ((U_{R_1} + Z_{R_1}) \cdot (U_{R_0} - Z_{R_0}) + (U_{R_1} - Z_{R_1}) \cdot (U_{R_0} + Z_{R_0}))^2$$
$$Z_{R_1} \leftarrow u_P \cdot ((U_{R_1} + Z_{R_1}) \cdot (U_{R_0} - Z_{R_0}) - (U_{R_1} - Z_{R_1}) \cdot (U_{R_0} + Z_{R_0}))^2.$$

Using the standard trick of making $Z_P = 1$ this can be computed at a cost of $2\mathbf{m} + 1\mathbf{m_{uP}} + 2\mathbf{s} + 6\mathbf{a}$

Algorithm 2 Left-to-right Montgomery ladder [Montgomery'87]

Require: $P = (u_P, v_P) \in E_A(\mathbb{F}_p)$, $k = (k_{n-1} = 1, k_{n-2}, \ldots, k_1, k_0)_2$
Ensure: $u_{Q=k\cdot P}$

```
R_0 ← O; R_1 ← u_P;
for i = n − 1 downto 0 do
    if k_i = 1 then
        R_0 ← R_0 +_(P) R_1;    R_1 ← 2R_1
    else
        R_1 ← R_0 +_(P) R_1;    R_0 ← 2R_0
    end if
end for
return u_Q ← R_0
```

Remark 2: Similarly, the operation of doubling the point R_0, can be efficiently computed as,

$$\begin{aligned} U_{R_0} &\leftarrow (U_{R_0} + Z_{R_0})^2 \cdot (U_{R_0} - Z_{R_0})^2 \\ T &\leftarrow (U_{R_0} + Z_{R_0})^2 - (U_{R_0} - Z_{R_0})^2 \\ Z_{R_0} &\leftarrow \left[a_{24} \cdot T + (U_{R_0} - Z_{R_0})^2\right] \cdot T, \end{aligned}$$

which can be computed at a cost of $2\mathbf{m} + 1\mathbf{m}_{\mathbf{a24}} + 2\mathbf{s} + 4\mathbf{a}$, where $\mathbf{m}_{\mathbf{a24}}$ stands for one multiplication by the constant $a_{24} = \frac{A+2}{4}$.

Algorithm 2 Left-to-right Montgomery ladder [Montgomery'87]

Require: $P = (u_P, v_P) \in E_A(\mathbb{F}_p),\ k = (k_{n-1} = 1, k_{n-2}, \ldots, k_1, k_0)_2$
Ensure: $u_{Q=k\cdot P}$

```
R_0 ← O; R_1 ← u_P;
for i = n − 1 downto 0 do
    if k_i = 1 then
        R_0 ← R_0 +_(P) R_1;   R_1 ← 2R_1
    else
        R_1 ← R_0 +_(P) R_1;   R_0 ← 2R_0
    end if
end for
return u_Q ← R_0
```

Total computational cost: In summary, the computational cost of the Montgomery ladder is,

$$n \cdot (4\mathbf{m} + 1\mathbf{m_{a24}} + 1\mathbf{m_{uP}} + 4\mathbf{s} + 8\mathbf{a}) + 1\mathbf{m} + 1\mathbf{i}.$$

In the RFC 7748 [essentially] this algorithm is called X25519 (with $n = 255$)

Algorithm 3 Low-level left-to-right Montgomery ladder

Require: $P = (u_P, v_P) \in E_A/\mathbb{F}_p$, $k = (k_{n-1} = 1, k_{n-2}, \ldots, k_1, k_0)_2$, $a_{24} = (A+2)/4$

Ensure: $u_{Q=kP}$

Initialization: $U_{R_0} \leftarrow 1$, $Z_{R_0} \leftarrow 0$, $U_{R_1} \leftarrow u_P$, $Z_{R_1} \leftarrow 1$, $s \leftarrow 0$

for $i \leftarrow n-1$ **downto** 0 **do**

    `# timing-attack countermeasure`

    $s \leftarrow s \oplus k_i$

    $U_{R_0}, U_{R_1} \leftarrow \mathrm{cswap}(s, U_{R_0}, U_{R_1})$

    $Z_{R_0}, Z_{R_1} \leftarrow \mathrm{cswap}(s, Z_{R_0}, Z_{R_1})$

    $s \leftarrow k_i$

    `# common operations`

    $A \leftarrow U_{R_0} + Z_{R_0}$; $B \leftarrow U_{R_0} - Z_{R_0}$

    `# addition`

    $C \leftarrow U_{R_1} + Z_{R_1}$; $D \leftarrow U_{R_1} - Z_{R_1}$

    $C \leftarrow C \times B$; $D \leftarrow D \times A$

    $U_{R_1} \leftarrow D + C$; $U_{R_1} \leftarrow U_{R_1}^2$

    $Z_{R_1} \leftarrow D - C$; $Z_{R_1} \leftarrow Z_{R_1}^2$; $Z_{R_1} \leftarrow u_P \times Z_{R_1}$

    `# doubling`

    $A \leftarrow A^2$; $B \leftarrow B^2$

    $U_{R_0} \leftarrow A \times B$

    $A \leftarrow A - B$

    $Z_{R_0} \leftarrow a_{24} \times A$; $Z_{R_0} \leftarrow Z_{R_0} + B$; $Z_{R_0} \leftarrow Z_{R_0} \times A$

end for

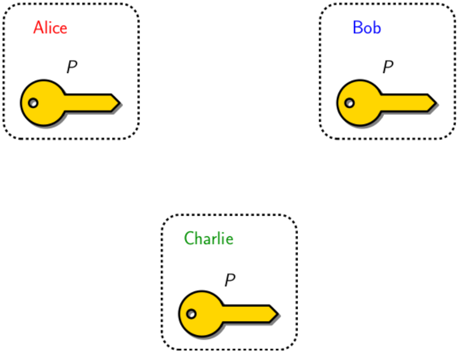
$U_{R_0}, U_{R_1} \leftarrow \mathrm{cswap}(s, U_{R_0}, U_{R_1})$

$Z_{R_0}, Z_{R_1} \leftarrow \mathrm{cswap}(s, Z_{R_0}, Z_{R_1})$

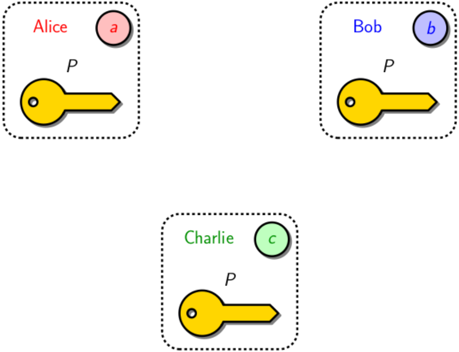
$Z_{R_0} \leftarrow Z_{R_0}^{-1}$; $u_{R_0} \leftarrow U_{R_0} \times Z_{R_0}$

return $u_Q \leftarrow u_{R_0}$

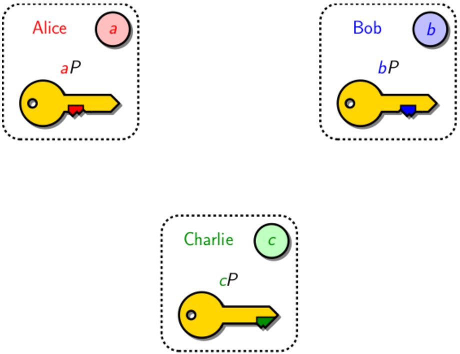
Design problem: How to establish a one-round tripartite shared-secret protocol?



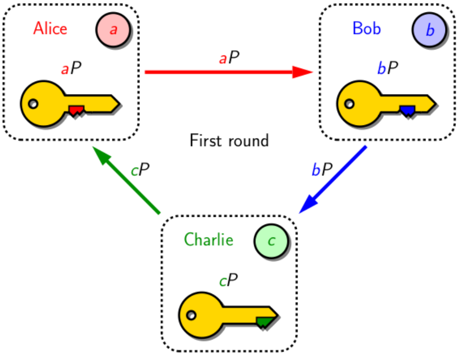
Design problem: How to establish a one-round tripartite shared-secret protocol?



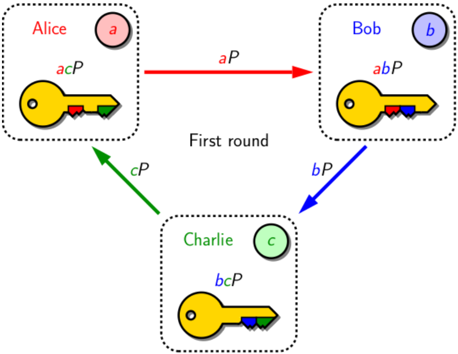
Design problem: How to establish a one-round tripartite shared-secret protocol?



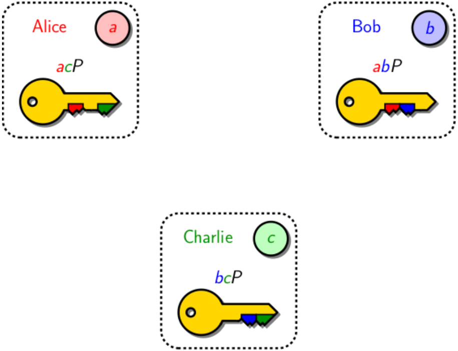
Design problem: How to establish a one-round tripartite shared-secret protocol?



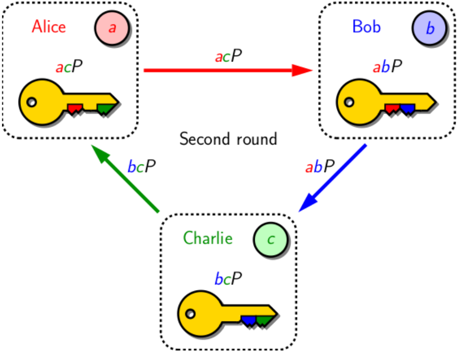
Design problem: How to establish a one-round tripartite shared-secret protocol?



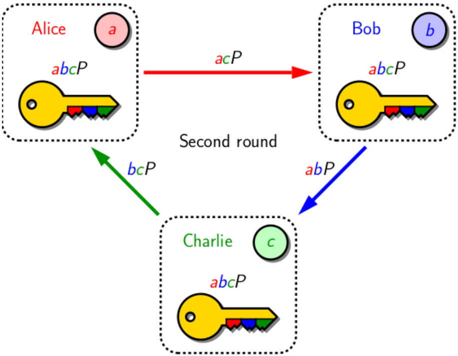
Design problem: How to establish a one-round tripartite shared-secret protocol?



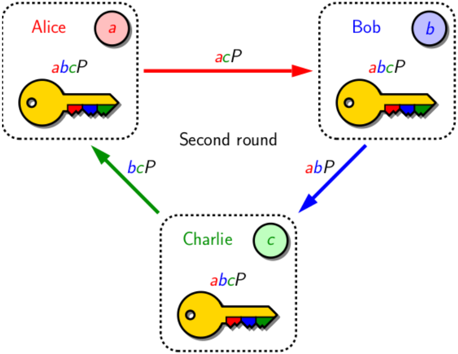
Design problem: How to establish a one-round tripartite shared-secret protocol?



Design problem: How to establish a one-round tripartite shared-secret protocol?



Design problem: How to establish a one-round tripartite shared-secret protocol?



- This problem remained open since the 1976 Diffie-Hellman paper,: There exists a tripartite Diffie-Hellman protocol that can be executed in just one round of public key exchanges?

Bilinear pairings

- $(\mathbb{G}_2, \times)$, a multiplicatively-written cyclic group of order $\#\mathbb{G}_2 = \#\mathbb{G}_1 = \ell$

Bilinear pairings

- $(\mathbb{G}_2, \times)$, a multiplicatively-written cyclic group of order $\#\mathbb{G}_2 = \#\mathbb{G}_1 = \ell$
- A bilinear pairing on $(\mathbb{G}_1, \mathbb{G}_2)$ is a map

$$\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$$

that satisfies the following conditions:

Bilinear pairings

- $(\mathbb{G}_2, \times)$, a multiplicatively-written cyclic group of order $\#\mathbb{G}_2 = \#\mathbb{G}_1 = \ell$
- A bilinear pairing on $(\mathbb{G}_1, \mathbb{G}_2)$ is a map

$$\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$$

that satisfies the following conditions:

  - non-degeneracy: $\hat{e}(P, P) \neq 1_{\mathbb{G}_2}$ (equivalently $\hat{e}(P, P)$ generates $\mathbb{G}_2$)

Bilinear pairings

- $(\mathbb{G}_2, \times)$, a multiplicatively-written cyclic group of order $\#\mathbb{G}_2 = \#\mathbb{G}_1 = \ell$
- A bilinear pairing on $(\mathbb{G}_1, \mathbb{G}_2)$ is a map

$$\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$$

that satisfies the following conditions:

  - non-degeneracy: $\hat{e}(P, P) \neq 1_{\mathbb{G}_2}$ (equivalently $\hat{e}(P, P)$ generates $\mathbb{G}_2$)
  - bilinearity:
    $\hat{e}(Q_1+Q_2, R) = \hat{e}(Q_1, R)\cdot\hat{e}(Q_2, R) \quad \hat{e}(Q, R_1+R_2) = \hat{e}(Q, R_1)\cdot\hat{e}(Q, R_2)$

Bilinear pairings

- $(\mathbb{G}_2, \times)$, a multiplicatively-written cyclic group of order $\#\mathbb{G}_2 = \#\mathbb{G}_1 = \ell$
- A bilinear pairing on $(\mathbb{G}_1, \mathbb{G}_2)$ is a map

$$\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$$

that satisfies the following conditions:

  - non-degeneracy: $\hat{e}(P, P) \neq 1_{\mathbb{G}_2}$ (equivalently $\hat{e}(P, P)$ generates $\mathbb{G}_2$)
  - bilinearity:
    $\hat{e}(Q_1+Q_2, R) = \hat{e}(Q_1, R)\cdot\hat{e}(Q_2, R) \quad \hat{e}(Q, R_1+R_2) = \hat{e}(Q, R_1)\cdot\hat{e}(Q, R_2)$
  - computability: $\hat{e}$ can be efficiently computed

Bilinear pairings

- $(\mathbb{G}_2, \times)$, a multiplicatively-written cyclic group of order $\#\mathbb{G}_2 = \#\mathbb{G}_1 = \ell$
- A bilinear pairing on $(\mathbb{G}_1, \mathbb{G}_2)$ is a map

$$\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$$

  that satisfies the following conditions:
  - non-degeneracy: $\hat{e}(P, P) \neq 1_{\mathbb{G}_2}$ (equivalently $\hat{e}(P, P)$ generates $\mathbb{G}_2$)
  - bilinearity:
    $\hat{e}(Q_1 + Q_2, R) = \hat{e}(Q_1, R) \cdot \hat{e}(Q_2, R) \quad \hat{e}(Q, R_1 + R_2) = \hat{e}(Q, R_1) \cdot \hat{e}(Q, R_2)$
  - computability: $\hat{e}$ can be efficiently computed
- Immediate property: for any two integers k_1 and k_2

$$\hat{e}(k_1 Q, k_2 R) = \hat{e}(Q, R)^{k_1 k_2}$$

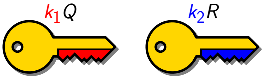
Bilinear pairings

- $(\mathbb{G}_2, \times)$, a multiplicatively-written cyclic group of order $\#\mathbb{G}_2 = \#\mathbb{G}_1 = \ell$
- A bilinear pairing on $(\mathbb{G}_1, \mathbb{G}_2)$ is a map

$$\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$$

  that satisfies the following conditions:
  - non-degeneracy: $\hat{e}(P, P) \neq 1_{\mathbb{G}_2}$ (equivalently $\hat{e}(P, P)$ generates $\mathbb{G}_2$)
  - bilinearity:
    $\hat{e}(Q_1+Q_2, R) = \hat{e}(Q_1, R)\cdot\hat{e}(Q_2, R) \quad \hat{e}(Q, R_1+R_2) = \hat{e}(Q, R_1)\cdot\hat{e}(Q, R_2)$
  - computability: $\hat{e}$ can be efficiently computed
- Immediate property: for any two integers k_1 and k_2

$$\hat{e}(k_1 Q, k_2 R) = \hat{e}(Q, R)^{k_1 k_2}$$



Bilinear pairings

- $(\mathbb{G}_2, \times)$, a multiplicatively-written cyclic group of order $\#\mathbb{G}_2 = \#\mathbb{G}_1 = \ell$
- A bilinear pairing on $(\mathbb{G}_1, \mathbb{G}_2)$ is a map

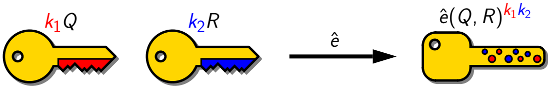
$$\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$$

that satisfies the following conditions:

  - non-degeneracy: $\hat{e}(P, P) \neq 1_{\mathbb{G}_2}$ (equivalently $\hat{e}(P, P)$ generates $\mathbb{G}_2$)
  - bilinearity:
    $\hat{e}(Q_1 + Q_2, R) = \hat{e}(Q_1, R) \cdot \hat{e}(Q_2, R) \quad \hat{e}(Q, R_1 + R_2) = \hat{e}(Q, R_1) \cdot \hat{e}(Q, R_2)$
  - computability: $\hat{e}$ can be efficiently computed

- Immediate property: for any two integers k_1 and k_2

$$\hat{e}(k_1 Q, k_2 R) = \hat{e}(Q, R)^{k_1 k_2}$$



Pairings in cryptography

- At first, used to attack supersingular elliptic curves
  - Menezes-Okamoto-Vanstone and Frey-Rück attacks, 1993 and 1994

$$\begin{array}{ccc} \mathrm{DLP}_{\mathbb{G}_1} & <_{\mathrm{P}} & \mathrm{DLP}_{\mathbb{G}_2} \\ kP & \longrightarrow & \hat{e}(kP, P) = \hat{e}(P, P)^k \end{array}$$

  - for cryptographic applications, we will also require the DLP in $\mathbb{G}_2$ to be hard

Pairings in cryptography

- At first, used to attack supersingular elliptic curves
  - Menezes-Okamoto-Vanstone and Frey-Rück attacks, 1993 and 1994

$$\begin{array}{ccc} \mathrm{DLP}_{\mathbb{G}_1} & <_{\mathrm{P}} & \mathrm{DLP}_{\mathbb{G}_2} \\ kP & \longrightarrow & \hat{e}(kP, P) = \hat{e}(P, P)^k \end{array}$$

  - for cryptographic applications, we will also require the DLP in $\mathbb{G}_2$ to be hard
- One-round three-party key agreement (Joux, 2000)

Pairings in cryptography

- At first, used to attack supersingular elliptic curves
  - Menezes-Okamoto-Vanstone and Frey-Rück attacks, 1993 and 1994

$$\begin{array}{ccc} \mathrm{DLP}_{\mathbb{G}_1} & <_{\mathrm{P}} & \mathrm{DLP}_{\mathbb{G}_2} \\ kP & \longrightarrow & \hat{e}(kP, P) = \hat{e}(P, P)^k \end{array}$$

  - for cryptographic applications, we will also require the DLP in $\mathbb{G}_2$ to be hard
- One-round three-party key agreement (Joux, 2000)
- Identity-based encryption
  - Boneh–Franklin, 2001
  - Sakai–Kasahara, 2001

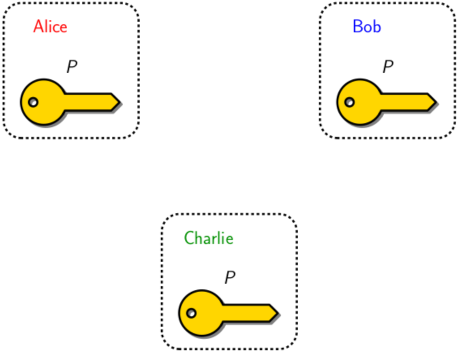
Pairings in cryptography

- At first, used to attack supersingular elliptic curves
  - Menezes-Okamoto-Vanstone and Frey-Rück attacks, 1993 and 1994

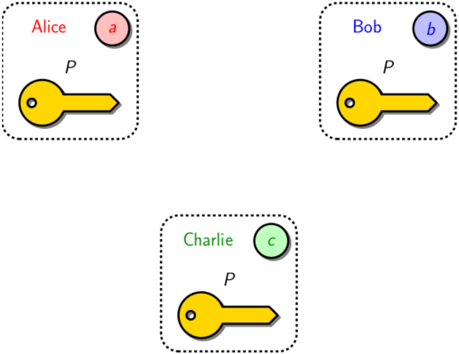
$$\begin{array}{ccc} \mathrm{DLP}_{\mathbb{G}_1} & <_{\mathrm{P}} & \mathrm{DLP}_{\mathbb{G}_2} \\ kP & \longrightarrow & \hat{e}(kP, P) = \hat{e}(P, P)^k \end{array}$$

  - for cryptographic applications, we will also require the DLP in $\mathbb{G}_2$ to be hard
- One-round three-party key agreement (Joux, 2000)
- Identity-based encryption
  - Boneh–Franklin, 2001
  - Sakai–Kasahara, 2001
- Short digital signatures, Aggregate signatures
  - Boneh–Lynn–Shacham, 2001
  - Boneh–Gentry–Lynn–Shacham, 2004
- cryptocurrencies, Pinocchio, Zcash 2013

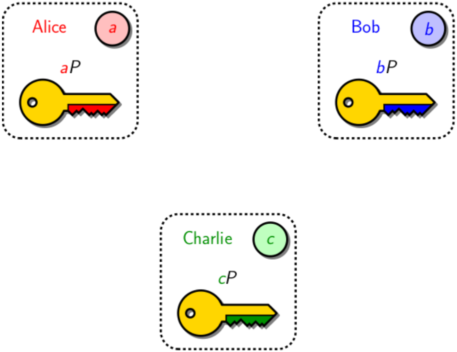
Design problem: How to establish a one-round tripartite shared-secret protocol? Solution:A One Round Protocol for Tripartite Diffie-Hellman.



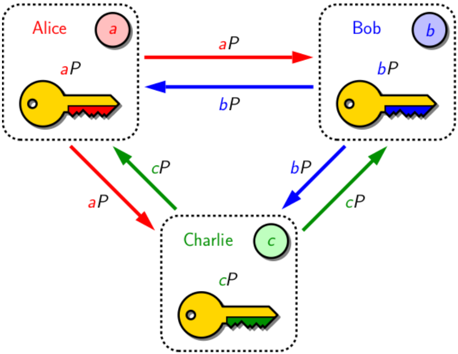
Design problem: How to establish a one-round tripartite shared-secret protocol? Solution:A One Round Protocol for Tripartite Diffie-Hellman.



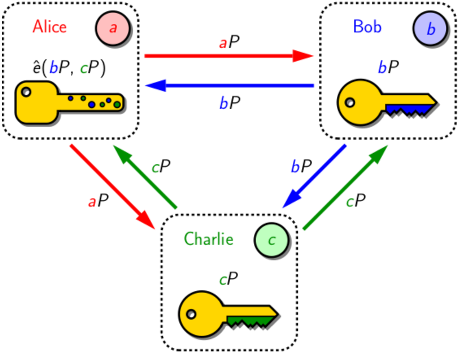
Design problem: How to establish a one-round tripartite shared-secret protocol? Solution: A One Round Protocol for Tripartite Diffie-Hellman.



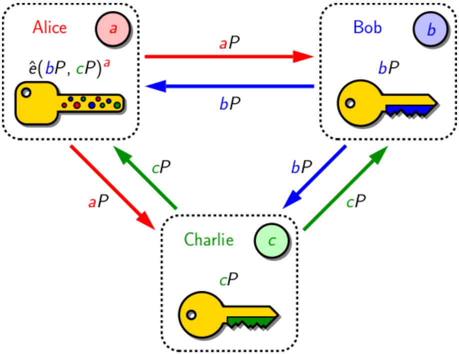
Design problem: How to establish a one-round tripartite shared-secret protocol? Solution: A One Round Protocol for Tripartite Diffie-Hellman.



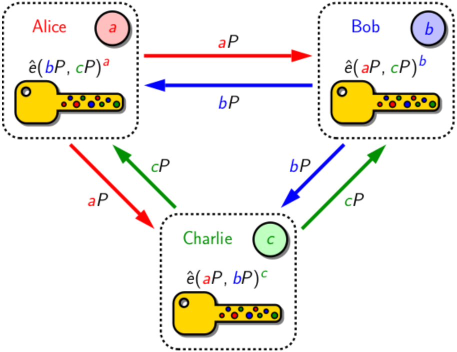
Design problem: How to establish a one-round tripartite shared-secret protocol? Solution: A One Round Protocol for Tripartite Diffie-Hellman.



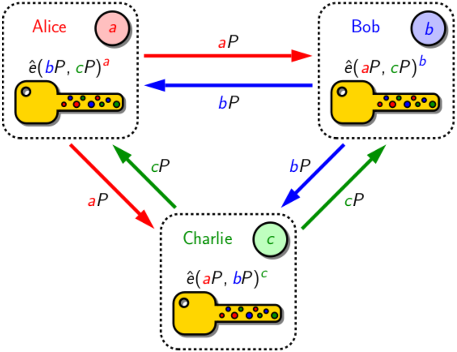
Design problem: How to establish a one-round tripartite shared-secret protocol? Solution: A One Round Protocol for Tripartite Diffie-Hellman.



Design problem: How to establish a one-round tripartite shared-secret protocol? Solution: A One Round Protocol for Tripartite Diffie-Hellman.

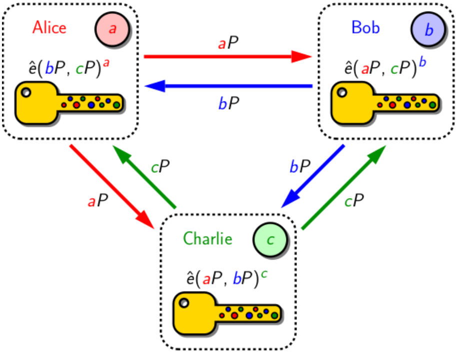


Design problem: How to establish a one-round tripartite shared-secret protocol? Solution:A One Round Protocol for Tripartite Diffie-Hellman.



- This problem remained open since the 1976 Diffie-Hellman paper,: There exists a tripartite Diffie-Hellman protocol that can be executed in just one round of public key exchanges?

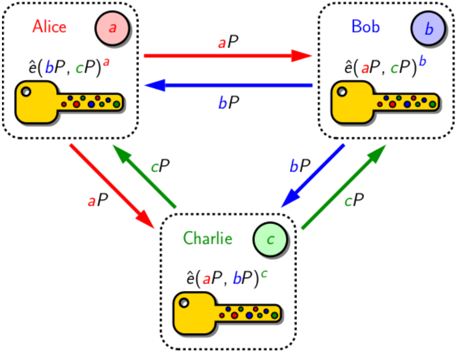
Design problem: How to establish a one-round tripartite shared-secret protocol? Solution: A One Round Protocol for Tripartite Diffie-Hellman.



- The protocol works because of,

$$\hat{e}(bP, cP)^a = \hat{e}(aP, cP)^b = \hat{e}(aP, bP)^c = \hat{e}(P, P)^{abc}$$

Design problem: How to establish a one-round tripartite shared-secret protocol? Solution: A One Round Protocol for Tripartite Diffie-Hellman.



Antoine Joux: "A One Round Protocol for Tripartite Diffie-Hellman". ANTS 2000: 385-394

Recommended key sizes (circa 2013)

Security in bits	RSA $\|\|N\|\|_2$	DL: $\mathbb{F}_p$ $\|\|p\|\|_2$	DL: $\mathbb{F}_{2^m}$ m	ECC $\|\|q\|\|_2$
80	1024	1024	1500	160
112	2048	2048	3500	224
128	3072	3072	4800	256
192	7680	7680	12500	384
256	15360	15360	25000	512

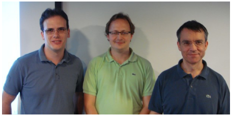
Recommended key sizes (2019)

Security in bits	RSA $\|\|N\|\|_2$	DL: $\mathbb{F}_p$ $\|\|p\|\|_2$	~~DL: $\mathbb{F}_{2^m}$~~ $\cancel{m}$	ECC $\|\|q\|\|_2$
≈ 74	1024	1024	~~1500~~	160
≈ 106	2048	2048	~~3500~~	224
128	3072	3072	~~4800*~~	256
192	7680	7680	~~12500~~	384
256	15360	15360	~~25000~~	512

Recommended key sizes (2019)

Security in bits	RSA $\|\|N\|\|_2$	DL: $\mathbb{F}_p$ $\|\|p\|\|_2$	~~DL: $\mathbb{F}_{2^m}$ m~~	ECC $\|\|q\|\|_2$
≈ 74	1024	1024	~~1500~~	160
≈ 106	2048	2048	~~3500~~	224
128	3072	3072	~~4800*~~	256
192	7680	7680	~~12500~~	384
256	15360	15360	~~25000~~	512

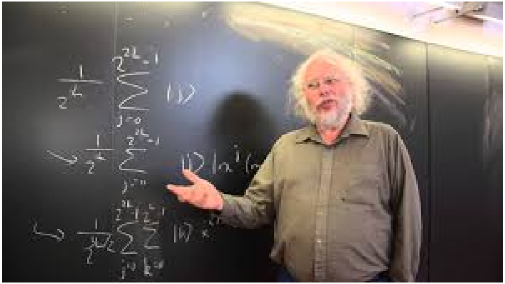
- * Nowadays, the extension $\mathbb{F}_{2^{4800}}$ is estimated to provide a security level of around 60 bits (see [Granger-Kleinjung-Zumbrägel'18], [AMOR'16]).



Barbulescu-Gaudry-Joux-Thomé: "A Heuristic Quasi-Polynomial Algorithm for Discrete Logarithm in Finite Fields of Small Characteristic". EUROCRYPT 2014: 1-16

[Apocalyptic] scenario for the next years: The arrival of large-scale quantum computers

[Apocalyptic] scenario for the next years: The arrival of large-scale quantum computers

-  

A quantum computer implementation of Peter Shor algorithm for factorization of integer numbers will produce that the computational effort for breaking elliptic-curve discrete logs would go from billions of years to hundred of hours.

[Apocalyptic] scenario for the next years: The arrival of large-scale quantum computers

-  

Along with ECC, RSA and DSA public key crypto-schemes will also go to extinction

Answers against the [Apocalyptic] scenario: Post-Quantum Cryptography (PQC)

- About two years ago, NIST launched a Post-Quantum Cryptography (PQC) standardization contest. NIST stated that

  'regardless of whether we can estimate the exact time of the arrival of the quantum computing era, we must begin now to prepare our information security systems to be able to resist quantum computing."

- The main focus of the contest is to find new PQC signature/verification and shared key establishment protocols. The latter task should be done using a scheme known as Key Encapsulation Mechanism (KEM).

Answers against the [Apocalyptic] scenario: Post-Quantum Cryptography (PQC)

- Out of 82 initial candidates only 23 made it to the second round. The surviving candidates have been classified in six categories.
- Here at Co-Crypto2019, we will be hearing a lot about,
  - Lattice-based cryptography 

Answers against the [Apocalyptic] scenario: Post-Quantum Cryptography (PQC)

- Out of 82 initial candidates only 23 made it to the second round. The surviving candidates have been classified in six categories.
- Here at Co-Crypto2019, we will be hearing a lot about,
  - Lattice-based cryptography 
  - Code-based crypto  

Answers against the [Apocalyptic] scenario: Post-Quantum Cryptography (PQC)

- Out of 82 initial candidates only 23 made it to the second round. The surviving candidates have been classified in six categories.
- Here at Co-Crypto2019, we will be hearing a lot about,
  - Lattice-based cryptography 
  - Code-based crypto  
  - Multivariate-based crypto 

Answers against the [Apocalyptic] scenario: Post-Quantum Cryptography (PQC)

- Out of 82 initial candidates only 23 made it to the second round. The surviving candidates have been classified in six categories.
- Here at Co-Crypto2019, we will be hearing a lot about,
  - Lattice-based cryptography 
  - Code-based crypto  
  - Multivariate-based crypto 
  - hash-based crypto  

Answers against the [Apocalyptic] scenario: Post-Quantum Cryptography (PQC)

- Out of 82 initial candidates only 23 made it to the second round. The surviving candidates have been classified in six categories.
- Here at Co-Crypto2019, we will be hearing a lot about,
  - Lattice-based cryptography 
  - Code-based crypto  
  - Multivariate-based crypto 
  - hash-based crypto  
  - isogeny-based crypto 

Design problem: How to construct a post-quantum Diffie-Hellman protocol?

Design problem: How to construct a post-quantum Diffie-Hellman protocol?

Secret Key

$[\mathfrak{a}] = \mathfrak{l}_1^{e_1^a} \cdots \mathfrak{l}_n^{e_n^a}$

$\{e_1^a \cdots e_n^a\}$

Public key

$E_A = E_0 \Big/ [\mathfrak{a}]$

Shared secret

$E_{AB} = E_b \Big/ [\mathfrak{a}]$

$\rightleftharpoons$

Secret Key

$[\mathfrak{b}] = \mathfrak{l}_1^{e_1^b} \cdots \mathfrak{l}_n^{e_n^b}$

$\{e_1^b \cdots e_n^b\}$

Public key

$E_B = E_0 \Big/ [\mathfrak{b}]$

Shared secret

$E_{AB} = E_a \Big/ [\mathfrak{b}]$

Castryck-Lange-Martindale-Panny-Renes: "CSIDH: An Efficient Post-Quantum Commutative Group Action". ASIACRYPT (3) 2018: 395-427

Thanks

- All pictures shown in this presentation were taken by the author in the Botero Museum and the Museo de oro at Bogotá
- Thanks are due to Dr. Jean-Luc Beuchat for designing several of the animations of this presentation