DL systems over finite fields I

ElGamal encryption and KEM-DEM framework

Tanja Lange

Eindhoven University of Technology

2MMC10 – Cryptology

Diffie–Hellman key exchange

- 1976 Diffie and Hellman introduce public-key cryptography.
- To use it, standardize group G and $g \in G$.
  Everybody knows G and g as well as how to compute in G.
- Warning #1: Many G are unsafe!
  - $G = (\mathbf{Q}, \cdot), g = 2$, $h_A = 65536$ means $a = 16$.
    In general, just check bitlength.
  - $G = (\mathbf{F}_p, +)$, i.e., A sends $h_A \equiv ag \bmod p$.
    Can recover a using XGCD.
- Diffie and Hellman suggested $G = (\mathbf{F}_p^*, \cdot)$ with g a primitive element, i.e., a generator of the whole group.
- Used in practice $G \subset (\mathbf{F}_p^*, \cdot)$ with g an element of large prime order.
- Miller and Koblitz suggested $G = E(\mathbf{F}_p, +)$, i,e., points on an elliptic curve over a finite field with addition of points.
- Used in practice $G \subset E(\mathbf{F}_p, +)$, i,e., prime-order subgroup of points on an elliptic curve over a finite field with addition of points.
  We have seen how to compute $+$ on different curve shapes,
  will now study security.

ElGamal encryption

KeyGen:

1. Pick random $0 < a < |G|$.
2. Compute $h_A = g^a$.
3. Output public key h_A, private key a.

Encryption:

1. Pick random k, compute $r = g^k$.
2. Encrypt $m \in G$ as $C = h_A^k \cdot m$.
3. Send (r, C).

Decryption:

1. Compute $m = C/(r^a)$

ElGamal encryption

KeyGen:

1. Pick random $0 < a < |G|$.
2. Compute $h_A = g^a$.
3. Output public key h_A, private key a.

Encryption:

1. Pick random k, compute $r = g^k$.
2. Encrypt $m \in G$ as $C = h_A^k \cdot m$.
3. Send (r, C).

Decryption:

1. Compute $m = C/(r^a) = (g^a)^k \cdot m/g^{ak}$.

- Positives:
  - Randomized DL-based encryption.
  - Is re-randomizable: $(rg^{k'}, Ch_A^{k'})$ decrypts to same m as (r, C).
  - Is homomorphic.

ElGamal encryption

KeyGen:

1. Pick random $0 < a < |G|$.
2. Compute $h_A = g^a$.
3. Output public key h_A, private key a.

Encryption:

1. Pick random k, compute $r = g^k$.
2. Encrypt $m \in G$ as $C = h_A^k \cdot m$.
3. Send (r, C).

Decryption:

1. Compute $m = C/(r^a) = (g^a)^k \cdot m/g^{ak}$.

- Positives:
  - Randomized DL-based encryption.
  - Is re-randomizable: $(rg^{k'}, Ch_A^{k'})$ decrypts to same m as (r, C).
  - Is homomorphic.
- Downsides:
  - Requires $m \in G$.
  - Is homomorphic. Not OW-CCA II secure.
  - Typically all we want is to share a symmetric key.
  - Beware of subgroups!

Security reduction – IND-CPA as hard as DDHP

Attacker goal

Break indistinguishability (IND),
i.e., learn which of two attacker-chosen messages m_0, m_1 was encrypted in $C = \mathrm{Enc}_{\mathrm{pk}}(m_i)$ (beyond 50% chance of guessing.)

Security reduction – IND-CPA as hard as DDHP

Attacker goal

Break indistinguishability (IND),
i.e., learn which of two attacker-chosen messages m_0, m_1 was encrypted in $C = \mathrm{Enc}_{\mathrm{pk}}(m_i)$ (beyond 50% chance of guessing.)

We get $(r, C) = (g^k, h_A^k m_i)$, need to decide if $i = 0$ or $i = 1$.

Security reduction – IND-CPA as hard as DDHP

Attacker goal

Break indistinguishability (IND),
i.e., learn which of two attacker-chosen messages m_0, m_1 was encrypted in $C = \mathrm{Enc}_{\mathrm{pk}}(m_i)$ (beyond 50% chance of guessing.)

We get $(r, C) = (g^k, h_A^k m_i)$, need to decide if $i = 0$ or $i = 1$.

Can compute C/m_0.

Security reduction – IND-CPA as hard as DDHP

Attacker goal

Break indistinguishability (IND),
i.e., learn which of two attacker-chosen messages m_0, m_1 was encrypted in $C = \mathrm{Enc}_{\mathrm{pk}}(m_i)$ (beyond 50% chance of guessing.)

We get $(r, C) = (g^k, h_A^k m_i)$, need to decide if $i = 0$ or $i = 1$.

Can compute C/m_0.

If $i = 0$: $(h_A, r, C/m_0) = (g^a, g^k, g^{ak})$ is a valid DH triple.

Security reduction – IND-CPA as hard as DDHP

Attacker goal

Break indistinguishability (IND),
i.e., learn which of two attacker-chosen messages m_0, m_1 was encrypted in $C = \text{Enc}_{\text{pk}}(m_i)$ (beyond 50% chance of guessing.)

We get $(r, C) = (g^k, h_A^k m_i)$, need to decide if $i = 0$ or $i = 1$.

Can compute C/m_0.

If $i = 0$: $(h_A, r, C/m_0) = (g^a, g^k, g^{ak})$ is a valid DH triple.
If $i = 1$: $(h_A, r, C/m_0) = (g^a, g^k, g^{ak} m_1/m_0)$ is not a valid DH triple.

Security reduction – IND-CPA as hard as DDHP

Attacker goal

Break indistinguishability (IND),
i.e., learn which of two attacker-chosen messages m_0, m_1 was encrypted in $C = \text{Enc}_{\text{pk}}(m_i)$ (beyond 50% chance of guessing.)

We get $(r, C) = (g^k, h_A^k m_i)$, need to decide if $i = 0$ or $i = 1$.

Can compute C/m_0.

If $i = 0$: $(h_A, r, C/m_0) = (g^a, g^k, g^{ak})$ is a valid DH triple.

If $i = 1$: $(h_A, r, C/m_0) = (g^a, g^k, g^{ak} m_1/m_0)$ is not a valid DH triple.

If we can solve DDHP we can break IND-CPA.

Security reduction – IND-CPA as hard as DDHP

Attacker goal

Break indistinguishability (IND),
i.e., learn which of two attacker-chosen messages m_0, m_1 was encrypted in $C = \text{Enc}_{\text{pk}}(m_i)$ (beyond 50% chance of guessing.)

We get $(r, C) = (g^k, h_A^k m_i)$, need to decide if $i = 0$ or $i = 1$.

Can compute C/m_0.

If $i = 0$: $(h_A, r, C/m_0) = (g^a, g^k, g^{ak})$ is a valid DH triple.
If $i = 1$: $(h_A, r, C/m_0) = (g^a, g^k, g^{ak} m_1/m_0)$ is not a valid DH triple.
If we can solve DDHP we can break IND-CPA.

Use IND-CPA attacker A to break DDHP:
Given (g^a, g^b, g^c) figure out whether $c = ab$.
Give $h_A = g^a$ to A as public key.
Upon input of m_0, m_1, pick random $i \in \{0, 1\}$, send $(r, C) = (g^b, g^c m_i)$.

Security reduction – IND-CPA as hard as DDHP

Attacker goal
Break indistinguishability (IND),
i.e., learn which of two attacker-chosen messages m_0, m_1 was encrypted in $C = \mathrm{Enc}_{\mathrm{pk}}(m_i)$ (beyond 50% chance of guessing.)

We get $(r, C) = (g^k, h_A^k m_i)$, need to decide if $i = 0$ or $i = 1$.

Can compute C/m_0.

If $i = 0$: $(h_A, r, C/m_0) = (g^a, g^k, g^{ak})$ is a valid DH triple.
If $i = 1$: $(h_A, r, C/m_0) = (g^a, g^k, g^{ak} m_1/m_0)$ is not a valid DH triple.
If we can solve DDHP we can break IND-CPA.

Use IND-CPA attacker A to break DDHP:
Given (g^a, g^b, g^c) figure out whether $c = ab$.
Give $h_A = g^a$ to A as public key.
Upon input of m_0, m_1, pick random $i \in \{0, 1\}$, send $(r, C) = (g^b, g^c m_i)$.
If A's reply matches i, output "valid triple"; else output "'not valid".

Security reduction – IND-CPA as hard as DDHP

Attacker goal

Break indistinguishability (IND),
i.e., learn which of two attacker-chosen messages m_0, m_1 was encrypted in $C = \mathrm{Enc}_{\mathrm{pk}}(m_i)$ (beyond 50% chance of guessing.)

We get $(r, C) = (g^k, h_A^k m_i)$, need to decide if $i = 0$ or $i = 1$.

Can compute C/m_0.

If $i = 0$: $(h_A, r, C/m_0) = (g^a, g^k, g^{ak})$ is a valid DH triple.

If $i = 1$: $(h_A, r, C/m_0) = (g^a, g^k, g^{ak} m_1/m_0)$ is not a valid DH triple.

If we can solve DDHP we can break IND-CPA.

Use IND-CPA attacker A to break DDHP:
Given (g^a, g^b, g^c) figure out whether $c = ab$.
Give $h_A = g^a$ to A as public key.
Upon input of m_0, m_1, pick random $i \in \{0, 1\}$, send $(r, C) = (g^b, g^c m_i)$.

If A's reply matches i, output "valid triple"; else output "'not valid".

For valid triple, inputs match IND-CPA game. Else A must guess.

Security reduction – IND-CPA as hard as DDHP

Attacker goal

Break indistinguishability (IND),
i.e., learn which of two attacker-chosen messages m_0, m_1 was encrypted in $C = \mathrm{Enc}_{\mathrm{pk}}(m_i)$ (beyond 50% chance of guessing.)

We get $(r, C) = (g^k, h_A^k m_i)$, need to decide if $i = 0$ or $i = 1$.

Can compute C/m_0.

If $i = 0$: $(h_A, r, C/m_0) = (g^a, g^k, g^{ak})$ is a valid DH triple.

If $i = 1$: $(h_A, r, C/m_0) = (g^a, g^k, g^{ak} m_1/m_0)$ is not a valid DH triple.

If we can solve DDHP we can break IND-CPA.

Use IND-CPA attacker A to break DDHP:
Given (g^a, g^b, g^c) figure out whether $c = ab$.
Give $h_A = g^a$ to A as public key.
Upon input of m_0, m_1, pick random $i \in \{0, 1\}$, send $(r, C) = (g^b, g^c m_i)$.

If A's reply matches i, output "valid triple"; else output "'not valid".

For valid triple, inputs match IND-CPA game. Else A must guess.

If A has advantage ε at IND-CPA we get advantage $\varepsilon/2$ at DDHP.

KEM–DEM framework

Formalize idea: use public-key system to transport symmetric key.

Data-encapsulation mechanism (DEM).

This is the regular symmetric-key authenticated encryption.

Key-encapsulation mechanism (KEM)

1. KeyGen: generate a public-key private-key pair.
2. Encapsulation: take public key, produce ciphertext and shared key.
3. Decapsulation: take private key and a ciphertext, produce shared key.

KEM–DEM framework

Formalize idea: use public-key system to transport symmetric key.

Data-encapsulation mechanism (DEM).

This is the regular symmetric-key authenticated encryption.

Key-encapsulation mechanism (KEM)

1. KeyGen: generate a public-key private-key pair.
2. Encapsulation: take public key, produce ciphertext and shared key.
3. Decapsulation: take private key and a ciphertext, produce shared key.

This formalizes nicely the semi-static DH system; here the $\mathbf{F}_p^*$ version:

Encapsulation:

1. Pick random $0 < r < |G|$.
2. Compute $h = g^r$.
3. Compute $K = \mathrm{KDF}(h_A^r)$.
4. Output (h, K).

Decapsulation:

1. Compute $K' = \mathrm{KDF}(h^a)$.
2. Output K'.

KEM–DEM framework

Formalize idea: use public-key system to transport symmetric key.

Data-encapsulation mechanism (DEM).
This is the regular symmetric-key authenticated encryption.

Key-encapsulation mechanism (KEM)

1. KeyGen: generate a public-key private-key pair.
2. Encapsulation: take public key, produce ciphertext and shared key.
3. Decapsulation: take private key and a ciphertext, produce shared key.

This formalizes nicely the semi-static DH system; here the $\mathbf{F}_p^*$ version:
Encapsulation:

1. Pick random $0 < r < |G|$.
2. Compute $h = g^r$.
3. Compute $K = \text{KDF}(h_A^r)$.
4. Output (h, K).

Decapsulation:

1. Compute $K' = \text{KDF}(h^a)$.
2. Output K'. Note $K' = \text{KDF}(h^a) = \text{KDF}((g^r)^a) = \text{KDF}(h_A^r) = K$.