UC SANTA CRUZ

Math 134: Cryptography

Lecture 12: the ElGamal Public-Key Cryptosystem

Eoin Mackall

February 5th, 2025

University of California, Santa Cruz

Announcements

- Midterm #1 results
- Homework #3 posted to canvas
- Final Project Rubric posted to canvas

Last time

Last time

Last week, we introduced two important concepts: *primitive roots* and *discrete logarithms*.

Last time

Last week, we introduced two important concepts: *primitive roots* and *discrete logarithms*.

Let $n > 1$ be an integer. Recall that we say that $a \in \mathbb{Z}$ is a primitive root modulo n if both $\gcd(a, n) = 1$ and

$$a^k \not\equiv 1 \pmod{n} \quad \text{for all } 1 \leq k < \phi(n).$$

Last time

Last week, we introduced two important concepts: *primitive roots* and *discrete logarithms*.

Let $n > 1$ be an integer. Recall that we say that $a \in \mathbb{Z}$ is a primitive root modulo n if both $\gcd(a, n) = 1$ and

$$a^k \not\equiv 1 \pmod{n} \quad \text{for all } 1 \leq k < \phi(n).$$

Equivalently, since $a^{\phi(n)} \equiv 1 \pmod{n}$ by Euler's theorem, a is a primitive root modulo n if $\operatorname{ord}_n(a) = \phi(n)$.

Last time

Example

Let $n = 101$ and $a = 2$. We'll check that a is a primitive root mod n.

Last time

Example

Let $n = 101$ and $a = 2$. We'll check that a is a primitive root mod n. Note that since n is prime we have $\phi(n) = 100 = 2^2 \cdot 5^2$.

Example

Let $n = 101$ and $a = 2$. We'll check that a is a primitive root mod n. Note that since n is prime we have $\phi(n) = 100 = 2^2 \cdot 5^2$.

Hence, it suffices to check that none of $2^2, 2^4, 2^5, 2^{10}, 2^{20}, 2^{25}, 2^{50}$, are congruent to 1 mod 101.

Last time

Example

Let $n = 101$ and $a = 2$. We'll check that a is a primitive root mod n. Note that since n is prime we have $\phi(n) = 100 = 2^2 \cdot 5^2$.

Hence, it suffices to check that none of $2^2, 2^4, 2^5, 2^{10}, 2^{20}, 2^{25}, 2^{50}$, are congruent to 1 mod 101.

$$2^2 \equiv 4 \pmod{101}$$

Last time

Example

Let $n = 101$ and $a = 2$. We'll check that a is a primitive root mod n. Note that since n is prime we have $\phi(n) = 100 = 2^2 \cdot 5^2$.

Hence, it suffices to check that none of $2^2, 2^4, 2^5, 2^{10}, 2^{20}, 2^{25}, 2^{50}$, are congruent to 1 mod 101.

$$2^2 \equiv 4 \pmod{101}$$
$$2^4 \equiv 16 \pmod{101}$$

Last time

Example

Let $n = 101$ and $a = 2$. We'll check that a is a primitive root mod n. Note that since n is prime we have $\phi(n) = 100 = 2^2 \cdot 5^2$.

Hence, it suffices to check that none of $2^2, 2^4, 2^5, 2^{10}, 2^{20}, 2^{25}, 2^{50}$, are congruent to 1 mod 101.

$$\begin{aligned} 2^2 &\equiv 4 \pmod{101} \\ 2^4 &\equiv 16 \pmod{101} \\ 2^5 &\equiv 32 \pmod{101} \end{aligned}$$

Last time

Example

Let $n = 101$ and $a = 2$. We'll check that a is a primitive root mod n. Note that since n is prime we have $\phi(n) = 100 = 2^2 \cdot 5^2$.

Hence, it suffices to check that none of $2^2, 2^4, 2^5, 2^{10}, 2^{20}, 2^{25}, 2^{50}$, are congruent to 1 mod 101.

$$2^2 \equiv 4 \pmod{101}$$
$$2^4 \equiv 16 \pmod{101}$$
$$2^5 \equiv 32 \pmod{101}$$
$$2^{10} \equiv 1024 \equiv 14 \pmod{101}$$

Last time

Example

Let $n = 101$ and $a = 2$. We'll check that a is a primitive root mod n. Note that since n is prime we have $\phi(n) = 100 = 2^2 \cdot 5^2$.

Hence, it suffices to check that none of $2^2, 2^4, 2^5, 2^{10}, 2^{20}, 2^{25}, 2^{50}$, are congruent to 1 mod 101.

$$\begin{aligned}
2^2 &\equiv 4 \pmod{101} \\
2^4 &\equiv 16 \pmod{101} \\
2^5 &\equiv 32 \pmod{101} \\
2^{10} &\equiv 1024 \equiv 14 \pmod{101} \\
2^{20} &\equiv (14)^2 \equiv 196 \equiv 95 \pmod{101}
\end{aligned}$$

Example

Let $n = 101$ and $a = 2$. We'll check that a is a primitive root mod n. Note that since n is prime we have $\phi(n) = 100 = 2^2 \cdot 5^2$.

Hence, it suffices to check that none of $2^2, 2^4, 2^5, 2^{10}, 2^{20}, 2^{25}, 2^{50}$, are congruent to 1 mod 101.

$$\begin{aligned}
2^2 &\equiv 4 \pmod{101} \\
2^4 &\equiv 16 \pmod{101} \\
2^5 &\equiv 32 \pmod{101} \\
2^{10} &\equiv 1024 \equiv 14 \pmod{101} \\
2^{20} &\equiv (14)^2 \equiv 196 \equiv 95 \pmod{101} \\
2^{25} &\equiv (95) \cdot 32 \equiv (-6) \cdot 32 \equiv -192 \equiv 10 \pmod{101}
\end{aligned}$$

Example

Let $n = 101$ and $a = 2$. We'll check that a is a primitive root mod n. Note that since n is prime we have $\phi(n) = 100 = 2^2 \cdot 5^2$.

Hence, it suffices to check that none of $2^2, 2^4, 2^5, 2^{10}, 2^{20}, 2^{25}, 2^{50}$, are congruent to 1 mod 101.

$$\begin{aligned}
2^2 &\equiv 4 \pmod{101} \\
2^4 &\equiv 16 \pmod{101} \\
2^5 &\equiv 32 \pmod{101} \\
2^{10} &\equiv 1024 \equiv 14 \pmod{101} \\
2^{20} &\equiv (14)^2 \equiv 196 \equiv 95 \pmod{101} \\
2^{25} &\equiv (95) \cdot 32 \equiv (-6) \cdot 32 \equiv -192 \equiv 10 \pmod{101} \\
2^{50} &\equiv (10)^2 \equiv 100 \equiv -1 \pmod{101}.
\end{aligned}$$

Last time

Example

Let $n = 101$ and $a = 2$. We'll check that a is a primitive root mod n. Note that since n is prime we have $\phi(n) = 100 = 2^2 \cdot 5^2$.

Hence, it suffices to check that none of $2^2, 2^4, 2^5, 2^{10}, 2^{20}, 2^{25}, 2^{50}$, are congruent to 1 mod 101.

$$\begin{aligned}
2^2 &\equiv 4 \pmod{101} \\
2^4 &\equiv 16 \pmod{101} \\
2^5 &\equiv 32 \pmod{101} \\
2^{10} &\equiv 1024 \equiv 14 \pmod{101} \\
2^{20} &\equiv (14)^2 \equiv 196 \equiv 95 \pmod{101} \\
2^{25} &\equiv (95) \cdot 32 \equiv (-6) \cdot 32 \equiv -192 \equiv 10 \pmod{101} \\
2^{50} &\equiv (10)^2 \equiv 100 \equiv -1 \pmod{101}.
\end{aligned}$$

Hence $\mathrm{ord}_{101}(2) = \phi(101) = 100$.

Last time

Let $n > 1$ be an integer and suppose that there exists a primitive root a for n.

Last time

Let $n > 1$ be an integer and suppose that there exists a primitive root a for n.

The *discrete logarithm* or *the index* of an integer *b to base a modulo n* is the unique integer k with $1 \leq k \leq \phi(n)$ such that $a^k \equiv b \pmod{n}$.

Last time

Let $n > 1$ be an integer and suppose that there exists a primitive root a for n.

The *discrete logarithm* or *the index* of an integer *b to base a modulo n* is the unique integer k with $1 \leq k \leq \phi(n)$ such that $a^k \equiv b \pmod{n}$.

Example

Let $n = 101$. Then $2^{57} \equiv 74 \pmod{101}$ and $2^{33} \equiv 35 \pmod{101}$.

Last time

Let $n > 1$ be an integer and suppose that there exists a primitive root a for n.

The *discrete logarithm* or *the index* of an integer *b to base a modulo n* is the unique integer k with $1 \leq k \leq \phi(n)$ such that $a^k \equiv b \pmod{n}$.

Example

Let $n = 101$. Then $2^{57} \equiv 74 \pmod{101}$ and $2^{33} \equiv 35 \pmod{101}$.

So the index of 74 to base 2 modulo 101 is $\mathrm{ind}_2(74) = 57$.

Last time

Let $n > 1$ be an integer and suppose that there exists a primitive root a for n.

The *discrete logarithm* or *the index* of an integer *b to base a modulo n* is the unique integer k with $1 \leq k \leq \phi(n)$ such that $a^k \equiv b \pmod{n}$.

Example

Let $n = 101$. Then $2^{57} \equiv 74 \pmod{101}$ and $2^{33} \equiv 35 \pmod{101}$.

So the index of 74 to base 2 modulo 101 is $\mathrm{ind}_2(74) = 57$.

The index of 35 to base 2 modulo 101 is $\mathrm{ind}_2(35) = 33$.

Table of contents

ElGamal Public-Key Cryptosystem

The ElGamal encryption algorithm

Alice wants to send a message to Bob. Bob has indicated that he wants to use the ElGamal public-key cryptosystem. What is the process for encryption?

The ElGamal encryption algorithm

Alice wants to send a message to Bob. Bob has indicated that he wants to use the ElGamal public-key cryptosystem. What is the process for encryption?

1. Bob chooses a large prime p and a primitive root α for p.

The ElGamal encryption algorithm

Alice wants to send a message to Bob. Bob has indicated that he wants to use the ElGamal public-key cryptosystem. What is the process for encryption?

1. Bob chooses a large prime p and a primitive root α for p.
2. Bob chooses a secret integer b with $1 < b < p - 1$ and computes $\beta \equiv \alpha^b \pmod{p}$.

The ElGamal encryption algorithm

Alice wants to send a message to Bob. Bob has indicated that he wants to use the ElGamal public-key cryptosystem. What is the process for encryption?

1. Bob chooses a large prime p and a primitive root α for p.
2. Bob chooses a secret integer b with $1 < b < p - 1$ and computes $\beta \equiv \alpha^b \pmod{p}$.
3. Bob sends Alice the triple (p, α, β). This is Bob's *public* key.

The ElGamal encryption algorithm

Alice wants to send a message to Bob. Bob has indicated that he wants to use the ElGamal public-key cryptosystem. What is the process for encryption?

1. Bob chooses a large prime p and a primitive root α for p.
2. Bob chooses a secret integer b with $1 < b < p - 1$ and computes $\beta \equiv \alpha^b \pmod{p}$.
3. Bob sends Alice the triple (p, α, β). This is Bob's *public* key.
4. Alice will send her message in the form of an integer $m \in \mathbb{Z}$. If $m > p$, then Alice breaks $m = m_1 m_2 m_3 ...$ into blocks m_i of sizes less than p.

The ElGamal encryption algorithm

Alice wants to send a message to Bob. Bob has indicated that he wants to use the ElGamal public-key cryptosystem. What is the process for encryption?

1. Bob chooses a large prime p and a primitive root α for p.
2. Bob chooses a secret integer b with $1 < b < p - 1$ and computes $\beta \equiv \alpha^b \pmod{p}$.
3. Bob sends Alice the triple (p, α, β). This is Bob's *public* key.
4. Alice will send her message in the form of an integer $m \in \mathbb{Z}$. If $m > p$, then Alice breaks $m = m_1 m_2 m_3 \ldots$ into blocks m_i of sizes less than p.
5. For each block m_i, Alice will generate a random integer k_i and she will compute $r_i \equiv \alpha^{k_i} \pmod{p}$ and $t_i \equiv \beta^{k_i} \cdot m_i \pmod{p}$.

The ElGamal encryption algorithm

Alice wants to send a message to Bob. Bob has indicated that he wants to use the ElGamal public-key cryptosystem. What is the process for encryption?

1. Bob chooses a large prime p and a primitive root α for p.
2. Bob chooses a secret integer b with $1 < b < p - 1$ and computes $\beta \equiv \alpha^b \pmod{p}$.
3. Bob sends Alice the triple (p, α, β). This is Bob's *public* key.
4. Alice will send her message in the form of an integer $m \in \mathbb{Z}$. If $m > p$, then Alice breaks $m = m_1 m_2 m_3 \ldots$ into blocks m_i of sizes less than p.
5. For each block m_i, Alice will generate a random integer k_i and she will compute $r_i \equiv \alpha^{k_i} \pmod{p}$ and $t_i \equiv \beta^{k_i} \cdot m_i \pmod{p}$.
6. Alice then sends all of the pairs (r_i, t_i) to Bob.

Example

Suppose Bob chooses $p = 101$ and $\alpha = 2$. Bob also randomly selects a secret number $b = 76$, and computes $\beta = 2^{76} \equiv 81 \pmod{101}$.

Example

Suppose Bob chooses $p = 101$ and $\alpha = 2$. Bob also randomly selects a secret number $b = 76$, and computes $\beta = 2^{76} \equiv 81 \pmod{101}$.

Bob asks Alice to send her favorite number to Bob, to test his new encryption algorithm. He sends Alice the triple
$(p, \alpha, \beta) = (101, 2, 81)$.

The ElGamal encryption algorithm

Example

Suppose Bob chooses $p = 101$ and $\alpha = 2$. Bob also randomly selects a secret number $b = 76$, and computes $\beta = 2^{76} \equiv 81 \pmod{101}$.

Bob asks Alice to send her favorite number to Bob, to test his new encryption algorithm. He sends Alice the triple $(p, \alpha, \beta) = (101, 2, 81)$.

Alice wants to send 8 to Bob, which is less than p. She then randomly generates an integer $k = 12$.

Example

Suppose Bob chooses $p = 101$ and $\alpha = 2$. Bob also randomly selects a secret number $b = 76$, and computes $\beta = 2^{76} \equiv 81 \pmod{101}$.

Bob asks Alice to send her favorite number to Bob, to test his new encryption algorithm. He sends Alice the triple $(p, \alpha, \beta) = (101, 2, 81)$.

Alice wants to send 8 to Bob, which is less than p. She then randomly generates an integer $k = 12$.

Alice computes both

$$r \equiv \alpha^k \equiv 2^{12} \pmod{101} \quad \text{and} \quad t \equiv \beta^k \cdot m \equiv (81)^{12} \cdot 8 \pmod{101}.$$

She finds $(r, t) = (56, 44)$ and sends this pair to Bob.

The ElGamal decryption algorithm

Bob receives a message (r, t) from Alice. What is the process for decryption?

The ElGamal decryption algorithm

Bob receives a message (r, t) from Alice. What is the process for decryption?

1. Bob computes a multiplicative inverse s to r modulo p (i.e. such that $sr \equiv 1 \pmod{p}$) using, for example, the Euclidean Algorithm.

The ElGamal decryption algorithm

Bob receives a message (r, t) from Alice. What is the process for decryption?

1. Bob computes a multiplicative inverse s to r modulo p (i.e. such that $sr \equiv 1 \pmod{p}$) using, for example, the Euclidean Algorithm.
2. Bob uses the secret integer b that he chose earlier to compute the message as $m \equiv ts^b \pmod{p}$.

The ElGamal decryption algorithm

Bob receives a message (r, t) from Alice. What is the process for decryption?

1. Bob computes a multiplicative inverse s to r modulo p (i.e. such that $sr \equiv 1 \pmod{p}$) using, for example, the Euclidean Algorithm.
2. Bob uses the secret integer b that he chose earlier to compute the message as $m \equiv ts^b \pmod{p}$.
3. This works since

$$\begin{aligned} ts^b \equiv (\beta^k m)s^b \equiv (\alpha^b)^k ms^b &\equiv (\alpha^k)^b ms^b \\ &\equiv r^b ms^b \equiv (rs)^b m \equiv (1)^b m \equiv m \pmod{p}. \end{aligned}$$

The ElGamal decryption algorithm

Bob receives a message (r, t) from Alice. What is the process for decryption?

1. Bob computes a multiplicative inverse s to r modulo p (i.e. such that $sr \equiv 1 \pmod p$) using, for example, the Euclidean Algorithm.
2. Bob uses the secret integer b that he chose earlier to compute the message as $m \equiv ts^b \pmod p$.
3. This works since

$$\begin{aligned} ts^b \equiv (\beta^k m)s^b \equiv (\alpha^b)^k ms^b &\equiv (\alpha^k)^b ms^b \\ &\equiv r^b ms^b \equiv (rs)^b m \equiv (1)^b m \equiv m \pmod p. \end{aligned}$$

Notation

We can write $s = r^{-1}$ (keeping in mind that $s \neq 1/r$ since we are working modulo p) and similarly $s^b = r^{-b}$. Then Bob computes m by $tr^{-b} \equiv m \pmod p$.

The ElGamal decryption algorithm

Example

Suppose Bob has selected $p = 101$, $\alpha = 2$, and $b = 76$ as before. He receives the pair $(r, t) = (56, 44)$ from Alice.

The ElGamal decryption algorithm

Example

Suppose Bob has selected $p = 101$, $\alpha = 2$, and $b = 76$ as before. He receives the pair $(r, t) = (56, 44)$ from Alice.

Bob uses the integer $r = 56$ to find a multiplicative inverse r^{-1} of r mod 101 using the Euclidean Algorithm. He can use $r^{-1} = -9$ since

$$(-9) \cdot 56 + (5) \cdot 101 = -504 + 505 = 1.$$

The ElGamal decryption algorithm

Example

Suppose Bob has selected $p = 101$, $\alpha = 2$, and $b = 76$ as before. He receives the pair $(r, t) = (56, 44)$ from Alice.

Bob uses the integer $r = 56$ to find a multiplicative inverse r^{-1} of r mod 101 using the Euclidean Algorithm. He can use $r^{-1} = -9$ since

$$(-9) \cdot 56 + (5) \cdot 101 = -504 + 505 = 1.$$

Bob then finds

$$tr^{-b} \equiv 44 \cdot (-9)^{76} \equiv 8 \pmod{101}.$$

So Alice must have sent the message $m = 8$ to Bob.

Security of ElGamal Encryption

The computational Diffie-Hellman problem

The security of the ElGamal public-key cryptosystem is based on the following problem:

The Computational Diffie-Hellman Problem

The computational Diffie-Hellman problem

The security of the ElGamal public-key cryptosystem is based on the following problem:

The Computational Diffie-Hellman Problem

Let p be an odd prime. Let α be a primitive root modulo p. Given the integers $\alpha^x \pmod{p}$ and $\alpha^y \pmod{p}$, find $\alpha^{xy} \pmod{p}$.

The computational Diffie-Hellman problem

The security of the ElGamal public-key cryptosystem is based on the following problem:

The Computational Diffie-Hellman Problem

Let p be an odd prime. Let α be a primitive root modulo p. Given the integers $\alpha^x \pmod{p}$ and $\alpha^y \pmod{p}$, find $\alpha^{xy} \pmod{p}$.

Indeed, suppose a third party Eve can solve the Computational Diffie-Hellman problem.

The computational Diffie-Hellman problem

The security of the ElGamal public-key cryptosystem is based on the following problem:

The Computational Diffie-Hellman Problem

Let p be an odd prime. Let α be a primitive root modulo p. Given the integers $\alpha^x \pmod{p}$ and $\alpha^y \pmod{p}$, find $\alpha^{xy} \pmod{p}$.

Indeed, suppose a third party Eve can solve the Computational Diffie-Hellman problem.

If Eve sees Bob's public key $(p, \alpha, \beta \equiv \alpha^b \pmod{p})$ and Alice's ciphertext $(r \equiv \alpha^k \pmod{p}, t \equiv \alpha^{bk} \cdot m \pmod{p})$, then Eve knows both $\alpha^b \pmod{b}$ and $\alpha^k \pmod{b}$.

The computational Diffie-Hellman problem

The security of the ElGamal public-key cryptosystem is based on the following problem:

The Computational Diffie-Hellman Problem

Let p be an odd prime. Let α be a primitive root modulo p. Given the integers $\alpha^x \pmod{p}$ and $\alpha^y \pmod{p}$, find $\alpha^{xy} \pmod{p}$.

Indeed, suppose a third party Eve can solve the Computational Diffie-Hellman problem.

If Eve sees Bob's public key $(p, \alpha, \beta \equiv \alpha^b \pmod{p})$ and Alice's ciphertext $(r \equiv \alpha^k \pmod{p}, t \equiv \alpha^{bk} \cdot m \pmod{p})$, then Eve knows both $\alpha^b \pmod{b}$ and $\alpha^k \pmod{b}$.

So Eve may calculate $\alpha^{bk} \pmod{p}$ and quickly find $\alpha^{-bk} \pmod{p}$ allowing her to recover $m \equiv \alpha^{-bk} \cdot t \pmod{p}$.

The computational Diffie-Hellman problem

The security of the ElGamal public-key cryptosystem is based on the following problem:

The Computational Diffie-Hellman Problem

Let p be an odd prime. Let α be a primitive root modulo p. Given the integers $\alpha^x \pmod{p}$ and $\alpha^y \pmod{p}$, find $\alpha^{xy} \pmod{p}$.

Conversely, assume that there is a method for converting ElGamal ciphertext (r, t) gotten from a public key (p, α, β) into the plaintext m.

The computational Diffie-Hellman problem

The security of the ElGamal public-key cryptosystem is based on the following problem:

The Computational Diffie-Hellman Problem

Let p be an odd prime. Let α be a primitive root modulo p. Given the integers $\alpha^x \pmod{p}$ and $\alpha^y \pmod{p}$, find $\alpha^{xy} \pmod{p}$.

Conversely, assume that there is a method for converting ElGamal ciphertext (r, t) gotten from a public key (p, α, β) into the plaintext m.

We could take $\beta \equiv \alpha^x \pmod{p}$. Then, by taking as input ciphertext $(r \equiv \alpha^y \pmod{p}, t \not\equiv 0 \pmod{p})$, we get $m \equiv t \cdot \alpha^{-xy} \pmod{p}$.

The computational Diffie-Hellman problem

The security of the ElGamal public-key cryptosystem is based on the following problem:

The Computational Diffie-Hellman Problem

Let p be an odd prime. Let α be a primitive root modulo p. Given the integers $\alpha^x \pmod{p}$ and $\alpha^y \pmod{p}$, find $\alpha^{xy} \pmod{p}$.

Conversely, assume that there is a method for converting ElGamal ciphertext (r, t) gotten from a public key (p, α, β) into the plaintext m.

We could take $\beta \equiv \alpha^x \pmod{p}$. Then, by taking as input ciphertext $(r \equiv \alpha^y \pmod{p}, t \not\equiv 0 \pmod{p})$, we get $m \equiv t \cdot \alpha^{-xy} \pmod{p}$.

Multiplying by $t^{-1} \pmod{p}$ allows us to find find $\alpha^{-xy} \pmod{p}$ and so also $\alpha^{xy} \pmod{p}$.

The Discrete Logarithm Problem

Often the security of ElGamal is related to the following (difficult) problem:

The Discrete Logarithm Problem

The Discrete Logarithm Problem

Often the security of ElGamal is related to the following (difficult) problem:

The Discrete Logarithm Problem

Let p be an odd prime. Let α be a primitive root modulo p. Given an integer $1 < \beta < n$, determine the unique integer $1 \leq x < \phi(n)$ such that $\beta \equiv \alpha^x \pmod{p}$.

The Discrete Logarithm Problem

Often the security of ElGamal is related to the following (difficult) problem:

The Discrete Logarithm Problem

Let p be an odd prime. Let α be a primitive root modulo p. Given an integer $1 < \beta < n$, determine the unique integer $1 \leq x < \phi(n)$ such that $\beta \equiv \alpha^x \pmod{p}$.

A solution to the Discrete Logarithm Problem gives a solution to the Computational Diffie-Hellman Problem.

The Discrete Logarithm Problem

Often the security of ElGamal is related to the following (difficult) problem:

The Discrete Logarithm Problem

Let p be an odd prime. Let α be a primitive root modulo p. Given an integer $1 < \beta < n$, determine the unique integer $1 \leq x < \phi(n)$ such that $\beta \equiv \alpha^x \pmod{p}$.

A solution to the Discrete Logarithm Problem gives a solution to the Computational Diffie-Hellman Problem.

Proof.

The Discrete Logarithm Problem

Often the security of ElGamal is related to the following (difficult) problem:

The Discrete Logarithm Problem

Let p be an odd prime. Let α be a primitive root modulo p. Given an integer $1 < \beta < n$, determine the unique integer $1 \le x < \phi(n)$ such that $\beta \equiv \alpha^x \pmod{p}$.

A solution to the Discrete Logarithm Problem gives a solution to the Computational Diffie-Hellman Problem.

Proof.

Suppose that you can solve the Discrete Logarithm Problem.

The Discrete Logarithm Problem

Often the security of ElGamal is related to the following (difficult) problem:

The Discrete Logarithm Problem

Let p be an odd prime. Let α be a primitive root modulo p. Given an integer $1 < \beta < n$, determine the unique integer $1 \leq x < \phi(n)$ such that $\beta \equiv \alpha^x \pmod{p}$.

A solution to the Discrete Logarithm Problem gives a solution to the Computational Diffie-Hellman Problem.

Proof.

Suppose that you can solve the Discrete Logarithm Problem.

Then given $\alpha^x \pmod{p}$ and $\alpha^y \pmod{p}$, you can solve for $x = \mathrm{ind}_\alpha(\alpha^x)$ and $y = \mathrm{ind}_\alpha(\alpha^y)$.

Often the security of ElGamal is related to the following (difficult) problem:

The Discrete Logarithm Problem

Let p be an odd prime. Let α be a primitive root modulo p. Given an integer $1 < \beta < n$, determine the unique integer $1 \leq x < \phi(n)$ such that $\beta \equiv \alpha^x \pmod{p}$.

A solution to the Discrete Logarithm Problem gives a solution to the Computational Diffie-Hellman Problem.

Proof.

Suppose that you can solve the Discrete Logarithm Problem.

Then given $\alpha^x \pmod{p}$ and $\alpha^y \pmod{p}$, you can solve for $x = \mathrm{ind}_\alpha(\alpha^x)$ and $y = \mathrm{ind}_\alpha(\alpha^y)$.

It is then easy to compute $\alpha^{xy} \equiv (\alpha^x)^y \equiv (\alpha^y)^x \pmod{p}$. $\square$