[CS304] Introduction to Cryptography and Network Security

Course Instructor: Dr. Dibyendu Roy
Scribed by: Manas Jitendrakumar Ingle (202151086)

Winter 2023-2024
[Week 11 : Lecture-April 16]

1 Discrete Logarithm Problem (DLP)

The Discrete Logarithm Problem (DLP) is a fundamental problem in the field of cryptography. It plays a crucial role in many cryptographic protocols, including those based on the Diffie-Hellman key exchange and the Digital Signature Algorithm (DSA).

Let G be a cyclic group of order q with generator g. Given an element $h \in G$, the discrete logarithm problem is to find an integer x, $0 \leq x < q$, such that $g^x \equiv h \pmod{q}$.

Difficulty

The security of many cryptographic protocols relies on the assumption that the DLP is computationally difficult to solve. In particular, if an efficient algorithm for solving the DLP were discovered, it would have significant implications for the security of these protocols.

Algorithms

Several algorithms have been developed for solving the DLP, including:

- **Brute Force**: Trying all possible values of x until the correct one is found. This approach becomes impractical for large values of q.
- **Baby-step Giant-step**: A classical algorithm that reduces the problem to one of finding a collision in a list of pairs of elements from the group.

The Discrete Logarithm Problem is a fundamental problem in cryptography with important implications for the security of cryptographic protocols. While no polynomial-time algorithm is known for solving the DLP, it remains an active area of research.

2 Baby-step Giant-step algorithm

The Baby-step Giant-step algorithm is a method for solving the Discrete Logarithm Problem (DLP) efficiently in cyclic groups. It was first proposed by Daniel Shanks in 1971 and is particularly useful

for small to medium-sized groups.

The Baby-step Giant-step algorithm works by breaking the search space into two smaller sets of elements:

- **Baby-steps**: A set of small powers of the generator g.
- **Giant-steps**: A set of larger powers of g that are multiples of a pre-chosen integer m.

By precomputing and storing these sets, the algorithm can efficiently search for a collision between a baby-step and a giant-step, which yields the solution to the DLP.

Algorithm

Let G be a cyclic group of order q with generator g, and let $h \in G$ be the element for which we want to find the discrete logarithm. The Baby-step Giant-step algorithm proceeds as follows:

1. Choose an integer m such that $m = \lceil \sqrt{q} \rceil$.
2. Compute and store the values $g^0, g^1, g^2, ..., g^{m-1}$ (baby-steps).
3. Compute the value g^{-m}.
4. For $j = 0$ to $m - 1$, compute $h \cdot (g^{-m})^j$ (giant-steps).
5. If a collision is found between a baby-step and a giant-step, i.e., if $h \cdot (g^{-m})^j = g^i$ for some $0 \leq i < m$ and $0 \leq j < m$, then the discrete logarithm x is given by $x = jm + i$.

Complexity

The Baby-step Giant-step algorithm has a time complexity of $O(\sqrt{q})$ and requires $O(\sqrt{q})$ storage space, making it particularly efficient for small to medium-sized groups.

3 Elgamal Encryption

ElGamal encryption is a public-key cryptosystem named after its inventor Taher ElGamal. It is based on the Diffie-Hellman key exchange and provides a way to securely exchange messages over an insecure channel without needing to share a secret key beforehand.

Key Generation

To use ElGamal encryption, each user generates a public-private key pair as follows:

1. Choose a large prime number p and a generator g of the multiplicative group $\mathbb{Z}_p^*$.

2. Select a random integer x, $1 < x < p - 1$, as the private key.
3. Compute the public key $y = g^x \mod p$.
4. The public key is (p, g, y), and the private key is x.

Encryption

To encrypt a message using ElGamal encryption, the sender performs the following steps:

1. Choose a random integer k, $1 < k < p - 1$, as the ephemeral key.
2. Compute the shared secret $s = y^k \mod p$.
3. Compute the ciphertext pair (c_1, c_2), where $c_1 = g^k \mod p$ and $c_2 = m \cdot s \mod p$, where m is the plaintext message.
4. Send (c_1, c_2) as the encrypted message.

Decryption

To decrypt a ciphertext (c_1, c_2), the receiver uses their private key x as follows:

1. Compute the shared secret $s = c_1^x \mod p$.
2. Compute the plaintext message $m = c_2 \cdot s^{-1} \mod p$.

Security

ElGamal encryption is semantically secure under the decisional Diffie-Hellman assumption. However, it is vulnerable to chosen ciphertext attacks if care is not taken during implementation.

4 Kerberos Version - 4 Message Exchange

Kerberos is a network authentication protocol designed to provide strong authentication for client-server applications. Kerberos Version 4 (Kerberos V4) is an early version of the protocol that operates based on a symmetric key system and uses a trusted third party, known as the Key Distribution Center (KDC), to authenticate users and servers.

4.1 Message Exchange

The Kerberos V4 message exchange involves the following steps:

4.2 Authentication

1. **Authentication Request**: The client sends an authentication request to the KDC, requesting a ticket-granting ticket (TGT) to access a specific service.
2. **TGT Issuance**: The KDC verifies the client's identity and issues a TGT encrypted with the client's secret key (K_c). The TGT contains a session key ($K_{c,s}$) for the client to use with the Ticket-Granting Server (TGS) and a timestamp.
3. **TGT Transmission**: The KDC sends the TGT back to the client.

4.3 Service Request

1. **Service Request**: The client sends a service request to the TGS, along with the TGT obtained in the previous step.
2. **Service Ticket Issuance**: The TGS verifies the TGT and issues a service ticket for the requested service, encrypted with the service's secret key (K_s). The service ticket contains a session key ($K_{c,s}$) for secure communication with the service and a timestamp.
3. **Service Ticket Transmission**: The TGS sends the service ticket back to the client.

4.4 Service Access

1. **Service Access**: The client sends the service ticket to the requested service, along with a timestamp encrypted with the session key ($K_{c,s}$).
2. **Service Verification**: The service verifies the authenticity of the service ticket and the timestamp. If valid, the service grants access to the client.

4.5 Security Considerations

The Kerberos V4 protocol relies on symmetric cryptography and assumes a trusted KDC. However, it is vulnerable to certain attacks, such as replay attacks and man-in-the-middle attacks. These vulnerabilities were addressed in later versions of the Kerberos protocol.