[CS309] Introduction to Cryptography and Network Security

Course Instructor: Dr. Dibyendu Roy — Autumn 2024-2025

Scribed by: Tanuj Saini(202251141) — Week 9,10

1 Diffie-Hellman Key Exchange

The Diffie-Hellman algorithm facilitates secure key exchange using public key cryptography, enabling two users to agree on a shared secret over an insecure channel. Consider a cyclic group $G = \langle g \rangle$ of prime order P. Alice selects a private key a ($1 \leq a < P$) and computes her public key $A = g^a \bmod P$. Similarly, Bob selects b and computes $B = g^b \bmod P$. They exchange A and B.

Key Computation:

Alice computes: $S = B^a \bmod P = g^{ab}$,
Bob computes: $S = A^b \bmod P = g^{ab}$.
Shared Secret Key: $S = g^{ab}$, which both parties independently compute.

Discrete Logarithm Problem: Given $g^x \bmod P$, finding x is computationally hard, ensuring the security of the exchange.

Note: The security of the Diffie-Hellman algorithm relies on the hardness of the discrete logarithm problem.

2 Modular Equations

We'll explore solving systems of linear equations to find x in the form:

$$a \cdot x \equiv b \bmod m \quad \text{(Eq.1)}$$

To begin, let's express Eq.1 as:

$$a \cdot x - m \cdot y = b \quad \text{(Eq.2)}$$

where y is an integer. Utilizing Bezout's Identity:

$$a \cdot x_0 + m \cdot y_0 = \gcd(a, m) \quad \text{(Eq.3)}$$

where x_0 and y_0 can be determined using the Extended Euclidean Algorithm.

Eq.2 is solvable if and only if $\gcd(a, m)$ divides b. Assuming $\gcd(a, m)$ divides b, we have:

$$t \cdot \gcd(a, m) = b$$

By multiplying Eq.3 by t, we get:

$$a \cdot (t \cdot x_0) + m \cdot (t \cdot y_0) = t \cdot \gcd(a, m) \implies a \cdot X_0 + m \cdot Y_0 = b$$

Hence, given an equation to solve, we first verify if $\gcd(a, m)$ divides b. If so, a solution exists. Then, we find x_0 and y_0 using the Extended Euclidean Algorithm and multiply them by $t = \frac{b}{\gcd(a,m)}$ to obtain X_0 and Y_0.

Once X_0 and Y_0 are identified as solutions of Eq.2, we can substitute x and y as follows:

$$x = X_0 + \frac{m}{\gcd(a, m)} \cdot n$$

$$y = Y_0 + \frac{a}{\gcd(a, m)} \cdot n$$

where n is an integer.

3 Exploring Elliptic Curve Cryptography (ECC)

- **Introduction:** While RSA offers a straightforward approach to cryptography with the Square and Multiply Algorithm, Elliptic Curve Cryptography (ECC) introduces a novel concept.
- **Computations on Curves:** ECC operates on elliptic curves rather than integers, leading to the development of modern cryptographic techniques.
- **Key Exchange:** ECC employs Elliptic Curve Diffie-Hellman (ECDH) for secure key exchange.
- **Digital Signatures:** Signatures in ECC are generated using Elliptic Curve Digital Signature Algorithm (ECDSA).
- **Security Benefits:** ECC's utilization of elliptic curves enables better security using smaller prime numbers.

Let's define two real numbers a and b such that:

$$a, b \in \mathbb{R} \text{ and } 4a^3 + 27b^2 \neq 0$$

Consider the curve:

$$y^2 = x^3 + ax + b$$

where $(x, y) \in \mathbb{R}^2$. This curve is known as an Elliptic Curve.

3.1 Elliptic Curve Mathematics

$$y^2 = x^3 + ax + b$$

$$4a^3 + 27b^2 \neq 0$$

Let us consider two points $P(x_1, y_1)$ and $Q(x_2, y_2)$. We have three cases:

1. $x_1 \neq x_2, y_1 \neq y_2$
2. $x_1 = x_2, y_1 = -y_2$
3. $x_1 = x_2, y_1 = y_2$

3.2 Elliptic Curve Diffie-Hellman (ECDH)

Let us consider a scenario where Alice and Bob want to exchange messages. They have a curve E and a point P, and (E, P) is public.

4 Discrete Logarithm Problem

The **Discrete Logarithm Problem (DLP)** is a foundational problem in cryptography that underpins the security of many algorithms, including the Diffie-Hellman key exchange and the Digital Signature Algorithm (DSA). It involves finding the exponent x in the equation:

$$g^x \equiv y \pmod{p}$$

where:

- g is a generator of the group,
- p is a large prime number, and
- y is a known value.

4.1 Why DLP is Hard

The DLP is computationally hard because:

- There is no efficient algorithm for solving it in general cases within polynomial time.
- It becomes infeasible as the prime p grows larger (e.g., 2048-bit primes).

4.2 Mathematical Properties

1. $g^a \cdot g^b \equiv g^{a+b} \pmod{p}$ (Closure under multiplication)
2. $g^{-a} \cdot g^a \equiv 1 \pmod{p}$ (Existence of inverse)
3. Exponentiation in modular arithmetic is easy, but finding x from g^x is hard

5 Kerberos Authentication Protocol

Kerberos is a secure, third-party authentication protocol designed for distributed networks. It provides mutual authentication between users and services by relying on a trusted third-party Key Distribution Center (KDC).

5.1 Key Components

1. **Key Distribution Center (KDC):** A trusted server that manages authentication by issuing tickets. It has two main components:
    - **Authentication Server (AS):** Verifies users and issues a Ticket Granting Ticket (TGT).
    - **Ticket Granting Server (TGS):** Issues service-specific tickets based on the TGT.
2. **Principal:** A user or service that participates in Kerberos authentication.
3. **Ticket:** An encrypted data structure containing user credentials, session keys, and timestamps.

5.2 Authentication Workflow

The Kerberos authentication process involves the following steps:

1. **Login and AS Request:**
    - The user enters their credentials (username and password).
    - The client sends an authentication request to the AS, encrypted using the user's password-derived key.

2. **Ticket Granting Ticket (TGT):**
   - The AS verifies the user's credentials and generates a TGT.
   - The TGT is encrypted using the KDC's secret key and sent to the client.
3. **Service Request to TGS:**
   - The client presents the TGT to the TGS along with a service request.
   - The TGS validates the TGT and issues a service ticket.
4. **Accessing the Service:**
   - The client sends the service ticket to the requested service.
   - The service validates the ticket and grants access.

5.3 Security Features

- Mutual Authentication: Both the client and the service verify each other.
- Replay Protection: Time-stamped tickets prevent reuse of old tickets.
- Session Keys: Unique keys are generated for each session, ensuring secure communication.

5.4 Advantages of Kerberos

- Centralized authentication improves efficiency and security.
- Mutual authentication ensures trust between users and services.
- Time-stamped tickets mitigate replay attacks.

5.5 Limitations of Kerberos

- Relies on synchronized clocks; discrepancies can lead to authentication failure.
- If the KDC is compromised, the entire system is at risk.
- Initial password entry can expose credentials if not securely transmitted.

5.6 Kerberos Ticket Structure

A typical Kerberos ticket contains:

- Principal's identity (user or service)
- Session key for secure communication
- Ticket lifetime (start and expiry times)
- Flags indicating ticket properties (e.g., renewable, forwardable)

5.7 Kerberos Versions

- **Kerberos v4:** Earlier version, used DES for encryption, now considered outdated.
- **Kerberos v5:** Enhanced version, supports various encryption types, cross-realm authentication, and extended ticket properties.

5.8 Applications of Kerberos

- Secure login for distributed systems
- Single Sign-On (SSO) implementation in corporate networks
- Securing services such as email, file servers, and databases

Kerberos ensures secure, efficient, and reliable authentication in distributed environments.