1 The ElGamal Signature

1.1 Algorithm

The ElGamal Signature Scheme

Public parameters:
\( g \) of order \( q \) inside \( \mathbb{Z}_p^* \) and \( h = g^x \); \( x \leftarrow R \{1, \ldots, q\} \) is the signing key.

Sign
\( l \leftarrow R \{1, \ldots, q\} \)
\( r \leftarrow g^l \mod p \)
\( s \leftarrow l^{-1}(m - x.r) \mod q \)

So the signature on a message \( m \) is the pair \((r, s)\).

To verify a signature \((r, s)\) on a message \( m \) we perform the check if:

\[ r^s h^r = g^m \]

Suppose you are given \( m \) (a message) and \( g, h \) and you want to forge a signature on \( m \), this means you have to come up with \( r, s \) s.t. \( r^s h^r = g^m \).

In reality \( m \leftarrow H(M) \) for large messages you sign the hash of the message. It is important that the hash function is collision resistant, hard for an attacker to find \( M \) s.t. \( H(M) = H(M') \).

2 Authentication

An entity verifies a claimed property of another entity.

1. Data Origin Authentication
2. Entity Authentication
3. Authenticated Key Exchange

Data Origin Authentication VS Data integrity

- Requires/involves communication between different parties
- Requires origin authentication
- It implies liveliness of origination
2.1 Data origin authentication

Transmitter ↔ Receiver

1. Validate the identity
2. Check data integrity
3. Ensure liveliness of transmitter

2.1.1 A solution based on digital signatures

Say we have a digital signature algorithm.

\[ \text{GEN}(v_k, s_k) \], where \( s_k \) is the signing key and \( v_k \) is the verification key.

\[ \text{SIGN}(s_k; M) \rightarrow \zeta \]

\[ \text{VERIFY}(v_k, M, \zeta) \in \{\text{TRUE}, \text{FALSE}\} \]

CERTIFICATION AUTHORITY : CA

\[ \text{GEN} \rightarrow (S_{kCA}, V_{kCA}) \]

Assumption: \( V_{kCA} \) can be available to interested parties in a trustworthy fashion.

Remark: In most cases it is usually preloaded in a user’s browser.

We can characterize the notion of data origin authentication as follows:

1. It consists of transmitting a message from a purported source (transmitter) to a receiver who will validate the message upon reception.
2. The message validation conducted by the receiver aims to establish the identity of the message transmitter.
3. The validation also aims to establish the data integrity of the message subsequent to its departure from the transmitter.
4. The validation further aims to establish liveliness of the message transmitter.

To understand this protocol let’s look a scenario when Alice wants to transmit a message \( M \) to Bob and see how Bob verifies if \( M \) was sent by Alice or not.

1. Alice generates her signing key \( s_{kA} \) and her validation key \( v_{kA} \)

\[ \text{GEN} \rightarrow (v_k, s_k) \]

2. Alice “applies for a certificate” on \( v_{kA} \) from a Certification Authority CA which has the structure:

\[ \text{Cert}_A = \text{Sig}_CA(V_{kA}; Alice; dates) \]

3. Alice transmits to Bob \( \{M; \text{Sig}_A(M; date); \text{Cert}_A; V_{kA}\} \)

4. Upon Reception Bob performs the following actions:

- Verifies \( \text{Cert}_A \)
- Verifies \( \text{Sig}_A(M; date) \)
• Checks date
• If verification is successful accepts M

\{Cert_A, V_{kA}\} is usually available in a trustworthy manner from a server on the greater network on which the users are a part of the public-key infrastructure.

2.2 Entity Authentication

One entity ensures the liveliness of another entity, that claims a certain identity.

Variants of Entity Authentication:

1. Host-Host
2. User-Host
3. Process-Host
4. Member-Club

2.3 Authenticated Key Exchange

Based on Diffie-Hellman key Exchange.

System wide parameters:

\[ \alpha \in \mathbb{Z}_p^*, \beta > \] is a cyclic subgroup of \( \mathbb{Z}_p^* \) of order \( q \). \( p \) and \( q \) are primes. Assume a CA as before.

Alice and Bob have obtained

\[ Cert_A = \text{Sig}_{CA}(Alice, V_{kA}; \text{desc}(< \alpha >)) \]

and

\[ Cert_B = \text{Sig}_{CA}(Bob, V_{kB}; \text{desc}(< \alpha >)) \]

respectively, where \( V_{kA} \) and \( V_{kB} \) are the verification keys for the digital signature scheme for Alice and Bob respectively. \text{desc}(< \alpha >) is the description of the cyclic group \( < \alpha >, \alpha, p, q \).

1. Alice picks \( x \leftarrow R \{1, 2, \ldots, q\} \) and transmits to Bob \( \alpha^x \)

2. Bob picks \( y \leftarrow R \{1, 2, \ldots, q\} \) and transmits to Alice \( \{\alpha^y; Cert_B; \epsilon_k(\text{Sig}_B(\alpha^y, \alpha^x))\} \)

Here, \( \epsilon \) stands for a symmetric encryption scheme such as AES using \( k = \alpha^{x:y} \)

3. Alice Computes \( k = \alpha^{x:y} = \alpha^{x:y} \), verifies \( Cert_B \) decrypts \( \epsilon_k(\text{Sig}_B(\alpha^y, \alpha^x)) \), verifies \( \text{Sig}_B(\alpha^y, \alpha^x) \), checks that \( \alpha^y, \alpha^x \) are ok.

4. Alice transmits to Bob \( Cert_A; \epsilon_k(\text{Sig}_A(\alpha^y, \alpha^x)) \)

5. Bob checks \( Cert_A, \) decrypts \( \epsilon_k(\text{Sig}_A(\alpha^y, \alpha^x)) \), checks \( \text{Sig}_A(\alpha^y, \alpha^x) \), checks that \( \alpha^y, \alpha^x \) are ok.

Some Attacks
2. Man-In-The-Middle Attack.
3. Parallel Session Attack.