Advanced Web Security

Secure Messaging (OTR)

OTR Messaging

- Off-the-Record messaging
  - No one else can hear the conversation
  - Neither Alice nor Bob can provide proof of what has been said

- Allow the following properties
  - Encryption
  - Authentication
  - Deniability
  - Perfect Forward Secrecy
- Protocol has high focus on usability and practical aspects

Authentication and Key Agreement (AKA)

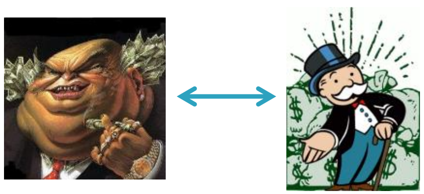
- Diffie-Hellman key agreement
  - Exchange signed Diffie-Hellman values and public keys

$$\begin{array}{lcll} A \rightarrow B & : & (g^{x_1})_{SK_A}, & PK_A \\ B \rightarrow A & : & (g^{y_1})_{SK_B}, & PK_B \end{array}$$

- How should we verify public keys?
  - PKI is not suitable in messaging protocols
  - We can not assume that they have met and exchanged public keys, or fingerprints
- Without knowing each other's public key they can not verify it.
  - Vulnerable to MitM-attacks
- Still, they probably have *some* shared *low entropy* secret

Socialist Millionaires Problem

- Millionaires problem
  - Two people wish to known who is richest – but they do not want to reveal their wealth
- Variant: Socialist Millionaires Problem
  - Two people want to know if they have the same wealth, but not to reveal how much they have.



Socialist Millionaires Problem (SMP)

- Alice has value x, Bob has value y.
  - Use a protocol that verifies if x = y
- Naïve solution: Exchange hash values.
  - Vulnerable to brute force, does not meet the low entropy requirement
- Use a protocol that allows exchange of values that do not give away *any* information
  - See lecture notes for a protocol.

SMP applied to AKA

- Add SMP to the protocol

$$\begin{array}{lcll} A \rightarrow B & : & (g^{x_1})_{SK_A}, & PK_A \\ B \rightarrow A & : & (g^{y_1})_{SK_B}, & PK_B \end{array}$$

$$x = y = H(PK_A \parallel PK_B \parallel g^{x_1 y_1} \parallel \text{"}shared\ secret\text{"})$$

SMP

- Eve now has only one chance to guess the secret in a MitM
  - If she fails, SMP will fail → Alice and Bob will know

Encryption and authentication of messages

- Diffie-Hellman provides Perfect Forward Secrecy
- However, if exponents are broken or leaked, the session is broken
- “Solution”: Make each message its own session

$$\begin{array}{lcll} & \vdots & & \\ A \rightarrow B & : & g^{x_i}, & E(M_j, k_{i-1,i-1}) \\ B \rightarrow A & : & g^{y_i}, & E(M_{j+1}, k_{i,i-1}) \\ A \rightarrow B & : & g^{x_{i+1}}, & E(M_{j+2}, k_{i,i}) \\ B \rightarrow A & : & g^{y_{i+1}}, & E(M_{j+3}, k_{i+1,i}) \end{array}$$

- Authenticate messages with MAC (derived from Diffie-Hellman)

Add Deniability

- With a MAC, only Alice or Bob can have created the message
- After verifying MAC, it is sent in clear in the next message.
- Make it possible to modify plaintexts
  - Not only repudiation, but also forgeability
- Use stream cipher so that it is also easy to modify known plaintexts to another known plaintext

$$c_i \oplus 1 = m_i \oplus k_i \oplus 1 = m_i \oplus 1 \oplus k_i.$$