Cryptography

Systems and Information Security
Informatics Engineering (3rd year, 2nd sem.)

José Bacelar Almeida

Applied Cryptography

Symmetric Crypto

Asymmetric Crypto

Applications

Roadmap

- Key-Agreement
  - Diffie-Hellman protocol
- Public-key cryptography
  - Public-key Encryption (PKE)
  - Digital Signatures (PKS)
  - Asymmetric primitives
- Elliptic-Curve Cryptography (ECC)
- Cryptography and Quantum Computation
  - Post-Quantum Cryptography (PQC)
- Certificates and Public-Key Infrastructure (PKI)

Key-Agreement

Context

- Symmetric crypto rely on shared secret-keys;
- Pre-agreement of keys is a costly procedure (it requires the use of secure channels…), and inflexible (e.g. consider adding an agent to the community…).
- Are there viable alternatives?
    - E.g.: Suppose we have a (symmetric) cipher in which the cipher operation is commutative, i.e. $E_{k1}(E_{k2}(X))=E_{k2}(E_{k1}(X))$
    - Each party (*A* and *B*) generate a secret key (*KA* and *KB*, respectively)
    - For *A* to communicate *M* with *B* it can
        - *A* sends $E_{KA}(M)$ to B — (note that *KA* is only known by *A*);
        - *B* returns $E_{KB}(E_{KA}(M))=E_{KA}(E_{KB}(M))$ to *A* — (again, *KB* is only known by *B*);
        - *A* decrypts the received message, and resends to *B* the result $E_{KB}(M)$;
        - *B* finally decrypts the ciphertext with its own key, obtaining message *M*.

    ... that is, *A* and *B* communicate securely without sharing secrets... (message *M* is always protected with at least one layer of encryption).
- Remark: but this scheme also displays important vulnerabilities... (c.f. man-in-the-middle attack studied below)

Key-Agreement

- The problem of key distribution can be circumvented if both parties agree on a common secret...
    - ...exchanging messages on a public channel...
    - ...but without it being possible to derive the secret knowing only the messages exchanged.
- A scheme that accommodates these requirements appeared in the article (New Directions in Cryptography, Diffie & Hellman 1976).
- Security follows from one-wayness of modular exponentiation (hardness of discrete logarithm).

Intermezzo: Algebraic Structures

- When p is a prime number, $GF(p)=(\mathbb{Z}_p, +, *)$ is a **field** (+ and * are respectively addition and multiplication modulo p).
  - Modular inverses can be computed by the (extended) Euclidean algorithm;
  - A primitive element is a **generator** g of the multiplicative group (that is, each non-zero element can be written as g^i, for some i.
- For composite n, $(\mathbb{Z}_n, +, *)$ forms a **ring**.
  - Elements with multiplicative inverse (**units**) are those x such that $gcd(x, n)=1$ (*primes relative* to n).
  - The number of those elements are given by $\varphi(n)$ (Euler function).

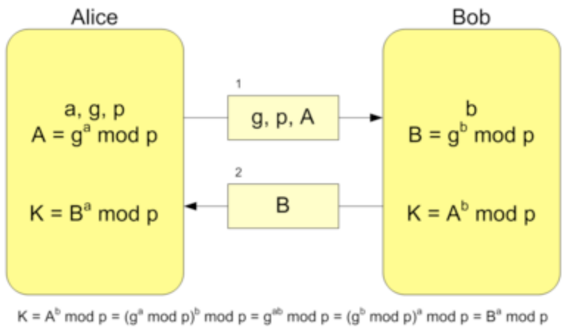
(Believed) Hard problems

Under appropriate conditions, the following problems are considered difficult.

- **Integer factorisation**
  - Given an integer n, determine its <u>prime number factorisation</u>. In other words, determine the prime numbers $p_1,...,p_i$ such that $p_1 \times ... \times p_i = n$.
- **Discrete logarithm**
  - Given a, b and n, compute x such that $a^x \bmod n = b$.
- **Discrete square-root**
  - Given y and n, compute x such that $x^2 \bmod n = y$.

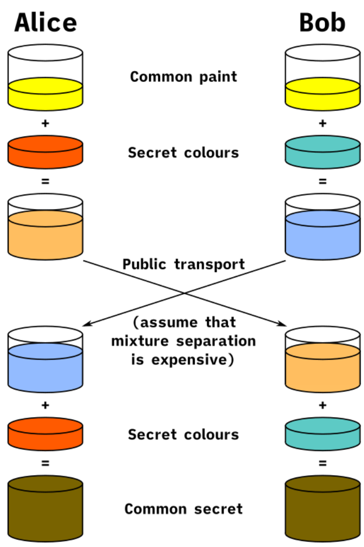
(Ephemeral) Diffie&Hellman protocol

- Parameters:
  - Let p be a prime and g a generator of a subgroup of Z_p^* of prime order q.
- Description:
  - A randomly chooses an integer $\boldsymbol{sk_A}$=x s.t. $1<x<q$ (**private key**), sending $\boldsymbol{pk_A}$=$g^x \bmod p$ (**public key**) to B.
  - Similarly, B randomly chooses $\boldsymbol{sk_B}$=y s.t. $1<y<q$, sending $\boldsymbol{pk_B}$=$g^y \bmod p$ to A.
  - **Shared secret**: $\boldsymbol{K}$=$g^{xy} \bmod p$ = $(g^y)^x \bmod p$ = $(g^x)^y \bmod p$.
- Notice that each party generates a key-pair for each execution of the protocol — hence the qualifier *Ephemeral*.
- Remark: shared secret K should never be used directly as a cryptographic key. It should instead be feeded into a KDF to derive needed secrets.



DH security wrt passive adversaries

- If the adversary is able to compute the discrete logarithm, then it could compute x from g^x, thus attacking the protocol.
- Strictly speaking, the security of the protocol is expressed as a security assumption of its own — the *Computational Diffie-Hellman* (CDH) problem: for random x and y, it is infeasable a PPT adversary to compute g^{xy} from g^x and g^y.



DH (in)security wrt active adversaries

- An *active adversary* might impersonate another party and so compromise the secrecy of the shared secret: known as **man-in-the-middle attack**.
- Example:

  Suppose A wants to agree on a secret with B:

  - A generates a key-pair $(sk_A, pk_A)=(x, g^x)$, and sends pk_A to B;
  - I intercepts A's message;
  - I generates its own key-pair $(sk_I, pk_I)=(z, g^z)$, returning pk_I to A:
  - A adopts the secret $K=(g^z)^x=g^{xz}$ which it presumes agreed with B;
  - I knows the secret $K=(g^x)^z=g^{xz}$ that A believes is shared with B.
- Most asymmetric cryptographic techniques are vulnerable to this attack!

the use of asymmetric cryptography depends on a reliable association between public-key and legitimate parties (identities).

Public-Key Cryptography