The opponent $O$ knows $u = a^x \mod p$, $v = a^y \mod p$, $a \neq 1$.

If $O$ is able to calculate discrete log's, the system is broken, i.e., breaking the DH problem is no harder than calculating discrete log's.

**Def 7.6** Diffie–Hellman–Problem (DHP)

Given $p$, $a \in \mathbb{Z}_p^*$, $a^x \mod p$, $a^y \mod p$, calculate $a^{xy} \mod p$ is the Diffie–Hellman problem.

An efficient alg. to solve the DHP would break the DH scheme.

Open question: Does an efficient alg. for solving DHP lead to an efficient alg. for discrete log's?

### 7.2 Shamir's no-key protocol

**Prop 7.7** Let $p$ be prime $a \in \mathbb{Z}_p^*$. Then

$$b \in \mathbb{Z}_p, \quad mab a^{-1} b^{-1} \equiv m \pmod{p}$$

**Proof:** $a^{-1}, b^{-1} \in \mathbb{Z}_p^*$ exist by Def. *unicity*

$$a \cdot a^{-1} \equiv 1 \pmod{p-1}, \quad b \cdot b^{-1} \equiv 1 \pmod{p-1} \iff \quad a \cdot a^{-1} = 2(p-1) + 1, \quad b \cdot b^{-1} = 2(p-1) + 1$$

Hence, for all $m \in \mathbb{Z}_p$,

$$mab a^{-1} b^{-1} = m (2(p-1) + 1) (2(p-1) + 1)$$

$$= m \cdot m (p-1) \equiv m \pmod{p} \iff \text{ Fermat}$$
A sends a key \( m \) to B as follows

- **Initial setup**: a prime \( p \) is chosen and published
- **Protocol actions**:

  1. A and B choose secret random numbers \( a, b \in \mathbb{Z}_p^* \) and calculate \( a^r \mod (p-1) \) and \( b^r \mod (p-1) \), respectively.

  2. **A:** Sends \( c_1 = a^m \mod p \) (A locks, sends to B)
  
  3. **B:** Sends \( c_2 = (c_1)^b \mod p \) (B locks, sends to A)
  
  4. **A:** Sends \( c_3 = (c_2)^a \mod p \) (A unlocks, returns to B)

  5. **B:** Deciphers \( m = (c_3)^{b^{-1}} \mod p \) (B unlocks, reads \( m \))

\[ (c_3)^{b^{-1}} = m \quad a^b a^{-n} b^{-n} \equiv m \mod p \]

**Observe**: no authentication provided; prevention from penins adversarial only.
8. Public key Encryption

Asymetric encryption which does not need to exchange secret keys.

Idea:[key Diffie Hellman (76), earlier but not published paper by James Ellis (70) paper released by British government 97)]

- All users share the same e, d (en- decryption function)
- Each user has a pair of keys (K, L) such that:

\[ d(e(M, K), L) = M \quad \forall M \in M \]

K is public, L is private key

- Requirements
  (i) \( c = e(M, K) \) "easy" given \( M, K \), solving for \( M \) "infeasible" given \( c \) and \( K \)
  (ii) \( M = d(c, L) \) "easy" given \( c \) and \( L \)

Hence, \( e(M, K) \) is a one-way function with "trapdoor" \( L \)

- Further requirements
  (i) \( (K, L) \) easy to generate
  (ii) There are sufficiently many pairs \( (K, L) \) exhaustivly search impossible
8.1 The RSA cryptosystem (Rivest, Shamir, Adleman, 1978)
(preceded indirectly by Coppersmith (1973), not published, released 1977)

**RSA - System**

(i) Choose \( p \neq q \) (large prime numbers), compute \( n = p \cdot q \)

(ii) Choose \( d \in \mathbb{Z}^*_n \), i.e. \( \gcd(d, \phi(n)) = 1 \)

\( \) (compute \( e = d^{-1} \mod \phi(n) \))

(iii) Public key \( (e, n) \), private key \( d \)

(iv) Message \( m \in \{1, \ldots, n-1\} \)

**Encryption** : \( c = m^e \mod n \)

**Decryption** : \( b = c^d \mod n \)

**Questions** : 1) \( b = m \)? 2) Security 3) Implementation

**Prop. 8.1** \( p \neq q\) prime, \( x, y \in \mathbb{Z}^*_n \)

\[ x \equiv y \pmod{p} \land x \equiv y \pmod{q} \implies x \equiv y \pmod{p \cdot q} \]

**Proof** : \( p \mid x - y \land q \mid x - y \implies p \cdot q \mid x - y \) (since \( p \cdot q \) are relatively prime)

**Prop. 8.2** Let \( p \neq q \) prime, \( n = p \cdot q \), \( d, d^{-1} \in \mathbb{Z}^*_n \)

\[ 0 \leq m < n \quad c = m \cdot d^{-1} \mod n \quad \text{Then} \quad m = c \cdot d \mod n \]

\( \Rightarrow \) Decrypption is the RSA system works

**Proof** : \( d^{-1} d \equiv 1 \pmod{\phi(n)} \implies \exists t : d \cdot d^{-1} = t \cdot (p-1) \cdot (q-1) + 1 \)

(i) \( \gcd(d(m_1, p)) = 1 \)

\( (m \cdot d^{-1}) \equiv m \cdot t(p-1)(q-1) + 1 \equiv m \cdot (p-1)(q-1) \equiv m \pmod{p} \)

\( \equiv 1 \pmod{p} \), \( \forall m \neq 0 \)
\( g(d(m, p)) = p \quad \text{if} \quad m \equiv 0 \pmod{p} \\
\Rightarrow \quad (m^{\phi(p)})^d \equiv 1 \equiv m \pmod{p} \)

Alternatively, \( (m^{\phi(p)})^d \equiv m \pmod{\phi(p)} \)

Using Prop 8.1: \( (m^{\phi(p)})^d \equiv m \pmod{\phi(n) = \phi(p) \cdot \phi(q)} \)

\textbf{Security of RSA}

Chosen-plaintext attacks are most relevant, since anybody can encrypt an arbitrary number of any messages using the public key. Hence, known: \( d^{-1} \cdot n \), arbitrary many pairs \((m, c)\).

a) Factoring of \( n \) to find \( p, q \) to compute

\[ d = (d^{-1})^{-1} \pmod{\phi(n)} = (p-1)(q-1) \]

the private key.

But: Factoring is infeasible.

b) Computing square roots modulo \( n \) allows factoring.

\textbf{Prop 8.3} Let \( n = p \cdot q \), \( p \neq q \) prime, \( x \) a nontrivial solution of \( x^2 \equiv 1 \pmod{n} \), i.e., \( x \equiv \pm 1 \pmod{n} \). Then

\[ g \mid d(x+1, n) \in \{p, q\} \]

\textbf{Proof: Exercise}

Hence: Computing square roots is no easier than factoring.

c) Computing \( \sqrt{n} \) without factoring \( n \),

any efficient algo for computing \( \sqrt{n} \) yields an efficient algo for factoring.

Hence, computing \( \sqrt{n} \) is no easier than factoring.
Proof: \( \eta = n \cdot p \cdot q \) prime (unknown)

\[ f(n) = (p-1)(q-1) \] is known

\[ f(n) = (p-1)(q-1) = \frac{p \cdot q - p - q + 1}{2} \]

(a) \( p + q = n - f(n) + 1 \) \quad (1)

\[ (p-q)^2 - (p+q)^2 = -4pq \] \quad (2)

\[ (p-q)^2 = (p+q)^2 - 4n \] \quad (2)

(b) \( q = \frac{1}{2}(p+q) - (p-q) \) \quad (3)

(1) yields \( p + q \) from (2) obtain \( p - q \), \( q \) follows by (3)

d) Computing \( p \) or \( q \) \quad \text{(without knowing} f(n)\text{)}

Proof: Let \( n = p \cdot q \cdot r \cdot q \) prime. Any efficient algorithm for computing \( b^n \mod f(n) \) leads to an efficient probabilistic algorithm for factoring \( n \) with error probability \( \leq \frac{1}{2} \)

Proof: Stein, p. 139 - 141

Repeat the above algorithm until a factorization is found.

Hence, computing \( b^n \mod f(n) \) is no easier than factoring.

Remark:

a) If \( d \) is known, \( n \) can be efficiently factored
   If the private key \( d \) is deleted, it is not sufficient to compute some new \( d \), \( d \equiv -1 \mod f(n) \), also change \( p \cdot q \).

b) Never let somebody observe your decryption process

c) \text{(an interesting} RSA (78):) An efficient algorithm for breaking the RSA system yields an efficient factoring algorithm.

(Still open question)