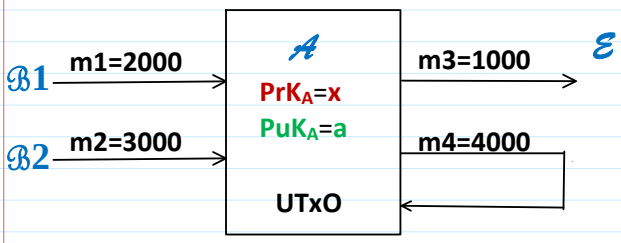
m_1, m_2 – certain sums of money

$$\left.\begin{array}{l} m_i = m_1 + m_2 \\ m_0 = m_3 + m_4 \end{array}\right\} \; m_i = m_e$$

Transactions confidentiality

To hide sums $m_1 = 2000$; $m_2 = 3000$.

Transaction - **Tx** in blockchain



Balance: $m_1 + m_2 = m_3 + m_4$

How to verify balance when inputs and expenses are encrypted?

To provide confidential transactions numbers

$n_1 = g^{m_1} \bmod p$ and $n_2 = g^{m_2} \bmod p$ are encrypted

$$n_1 \cdot n_2 \bmod p = (g^{m_1} \bmod p \cdot g^{m_2} \bmod p) \bmod p = g^{(m_1+m_2) \bmod p-1} \bmod p$$

$m_3 = 1000$; $m_4 = 4000$

$n_3 = g^{m_3} \bmod p$ and $n_4 = g^{m_4} \bmod p$.

$n_3 \cdot n_4 \bmod p = g^{(m_3+m_4) \bmod (p-1)} \bmod p$ ← Fermat theorem:

Operations in exponents can be reduced mod $(p-1)$ ⇐ If p – is prime: $z^{p-1} = 1 \bmod p \Rightarrow 0 \equiv p-1$

If $(m_1 + m_2) \bmod (p-1) = (m_3 + m_4) \bmod (p-1)$

$n_1 \cdot n_2 \bmod p = n_3 \cdot n_4 \bmod p$

Paillier Encryption-Decryption

p, q – prime number, generated at random >> p = genprime (28)

$N = p \cdot q$; $N^2 = p^2 \cdot q^2$; E.g. $p = 3$; $q = 5$; $N = 15$.

$Enc_N : \mathbb{Z}_N \times \mathbb{Z}_N^* \rightarrow \mathbb{Z}_{N^2}^*$; $\mathbb{Z}_N = \{0, 1, 2, \ldots, N-1\}$; $+ \bmod N$; $* \bmod N$

$Enc_N : \mathcal{Z}_N \times \mathcal{Z}_N^* \rightarrow \mathcal{Z}_{N^2}^*$; $\mathcal{Z}_N = \{0, 1, 2, \ldots, N-1\}$; $+ \bmod N$; $* \bmod N$

$\mathcal{Z}_p^* = \{1, 2, 3, \ldots, p-1\}$; $\mathcal{Z}_N^* = \{z \mid \gcd(z, N) = 1\}$; group: $* \bmod N$

$\mathcal{Z}_{N^2}^* = \{w \mid \gcd(w, N^2) = 1\}$; group: $\circledast \bmod N^2$

Example : $N = 15$

$\mathcal{Z}_{15} = \{0, 1, 2, \ldots, 14\}$;

$|\mathcal{Z}_{15}| = 15$

$\mathcal{Z}_{15}^* = \{1, 2, 4, 7, 8, 11, 13, 14\}$;

$|\mathcal{Z}_{15}^*| = 8$. Euler function $\phi(N) = \phi$

If $N = p \cdot q$ then $\phi(N) = (p-1) \cdot (q-1)$.

$\phi(15) = (3-1) \cdot (5-1) = 8$;

```
>> z=7;
>> n=15;
>> z_m1=mulinv(z,n)
z_m1 = 13
>> mod(z*z_m1,n)
ans = 1
>> z=10;
>> z_m1=mulinv(z,n)
z_m1 = Inverse element does not exist
>> gcd(z,n)
ans = 5
```

$\phi = (p-1) \cdot (q-1) = \phi(N)$: $\phi = (3-1) \cdot (5-1) = 2 \cdot 4 = 8$

$m \in \mathcal{Z}_N = \{0, 1, 2, \ldots, N-1\}$; $\mathcal{Z}_{15} = \{0, 1, 2, \ldots, 14\}$

$r \in \mathcal{Z}_N^* = \{z ; \gcd(z, N) = 1\}$; $\mathcal{Z}_{15}^* = \{1, 2, 4, 7, 8, 11, 13, 14\}$

$|\mathcal{Z}_{15}^*| = 8$

>> gcd (4, 15) = 1

>> gcd (9, 15) = 3 ≠ 1.

>> gcd (7, 15) = 1

$Enc_N(m, r) = c \in \mathcal{Z}_{N^2}^*$; $|\mathcal{Z}_N \times \mathcal{Z}_N^*| = |\mathcal{Z}_{N^2}^*|$.

Enc_N: is 1-to-1 mapping.

Enc_N: is additively-multiplicative isomorphic

Enc_N: $(m_1 + m_2) \bmod N, (r_1 * r_2) \bmod N) = c = c_1 * c_2 \bmod N^2$

$c_1 = Enc_N(m_1, r_1)$; $c_2 = Enc_N(m_2, r_2)$.

$c = Enc_N((m_1 + m_2) \bmod N, (r_1 * r_2) \bmod N)$; $c \in \mathcal{Z}_{N^2}^*$

To encrypt message m (probabilistically) the random number r must be generated.

To achieve additively-homomorphic encryption the sum $m_1 + m_2$ must be encrypted with parameter $r = (r_1 * r_2) \bmod N$.

To achieve additively-homomorphic encryption the sum $m_1 + m_2$ must be encrypted with parameter $r = (r_1 * r_2) \bmod N$.

$Enc(N, m, r) = c$ $\qquad$ $Dec(\phi, c) = m$

$PuK = N = p \cdot q$; $\qquad$ $PrK = \phi = (p-1)\cdot(q-1)$.

$(m, r) \in \mathcal{I}_N \times \mathcal{I}_N^*$; $\quad c \in \mathcal{I}_{N^2}^* = \{z;\ gcd(z, N^2) = 1\}$;

$PuK = N$. $\Rightarrow$ factor $(N) = (p, q)$

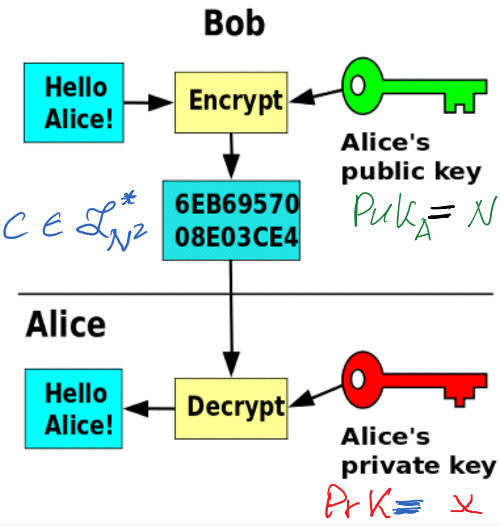
to achieve security equivalent to security of ECDSA with EC having $|PrK| = 256$ bits $|N| \cong 3000$ bits.

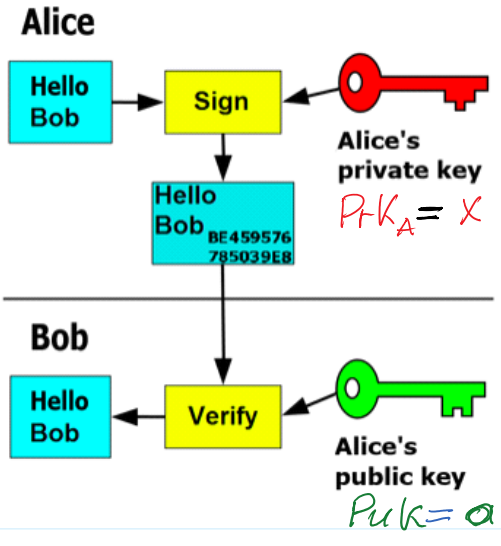
$|p| \simeq 1500$ bits; $|q| \approx 1500$ bits

$\sqrt{2^{3000}} \simeq 2^{1500}$.

$Enc(N, m, r) = c \in \mathcal{I}_{N^2}^*$

$Dec(\phi, c) = m \in \mathcal{I}_N$.





$\phi(N) = \phi(p \cdot q) = \phi(p) \cdot \phi(q) = (p-1)\cdot(q-1) = \phi = PrK_A$

$\phi(N^2) = p\cdot(p-1)\cdot q\cdot(q-1) = p\cdot q\cdot(p-1)\cdot(q-1)$

$= |\mathcal{I}_N| \cdot |\mathcal{I}_N^*| = |\mathcal{I}_N \times \mathcal{I}_N^*| = |\mathcal{I}_{N^2}^*|$.

PuK = N --> PrK = fy = ϕ ; $N = p \cdot q$; When $N \sim 2^{2048} \Rightarrow$ to find p, q

is infeasible → RSA problem.

Security:

When PuK = N it is infeasible to find $\phi = (p-1)\cdot(q-1)$ ⇒

⇒ RSA problem is infeasible ⇔ factorization problem.

>> factor(15) → ans 3, 5

Factorization problem is infeasible if N is sufficiently large.

$N \sim 2^{2048}$ → $|N| = 2048$ bits.

CONSTRUCTION 11.32

Let GenModulus be a polynomial-time algorithm that, on input 1^n, outputs (N, p, q) where $N = pq$ and p and q are n-bit primes (except with probability negligible in n). Define a public-key encryption scheme as follows:

- Gen: on input 1^n run GenModulus(1^n) to obtain (N, p, q). The public key is N, and the private key is $\langle N, \phi(N)\rangle = \langle N, \phi\rangle$.
- Enc: on input a public key N and a message $m \in \mathbb{Z}_N$, choose a random $r \leftarrow \mathbb{Z}_N^*$ and output the ciphertext

$$c := [(1+N)^m \cdot r^N \bmod N^2].$$

- Dec: on input a private key $\langle N, \phi(N)\rangle$ and a ciphertext c, compute

$$m := \left[\frac{[c^{\phi(N)} \bmod N^2] - 1}{N} \cdot \phi(N)^{-1} \bmod N\right].$$

The Paillier encryption scheme.

$c \in \mathbb{Z}_{N^2}^* = \{w \mid \gcd(w, N^2) = 1\}$

$m \in \mathbb{Z}_N = \{0, 1, 2, \ldots, N-1\}$

Parameters generation

```
>> p=127
p = 127
>> isprime(p)
ans = 1
>> dec2bin(p)
ans = 1111111
>> q=113
q = 113
>> isprime(q)
ans = 1
```

```
>> N=p*q
N = 14351        % PuK=N
>> N_2=int64(N*N)
N_2 = 205951201
>> fy=(p-1)*(q-1)
fy = 14112       % PrK=fy=φ
```

Encryption

$$c := [\underbrace{(1+N)^m}_{e_1} \cdot \underbrace{r^N}_{e_2} \bmod N^2].$$

$c = e_1 \cdot e_2 \bmod N^2$

```
>> m=11111
m = 11111
>> r=genprime(14)
r = 9049
>> gcd(r,N)
ans = 1
```

```
>> e1=mod_exp((1+N),m,N_2)
e1 = 159453962
>> e2=mod_exp(r,N,N_2)
e2 = 73833387
>> c=mod(e1*e2,N_2)
c = 120531541
```

Decryption

$$m := \left[\frac{[\overbrace{c^{\phi(N)}}^{d_1} \bmod N^2] - 1}{N} \cdot \phi(N)^{-1} \bmod N\right].$$

$d_2 \bmod N$ $\quad$ $d_3 \bmod N$

```
>> d1=mod_exp(c,fy,N_2)
d1 = 197426708
>> d2=mod((d1-1)/N,N)
d2 = 13757
>> d3=mulinv(fy,N)
d3 = 5224
```

$m = d_2 \cdot d_3 \bmod N$

$d_2 \bmod N$ $\quad$ $d_3 \bmod N$

d2 = 13757
>> d3=mulinv(fy,N)
d3 = 5224
>> mod(fy*d3,N) % verification of inverse value
ans = 1
>> mm=mod(d2*d3,N)
mm = 11111

eVoting System must guarantee:
- Conseal the Vote
- Conseal the Ballots

Table of eVoting in Google drive
https://docs.google.com/spreadsheets/d/1joAuG6oh1arcQqc7zPTPbLBH48wG8Fn6/edit?usp=sharing&ouid=111502255533491874828&rtpof=true&sd=true

Voter Victor - W

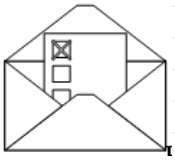
PrK_W $\quad$ PuK_W

Conseal the Vote $\mathbf{V_W}$ in the Ballot:

Election Committee - EC

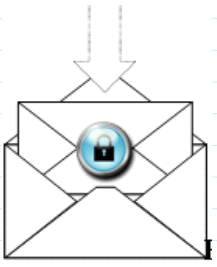
PuK_{EC} $\quad$ PrK_{EC}

N $\quad$ $\phi \equiv fy$



$Enc(PuK_{EC}, V_W) = E_V$

Internal Envelope
IE



$Sign(PrK_W, EE) = \sigma_W$

EC

External Envelope
EE

Signed **EE** with Voter's $\mathbf{V_W}$ signature σ_W and PuK_W

After eVoting it is the time to calculate the results.
1. **EC** verifies Voter's $\mathbf{V_W}$ PuK_W and if PuK_W is registered in **EC** database then goes to step 2.
2. **EC** verifies PuK_W certificate and if it is valid then goes to step 3.
3. **EC** verifies signature σ_W on **EE** and if it is valid then extracts **IE** and decrypts **IE** with its PrK_{EC}
4. **EC** prceeds with ballots computation.

Till this place

CONSTRUCTION 11.32

Let GenModulus be a polynomial-time algorithm that, on input 1^n, outputs (N, p, q) where $N = pq$ and p and q are n-bit primes (except with probability negligible in n). Define a public-key encryption scheme as follows:

- **Gen:** on input 1^n run GenModulus(1^n) to obtain (N, p, q). The public key is N, and the private key is $\langle N, \phi(N)\rangle$: $\phi(N) = \phi = ty.$
- **Enc:** on input a public key N and a message $m \in \mathbb{Z}_N$, choose a random $r \leftarrow \mathbb{Z}_N^*$ and output the ciphertext

$$c := [(1+N)^m \cdot r^N \bmod N^2].$$

- **Dec:** on input a private key $\langle N, \phi(N)\rangle$ and a ciphertext c, compute

$$m := \left[\frac{[c^{\phi(N)} \bmod N^2] - 1}{N} \cdot \phi(N)^{-1} \bmod N\right].$$

The Paillier encryption scheme.

B: PuK_A

A: PrK

$1^n = 1 \cdot 1 \cdot 1 \cdot \ldots \cdot 1$

n - required number of bits

$m < N$; $|N| = 2048$b.

Additively-Multiplicative encryption: V_1, V_2 – votes to be encrypted.

$PuK = N$; $PrK = \phi$.

Voter 1: $r_1 \leftarrow randi(\mathbb{Z}_N^*)$

$Enc(N, V_1, r_1) = c_1$

Voter 1: $r_2 \leftarrow randi(\mathbb{Z}_N^*)$

$Enc(N, V_2, r_2) = c_2$

$\Rightarrow c_1 * c_2 \bmod N^2 =$

$= Enc(N, V_1 + V_2, r_1 \cdot r_2)$

$gcd(r, n) = 1$

$r \in \mathbb{Z}_N^* = \{r;\ gcd(r, N) = 1\}$

$f_1 = (1+n)^m \bmod n^2$

$f_2 = r^n \bmod n^2$

$c = (1+n)^m \cdot r^n \bmod n^2$

$f_1 \cdot f_2$

A: $Enc_{PuK_B}(m) = c$

$m \in \mathcal{M} = \mathcal{Z}_N = \{0, 1, 2, \ldots, N-1\}$; $c \in \mathcal{C} = \mathcal{Z}^*_{N^2}$

B: $Dec_{PrK_B}(c) = m$

ϕ^{-1} mod N computation with Octave software

```
>> fyem1 = mulinv(fy, N)
```

$$Enc(m_1 + m_2) = Enc(m_1) \cdot Enc(m_2) \quad \text{mod } N^2$$
$$c = c_1 \cdot c_2$$

$$Enc(m_1 \cdot m_2) = Enc(m_1) \cdot Enc(m_2)$$
$$Enc(m_1 + m_2) = Enc(m_1) + Enc(m_2)$$

} Full homomorphic (isomorphic)

Encrypted Data Base

$m_1 \longrightarrow$ randomas c_1

$m_2 \longrightarrow$ randomas c_2

server - cloud

c

$m_1, m_2 < N$

$Dec(c) = m_1 \cdot m_2$