Cryptography

Public Key Cryptography
University of Birmingham
Autumn Term 2017

Lecturer: David Galindo

UNIVERSITY OF BIRMINGHAM | Security and Privacy

Arrangements

symmetric key cryptography was given by Mark Ryan;
public key cryptography will be given by myself

A total of **two summative assessments** plus exam

Exam counts **80%**
Continuous Assessment counts **20%**
of final mark

Assessment & Office hours

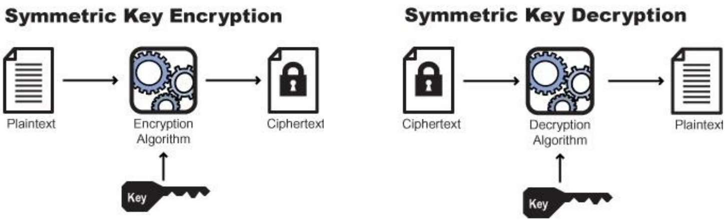
2nd assessment: distributed 16 Nov, deadline 27 Nov

Where&how to find me:

- My Office is Room 116, 1st floor, SCoS
- Office hours:
  - Wednesdays 2pm-4pm
- Contact: `D.Galindo@cs.bham.ac.uk`

Secret key encryption

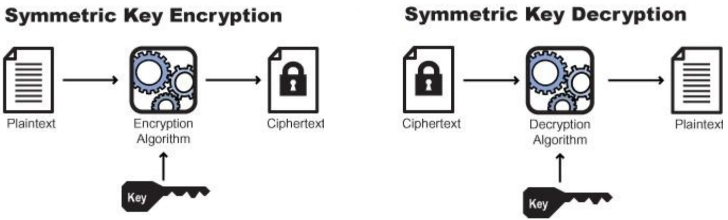
so far: we covered **symmetric encryption**, where encryption key K and decryption key K are equal



http://www.infosectoday.com

Secret key encryption

so far: we covered **symmetric encryption**, where encryption key K and decryption key K are equal



http://www.infosectoday.com

Question 1: give examples of symmetric key encryption where encryption and decryption algorithms are equal

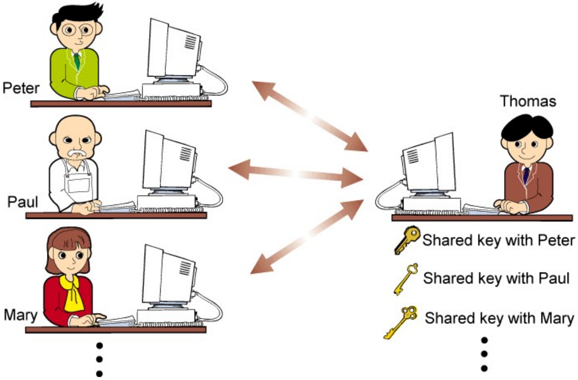
Some responses to Question 1

Definition 5.1.5 Counter mode (CTR)
Let $e()$ be a block cipher of block size b, and let x_i and y_i be bit strings of length b. The concatenation of the initialization value IV and the counter CTR_i is denoted by $(IV||CTR_i)$ and is a bit string of length b.
Encryption: $y_i = e_k(IV||CTR_i) \oplus x_i, \quad i \geq 1$
Decryption: $x_i = e_k(IV||CTR_i) \oplus y_i, \quad i \geq 1$

Definition 5.1.4 Cipher feedback mode (CFB)
Let $e()$ be a block cipher of block size b; let x_i and y_i be bit strings of length b; and IV be a nonce of length b.
Encryption (first block): $y_1 = e_k(IV) \oplus x_1$
Encryption (general block): $y_i = e_k(y_{i-1}) \oplus x_i, \quad i \geq 2$
Decryption (first block): $x_1 = e_k(IV) \oplus y_1$
Decryption (general block): $x_i = e_k(y_{i-1}) \oplus y_i, \quad i \geq 2$

from *Understanding Cryptography.* Paar, Pelzl (2010)

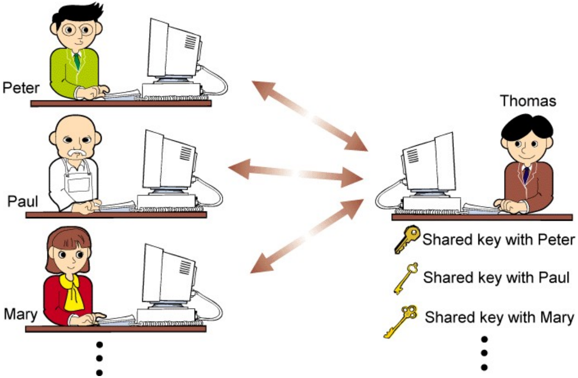
Key management problem



http://www.csis.hku.hk

Question: how many keys needed for pairwise **secret communication** between *n* parties?

Key management problem

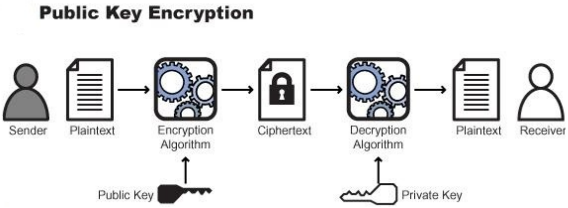


http://www.csis.hku.hk

Question: how many keys needed for pairwise **secret communication** between n parties? $\frac{n(n-1)}{2}$

Public key encryption

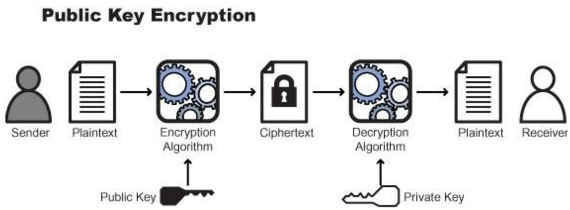
Can do differently: can use **asymmetric encryption**, where encryption key K and decryption key K' are different



http://www.infosectoday.com

Public key encryption

Can do differently: can use **asymmetric encryption**, where encryption key K and decryption key K' are different

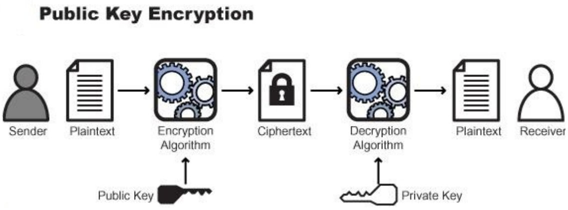


`http://www.infosectoday.com`

Question 1: Would it make sense to make both keys *public*?

Public key encryption

Can do differently: can use **asymmetric encryption**, where encryption key K and decryption key K' are different



http://www.infosectoday.com

Question 1: Would it make sense to make both keys *public*?

Question 2: In asymmetric encryption, can encryption and decryption algorithms *be equal*?

Public key encryption - physical analogy

http://csunplugged.org

Alice encrypts to Bob's public key

Assume Alice has **padlock** and Bob has the **key**

Alice places her **message** in a **safe box**, applies padlock

Bob **unlocks** padlock with **key** and takes out **message** from **safe box**

Public key encryption - physical analogy

http://csunplugged.org

Alice encrypts to Bob's public key

Assume Alice has **padlock** and Bob has the **key**

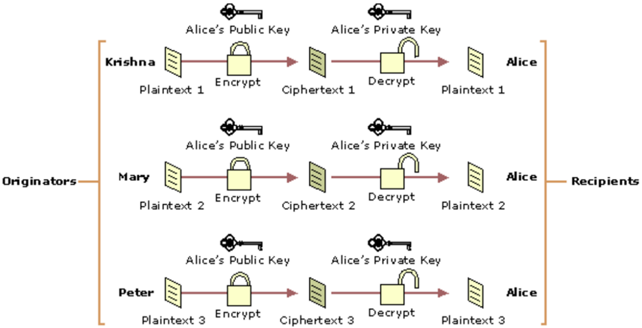
Alice places her **message** in a **safe box**, applies padlock

Bob **unlocks** padlock with **key** and takes out **message** from **safe box**

Question: Find a variation of this analogy for *symmetric key* crypto

Public key encryption and Key Management

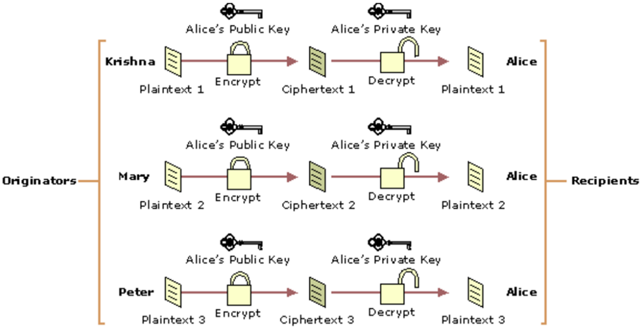
Consider key management for n communicating parties:



https://technet.microsoft.com

Public key encryption and Key Management

Consider key management for *n* communicating parties:

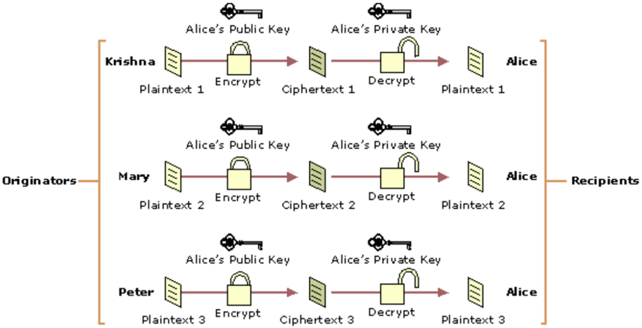


https://technet.microsoft.com

Question: how many keys needed for pairwise **secret communication** between *n* parties?

Public key encryption and Key Management

Consider key management for n communicating parties:



https://technet.microsoft.com

Question: how many keys needed for pairwise **secret communication** between n parties? n public keys

A public key encryption scheme consists of the following algorithms $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$:

- $\mathsf{KG}(\lambda)$ on input a **security parameter** λ outputs a pair of encryption/decryption keys (PK, SK)
- $\mathsf{Enc}(PK, m; r)$ on inputs a public key PK, plaintext m outputs a ciphertext C (eventually local randomness r)
- $\mathsf{Dec}(SK, C)$ on inputs a decryption key SK and a ciphertext C outputs a plaintext m

Modular Arithmetic - Recap

$\mathbb{Z}_N$ and modular arithmetic

Definition (mod N)

Fix a positive integer N which we call the *modulus*. Let $a, b \in \mathbb{Z}$ two integers. We write $a = b \pmod N$ or $a \equiv b \mod N$ if N divides $b - a$. Equivalently if $b - a = q \cdot N$ for an integer q. We say that a and b are **congruent modulo N** or that the **modular reduction modulo N** of a is b

Definition ($\mathbb{Z}_N$)

$\mathbb{Z}_N$ for $N \in \mathbb{Z}, N > 0$ is defined as $\mathbb{Z}_N = \{0, 1, \ldots, N-2, N-1\}$. We call it the **ring of integers modulo N**

Basic modular arithmetic

The set $\mathbb{Z}_N$ has two modular operations, namely **addition** and **multiplication.**

For example, for $N = 16$

$11 + 13 \mod 16 = 24 = 8 \mod 16$ **since** $24 - 8 = 16 \cdot 1$

$11 \cdot 13 \mod 16 = 143 \mod 16 = 15$ **since** $143 - 15 = 16 \cdot 8$

$27 \cdot 45 \mod 16 = 15$ **since** $27 \cdot 45 \mod 16 = 11 \cdot 13 \mod 16$

Greatest Common Divisor (Euclidean Algorithm)

Definition (GCD)

Let $a, b \in \mathbb{Z}$ be two integers with $a \neq 0$ and $b \neq 0$. The **greatest common divisor** for a and b, written $\gcd(a, b)$, is the largest positive integer that divides both numbers without remainder

compute $\gcd(a, b)$

1. **read** a, b
2. **while** $b \neq 0$ **do**
   - $r \leftarrow a \mod b$
   - $a \leftarrow b$
   - $b \leftarrow r$
3. **return** $|a|$

Examples:
$\gcd(100, 76) = \gcd(76, 24) = \gcd(24, 4) = 4$

$\gcd(1665, 910) = \gcd(1665, 910) = \gcd(910, 755)$
$\gcd(155, 135) = \gcd(135, 20) = \gcd(20, 15) =$
$\gcd(15, 5) = \gcd(5, 0) = 5$

Question: Show that $\gcd(1426668559730, 810653094756) = 1417082$

Solution to GCD calculation

$$
\begin{aligned}
\gcd(1\,426\,668\,559\,730,\ 810\,653\,094\,756) &= \gcd(810\,653\,094\,756,\ 616\,015\,464\,974),\\
&= \gcd(616\,015\,464\,974,\ 194\,637\,629\,782),\\
&= \gcd(194\,637\,629\,782,\ 32\,102\,575\,628),\\
&= \gcd(32\,102\,575\,628,\ 2\,022\,176\,014),\\
&= \gcd(2\,022\,176\,014,\ 1\,769\,935\,418),\\
&= \gcd(1\,769\,935\,418,\ 252\,240\,596),\\
&= \gcd(252\,240\,596,\ 4\,251\,246),\\
&= \gcd(4\,251\,246,\ 1\,417\,082),\\
&= \gcd(1\,417\,082,\ 0),\\
&= 1\,417\,082.
\end{aligned}
$$

from *Cryptography Made Simple.* N.P. Smart (2016)

The Extended Euclidean Algorithm

Let $a > b$ be two integers such that $a > 0$ and $b > 0$. Then the following algorithm computes integers α and β such that

$$\gcd(a, b) = \alpha \cdot a + \beta \cdot b$$

read a, b

$\lambda_{11} \leftarrow 1, \lambda_{22} \leftarrow 1, \lambda_{12} \leftarrow 0, \lambda_{21} \leftarrow 0$

while $b \neq 0$ **do**

  $q \leftarrow a \div b$

  $r \leftarrow a \mod b$

  $a \leftarrow b$

  $b \leftarrow r$

  $t_{21} \leftarrow \lambda_{21};\ t_{22} \leftarrow \lambda_{22}$

  $\lambda_{21} \leftarrow \lambda_{11} - q \cdot \lambda_{21}$

  $\lambda_{22} \leftarrow \lambda_{12} - q \cdot \lambda_{22}$

  $\lambda_{11} \leftarrow t_{21}$

  $\lambda_{12} \leftarrow t_{22}$

return $(\gcd(a, b), \alpha, \beta) \leftarrow (|a|, \lambda_{11}, \lambda_{12})$

Inverses modulo N

Theorem

$x \in \mathbb{Z}_N$ *has an inverse* y *(i.e. there exists* $y \in \mathbb{Z}_N$ *such that* $x \cdot y = 1 \mod N$*) if and only if* $\gcd(N, x) = 1$. *We say* y ***is the inverse*** *of* x ***modulo N*** *and write it as* $y = x^{-1} = 1/x \mod N$

Fact

$y = x^{-1} \mod N$ *can be computed with the Extended Euclidean algorithm: let* $\gcd(N, x) = \alpha \cdot N + \beta \cdot x = 1$. *Then* $y := \beta$

Example: Inverses modulo 19

We compute $7^{-1} \mod 19$ using $\gcd(19,7) = \alpha \cdot 19 + \beta \cdot 7$

0. $\lambda_{11} \leftarrow 1, \lambda_{22} \leftarrow 1, \lambda_{12} \leftarrow 0, \lambda_{21} \leftarrow 0, a \leftarrow 19, b \leftarrow 7$

Example: Inverses modulo 19

We compute $7^{-1} \mod 19$ using $\gcd(19, 7) = \alpha \cdot 19 + \beta \cdot 7$

0. $\lambda_{11} \leftarrow 1, \lambda_{22} \leftarrow 1, \lambda_{12} \leftarrow 0, \lambda_{21} \leftarrow 0, a \leftarrow 19, b \leftarrow 7$
1. $q \leftarrow 2, r \leftarrow 5, a \leftarrow 7, b \leftarrow 5, t_{21} \leftarrow 0, t_{22} \leftarrow 1, \lambda_{21} \leftarrow 1,$
   $\lambda_{22} \leftarrow -2, \lambda_{11} \leftarrow 0, \lambda_{12} \leftarrow 1$

Example: Inverses modulo 19

We compute $7^{-1} \mod 19$ using $\gcd(19,7) = \alpha \cdot 19 + \beta \cdot 7$

0. $\lambda_{11} \leftarrow 1, \lambda_{22} \leftarrow 1, \lambda_{12} \leftarrow 0, \lambda_{21} \leftarrow 0, a \leftarrow 19, b \leftarrow 7$
1. $q \leftarrow 2, r \leftarrow 5, a \leftarrow 7, b \leftarrow 5, t_{21} \leftarrow 0, t_{22} \leftarrow 1, \lambda_{21} \leftarrow 1,$
$\lambda_{22} \leftarrow -2, \lambda_{11} \leftarrow 0, \lambda_{12} \leftarrow 1$
2. $q \leftarrow 1, r \leftarrow 2, a \leftarrow 5, b \leftarrow 2, t_{21} \leftarrow 1, t_{22} \leftarrow -2, \lambda_{21} \leftarrow -1,$
$\lambda_{22} \leftarrow 3, \lambda_{11} \leftarrow 1, \lambda_{12} \leftarrow -2$

Example: Inverses modulo 19

We compute $7^{-1} \mod 19$ using $\gcd(19, 7) = \alpha \cdot 19 + \beta \cdot 7$

0. $\lambda_{11} \leftarrow 1, \lambda_{22} \leftarrow 1, \lambda_{12} \leftarrow 0, \lambda_{21} \leftarrow 0, a \leftarrow 19, b \leftarrow 7$
1. $q \leftarrow 2, r \leftarrow 5, a \leftarrow 7, b \leftarrow 5, t_{21} \leftarrow 0, t_{22} \leftarrow 1, \lambda_{21} \leftarrow 1,$
   $\lambda_{22} \leftarrow -2, \lambda_{11} \leftarrow 0, \lambda_{12} \leftarrow 1$
2. $q \leftarrow 1, r \leftarrow 2, a \leftarrow 5, b \leftarrow 2, t_{21} \leftarrow 1, t_{22} \leftarrow -2, \lambda_{21} \leftarrow -1,$
   $\lambda_{22} \leftarrow 3, \lambda_{11} \leftarrow 1, \lambda_{12} \leftarrow -2$
3. $q \leftarrow 2, r \leftarrow 1, a \leftarrow 2, b \leftarrow 1, t_{21} \leftarrow -1, t_{22} \leftarrow 3, \lambda_{21} \leftarrow 3,$
   $\lambda_{22} \leftarrow -8, \lambda_{11} \leftarrow 1, \lambda_{12} \leftarrow -2$

Example: Inverses modulo 19

We compute $7^{-1} \mod 19$ using $\gcd(19,7) = \alpha \cdot 19 + \beta \cdot 7$

0. $\lambda_{11} \leftarrow 1, \lambda_{22} \leftarrow 1, \lambda_{12} \leftarrow 0, \lambda_{21} \leftarrow 0, a \leftarrow 19, b \leftarrow 7$
1. $q \leftarrow 2, r \leftarrow 5, a \leftarrow 7, b \leftarrow 5, t_{21} \leftarrow 0, t_{22} \leftarrow 1, \lambda_{21} \leftarrow 1,$ $\lambda_{22} \leftarrow -2, \lambda_{11} \leftarrow 0, \lambda_{12} \leftarrow 1$
2. $q \leftarrow 1, r \leftarrow 2, a \leftarrow 5, b \leftarrow 2, t_{21} \leftarrow 1, t_{22} \leftarrow -2, \lambda_{21} \leftarrow -1,$ $\lambda_{22} \leftarrow 3, \lambda_{11} \leftarrow 1, \lambda_{12} \leftarrow -2$
3. $q \leftarrow 2, r \leftarrow 1, a \leftarrow 2, b \leftarrow 1, t_{21} \leftarrow -1, t_{22} \leftarrow 3, \lambda_{21} \leftarrow 3,$ $\lambda_{22} \leftarrow -8, \lambda_{11} \leftarrow 1, \lambda_{12} \leftarrow -2$
4. $q \leftarrow 2, r \leftarrow 0, a \leftarrow 1, b \leftarrow 0, t_{21} \leftarrow 3, t_{22} \leftarrow -8, \lambda_{21} \leftarrow -7,$ $\lambda_{22} \leftarrow 14, \lambda_{11} \leftarrow 3, \lambda_{12} \leftarrow -8$

Example: Inverses modulo 19

We compute $7^{-1} \mod 19$ using $\gcd(19,7) = \alpha \cdot 19 + \beta \cdot 7$

0. $\lambda_{11} \leftarrow 1, \lambda_{22} \leftarrow 1, \lambda_{12} \leftarrow 0, \lambda_{21} \leftarrow 0, a \leftarrow 19, b \leftarrow 7$
1. $q \leftarrow 2, r \leftarrow 5, a \leftarrow 7, b \leftarrow 5, t_{21} \leftarrow 0, t_{22} \leftarrow 1, \lambda_{21} \leftarrow 1, \lambda_{22} \leftarrow -2, \lambda_{11} \leftarrow 0, \lambda_{12} \leftarrow 1$
2. $q \leftarrow 1, r \leftarrow 2, a \leftarrow 5, b \leftarrow 2, t_{21} \leftarrow 1, t_{22} \leftarrow -2, \lambda_{21} \leftarrow -1, \lambda_{22} \leftarrow 3, \lambda_{11} \leftarrow 1, \lambda_{12} \leftarrow -2$
3. $q \leftarrow 2, r \leftarrow 1, a \leftarrow 2, b \leftarrow 1, t_{21} \leftarrow -1, t_{22} \leftarrow 3, \lambda_{21} \leftarrow 3, \lambda_{22} \leftarrow -8, \lambda_{11} \leftarrow 1, \lambda_{12} \leftarrow -2$
4. $q \leftarrow 2, r \leftarrow 0, a \leftarrow 1, b \leftarrow 0, t_{21} \leftarrow 3, t_{22} \leftarrow -8, \lambda_{21} \leftarrow -7, \lambda_{22} \leftarrow 14, \lambda_{11} \leftarrow 3, \lambda_{12} \leftarrow -8$

   Hence $\gcd(19,7) = 3 \cdot 19 + (-8) \cdot 7$

Example: Inverses modulo 19

We compute $7^{-1} \mod 19$ using $\gcd(19,7) = \alpha \cdot 19 + \beta \cdot 7$

0. $\lambda_{11} \leftarrow 1, \lambda_{22} \leftarrow 1, \lambda_{12} \leftarrow 0, \lambda_{21} \leftarrow 0, a \leftarrow 19, b \leftarrow 7$
1. $q \leftarrow 2, r \leftarrow 5, a \leftarrow 7, b \leftarrow 5, t_{21} \leftarrow 0, t_{22} \leftarrow 1, \lambda_{21} \leftarrow 1,$ $\lambda_{22} \leftarrow -2, \lambda_{11} \leftarrow 0, \lambda_{12} \leftarrow 1$
2. $q \leftarrow 1, r \leftarrow 2, a \leftarrow 5, b \leftarrow 2, t_{21} \leftarrow 1, t_{22} \leftarrow -2, \lambda_{21} \leftarrow -1,$ $\lambda_{22} \leftarrow 3, \lambda_{11} \leftarrow 1, \lambda_{12} \leftarrow -2$
3. $q \leftarrow 2, r \leftarrow 1, a \leftarrow 2, b \leftarrow 1, t_{21} \leftarrow -1, t_{22} \leftarrow 3, \lambda_{21} \leftarrow 3,$ $\lambda_{22} \leftarrow -8, \lambda_{11} \leftarrow 1, \lambda_{12} \leftarrow -2$
4. $q \leftarrow 2, r \leftarrow 0, a \leftarrow 1, b \leftarrow 0, t_{21} \leftarrow 3, t_{22} \leftarrow -8, \lambda_{21} \leftarrow -7,$ $\lambda_{22} \leftarrow 14, \lambda_{11} \leftarrow 3, \lambda_{12} \leftarrow -8$

   Hence $\gcd(19,7) = 3 \cdot 19 + (-8) \cdot 7$

Finally $7^{-1} \mod 19 = -8 = 11 \mod 19$

Example: Inverses modulo 19

We compute $7^{-1} \mod 19$ using $\gcd(19, 7) = \alpha \cdot 19 + \beta \cdot 7$

0. $\lambda_{11} \leftarrow 1, \lambda_{22} \leftarrow 1, \lambda_{12} \leftarrow 0, \lambda_{21} \leftarrow 0, a \leftarrow 19, b \leftarrow 7$
1. $q \leftarrow 2, r \leftarrow 5, a \leftarrow 7, b \leftarrow 5, t_{21} \leftarrow 0, t_{22} \leftarrow 1, \lambda_{21} \leftarrow 1,$ $\lambda_{22} \leftarrow -2, \lambda_{11} \leftarrow 0, \lambda_{12} \leftarrow 1$
2. $q \leftarrow 1, r \leftarrow 2, a \leftarrow 5, b \leftarrow 2, t_{21} \leftarrow 1, t_{22} \leftarrow -2, \lambda_{21} \leftarrow -1,$ $\lambda_{22} \leftarrow 3, \lambda_{11} \leftarrow 1, \lambda_{12} \leftarrow -2$
3. $q \leftarrow 2, r \leftarrow 1, a \leftarrow 2, b \leftarrow 1, t_{21} \leftarrow -1, t_{22} \leftarrow 3, \lambda_{21} \leftarrow 3,$ $\lambda_{22} \leftarrow -8, \lambda_{11} \leftarrow 1, \lambda_{12} \leftarrow -2$
4. $q \leftarrow 2, r \leftarrow 0, a \leftarrow 1, b \leftarrow 0, t_{21} \leftarrow 3, t_{22} \leftarrow -8, \lambda_{21} \leftarrow -7,$ $\lambda_{22} \leftarrow 14, \lambda_{11} \leftarrow 3, \lambda_{12} \leftarrow -8$

   Hence $\gcd(19, 7) = 3 \cdot 19 + (-8) \cdot 7$

Finally $7^{-1} \mod 19 = -8 = 11 \mod 19$

Question: Compute $11^{-1} \mod 19$ and $5^{-1} \mod 19$

Inverses modulo N

Definition

$\mathbb{Z}_N^*$ is the subset of $\mathbb{Z}_N$ containing all its invertible elements

Definition

We call the function $\phi(N)$, which assigns to an integer N the number of invertible elements in $\mathbb{Z}_N^*$ *Euler's Totient function*

Properties ϕ function

- If $p \geq 2$ is prime, then

$$\phi(p) = p - 1$$

- More generally, for any $e \geq 1$,

$$\phi(p^e) = p^{e-1} \cdot (p - 1)$$

- For $n, m > 0$ such that $\gcd(n, m) = 1$, we have:

$$\phi(n \cdot m) = \phi(n) \cdot \phi(m)$$

Euler's theorem

Theorem

Let $N \in \mathbb{N}$ *and* $a \in \mathbb{Z}$*, with* $\gcd(a, N) = 1$*, then we have* $a^{\phi(N)} \equiv 1 (\bmod N)$

- Proof
  - Consider the map $f_a : \mathbb{Z}_N^* \rightarrow \mathbb{Z}_N^*$, such that $f_a(b) = a \cdot b$ for any $b \in \mathbb{Z}_N^*$
  - f_a is a bijection (also called permutation in crypto language)
    - f_a is injective, i.e. $f_a(b_1) = f_a(b_2)$ iff $b_1 = b_2 \mod N$
    - f_a is exhaustive, i.e. for any $b \in \mathbb{Z}_N$ there exists $b' \in \mathbb{Z}_N$ such that $f_a(b') = b$
  - therefore

$$\prod_{b \in \mathbb{Z}_N^*} b = \prod_{b' \in \mathbb{Z}_N^*} (a \cdot b') = a^{\phi(N)} \cdot \prod_{b' \in \mathbb{Z}_N^*} b'$$

  - We can conclude that $a^{\phi(N)} \equiv 1 \mod N$

Fermat's little theorem

- Theorem
  - For any prime p and any integer $a \neq 0 \mod p$, we have $a^{p-1} \equiv 1 \mod p$. Moreover, for any integer a, we have $a^p \equiv a \mod p$
- Proof
  - *Hint*: Use Euler's theorem

Fermat's little theorem

- Theorem
  - For any prime p and any integer $a \neq 0 \mod p$, we have $a^{p-1} \equiv 1 \mod p$. Moreover, for any integer a, we have $a^p \equiv a \mod p$
- Proof
  - *Hint*: Use Euler's theorem
  - *Answer*: From Euler's theorem we know that if $\gcd(a, N) = 1$ then $a^{\phi(N)} \equiv 1 (\mathrm{mod}\ N)$. Since p is prime and $a \neq 0 \mod p$ then $\gcd(a, N) = 1$. Finally, $\phi(p) = p - 1$

Solving modular linear equations

Let $a, x, b \in \mathbb{Z}_N^\star$ for a positive integer N

Question: How to solve the equation

$$ax = b \mod N \qquad ?$$

Solving modular linear equations

Let $a, x, b \in \mathbb{Z}_N^\star$ for a positive integer N

Question: How to solve the equation

$$ax = b \mod N \qquad ?$$

Answer: $x = a^{-1} \cdot b \mod N$

Question: Confirm x satisfies the equation above

Solving modular linear equations

Let $a, x, b \in \mathbb{Z}_N^\star$ for a positive integer N

Question: How to solve the equation

$$ax = b \mod N \qquad ?$$

Answer: $x = a^{-1} \cdot b \mod N$

Question: Confirm x satisfies the equation above

Answer: Indeed $a \cdot (a^{-1} \cdot b) = a \cdot a^{-1} \cdot b = 1 \cdot b = b \mod N$

Solving modular linear equations

Let $a, x, b \in \mathbb{Z}_N^\star$ for a positive integer N

Question: How to solve the equation

$$ax = b \mod N \qquad ?$$

Answer: $x = a^{-1} \cdot b \mod N$

Question: Confirm x satisfies the equation above

Answer: Indeed $a \cdot (a^{-1} \cdot b) = a \cdot a^{-1} \cdot b = 1 \cdot b = b \mod N$

Question: Solve the equation $7 \cdot x + 3 = 7 \mod 19$

Solving modular linear equations

Let $a, x, b \in \mathbb{Z}_N^\star$ for a positive integer N

Question: How to solve the equation

$$ax = b \mod N \qquad ?$$

Answer: $x = a^{-1} \cdot b \mod N$

Question: Confirm x satisfies the equation above

Answer: Indeed $a \cdot (a^{-1} \cdot b) = a \cdot a^{-1} \cdot b = 1 \cdot b = b \mod N$

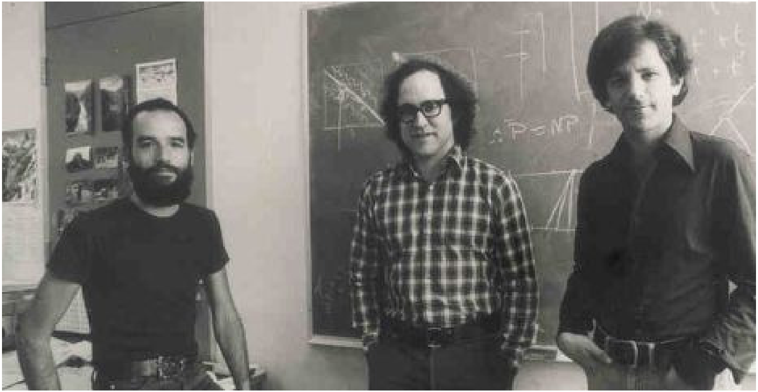
Question: Solve the equation $7 \cdot x + 3 = 7 \mod 19$

Answer: $x = 7^{-1} \cdot 4 = 6 \mod 19$

RSA - A Trapdoor One-Way Permutation

The RSA algorithm

- The RSA algorithm is the most widely-used public-key encryption algorithm
  - Invented in 1977 by Rivest, Shamir and Adleman
  - Used for encryption and signature
  - Widely used in electronic commerce protocols (TLS, PKI)



RSA cryptosystem

- Key generation $\mathsf{KG}(\lambda)$
  - Generate two distinct primes p and q of same bit-size λ
  - Compute $N = p \cdot q$ and $\phi = (p-1)(q-1)$
  - Select a random integer e, $1 < e < \phi$ such that $\gcd(e, \phi) = 1$
  - Compute the unique integer d such that

$$e \cdot d \equiv 1 \mod \phi$$

    using the Extended Euclidean algorithm
  - The public key is $PK = (N, e)$. The private key is $SK = d$*

*The convention is that SK includes PK

RSA cryptosystem

- Encryption $\mathrm{Enc}(PK, m)$
  - Given a message $m \in \mathbb{Z}_N^\star$ and the recipient's public-key $PK = (N, e)$ compute the ciphertext:

$$c = m^e \mod N$$

- Decryption $\mathrm{Dec}(SK, c)^\dagger$
  - Given a ciphertext c, to recover m, compute:

$$m = c^d \mod N$$

†Knowledge of PK is needed to perform this computation

Toy Example

- $p = 3, q = 11, N = 33, \phi = ?$
- Let e s.t. $\mathrm{gcd}(e, \phi) = 1$. For instance $e = 7$
- $d = ??$
- $PK = (N, e) = (33, 7)$ and $SK = d = ??$
- The message space is $\mathbb{Z}_{33}^{\star} = ??$
- Encrypt $m = 4$ using RSA encryption with $PK = (33, 7)$
  - $C = ??$
- Recover m from C using RSA decryption with SK

Toy Example

- $p = 3, q = 11, N = 33, \phi = 20$

Toy Example

- $p = 3, q = 11, N = 33, \phi = 20$
- Choose e s.t. $\gcd(e, \phi) = 1$. For instance $e = 7$
- $1 = \gcd(e, \phi) = 3 \cdot e + (-1) \cdot \phi$. Hence $d = 3$
- $PK = (n, e) = (33, 7)$ and $SK = d = 3$

Toy Example

- $p = 3, q = 11, N = 33, \phi = 20$
- Choose e s.t. $\gcd(e, \phi) = 1$. For instance $e = 7$
- $1 = \gcd(e, \phi) = 3 \cdot e + (-1) \cdot \phi$. Hence $d = 3$
- $PK = (n, e) = (33, 7)$ and $SK = d = 3$
- The message space is $\mathbb{Z}_{33}^{\star} = \{1, 2, 4, 5, 7, 8, 10, 13, 14, 16, 17, 19, 20, 23, 25, 26, 28, 29, 31, 32\}$

Toy Example

- $p = 3, q = 11, N = 33, \phi = 20$
- Choose e s.t. $\gcd(e, \phi) = 1$. For instance $e = 7$
- $1 = \gcd(e, \phi) = 3 \cdot e + (-1) \cdot \phi$. Hence $d = 3$
- $PK = (n, e) = (33, 7)$ and $SK = d = 3$
- The message space is $\mathbb{Z}_{33}^{\star} = \{1, 2, 4, 5, 7, 8, 10, 13, 14, 16, 17, 19, 20, 23, 25, 26, 28, 29, 31, 32\}$
- Encrypt $m = 4$ using RSA encryption with $PK = (33, 7)$
  - $C = 4^7 \mod 33 \equiv 4^3 \cdot 4^3 \cdot 4 \equiv (-2) \cdot (-2) \cdot 4 \equiv 16 \mod 33$

Toy Example

- $p = 3, q = 11, N = 33, \phi = 20$
- Choose e s.t. $\gcd(e, \phi) = 1$. For instance $e = 7$
- $1 = \gcd(e, \phi) = 3 \cdot e + (-1) \cdot \phi$. Hence $d = 3$
- $PK = (n, e) = (33, 7)$ and $SK = d = 3$
- The message space is $\mathbb{Z}_{33}^{\star} = \{1, 2, 4, 5, 7, 8, 10, 13, 14, 16, 17, 19, 20, 23, 25, 26, 28, 29, 31, 32\}$
- Encrypt $m = 4$ using RSA encryption with $PK = (33, 7)$
  - $C = 4^7 \mod 33 \equiv 4^3 \cdot 4^3 \cdot 4 \equiv (-2) \cdot (-2) \cdot 4 \equiv 16 \mod 33$
- Recover m using RSA decryption with $SK = 3$
  - $m = 16^3 \mod 33 \equiv 2^{12} \equiv 2^5 \cdot 2^5 \cdot 4 \equiv (-1) \cdot (-1) \cdot 4 \equiv 4 \mod 33$

Modular exponentiation

Let $e_{k-1}e_{k-2}\ldots e_1e_0$ be the binary representation of $e \in \mathbb{N}$

- We need to compute $x^e \mod N$
- Naive method: multiplying x in total e times by itself modulo N
- Slow: if e is 100 bits, roughly 2^{100} multiplications!

Modular exponentiation

Let $e_{k-1}e_{k-2}\dots e_1e_0$ be the binary representation of $e \in \mathbb{N}$

- We need to compute $x^e \mod N$
- Naive method: multiplying x in total e times by itself modulo N
- Slow: if e is 100 bits, roughly 2^{100} multiplications!

Better: use the **square-and-multiply** algorithm for fast modular exponentiation:

`int` **ModPower**(`int` x, N, `bit-string` $e_{k-1}e_{k-2}\dots e_1e_0$)

> $y \leftarrow x$
> for $i \leftarrow k-2$ downto 0 do
>> $y \leftarrow y^2 \cdot x^{e_i} \mod N$
>
> return y

http://wrean.ca/cazelais/rsa.pdf
http://pages.pacificcoast.net/~cazelais/documents.html
http://www.sfs.uni-tuebingen.de/~adriane/2006/winter/384/handouts/decimal-binary.pdf

Alternative RSA decryption

We need to compute

- Given a ciphertext c, to recover m, compute:

$$m = c^d \mod N$$

where $N = p \cdot q$. With knowledge of p, q we can proceed as follows:

- Compute
$$c_p = c^{d_p} \mod p$$
$$c_q = c^{d_q} \mod q$$

  where
$$d_p = d \mod p - 1$$
$$d_q = d \mod q - 1$$

and apply the Chinese Remainder Theorem as

$$m = q \cdot (q^{-1} \mod p) \cdot c_p + p \cdot (p^{-1} \mod q) \cdot c_q$$

Alternative RSA decryption: an example

- We need to compute $m = c^d \mod N$, where $c = 82, d = 29, N = 91$
- We'll start by computing $c_p = 91^{d_p} \mod 7$ and $c_q = 91^{d_q} \mod 13$
- $d_p = d \mod p - 1 = 29 \mod 6 \equiv 5$
- $d_q = d \mod q - 1 = 29 \mod 12 \equiv 5$
- $c_p = 82^5 \mod 7 \equiv 5^5 \equiv (-2)^5 \equiv -4 \equiv 3 \mod 7$
- $c_q = 82^5 \mod 13 \equiv 4^5 \equiv (2)^5 \cdot (2)^5 \equiv 6 \cdot 6 \equiv 10 \mod 13$
- $7^{-1} \mod 13 = 2$ since $7 \cdot 2 \mod 13 = 1$
- $13^{-1} \mod 7 = 6$ since $13 \cdot 6 \equiv 6 \cdot 6 \mod 7 = 1$
- $m = q \cdot (q^{-1} \mod p) \cdot c_p + p \cdot (p^{-1} \mod q) \cdot c_q$
- $m \mod 91 = 13 \cdot 6 \cdot 3 + 7 \cdot 2 \cdot 10 \equiv 374 \equiv 10 \mod 91$

Chinese Remainder Theorem

Theorem

Let $n_1, n_2 > 0$ *integers such that* $\gcd(n_1, n_2) = 1$. *For all* $a, b \in \mathbb{Z}$ *there exists a unique solution in* $\mathbb{Z}_{n_1 \cdot n_2}$ *to the equation*

$$\begin{aligned} x &\equiv a \mod n_1 \\ x &\equiv b \mod n_2 \end{aligned}$$

Furthermore $x = n_2 \cdot i_1 \cdot a + n_1 \cdot i_2 \cdot b$, *where*

$$\begin{aligned} i_1 &= (n_2)^{-1} \mod n_1 \\ i_2 &= (n_1)^{-1} \mod n_2 \end{aligned}$$

i.e. $x = n_2 \cdot ((n_2)^{-1} \mod n_1) \cdot a + n_1 \cdot ((n_1)^{-1} \mod n_2) \cdot b$

Chinese Remainder Theorem

Indeed, let

$$x = n_2 \cdot ((n_2)^{-1} \mod n_1) \cdot a + n_1 \cdot ((n_1)^{-1} \mod n_2) \cdot b$$

then

- $x \mod n_1 \equiv n_2 \cdot ((n_2)^{-1} \mod n_1) \cdot a \equiv a \mod n_1$
- $x \mod n_2 \equiv n_1 \cdot ((n_1)^{-1} \mod n_2) \cdot b \equiv b \mod n_2$

Proof that decryption works

- Since $e \cdot d \equiv 1 \mod \phi$, there is an integer k such that $e \cdot d = 1 + k \cdot \phi$.
- If $m \neq 0 \mod p$, then by Fermat's little theorem $m^{p-1} \equiv 1 \mod p$, which gives :

$$m^{1+k\cdot(p-1)\cdot(q-1)} \equiv m \mod p$$

  - This gives $m^{ed} \equiv m \mod p$ for all m.
  - Similarly, $m^{ed} \equiv m \mod q$ for all m.
  - By the Chinese Remainder Theorem, if $p \neq q$, then

$$m^{ed} \equiv m \mod N$$

Attacks against RSA

- **Factoring large integers**: given $N = p \cdot q$ compute p, q
  - Best factoring algorithm: Number Field Sieve
  - Sub-exponential complexity

$$\exp\left((c + \circ(1))\, n^{1/3} \log^{2/3} n\right)$$

    for n-bit integer.
  - Current factoring record (2009): 768-bit RSA modulus (232 digits)
- **Equivalence** between factoring and breaking RSA **?**

Conjecture: breaking RSA is hard

- Breaking RSA (with λ bits security):
  - Given (N, e) and y chosen at random, for λ-bit N, find x such that $y \equiv x^e \mod N$
- Factoring : given $N = p \cdot q$ for p, q chosen at random and λ-bit N, compute p, q
- Open problem
  - Is breaking RSA equivalent to factoring?
- Knowing d is equivalent to factoring
  - Probabilistic algorithm (RSA, 1978)
  - Deterministic algorithm (A. May 2004, J.S. Coron and A. May 2007)

Key sizes (NIST 2012 recommendations)

Date	Minimum of Strength	Symmetric Algorithms	Factoring Modulus	Discrete Logarithm Key	Discrete Logarithm Group	Elliptic Curve	Hash (A)	Hash (B)
2010 (Legacy)	80	2TDEA*	1024	160	1024	160	SHA-1** SHA-224 SHA-256 SHA-384 SHA-512	SHA-1 SHA-224 SHA-256 SHA-384 SHA-512
2011 - 2030	112	3TDEA	2048	224	2048	224	SHA-224 SHA-256 SHA-384 SHA-512	SHA-1 SHA-224 SHA-256 SHA-384 SHA-512
> 2030	128	AES-128	3072	256	3072	256	SHA-256 SHA-384 SHA-512	SHA-1 SHA-224 SHA-256 SHA-384 SHA-512
>> 2030	192	AES-192	7680	384	7680	384	SHA-384 SHA-512	SHA-224 SHA-256 SHA-384 SHA-512
>>> 2030	256	AES-256	15360	512	15360	512	SHA-512	SHA-256 SHA-384 SHA-512

`https://www.keylength.com/`

Attacks against RSA

- Factoring
  - Equivalence between factoring and breaking RSA ?
- Mathematical attacks
  - Attacks against plain RSA encryption (and signature)
  - Low private / public exponent attacks
  - **Solution:** Provably secure constructions

Elementary attacks against plain RSA encryption

- Plain RSA encryption: dictionary attack
  - If only two possible messages m_0 and m_1, then only $c_0 = (m_0)^e \mod N$ and $c_1 = (m_1)^e \mod N$
    $\Rightarrow$ encryption must be probabilistic
- Plain RSA encryption: malleability attack
  - Given an encryption $c = m^e \mod N$ for an unknown message m it is possible to create a encryption of $m' = \lambda \cdot m \mod N$ by computing

$$c' = (m)^e \cdot \lambda^e = (m \cdot \lambda)^e \mod N$$

$\Rightarrow$ encryption must be non-malleable

Elementary attacks against RSA with padding

```
Network Working Group                                         B. Kaliski
Request for Comments: 2313                       RSA Laboratories East
Category: Informational                                      March 1998
```

PKCS #1: RSA Encryption
Version 1.5

Status of this Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Overview

This document describes a method for encrypting data using the RSA public-key cryptosystem.

- PKCS#1 v1.5
  - $\mu(m) = 0002 \| r \| 00 \| m$
  - $c = \mu(m)^e \mod N$
  - Still insufficient (Bleichenbacher's attack, 1998)

Basic Encryption Security

Definition (One-Wayness)

One-wayness under chosen-plaintext attack (OW-CPA) game is played between the challenger and an attacker

- The challenger runs $(PK, SK) \leftarrow \mathsf{KG}(\lambda)$ and passes the public key PK to the attacker
- The challenger selects a m from the message space at random
- The challenger returns $C = \mathsf{Enc}(PK, m)$ to the attacker, where r is randomness local to running Enc
- The attacker performs a polynomial number of computations and outputs a message m'

The attacker wins this game if $m' = m$

One-wayness

Intuitively, we call a public key encryption scheme *OW-CPA secure*, simply **one-way**, if the attacker cannot compute m correctly, i.e. wins the game almost never.

Definition

Let $Pr[m = m']$ be the probability that the attacker wins the OW-CPA game, taken over all randomness involved in the game. A PKE scheme satisfies *one-wayness* (OW) if

$$\left|Pr[m = m']\right|$$

is negligible as a function of λ

Defense against dictionary attacks

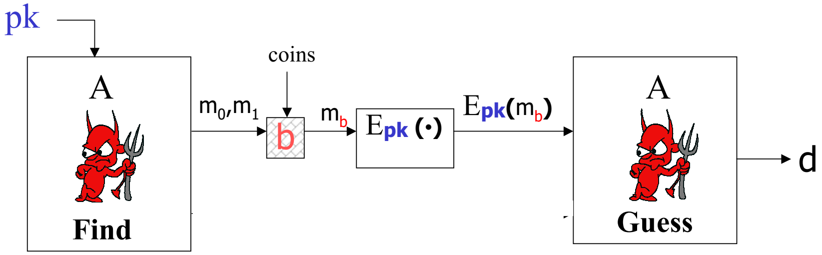
Definition (Semantic security)

Indistinguishability under chosen-plaintext attack (IND-CPA) game is played between the challenger and an attacker

- The challenger runs $(PK, SK) \leftarrow \mathrm{KG}(\lambda)$ and passes the public key pk to the attacker
- The attacker performs a polynomial number of computations
- The attacker submits two messages m_0 and m_1 of equal length to the challenger
- The challenger selects a bit $b \in \{0, 1\}$ at random
- The challenger returns $C_b = \mathrm{Enc}(PK, m_b)$ to the attacker
- The attacker performs a polynomial number of computations and outputs a bit b'

The attacker wins this game if $b' = b$

IND-CPA game



IND-CPA security

Intuitively, we call a public key encrytion scheme *IND-CPA secure* if the attacker cannot do better than guessing the bit b, i.e. wins the game at most half the time.

Definition

Let $Pr[b = b']$ be the probability that the attacker wins the IND-CPA game, taken over all randomness involved in the game. A PKE scheme satisfies *indistinguishability under chosen-plaintext attack* (IND-CPA) if

$$\left| Pr[b = b'] - \frac{1}{2} \right|$$

is negligible as a function of λ

Encrypting messages of arbitrary length

Can encrypt arbitrarily large messages by splitting them up into blocks of suitable size and encrypting each block separately

Theorem

If public-key encryption scheme is IND-CPA-secure, encrypting arbitrarily large messages by splitting them into blocks of suitable size and encrypting each block separately with the same key is IND-CPA secure

IND-CPA secure public-key encryption

Several possibilities to achieve IND-CPA secure public-key encryption

First possibility: add suitable padding (PKCS) to RSA

IND-CPA secure public-key encryption

Second possibility: encrypt random number rather than message
(H is hash function)

- Encryption: choose random r, ciphertext is $(E_{PK}(r), H(r) \oplus m)$
- Decryption: Given (c_1, c_2), compute message as $H(D_{SK}(c_1)) \oplus c_2$

Intuitively: IND-CPA satisfied because attacker cannot decrypt c_1, hence second component looks like one-time pad
Formal proof surprisingly difficult - requires new ideas

RSA-based IND-CPA encryption

Let $F_{(N,e)} : \mathbb{Z}_N^\star \to \mathbb{Z}_N^\star$ be $F_{(N,e)} = x^e \mod N$ be the RSA Trapdoor One-Way Permutation.

Let $H : \{0,1\}^\star \to \{0,1\}^\tau$ for $\tau \in \mathbb{Z}_+$ Let us build an IND-CPA PKE scheme:

- $\mathrm{Enc}((N,e), m; r)$: to encrypt $m \in \{0,1\}^\tau$, choose random r in $\mathbb{Z}_n^\star$. The ciphertext is $\left(F_{(N,e)}(r), H(r) \oplus m\right) \in \mathbb{Z}_n^\star \times \{0,1\}^\tau$
- $\mathrm{Dec}((N,e,d), C)$: Given $C = (c_1, c_2) \in \mathbb{Z}_n^\star \times \{0,1\}^\tau$, compute message as $m = H\left(F_{(N,e,d)}^{-1}(c_1)\right) \oplus c_2$

Intuitively: IND-CPA satisfied because attacker cannot decrypt c_1, hence second component looks like one-time pad

Formal proof is involved - requires tools and formal reasoning we have not used yet

Attacks against RSA

- Factoring
  - Equivalence between factoring and breaking RSA ?
- Mathematical attacks
  - Attacks against plain RSA encryption (and signature)
  - Low private / public exponent attacks
  - **Solution:** Provably secure constructions
- Implementation attacks
  - Timing attacks, power attacks and fault attacks
  - **Solution:** Countermeasures