UC SANTA CRUZ

Math 134: Cryptography

Lecture 7: RSA encryption

Eoin Mackall

January 22, 2025

University of California, Santa Cruz

Last time

Last time

Definition

Let $\phi : \mathbb{N} \to \mathbb{N}$ *be defined on* $n \in \mathbb{N}$ *by*

$$\phi(n) = \#\{d \in \mathbb{N} : 1 \leq d \leq n \text{ and } \gcd(d, n) = 1\}.$$

The function ϕ *is called Euler's totient function.*

Last time

Definition

Let $\phi : \mathbb{N} \to \mathbb{N}$ be defined on $n \in \mathbb{N}$ by

$$\phi(n) = \#\{d \in \mathbb{N} : 1 \leq d \leq n \text{ and } \gcd(d, n) = 1\}.$$

The function ϕ is called Euler's totient function.

Example

Let $n = 30$. The integers between 1 and 30 are as follows:

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

20 21 22 23 24 25 26 27 28 29 30.

Last time

Definition

Let $\phi : \mathbb{N} \to \mathbb{N}$ be defined on $n \in \mathbb{N}$ by

$$\phi(n) = \#\{d \in \mathbb{N} : 1 \leq d \leq n \text{ and } \gcd(d, n) = 1\}.$$

The function ϕ is called Euler's totient function.

Example

Let $n = 30$. The integers between 1 and 30 are as follows:

$$1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 10\ 11\ 12\ 13\ 14\ 15\ 16\ 17\ 18\ 19$$

$$20\ 21\ 22\ 23\ 24\ 25\ 26\ 27\ 28\ 29\ 30.$$

We can write $30 = 2 \cdot 3 \cdot 5$. So a number d with $1 \leq d \leq 30$ which is coprime to 30 is not a multiple of $2, 3$ or 5.

Last time

Definition

Let $\phi : \mathbb{N} \to \mathbb{N}$ be defined on $n \in \mathbb{N}$ by

$$\phi(n) = \#\{d \in \mathbb{N} : 1 \leq d \leq n \text{ and } \gcd(d, n) = 1\}.$$

The function ϕ is called Euler's totient function.

Example

Let $n = 30$. The integers between 1 and 30 are as follows:

$$1\ \cancel{2}\ \cancel{3}\ \cancel{4}\ \cancel{5}\ \cancel{6}\ 7\ \cancel{8}\ \cancel{9}\ \cancel{10}\ 11\ \cancel{12}\ 13\ \cancel{14}\ \cancel{15}\ \cancel{16}\ 17\ \cancel{18}\ 19$$

$$\cancel{20}\ \cancel{21}\ \cancel{22}\ 23\ \cancel{24}\ \cancel{25}\ \cancel{26}\ \cancel{27}\ \cancel{28}\ 29\ \cancel{30}.$$

We can write $30 = 2 \cdot 3 \cdot 5$. So a number d with $1 \leq d \leq 30$ which is coprime to 30 is not a multiple of $2, 3$ or 5.

Last time

Definition

Let $\phi : \mathbb{N} \to \mathbb{N}$ be defined on $n \in \mathbb{N}$ by

$$\phi(n) = \#\{d \in \mathbb{N} : 1 \leq d \leq n \text{ and } \gcd(d, n) = 1\}.$$

The function ϕ is called Euler's totient function.

Example

Let $n = 30$. The integers between 1 and 30 are as follows:

$$1\ \cancel{2}\ \cancel{3}\ \cancel{4}\ \cancel{5}\ \cancel{6}\ 7\ \cancel{8}\ \cancel{9}\ \cancel{10}\ 11\ \cancel{12}\ 13\ \cancel{14}\ \cancel{15}\ \cancel{16}\ 17\ \cancel{18}\ 19$$

$$\cancel{20}\ \cancel{21}\ \cancel{22}\ 23\ \cancel{24}\ \cancel{25}\ \cancel{26}\ \cancel{27}\ \cancel{28}\ 29\ \cancel{30}.$$

We can write $30 = 2 \cdot 3 \cdot 5$. So a number d with $1 \leq d \leq 30$ which is coprime to 30 is not a multiple of $2, 3$ or 5. So $\phi(30) = \#\{1, 7, 11, 13, 17, 19, 23, 29\} = 8$

Properties of Euler's ϕ function

- If p is a prime number, then $\phi(p) = p - 1$.

Last time

Properties of Euler's ϕ function

- If p is a prime number, then $\phi(p) = p - 1$.
- If p^r is a prime power, for some $r \geq 1$, then

$$\phi(p^r) = \left(1 - \frac{1}{p}\right) p^r.$$

Last time

Properties of Euler's ϕ function

- If p is a prime number, then $\phi(p) = p - 1$.
- If p^r is a prime power, for some $r \geq 1$, then

$$\phi(p^r) = \left(1 - \frac{1}{p}\right) p^r.$$

- If $a, b \in \mathbb{N}$ are coprime, then $\phi(ab) = \phi(a) \cdot \phi(b)$. In particular, for any number $n \in \mathbb{N}$ we have

$$\phi(n) = n \cdot \prod_{p|n,\, p \text{ prime}} \left(1 - \frac{1}{p}\right).$$

Last time

Example

For $30 = 2 \cdot 3 \cdot 5$ we have

$$\phi(30) = 30 \cdot (1 - 1/2)(1 - 1/3)(1 - 1/5)$$

Last time

Example

For $30 = 2 \cdot 3 \cdot 5$ we have

$$\begin{aligned} \phi(30) &= 30 \cdot (1 - 1/2)(1 - 1/3)(1 - 1/5) \\ &= 30 \cdot (1/2)(2/3)(4/5) \end{aligned}$$

Example

For $30 = 2 \cdot 3 \cdot 5$ we have

$$\begin{aligned}\phi(30) &= 30 \cdot (1 - 1/2)(1 - 1/3)(1 - 1/5) \\ &= 30 \cdot (1/2)(2/3)(4/5) \\ &= 8.\end{aligned}$$

Last time

Example

For $30 = 2 \cdot 3 \cdot 5$ we have

$$\begin{aligned}\phi(30) &= 30 \cdot (1 - 1/2)(1 - 1/3)(1 - 1/5) \\ &= 30 \cdot (1/2)(2/3)(4/5) \\ &= 8.\end{aligned}$$

Example

For $162 = 2 \cdot 3^4$ we have

$$\phi(162) = 162 \cdot (1 - 1/2)(1 - 1/3)$$

Last time

Example

For $30 = 2 \cdot 3 \cdot 5$ we have

$$\begin{aligned}\phi(30) &= 30 \cdot (1 - 1/2)(1 - 1/3)(1 - 1/5) \\ &= 30 \cdot (1/2)(2/3)(4/5) \\ &= 8.\end{aligned}$$

Example

For $162 = 2 \cdot 3^4$ we have

$$\begin{aligned}\phi(162) &= 162 \cdot (1 - 1/2)(1 - 1/3) \\ &= 162 \cdot (1/2)(2/3)\end{aligned}$$

Last time

Example

For $30 = 2 \cdot 3 \cdot 5$ we have

$$\begin{aligned}\phi(30) &= 30 \cdot (1 - 1/2)(1 - 1/3)(1 - 1/5) \\ &= 30 \cdot (1/2)(2/3)(4/5) \\ &= 8.\end{aligned}$$

Example

For $162 = 2 \cdot 3^4$ we have

$$\begin{aligned}\phi(162) &= 162 \cdot (1 - 1/2)(1 - 1/3) \\ &= 162 \cdot (1/2)(2/3) \\ &= 54.\end{aligned}$$

Last time

Let $n > 1$ be an integer, and let $a \in \mathbb{Z}$ be an integer with $\gcd(a, n) = 1$.

Last time

Let $n > 1$ be an integer, and let $a \in \mathbb{Z}$ be an integer with $\gcd(a, n) = 1$.

Theorem (Euler's Theorem on modular exponentiation)

For integers n and a as above, we have

$$a^{\phi(n)} \equiv 1 \pmod{n}$$

where $\phi(n)$ is Euler's totient function.

Last time

Example

We have $\phi(30) = 8$. Also, $\gcd(7, 30) = 1$.

Example

We have $\phi(30) = 8$. Also, $\gcd(7, 30) = 1$.

$$7^2 \equiv 49 \equiv 19 \pmod{30}$$

Last time

Example

We have $\phi(30) = 8$. Also, $\gcd(7, 30) = 1$.

$$\begin{aligned} 7^2 &\equiv 49 \equiv 19 \pmod{30} \\ 7^4 &\equiv (19)^2 \equiv 361 \equiv 1 \pmod{30} \end{aligned}$$

Example

We have $\phi(30) = 8$. Also, $\gcd(7, 30) = 1$.

$$\begin{aligned} 7^2 &\equiv 49 \equiv 19 \pmod{30} \\ 7^4 &\equiv (19)^2 \equiv 361 \equiv 1 \pmod{30} \\ 7^8 &\equiv 1^2 \equiv 1 \pmod{30}. \end{aligned}$$

Example

We have $\phi(30) = 8$. Also, $\gcd(7, 30) = 1$.

$$7^2 \equiv 49 \equiv 19 \pmod{30}$$
$$7^4 \equiv (19)^2 \equiv 361 \equiv 1 \pmod{30}$$
$$7^8 \equiv 1^2 \equiv 1 \pmod{30}.$$

Example

We have $\phi(15) = 8$ also. And $\gcd(11, 15) = 1$.

Last time

Example

We have $\phi(30) = 8$. Also, $\gcd(7, 30) = 1$.

$$7^2 \equiv 49 \equiv 19 \pmod{30}$$
$$7^4 \equiv (19)^2 \equiv 361 \equiv 1 \pmod{30}$$
$$7^8 \equiv 1^2 \equiv 1 \pmod{30}.$$

Example

We have $\phi(15) = 8$ also. And $\gcd(11, 15) = 1$.

$$(11)^2 \equiv 121 \equiv 1 \pmod{15}$$

Last time

Example

We have $\phi(30) = 8$. Also, $\gcd(7, 30) = 1$.

$$\begin{aligned} 7^2 &\equiv 49 \equiv 19 \pmod{30} \\ 7^4 &\equiv (19)^2 \equiv 361 \equiv 1 \pmod{30} \\ 7^8 &\equiv 1^2 \equiv 1 \pmod{30}. \end{aligned}$$

Example

We have $\phi(15) = 8$ also. And $\gcd(11, 15) = 1$.

$$\begin{aligned} (11)^2 &\equiv 121 \equiv 1 \pmod{15} \\ (11)^8 &\equiv 1^4 \equiv 1 \pmod{15} \end{aligned}$$

Table of contents

RSA encrytpion scheme

Private Key Encryption

Alice and Bob want to privately communicate.

Private Key (Symmetric)

Private Key Encryption

Alice and Bob want to privately communicate.

Private Key (Symmetric)

- In a private key encryption scheme, Alice and Bob agree on an encryption algorithm beforehand.

Private Key Encryption

Alice and Bob want to privately communicate.

Private Key (Symmetric)

- In a private key encryption scheme, Alice and Bob agree on an encryption algorithm beforehand.
- Alice and Bob share a *private key* that *they both use* for encryption and decryption.

Alice and Bob want to privately communicate.

Private Key (Symmetric)

- In a private key encryption scheme, Alice and Bob agree on an encryption algorithm beforehand.
- Alice and Bob share a *private key* that *they both use* for encryption and decryption.
- As an analogy, this is like the following:

Private Key Encryption

Alice and Bob want to privately communicate.

Private Key (Symmetric)

- In a private key encryption scheme, Alice and Bob agree on an encryption algorithm beforehand.
- Alice and Bob share a *private key* that *they both use* for encryption and decryption.
- As an analogy, this is like the following:
  - Alice and Bob meet up and make two keys for one lock.

Private Key Encryption

Alice and Bob want to privately communicate.

Private Key (Symmetric)

- In a private key encryption scheme, Alice and Bob agree on an encryption algorithm beforehand.
- Alice and Bob share a *private key* that *they both use* for encryption and decryption.
- As an analogy, this is like the following:
  - Alice and Bob meet up and make two keys for one lock.
  - Later, if Alice wants to send a message to Bob, she sends a box with a message in it locked with their shared lock.

Private Key Encryption

Alice and Bob want to privately communicate.

Private Key (Symmetric)

- In a private key encryption scheme, Alice and Bob agree on an encryption algorithm beforehand.
- Alice and Bob share a *private key* that *they both use* for encryption and decryption.
- As an analogy, this is like the following:
  - Alice and Bob meet up and make two keys for one lock.
  - Later, if Alice wants to send a message to Bob, she sends a box with a message in it locked with their shared lock.
  - When Bob gets this box, he may open it with his copy of the key.

Public Key Encryption

Alice wants to send a message to Bob.

Public Key (Asymmetric)

Public Key Encryption

Alice wants to send a message to Bob.

Public Key (Asymmetric)

- In a public encryption scheme, one party (Bob) picks the encryption algorithm that will be used.

Public Key Encryption

Alice wants to send a message to Bob.

Public Key (Asymmetric)

- In a public encryption scheme, one party (Bob) picks the encryption algorithm that will be used.
- Before a message is sent, Bob chooses two keys. One is for encryption; anyone is allowed to see this one. The other is for decryption; Bob keeps this one secret.

Public Key Encryption

Alice wants to send a message to Bob.

Public Key (Asymmetric)

- In a public encryption scheme, one party (Bob) picks the encryption algorithm that will be used.
- Before a message is sent, Bob chooses two keys. One is for encryption; anyone is allowed to see this one. The other is for decryption; Bob keeps this one secret.
- For Alice to send a message to Bob, she first needs Bob to give her the encryption key. She then encrypts her message using this key and sends the result to Bob.

Public Key Encryption

Alice wants to send a message to Bob.

Public Key (Asymmetric)

- In a public encryption scheme, one party (Bob) picks the encryption algorithm that will be used.
- Before a message is sent, Bob chooses two keys. One is for encryption; anyone is allowed to see this one. The other is for decryption; Bob keeps this one secret.
- For Alice to send a message to Bob, she first needs Bob to give her the encryption key. She then encrypts her message using this key and sends the result to Bob.
- As an analogy, this is like the following:

Public Key Encryption

Alice wants to send a message to Bob.

Public Key (Asymmetric)

- In a public encryption scheme, one party (Bob) picks the encryption algorithm that will be used.
- Before a message is sent, Bob chooses two keys. One is for encryption; anyone is allowed to see this one. The other is for decryption; Bob keeps this one secret.
- For Alice to send a message to Bob, she first needs Bob to give her the encryption key. She then encrypts her message using this key and sends the result to Bob.
- As an analogy, this is like the following:
    - Alice notifies Bob that she wants to send him a message.

Public Key Encryption

Alice wants to send a message to Bob.

Public Key (Asymmetric)

- In a public encryption scheme, one party (Bob) picks the encryption algorithm that will be used.
- Before a message is sent, Bob chooses two keys. One is for encryption; anyone is allowed to see this one. The other is for decryption; Bob keeps this one secret.
- For Alice to send a message to Bob, she first needs Bob to give her the encryption key. She then encrypts her message using this key and sends the result to Bob.
- As an analogy, this is like the following:
    - Alice notifies Bob that she wants to send him a message.
    - Bob sends Alice a box together with an unlocked lock.

Public Key Encryption

Alice wants to send a message to Bob.

Public Key (Asymmetric)

- In a public encryption scheme, one party (Bob) picks the encryption algorithm that will be used.
- Before a message is sent, Bob chooses two keys. One is for encryption; anyone is allowed to see this one. The other is for decryption; Bob keeps this one secret.
- For Alice to send a message to Bob, she first needs Bob to give her the encryption key. She then encrypts her message using this key and sends the result to Bob.
- As an analogy, this is like the following:
    - Alice notifies Bob that she wants to send him a message.
    - Bob sends Alice a box together with an unlocked lock.
    - Alice puts her message in the box, locks it with Bob's lock, and sends it to Bob.

Public Key Encryption

Alice wants to send a message to Bob.

Public Key (Asymmetric)

- In a public encryption scheme, one party (Bob) picks the encryption algorithm that will be used.
- Before a message is sent, Bob chooses two keys. One is for encryption; anyone is allowed to see this one. The other is for decryption; Bob keeps this one secret.
- For Alice to send a message to Bob, she first needs Bob to give her the encryption key. She then encrypts her message using this key and sends the result to Bob.
- As an analogy, this is like the following:
    - Alice notifies Bob that she wants to send him a message.
    - Bob sends Alice a box together with an unlocked lock.
    - Alice puts her message in the box, locks it with Bob's lock, and sends it to Bob.
    - When Bob gets this box, he may open it with his secret key.

Public Key Encryption

There are some subtle differences between private and public key encryption outside of the structure of the schemes themselves.

Public vs private

Public Key Encryption

There are some subtle differences between private and public key encryption outside of the structure of the schemes themselves.

Public vs private

- In private key encryption, Bob can be sure that the message he receives came from Alice (or someone with Alice's key).

Public Key Encryption

There are some subtle differences between private and public key encryption outside of the structure of the schemes themselves.

Public vs private

- In private key encryption, Bob can be sure that the message he receives came from Alice (or someone with Alice's key).
- On the other hand, in a public key encryption scheme, Bob has no way to know if Alice really sent the message he receives.

The RSA encryption algorithm

Alice wants to send a message to Bob. Bob has indicated that he wants to use RSA. What is the process for encryption?

The RSA encryption algorithm

Alice wants to send a message to Bob. Bob has indicated that he wants to use RSA. What is the process for encryption?

1. Bob picks beforehand two (large) prime numbers p, q and computes their product $n = pq$.

The RSA encryption algorithm

Alice wants to send a message to Bob. Bob has indicated that he wants to use RSA. What is the process for encryption?

1. Bob picks beforehand two (large) prime numbers p, q and computes their product $n = pq$.
2. Bob chooses an integer e with $\gcd(e, \phi(n)) = 1$.

The RSA encryption algorithm

Alice wants to send a message to Bob. Bob has indicated that he wants to use RSA. What is the process for encryption?

1. Bob picks beforehand two (large) prime numbers p, q and computes their product $n = pq$.
2. Bob chooses an integer e with $\gcd(e, \phi(n)) = 1$.
3. Bob sends Alice the pair (n, e). The pair (n, e) is Bob's *public* key.

The RSA encryption algorithm

Alice wants to send a message to Bob. Bob has indicated that he wants to use RSA. What is the process for encryption?

1. Bob picks beforehand two (large) prime numbers p, q and computes their product $n = pq$.
2. Bob chooses an integer e with $\gcd(e, \phi(n)) = 1$.
3. Bob sends Alice the pair (n, e). The pair (n, e) is Bob's *public* key.
4. Alice will send her message in the form of an integer $m \in \mathbb{Z}$. If $m > n$, then Alice breaks $m = m_1 m_2 m_3 ...$ into blocks m_i of sizes less than n.

The RSA encryption algorithm

Alice wants to send a message to Bob. Bob has indicated that he wants to use RSA. What is the process for encryption?

1. Bob picks beforehand two (large) prime numbers p, q and computes their product $n = pq$.
2. Bob chooses an integer e with $\gcd(e, \phi(n)) = 1$.
3. Bob sends Alice the pair (n, e). The pair (n, e) is Bob's *public* key.
4. Alice will send her message in the form of an integer $m \in \mathbb{Z}$. If $m > n$, then Alice breaks $m = m_1 m_2 m_3 ...$ into blocks m_i of sizes less than n.
5. For each block m_i, Alice computes the number $0 \leq r_i < n$ with $m_i^e \equiv r_i \pmod{n}$. She then sends Bob the numbers $r_1, r_2, r_3, ...$.

The RSA encryption algorithm

Alice wants to send a message to Bob. Bob has indicated that he wants to use RSA. What is the process for encryption?

1. Bob picks beforehand two (large) prime numbers p, q and computes their product $n = pq$.
2. Bob chooses an integer e with $\gcd(e, \phi(n)) = 1$.
3. Bob sends Alice the pair (n, e). The pair (n, e) is Bob's *public* key.
4. Alice will send her message in the form of an integer $m \in \mathbb{Z}$. If $m > n$, then Alice breaks $m = m_1 m_2 m_3 ...$ into blocks m_i of sizes less than n.
5. For each block m_i, Alice computes the number $0 \leq r_i < n$ with $m_i^e \equiv r_i \pmod{n}$. She then sends Bob the numbers $r_1, r_2, r_3, ...$

Note

$\phi(n) = \phi(pq) = \phi(p)\phi(q) = (p-1)(q-1)$

Example

Let's encrypt the message `Banana slugs actually do have known predators` using an RSA encryption scheme with public key $(n, e) = (1125897758834689, 65537)$.

Example

Let's encrypt the message `Banana slugs actually do have known predators` using an RSA encryption scheme with public key $(n, e) = (1125897758834689, 65537)$.

Remark

Where did these numbers come from? The number $1125897758834689 = 524287 \cdot 2147483647$ is the product of two Mersenne primes. The number 65537 is a Fermat prime. In general, picking these numbers is its own problem.

Example

Let's encrypt the message `Banana slugs actually do have known predators` using an RSA encryption scheme with public key $(n, e) = (1125897758834689, 65537)$.

The phrase `Banana slugs actually do have known predators` converts to the following Binary using Ascii encoding:

```
01000010 01100001 01101110 01100001 01101110
01100001 00100000 01110011 01101100 01110101
01100111 01110011 00100000 01100001 01100011
01110100 01110101 01100001 01101100 01101100
01111001 00100000 01100100 01101111 00100000
01101000 01100001 01110110 01100101 00100000
01101011 01101110 01101111 01110111 01101110
00100000 01110000 01110010 01100101 01100100
01100001 01110100 01101111 01110010 01110011
```

Example

Let's encrypt the message `Banana slugs actually do have known predators` using an RSA encryption scheme with public key $(n, e) = (1125897758834689, 65537)$.

Converting each row of binary (a concatenation of 5 bytes) to an integer gives the message:

```
285102465390 417156263029 444313133411 500185525356
520234495776 448311747872 461414299502 139325498724
418565288563
```

The RSA encryption algorithm

The message to be sent is:

285102465390 417156263029 444313133411 500185525356 520234495776 448311747872 461414299502 139325498724 418565288563.

The RSA encryption algorithm

The message to be sent is:

`285102465390 417156263029 444313133411 500185525356 520234495776 448311747872 461414299502 139325498724 418565288563`.

We compute:

The message to be sent is:

`285102465390 417156263029 444313133411 500185525356 520234495776 448311747872 461414299502 139325498724 418565288563`.

We compute:

$$285102465390^{65537} \equiv 1060290675860014 \pmod{1125897758834689}$$

The RSA encryption algorithm

The message to be sent is:

`285102465390 417156263029 444313133411 500185525356 520234495776 448311747872 461414299502 139325498724 418565288563`.

We compute:

$$285102465390^{65537} \equiv 1060290675860014 \pmod{1125897758834689}$$
$$417156263029^{65537} \equiv 683711618526171 \pmod{1125897758834689}$$

The message to be sent is:

`285102465390 417156263029 444313133411 500185525356 520234495776 448311747872 461414299502 139325498724 418565288563`.

We compute:

$$285102465390^{65537} \equiv 1060290675860014 \pmod{1125897758834689}$$
$$417156263029^{65537} \equiv 68371161852617 \pmod{1125897758834689}$$
$$444313133411^{65537} \equiv 781739137181126 \pmod{1125897758834689}$$

The RSA encryption algorithm

The message to be sent is:

`285102465390 417156263029 444313133411 500185525356 520234495776 448311747872 461414299502 139325498724 418565288563`.

We compute:

$$
\begin{aligned}
285102465390^{65537} &\equiv 1060290675860014 \pmod{1125897758834689} \\
417156263029^{65537} &\equiv 68371161852617 \pmod{1125897758834689} \\
444313133411^{65537} &\equiv 781739137181126 \pmod{1125897758834689} \\
&\vdots
\end{aligned}
$$

The RSA encryption algorithm

The message to be sent is:

`285102465390 417156263029 444313133411 500185525356 520234495776 448311747872 461414299502 139325498724 418565288563`.

We compute:

$$\begin{aligned} 285102465390^{65537} &\equiv 1060290675860014 \pmod{1125897758834689} \\ 417156263029^{65537} &\equiv 68371161852617 \pmod{1125897758834689} \\ 444313133411^{65537} &\equiv 781739137181126 \pmod{1125897758834689} \\ &\vdots \end{aligned}$$

And we send the encrypted ciphertext message:
`1060290675860014 68371161852617 781739137181126 1061754703811626 791604339591861 22286202176830 4 236496462528961 540900973748831 1023668662869907`.

The RSA decryption algorithm

Bob has received a message from Alice. Bob had indicated that he wanted to use RSA for communication. What is the process for decryption?

The RSA decryption algorithm

Bob has received a message from Alice. Bob had indicated that he wanted to use RSA for communication. What is the process for decryption?

1. Prior to their communication, Bob had selected an integer d with $ed \equiv 1 \pmod{\phi(n)}$. Bob probably did this while computing $\gcd(e, \phi(n)) = 1$ (using, e.g. the Euclidean algorithm). The integer d is Bob's *secret* key.

The RSA decryption algorithm

Bob has received a message from Alice. Bob had indicated that he wanted to use RSA for communication. What is the process for decryption?

1. Prior to their communication, Bob had selected an integer d with $ed \equiv 1 \pmod{\phi(n)}$. Bob probably did this while computing $\gcd(e, \phi(n)) = 1$ (using, e.g. the Euclidean algorithm). The integer d is Bob's *secret* key.
2. Bob receives a sequence of integers $r_1, r_2, r_3, \ldots$ from Alice.

The RSA decryption algorithm

Bob has received a message from Alice. Bob had indicated that he wanted to use RSA for communication. What is the process for decryption?

1. Prior to their communication, Bob had selected an integer d with $ed \equiv 1 \pmod{\phi(n)}$. Bob probably did this while computing $\gcd(e, \phi(n)) = 1$ (using, e.g. the Euclidean algorithm). The integer d is Bob's *secret* key.
2. Bob receives a sequence of integers $r_1, r_2, r_3, \ldots$ from Alice.
3. For each integer r_i, bob computes

$$r_i^d \equiv (m_i^e)^d \equiv m_i^{ed} \equiv m_i \pmod{n}.$$

The RSA decryption algorithm

Bob has received a message from Alice. Bob had indicated that he wanted to use RSA for communication. What is the process for decryption?

1. Prior to their communication, Bob had selected an integer d with $ed \equiv 1 \pmod{\phi(n)}$. Bob probably did this while computing $\gcd(e, \phi(n)) = 1$ (using, e.g. the Euclidean algorithm). The integer d is Bob's *secret* key.
2. Bob receives a sequence of integers $r_1, r_2, r_3, \ldots$ from Alice.
3. For each integer r_i, bob computes

$$r_i^d \equiv (m_i^e)^d \equiv m_i^{ed} \equiv m_i \pmod{n}.$$

4. Bob combines these integers to obtain Alice's message $m = m_1 m_2 m_3 \ldots.$

The RSA decryption algorithm

Example

Let's decrypt the ciphertext message:

`112330206646214 5451538274036662 247266778462632`
`514155951398710 812340948062451`

using an RSA encryption scheme with the same public key $(n, e) = (1125897758834689, 65537)$.

The RSA decryption algorithm

Example

Let's decrypt the ciphertext message:

`112330206646214 5 451538274036662 247266778462632 514155951398710 812340948062451`

using an RSA encryption scheme with the same public key $(n, e) = (1125897758834689, 65537)$.

Using the Euclidean algorithm, we can find $d = 600476513804837$ is an integer with

$$65537 \cdot 600476513804837 \equiv 1 \pmod{524286 \cdot 2147483646}.$$

The RSA decryption algorithm

Example

Let's decrypt the ciphertext message:

`112330206646214 5451538274036662 247266778462632`
`514155951398710 812340948062451`

using an RSA encryption scheme with the same public key $(n, e) = (1125897758834689, 65537)$.

Using the Euclidean algorithm, we can find $d = 600476513804837$ is an integer with

$$65537 \cdot 600476513804837 \equiv 1 \pmod{524286 \cdot 2147483646}.$$

Here we are using the factorization

$$1125897758834689 = 524287 \cdot 2147483647$$

to compute $\phi(1125897758834689) = 524286 \cdot 2147483646$.

Example (Ciphertext)

112330206646214 5451538274036662 2472667784626332
514155951398710 812340948062451.

The RSA decryption algorithm

Example (Ciphertext)

`112330206646214 5451538274036662 247266778462632`
`514155951398710 812340948062451`.

We compute:

The RSA decryption algorithm

Example (Ciphertext)

`112330206646214 5451538274036662 247266778462632 514155951398710 812340948062451`.

We compute:

$$112330206646214 5^{600476513804837} \equiv 289397830432 \pmod{1125897758834689}$$

Example (Ciphertext)

`112330206646214 5451538274036662 247266778462632 514155951398710 812340948062451`.

We compute:

$$112330206646214 5^{600476513804837} \equiv 289397830432 \pmod{1125897758834689}$$

$$451538274036662^{600476513804837} \equiv 418531057766 \pmod{1125897758834689}$$

The RSA decryption algorithm

Example (Ciphertext)

`112330206646214 5 451538274036662 247266778462632 514155951398710 812340948062451`.

We compute:

$$
\begin{aligned}
112330206646214 5^{600476513804837} &\equiv 289397830432 \pmod{1125897758834689}\\
451538274036662^{600476513804837} &\equiv 418531057766 \pmod{1125897758834689}\\
247266778462632^{600476513804837} &\equiv 504363907360 \pmod{1125897758834689}
\end{aligned}
$$

The RSA decryption algorithm

Example (Ciphertext)

`112330206646214 5451538274036662 247266778462632 514155951398710 812340948062451`.

We compute:

$$
\begin{aligned}
112330206646214 5^{600476513804837} &\equiv 289397830432 \pmod{1125897758834689} \\
451538274036662^{600476513804837} &\equiv 418531057766 \pmod{1125897758834689} \\
247266778462632^{600476513804837} &\equiv 504363907360 \pmod{1125897758834689} \\
&\vdots
\end{aligned}
$$

The RSA decryption algorithm

Example (Ciphertext)

`112330206646214 5451538274036662 247266778462632 514155951398710 812340948062451`.

We compute:

$$
\begin{aligned}
112330206646214 5^{600476513804837} &\equiv 289397830432 \pmod{1125897758834689} \\
451538274036662^{600476513804837} &\equiv 418531057766 \pmod{1125897758834689} \\
247266778462632^{600476513804837} &\equiv 504363907360 \pmod{1125897758834689} \\
&\vdots
\end{aligned}
$$

Example (Cont'd)

We find the plaintext message: `289397830432 418531057766 504363907360 418464230753 27763`.

The RSA decryption algorithm

Example (Cont'd)

We find the plaintext message: `289397830432 418531057766 504363907360 418464230753 27763`.

The RSA decryption algorithm

Example (Cont'd)

We find the plaintext message: 289397830432 418531057766 504363907360 418464230753 27763.

Converting these decimal numbers into binary gives:

```
1000011011000001011101000111001100100000
1100001011100100110010100100000001100110
1110101011011100110111001111001001000000
1100001011011100110100101101101011000001
110110001110011
```

The RSA decryption algorithm

Example (Cont'd)

We find the plaintext message: 289397830432 418531057766 504363907360 418464230753 27763.

Converting these decimal numbers into binary gives:

```
100001101100001011101000111001100100000
110000101110010011001010010000001100110
111010101101110011011100111100100100000
110000101101110011010010110110101100001
11011000111001 1
```

Processing this binary sequence to a sequence of bytes gives:

```
01000011 01100001 01110100 01110011 00100000
01100001 01110010 01100101 00100000 01100110
01110101 01101110 01101110 01111001 00100000
01100001 01101110 01101001 01101101 01100001
01101100 01110011
```

Example (Cont'd)

We find the plaintext message: 289397830432 418531057766 504363907360 418464230753 27763.

Processing this binary sequence to a sequence of bytes gives:

01000011 01100001 01110100 01110011 00100000
01100001 01110010 01100101 00100000 01100110
01110101 01101110 01101110 01111001 00100000
01100001 01101110 01101001 01101101 01100001
01101100 01110011

The RSA decryption algorithm

Example (Cont'd)

We find the plaintext message: `289397830432 418531057766 504363907360 418464230753 27763`.

Processing this binary sequence to a sequence of bytes gives:

```
01000011 01100001 01110100 01110011 00100000
01100001 01110010 01100101 00100000 01100110
01110101 01101110 01101110 01111001 00100000
01100001 01101110 01101001 01101101 01100001
01101100 01110011
```

Example (Cont'd)

Converting this sequence of bytes to a string gives: `Cats are funny animals`.