

Forum:[Hack Forums](#)

Thread: [How does RSA work?](#)

rjusticemi - August 31, 2025, 1:27 am

I know how asymmetric encryption works; it works by exchanging public keys, and when someone wants to send a message, the message is encrypted with the recipient's public key, which can only be decrypted with the recipient's private key. I know RSA is one of the most popular asymmetric encryption algorithms, but I wanna know more about how it works

When I search "how does RSA work" on Google, it doesn't actually tell me how the RSA algorithm itself works, but it just tells me the basics of how asymmetric encryption in general works (what I just explained above).

Can someone simply explain how RSA works, and the process of encrypting and decrypting messages using the RSA algorithm?

DramaKing - August 31, 2025, 5:56 am

(Aug 31, 2025 01:27 AM)rjusticemi Wrote: [\[CITE\]](#) I know how asymmetric encryption works; it works by exchanging public keys, and when someone wants to send a message, the message is encrypted with the recipient's public key, which can only be decrypted with the recipient's private key. I know RSA is one of the most popular asymmetric encryption algorithms, but I wanna know more about how it works

When I search "how does RSA work" on Google, it doesn't actually tell me how the RSA algorithm itself works, but it just tells me the basics of how asymmetric encryption in general works (what I just explained above).

Can someone simply explain how RSA works, and the process of encrypting and decrypting messages using the RSA algorithm?

At a basic level, RSA involves generating two large prime numbers and calculating keys from those. Information on the key generation process is easy to find.

<https://www.geeksforgeeks.org/dsa/how-to-generate-large-prime-numbers-for-rsa-algorithm/>

h4ckn4xus - September 10, 2025, 9:52 am

RSA uses two keys: a public key (anyone can use it to send you messages) and a private key (you keep it secret to read those messages).

Choose two big prime numbers:
Let's call them p and q .

Example: $p = 11$, $q = 13$ (small numbers just for illustration).

Compute $n = p \times q$:

$$n = 11 \times 13 = 143$$

n is part of both the public and private keys.

Compute Euler's totient function $\phi(n) = (p-1) \times (q-1)$:

$$\phi(n) = (11-1) \times (13-1) = 10 \times 12 = 120$$

Choose a public exponent e :

e must be coprime with $\phi(n)$ (no common factors besides 1).

Example: $e = 7$ ($\gcd(7, 120) = 1$)

Compute the private exponent d:

d is the multiplicative inverse of e modulo $\phi(n)$

Solve for d such that $(d \times e) \% \phi(n) = 1$

Here, $d = 103$ because $(103 \times 7) \% 120 = 1$

Now you have:

Public key: $(n = 143, e = 7)$

Private key: $(n = 143, d = 103)$

ENCRYPTION:

To send a message m (as a number smaller than n):

Take the message m -> example: $m = 9$

Encrypt it using the public key (n, e) :

$c = 9^7 \% 143$

Calculate:

$9^7 = 4782969$

$4782969 \% 143 = 48$

So, the ciphertext $c = 48$. This is what you send.

DECRYPT:

The receiver uses the private key (n, d) to decrypt:

$m = 48^{103} \% 143$

Using modular arithmetic tricks (you usually use software for big numbers), the result is $m = 9$, recovering the original message.

RESUME:

1-> Generate keys using large primes -> public and private key.

2 -> Encrypt: $c = m^e \% n$ with the public key.

3 -> Decrypt: $m = c^d \% n$ with the private key.

The security of RSA comes from the fact that factoring n into p and q is extremely hard if p and q are large (hundreds of digits).

??????????? - September 10, 2025, 10:08 am

(Sep 10, 2025 09:52 AM)h4ckn4xus Wrote: [\[CITE\]](#) RSA uses two keys: a public key (anyone can use it to send you messages) and a private key (you keep it secret to read those messages).

Choose two big prime numbers:

Let's call them p and q.

Example: $p = 11, q = 13$ (small numbers just for illustration).

Compute $n = p \times q$:

$n = 11 \times 13 = 143$

n is part of both the public and private keys.

Compute Euler's totient function $\phi(n) = (p-1) \times (q-1)$:
 $\phi(n) = (11-1) \times (13-1) = 10 \times 12 = 120$

Choose a public exponent e :
 e must be coprime with $\phi(n)$ (no common factors besides 1).
Example: $e = 7$ ($\gcd(7, 120) = 1$)
Compute the private exponent d :
 d is the multiplicative inverse of e modulo $\phi(n)$

Solve for d such that $(d \times e) \% \phi(n) = 1$
Here, $d = 103$ because $(103 \times 7) \% 120 = 1$

Now you have:
Public key: $(n = 143, e = 7)$
Private key: $(n = 143, d = 103)$

ENCRYPTION:
To send a message m (as a number smaller than n):
Take the message $m \rightarrow$ example: $m = 9$
Encrypt it using the public key (n, e) :
 $c = 9^7 \% 143$

Calculate:
 $9^7 = 4782969$
 $4782969 \% 143 = 48$
So, the ciphertext $c = 48$. This is what you send.

DECRYPT:
The receiver uses the private key (n, d) to decrypt:

$$m = 48^{103} \% 143$$

Using modular arithmetic tricks (you usually use software for big numbers), the result is $m = 9$, recovering the original message.

RESUME:
1-> Generate keys using large primes -> public and private key.

2 -> Encrypt: $c = m^e \% n$ with the public key.

3 -> Decrypt: $m = c^d \% n$ with the private key.

The security of RSA comes from the fact that factoring n into p and q is extremely hard if p and q are large (hundreds of digits).

RSA Encryption - More Simplified

Quote:RSA uses two keys – a public one for encryption, and a private one for decryption.

Example:
Code $p = 11$
 $q = 13$
 $n = p \times q = 143$

$$\phi(n) = (p-1)(q-1) = 120$$

$e = 7$ (coprime with 120)

$d = 103$ (since $(e \times d) \bmod \phi(n) = 1$)

Encrypt: $c = m^e \bmod n$

-> Example: Code $m = 9 \Rightarrow c = 9^7 \bmod 143 = 48$

Decrypt: $m = c^d \bmod n$

-> Code $m = 48^{103} \bmod 143 = 9$

Summary: Public key: $(n=143, e=7)$

Private key: $(n=143, d=103)$

Secure when using large primes + padding (e.g., OAEP)

Keep it padded, keep it safe. Always stay sharp ?????????

KAPALfx - September 10, 2025, 10:10 am

RSA encryption is based on modular arithmetic. You start by generating two large primes, multiply them to form n , and then choose an exponent e as part of the public key. To encrypt, you raise the message to the power of $e \bmod n$. To decrypt, you use the private exponent d to reverse the process. The difficulty of breaking RSA comes from factoring n back into its prime factors.

rjusticemi - September 10, 2025, 10:24 am

(Sep 10, 2025 09:52 AM)h4ckn4xus Wrote: [\[CITE\]](#) RSA uses two keys: a public key (anyone can use it to send you messages) and a private key (you keep it secret to read those messages).

Choose two big prime numbers:

Let's call them p and q .

Example: $p = 11, q = 13$ (small numbers just for illustration).

Compute $n = p \times q$:

$$n = 11 \times 13 = 143$$

n is part of both the public and private keys.

Compute Euler's totient function $\phi(n) = (p-1) \times (q-1)$:

$$\phi(n) = (11-1) \times (13-1) = 10 \times 12 = 120$$

Choose a public exponent e :

e must be coprime with $\phi(n)$ (no common factors besides 1).

Example: $e = 7$ ($\gcd(7, 120) = 1$)

Compute the private exponent d :

d is the multiplicative inverse of e modulo $\phi(n)$

Solve for d such that $(d \times e) \% \phi(n) = 1$

Here, $d = 103$ because $(103 \times 7) \% 120 = 1$

Now you have:

Public key: $(n = 143, e = 7)$

Private key: $(n = 143, d = 103)$

ENCRYPTION:

To send a message m (as a number smaller than n):

Take the message m -> example: $m = 9$

Encrypt it using the public key (n, e) :

$$c = 9^7 \% 143$$

Calculate:

$$9^7 = 4782969$$

$$4782969 \% 143 = 48$$

So, the ciphertext $c = 48$. This is what you send.

DECRYPT:

The receiver uses the private key (n, d) to decrypt:

$$m = 48^{103} \% 143$$

Using modular arithmetic tricks (you usually use software for big numbers), the result is $m = 9$, recovering the original message.

RESUME:

1-> Generate keys using large primes -> public and private key.

2 -> Encrypt: $c = m^e \% n$ with the public key.

3 -> Decrypt: $m = c^d \% n$ with the private key.

The security of RSA comes from the fact that factoring n into p and q is extremely hard if p and q are large (hundreds of digits).

Thanks for the explanation, this explains everything and is quite easy to understand