How Does Public Key Encryption Work? | Public Key Cryptography and SSL

Public key encryption, also known as asymmetric encryption, uses two separate keys instead of one shared one: a public key and a private key. Public key encryption is an important technology for Internet security.

Share

Public Key Encryption Learning Objectives

After reading this article you will be able to:

- Define public key encryption
- Understand how public key encryption works
- Learn why public key encryption is essential for the TLS/SSL protocol

Related Content

Keyless SSL

What is SSL?

SSL Handshake

What is an SSL Certificate?

What is Mixed Content?

What is public key encryption?

Public key encryption, or public key cryptography, is a method of encrypting data with two different keys and making one of the keys, the public key, available for anyone to use. The other key is known as the private key. Data encrypted with the public key can only be decrypted with the private key, and data encrypted with the private key can only be decrypted with the public key. Public key encryption is also known as asymmetric encryption. It is widely used, especially for TLS/SSL, which makes HTTPS possible.

What is a cryptographic key?

In cryptography, a key is a piece of information used for scrambling data so that it appears random; often it's a large number, or string of numbers and letters. When unencrypted data, also called plaintext, is put into an encryption algorithm using the key, the plaintext comes out the other side as random-looking data. However, anyone with the right key for decrypting the data can put it back into plaintext form.

For example, suppose we take a plaintext message, "hello," and encrypt it with a key*; let's say the key is "2jd8932kd8." Encrypted with this key, our simple "hello" now reads "X5xJCSycg14=", which seems like random garbage data. However, by decrypting it with that same key, we get "hello" back.

Plaintext + key = ciphertext:

```
hello + 2jd8932kd8 = X5xJCSycg14=
```

Ciphertext + key = plaintext:

```
X5xJCSycg14= + 2jd8932kd8 = hello
```

(This is an example of symmetric encryption, in which only one key is used.)

*Using Blowfish algorithm, CBC mode, Base64 encoding.

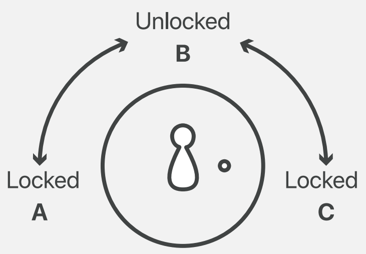
How does public key encryption work?

Public key cryptography can seem complex for the uninitiated; fortunately a writer named Panayotis Vryonis came up with an analogy that roughly goes as follows.

Imagine a trunk with a lock that two people, Bob and Alice, use to ship documents back and forth. A typical lock has only two states: locked and unlocked. Anyone with a copy of the key can unlock the trunk if it's locked, and vice versa. When Bob locks the trunk and sends it to Alice, he knows that Alice can use her copy of the key to unlock the trunk. This is essentially how what's known as symmetric cryptography works: one secret key is used for both encrypting and decrypting, and both sides of a conversation use the same key.

Now imagine, instead, that Bob makes a trunk with a special kind of lock. This lock has three states instead of two:

- A. Locked, key turned all the way to the left
- B. Unlocked, in the middle.
- C. Locked, key turned all the way to the right.



Instead of one key, two keys go with this lock:

- Key No. 1 can only turn to the left
- Key No. 2 can only turn to the right

This means that if the trunk is locked and the key is turned to position A, only key No. 2 can unlock it by turning right, to position B (unlocked). If the trunk is locked in position C, only key No. 1 can unlock it by turning the lock left, to position B.

In other words, either key can lock the trunk – but once it is locked, only the other key can unlock it.

Now let's say Bob makes a few dozen copies of key No. 2, the key that only turns right, and shares them with everyone he knows and anyone who wants a copy, making it his public key. He keeps key No. 1 for himself – it's his private key. What does this accomplish?

1. Alice can send Bob confidential data via the trunk and be confident that only Bob can unlock it. Once Alice has locked the trunk with the public key, which turns from left to right, only a key that can turn right to left can unlock it. That means only Bob's private key can unlock it.
2. Alice can be sure that the trunk is actually from Bob, and not an impersonator, if it's locked with his private key. There's only one key that can lock the trunk so that the lock is in position A, or turned all the way to the left: Bob's private key. True, anyone can unlock it with the public key by turning the key to the right, but it's guaranteed that the trunk is from Bob.

Substitute plaintext data for the trunk and cryptographic keys for the physical keys from this analogy, and this is how public key cryptography works. Only the owner of the private key can encrypt data so that the public key decrypts it; meanwhile, anyone can encrypt data with the public key, but only the owner of the private key can decrypt it.

Therefore, anyone can send data securely to the private key owner. Also, anyone can verify that data they receive from the owner of the private key is actually from that source, and not from an impersonator (see What is an on-path attack?).

How does TLS/SSL use public key encryption?

Public key encryption is extremely useful for establishing secure communications over the Internet (via HTTPS). A website's SSL/TLS certificate, which is shared publicly, contains the public key, and the private key is installed on the origin server – it's "owned" by the website.

TLS handshakes use public key cryptography to authenticate the identity of the origin server, and to exchange data that is used for generating the session keys. A key exchange algorithm, such as RSA or Diffie-Hellman, uses the public-private key pair to agree upon session keys, which are used for symmetric encryption once the handshake is complete. Clients and servers are able to agree upon new session keys for each communication session, so that bad actors are unable to decrypt communications even if they identify or steal one of the session keys.

About SSL
What is SSL?
What is TLS?
How SSL Works

Contact Sales:
+1 (888) 99 FLARE

About HTTPS
What is HTTPS?
Why Use HTTPS?
HTTP Security Gaps
Connection Not Private

About Encryption
What is Encryption?
Public Key Encryption
Asymmetric Encryption
Lava Lamp Encryption
Cryptographic Key
What is a Session Key?

SSL Glossary
What is Mixed Content?
SSL Handshake
What is an SSL Certificate?
SSL Certificate Types
Why Use TLS 1.3?
What is SNI?
What is Encrypted SNI?
What is Domain Spoofing?

Learning Center Navigation
Learning Center Home
DDoS Learning Center
CDN Learning Center
DNS Learning Center
Performance Learning Center
Security Learning Center
Serverless Learning Center
Bots Learning Center
Cloud Learning Center
Access Management Learning Center
Network Layer Learning Center
Privacy Learning Center
Video Streaming Learning Center