Understanding SSH Key Pairs

In every SSH/SFTP connection there are four keys (or two key-pairs) involved. This article explains a difference between them and what keys an SFTP client user needs to care about.

The SSH employs a public key cryptography. A public-key cryptography, also known as asymmetric cryptography, is a class of cryptographic algorithms which requires two separate keys, one of which is secret (or private) and one of which is public.[1] Together they are known as a key-pair. In SSH, the public key cryptography is used in both directions (client to server and server to client), so two key pairs are used. One key pair is known as a host (server) key, the other as a user (client) key.

- User Private Key
- User Public Key
- Host Private Key
- Host Public Key

A *user private key* is key that is kept secret by the SSH user on his/her client machine. The user must never reveal the private key to anyone, including the server (server administrator), not to compromise his/her identity.

To protect the private key, it should be generated locally on a user's machine (e.g. using PuTTYgen) and stored encrypted by a passphrase. The passphrase should be long enough (that's why it's called passphrase, not password) to withstand a brute-force attack for a reasonably long time, in case an attacker obtains the private key file.

Different file formats are used to store private keys. WinSCP supports PuTTY format, with `.ppk` extension.

A user public key is a counterpart to *user private key*. They are generated at the same time. The *user public key* can be safely revealed to anyone, without compromising user identity.

To allow authorization of the user on a server, the user public key is registered on the server. In the most widespread SSH server implementation, the OpenSSH, file `~/.ssh/authorized_keys` is used for that.

Learn more about public key authentication in general and how to setup authentication with public keys.

A *host private key* is generated when the SSH server is set up. It is safely stored in a location that should be accessible by a server administrator only. The user connecting to the SSH server does not need to care about *host private key* in general.

A *host public key* is a counterpart to *host private key*. They are generated at the same time. The *host public key* can be safely revealed to anyone, without compromising host identity.

To allow authorizing the host to the user, the user should be provided with host public key in advance, before connecting. The client application typically prompts the user with *host public key* on the first connection to allow the user to verify/authorize the key. The *host public key* is then saved and verified automatically on further connections. The client application warns the user, if the host key changes.

---

1. The text is partially copied from Wikipedia article on Public-key cryptography. The text is licensed under GNU Free Documentation License. ^