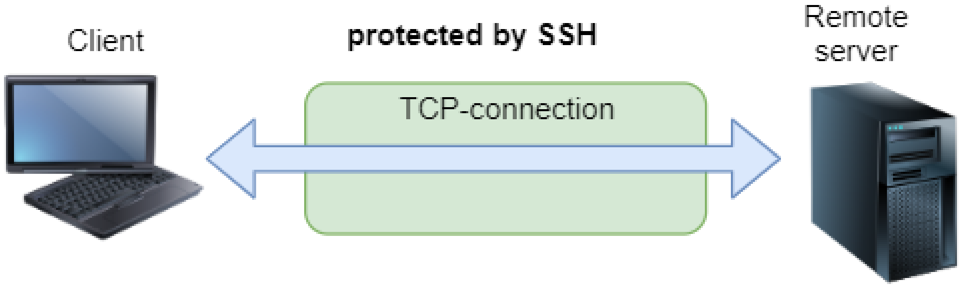
Linux

Remote control using SSH

What is SSH?

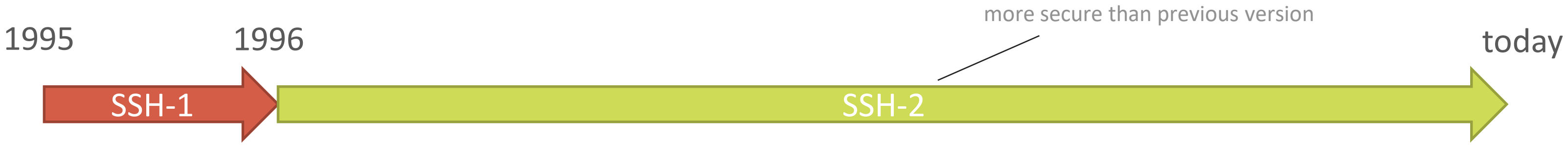
SSH (Secure Shell) – a cryptographic network protocol for operating network services securely over an unsecured network.
It replaces unsecure Telnet and unsecure rsh/rexec/rlogin protocols

Implementations



Overview

Versions



more secure than previous version

1995

1996

today

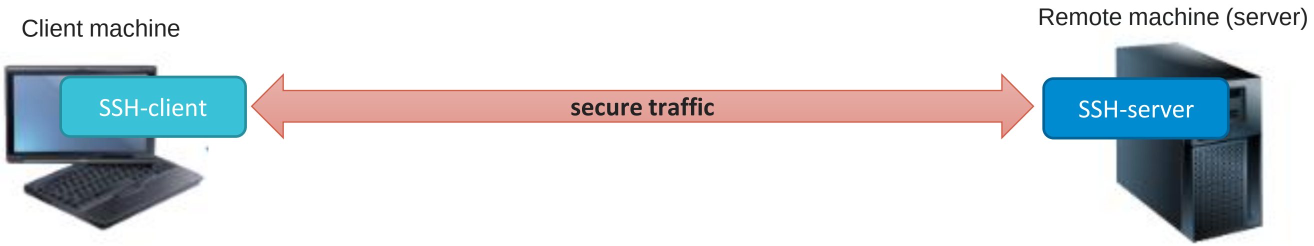
SSH-1

SSH-2

Usage

- *log into remote machine and execute commands*
- *secure transfer files*
- *tunneling, forwarding TCP ports and X11 connections and compression traffic*

Components



Client machine

Remote machine (server)

SSH-client

secure traffic

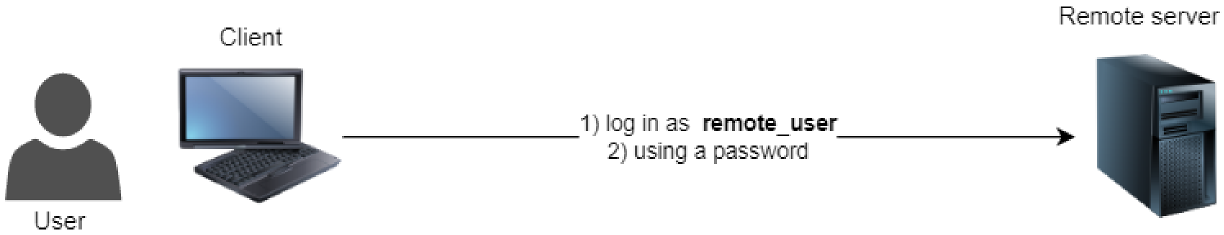
SSH-server

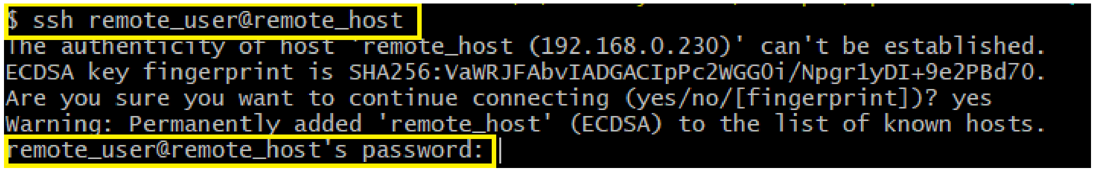
Overview

Basic authentication methods

- password
- public/private key pair
- PAM
- Kerberos
- …

Password authentication





```
$ ssh remote_user@remote_host
The authenticity of host 'remote_host (192.168.0.230)' can't be established.
ECDSA key fingerprint is SHA256:VaWRJFAbvIADGACIpPc2WGG0i/Npgr1yDI+9e2PBd70.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'remote_host' (ECDSA) to the list of known hosts.
remote_user@remote_host's password:
```

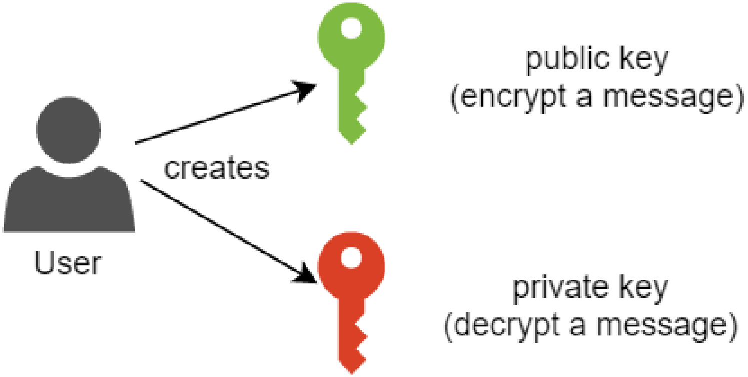
Overview

Public/private key pair authentication

Encryption types:

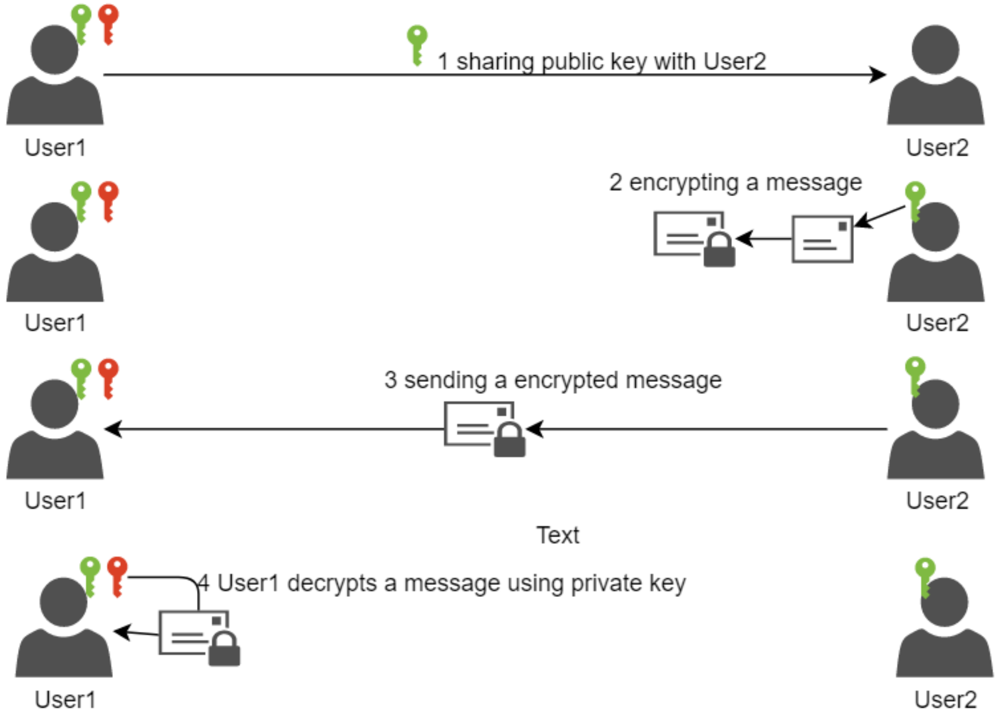
- Symmetric (single key for encryption and decryption)
- Asymmetric (two keys: for encryption and for decryption)

Asymmetric encryption



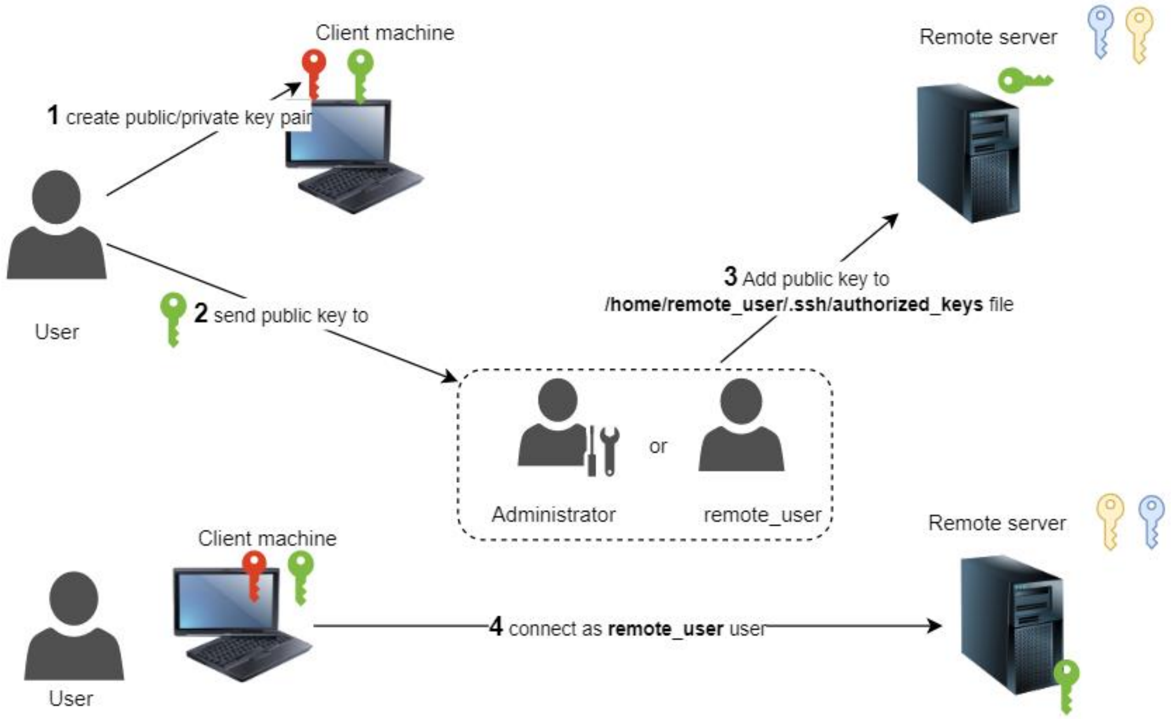
Overview

Public/private key pair authentication


1 sharing public key with User2
User1
User2
2 encrypting a message
User1
User2
3 sending a encrypted message
User1
User2
Text
4 User1 decrypts a message using private key
User1
User2

Connection organization

Public/private key pair authentication



Basics

Setup | Key pair generation | Connection | Settings

CLIENT **SERVER**

RedHat package

CLIENT:

```
openssh-clients
```

SERVER:

```
openssh-server
```

Debian package

CLIENT:

```
openssh-client
```

SERVER:

```
openssh-server
```

Installation

CLIENT:

RedHat/CentOS

```
yum install openssh-clients
```

Debian

```
apt install openssh-client
```

SERVER:

RedHat/CentOS

```
yum install openssh-server
```

Debian

```
apt install openssh-server
```

Basics

Setup | **Key pair generation** | Connection | Settings

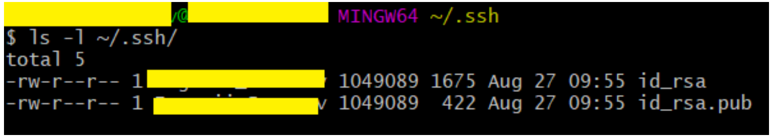
Command which creates public and private keys

```
ssh-keygen -t <type of key>
```

Example usage

```
ssh-keygen -t rsa
```

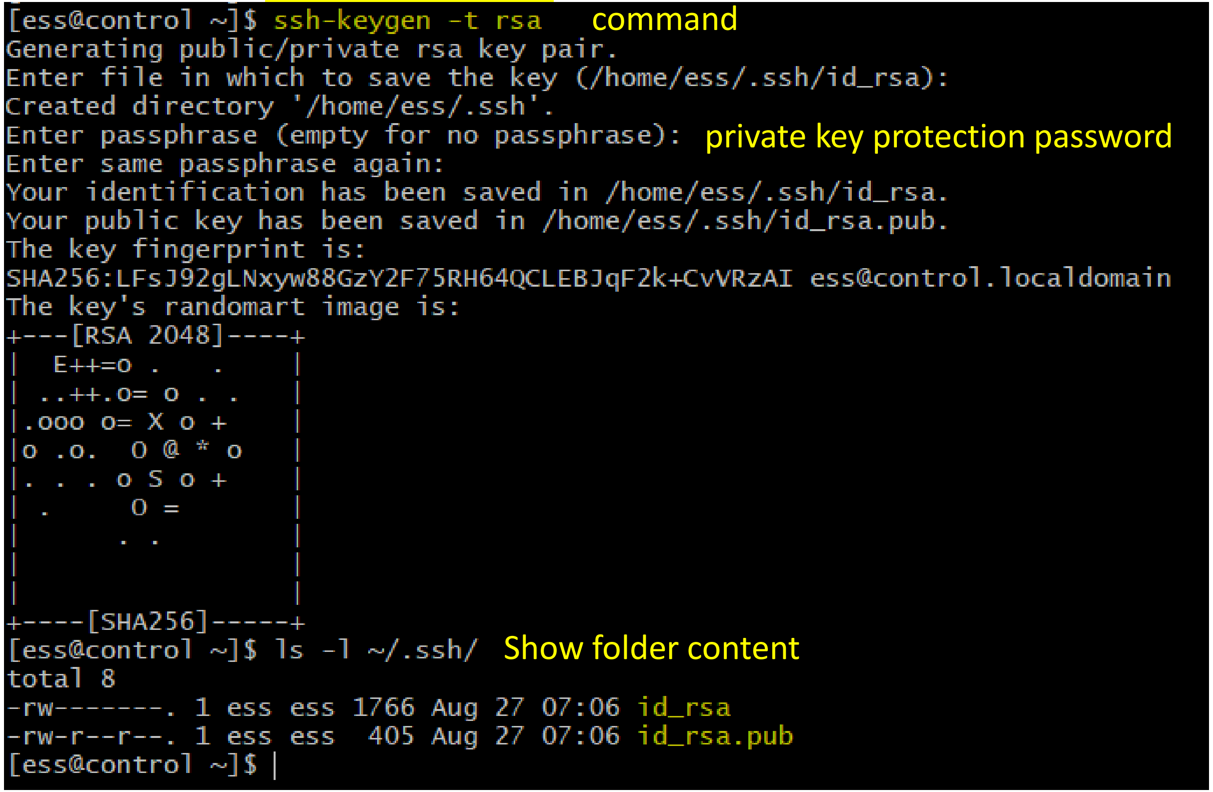
and it’s result



```
[redacted]@[redacted] MINGW64 ~/.ssh
$ ls -l ~/.ssh/
total 5
-rw-r--r-- 1 [redacted] 1049089 1675 Aug 27 09:55 id_rsa
-rw-r--r-- 1 [redacted] 1049089  422 Aug 27 09:55 id_rsa.pub
```

Default SSH client settings folder is **~/.ssh/**

Basics (example)



```
[ess@control ~]$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ess/.ssh/id_rsa):
Created directory '/home/ess/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ess/.ssh/id_rsa.
Your public key has been saved in /home/ess/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:LFsJ92gLNxyw88GzY2F75RH64QCLEBJqF2k+CvVRzAI ess@control.localdomain
The key's randomart image is:
+---[RSA 2048]----+
|  E++=o .   .    |
| ..++.o= o . .   |
|.ooo o= X o +    |
|o .o.  O @ * o   |
|. . . o S o +    |
| .     O =       |
|      . .        |
|                 |
|                 |
+----[SHA256]-----+
[ess@control ~]$ ls -l ~/.ssh/
total 8
-rw-------. 1 ess ess 1766 Aug 27 07:06 id_rsa
-rw-r--r--. 1 ess ess  405 Aug 27 07:06 id_rsa.pub
[ess@control ~]$ 
```

command

private key protection password

Show folder content

Basics

Setup | Key pair generation | **Connection** | Settings

Command to establish SSH connection

```
ssh (options) remote_user@remote_server (command)
```

Usage examples

- connect to *remote_host* server as current user and run a remote shell

```
ssh remote_host
```

- connect to *remote_host* server as *remote_user user* and run a remote shell

```
ssh remote_user@remote_host
```

- connect to *remote_host* server and run ‘who’ command as *remote_user* user

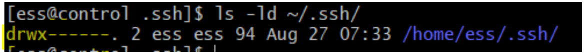
```
ssh remote_user@remote_host "who"
```

Basics

CLIENT

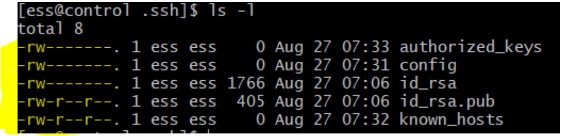
Client configuration

- Configuration folder **~/.ssh/** is created manually or during key pair generation, connection and etc.



```
[ess@control .ssh]$ ls -ld ~/.ssh/
drwx------. 2 ess ess 94 Aug 27 07:33 /home/ess/.ssh/
```

- Permission folder **~/.ssh/** is documented and mandatory



```
[ess@control .ssh]$ ls -l
total 8
-rw-------. 1 ess ess    0 Aug 27 07:33 authorized_keys
-rw-------. 1 ess ess    0 Aug 27 07:31 config
-rw-------. 1 ess ess 1766 Aug 27 07:06 id_rsa
-rw-r--r--. 1 ess ess  405 Aug 27 07:06 id_rsa.pub
-rw-r--r--. 1 ess ess    0 Aug 27 07:32 known_hosts
```

- Client config-file is located in **~/.ssh/config**

SERVER

Server configuration

- Configuration folder is **/etc/ssh/**, configuration file is **/etc/ssh/sshd_config**
- sshd is service which handle SSH connection (gracefully restart)
- Remote server can be a SSH client to connect external services and applications. Configuration file is stored in **/etc/ssh/ssh_config** for client configuration.
- Service is gracefully reloaded

```
systemctl reload sshd
```

Basics

Setup

Key pair generation

Connection

Settings

Client configuration folder

Folder and it’s content is represented below

Folder/File	Description
~/.ssh/	This directory is the default location for all user-specific configuration and authentication information
~/.ssh/known_hosts	Contains a list of host keys for all hosts the user has logged into that are not already in the systemwide list of known host keys
~/.ssh/config	This is the per-user configuration file
~/.ssh/authorized_keys	Lists the public keys (RSA/DSA) that can be used for logging in as this user.

Basics

Setup | Key pair generation | Connection | Settings

CLIENT CONFIGURATION EXAMPLE

```
# server 192.168.0.230 and remoteuser1 user
Host remote_server1
   HostName 192.168.0.230
   User remoteuser1
   IdentityFile ~/.ssh/id_rsa1

#  server 192.168.0.230 and remoteuser2 user
Host remote_server2
   HostName 192.168.0.230
   User remoteuser2
   IdentityFile ~/.ssh/id_rsa2

# Common SSH config
Host *
   User vagrant
```

SERVER CONFIGURATION EXAMPLE

```
AddressFamily inet
ListenAddress 0.0.0.0

Protocol 2

HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_ed25519_key

# Logging
LogLevel INFO

# Authentication:
LoginGraceTime 120
PermitRootLogin without-password
StrictModes yes
```

SERVER CONFIGURATION EXAMPLE
(continue)

```
ClientAliveInterval 1800
ClientAliveCountMax 0

AuthorizedKeysFile %h/.ssh/authorized_keys

IgnoreUserKnownHosts yes
IgnoreRhosts yes

PasswordAuthentication yes
PermitEmptyPasswords no

PubkeyAuthentication yes
UsePAM yes
```

Basics

SERVER CONFIGURATION EXAMPLE
(continue)

```
GSSAPIAuthentication yes
ChallengeResponseAuthentication no

AuthorizedKeysCommand
/usr/bin/sss_ssh_authorizedkeys
AuthorizedKeysCommandUser nobody


AllowAgentForwarding yes
AllowTcpForwarding yes
X11Forwarding yes

Banner /etc/disclaimer
```

Useful commands and tips

Output verbose information (troubleshooting a connection issue, *learning SSH connection*)

```
ssh –v ….
```

Keys control management

Authentication agent

```
ssh-agent [-c | -s] [-d] [-a bind_address] [-t life] [command [arg ...]]
```

Adds RSA or DSA identities to the authentication agent

```
ssh-add [-cDdLlXx] [-t life] [file …]
```

Copy public key to server

```
ssh-copy-id [-i [identity_file]] [user@]machine
```

Secure copying data

```
scp [options] [source user@IP source]:[source folder] \
      [destination user@IP destination]:[destination path]
```

Manuals

```
man ssh
man sshd
```

Secure copying data (scp)

Command **scp** allows to transfer data between

- local and remote machines
- remote machine and remote machine

Command

```
scp [-12346BCpqrv] [-c cipher] [-F ssh_config] [-i identity_file]
    [-l limit] [-o ssh_option] [-P port] [-S program]
    [[user@]host1:]file1 ... [[user@]host2:]file2
```

Execution stages

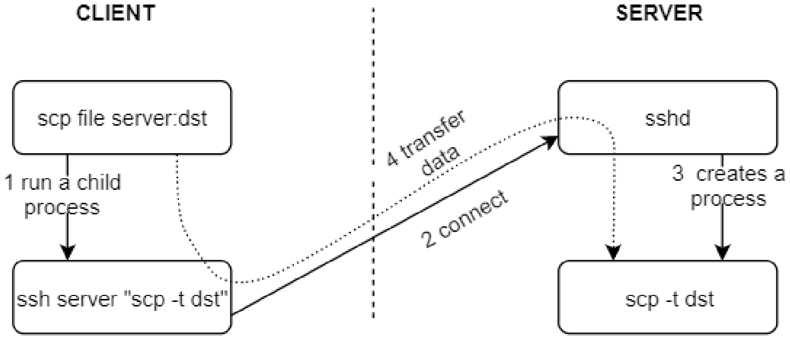
- “connection” establishment
- data transfer

Secure copying data (scp)

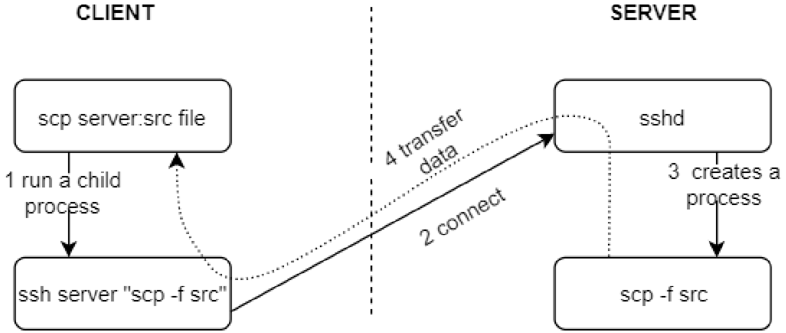
“Connection” establishment
It includes steps 1-3

Data transfer
It’s represented by step 4

Upload file



Download file



Secure copying data (scp)

Command and it’s arguments

upload data

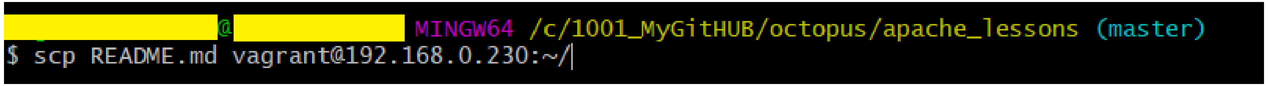
```
scp [options] /local/path/file [user2@IP2]:[remote path2]
```

download data

```
scp [options] [user1@IP1]:[remote path1] /local/path/file
```

Usage example

Copying README.md to server192.168.0.230 using vagrant user. Destination is remote user home folder.



```
[redacted]@[redacted] MINGW64 /c/1001_MyGitHUB/octopus/apache_lessons (master)
$ scp README.md vagrant@192.168.0.230:~/
```

Useful links and commands

Public sources

- IBM’s SSH Guide https://developer.ibm.com/articles/au-sshsecurity/#
- Securing OpenSSH https://wiki.centos.org/HowTos/Network/SecuringSSH
- SSH wiki page https://en.wikipedia.org/wiki/Secure_Shell
- Ubuntu's documentation https://help.ubuntu.ru/wiki/ssh
- Online manual (ssh) https://www.opennet.ru/cgi-bin/opennet/man.cgi?topic=ssh
- Online manual (ssh-keygen) https://www.opennet.ru/man.shtml?topic=ssh-keygen
- Online manual (ssh-add) https://www.opennet.ru/man.shtml?topic=ssh-add
- Online manual (ssh-agent) https://www.opennet.ru/man.shtml?topic=ssh-agent

Linux manuals (commands)

```
man ssh
man sshd
```

Practice

#	Step	Where
1	Create a new virtual machine (VM) or use existing one	local machine
2	Install SSH client (if it’s installed yet)	local machine
3	Create private and public key pair	local machine
4	Install SSH server package (if it’s installed yet)	remote server
5	Create *mysshfriend* user with password	remote server
6	Connect to VM as *mysshfriend* user	local machine
7	Add public key to *mysshfriend* user	remote server
8	Connect to VM as *mysshfriend* user	local machine

THANK YOU