

SSH and SCP

<https://github.com/heig-vd-dai-course>

[Web](#) • [PDF](#)

L. Delafontaine and H. Louis, with the help of GitHub Copilot.

This work is licensed under the [CC BY-SA 4.0](#) license.

Objectives

- Refresh on security
- Learn how to use the SSH protocol to connect to a remote server
- Learn how to use the SCP protocol to transfer files to a remote server

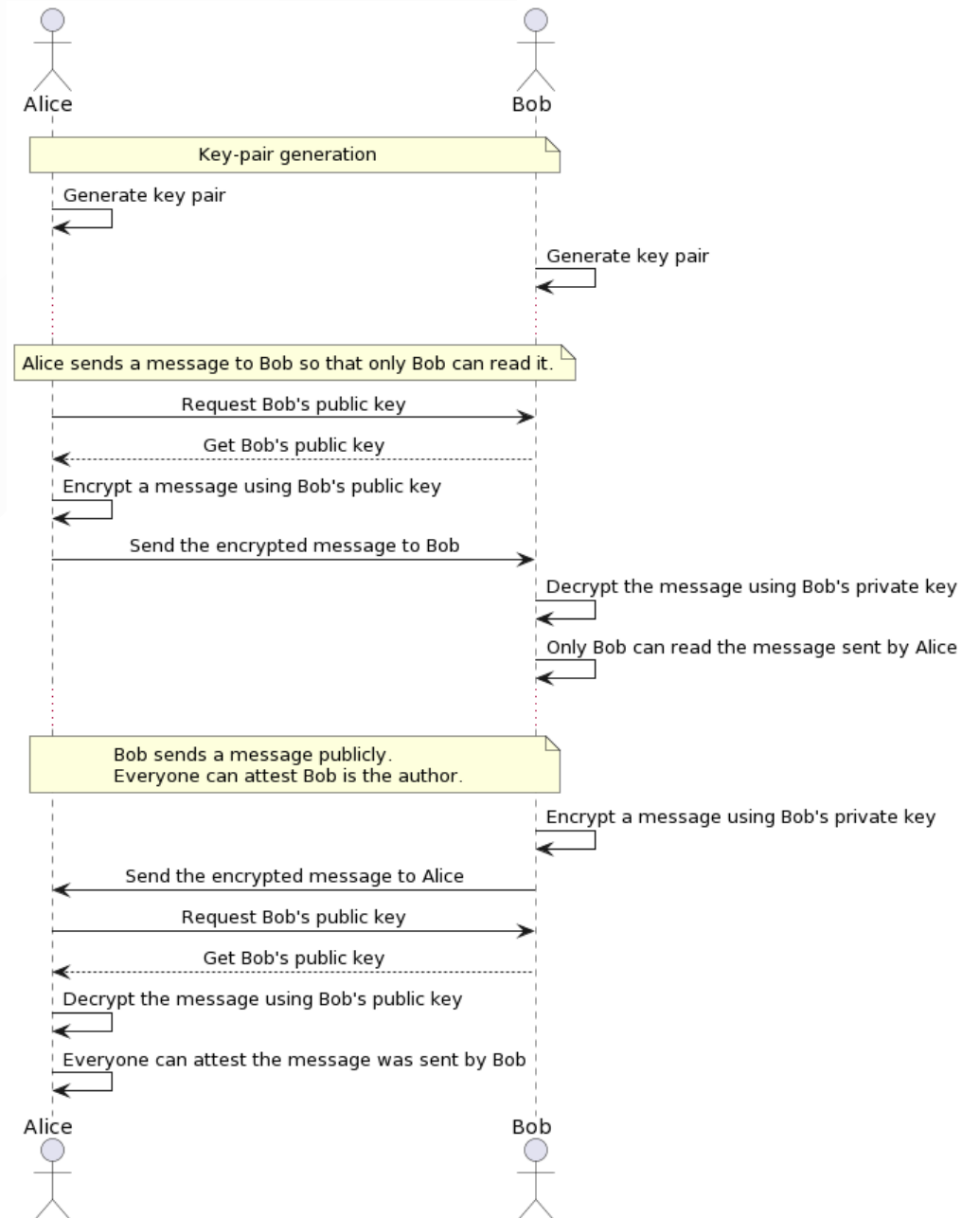


A quick reminder about security

More details for this section in the [course material](#). You can find other resources and alternatives as well.

A quick reminder about security

- A secure protocol ensures the confidentiality of the data exchanged
- Most secure protocols rely on cryptography
- Cryptography is based on algorithms and keys



SSH

More details for this section in the [course material](#). You can find other resources and alternatives as well.

SSH

- Secure Shell
- Uses TCP port 22
- A protocol to connect to a remote server
- Can be used to execute commands on a remote server
- The standard way to connect to a remote server

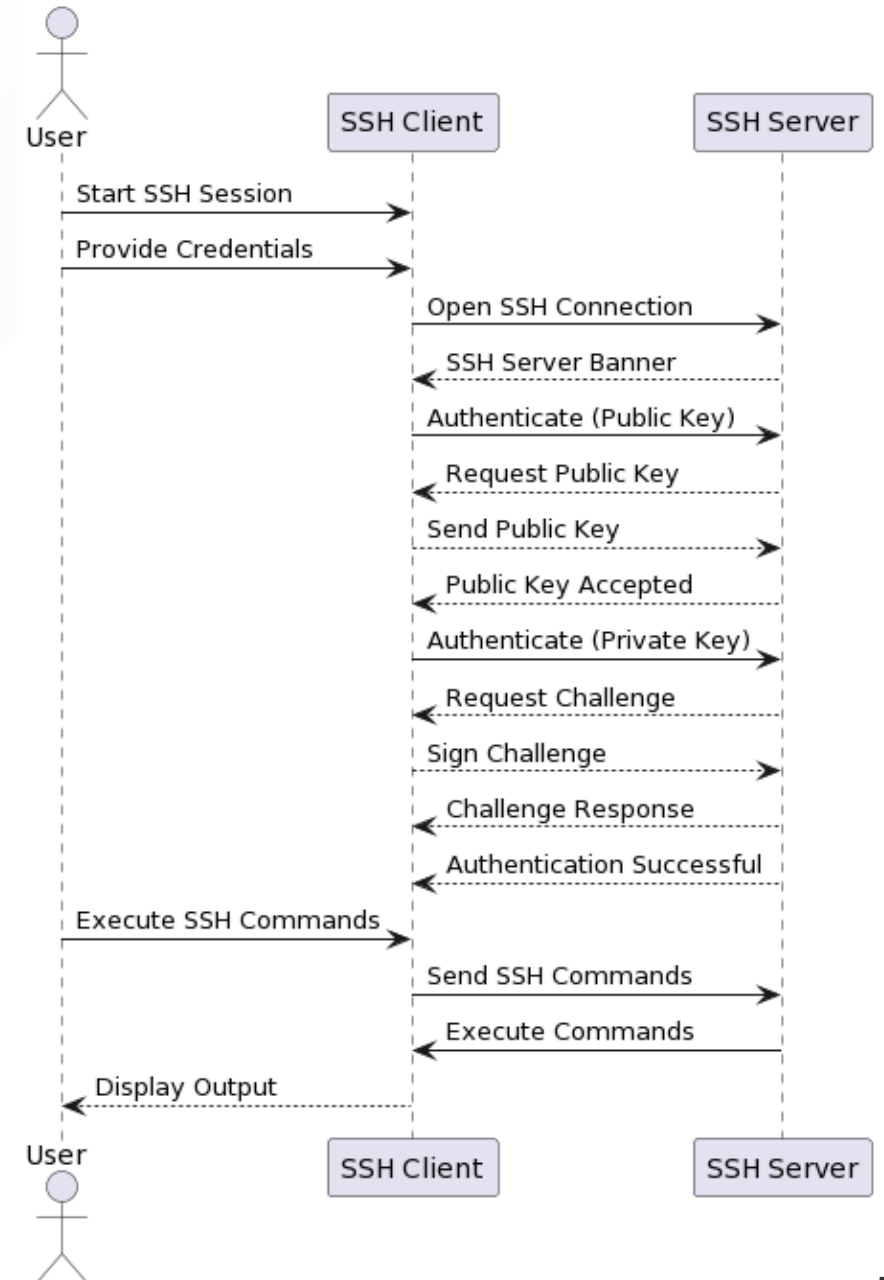
The screenshot shows the IETF Datatracker page for RFC 4253. The page is titled "The Secure Shell (SSH) Transport Layer Protocol" and is categorized as a "Proposed Standard". The authors listed are T. Ylonen, C. Lonvick, and Cisco Systems, Inc., dated January 2006. The page includes sections for "Status of This Memo", "Copyright Notice", and "Abstract". The "Status of This Memo" section states that the document specifies an Internet standards track protocol for the Internet community and requests discussion and suggestions for improvements. The "Copyright Notice" section states that the copyright is held by The Internet Society (2006). The "Abstract" section describes the SSH transport layer protocol, which typically runs on top of TCP/IP and provides strong encryption, server authentication, and integrity protection. It also mentions that the document describes the Diffie-Hellman key exchange method and the minimal set of algorithms needed to implement the SSH transport layer protocol. On the right side of the page, there is a sidebar with navigation options like "Info", "Contents", and "Prefs", and a "Document type" section indicating it is an "RFC Proposed Standard" from January 2006. There are also buttons for "View errata", "Report errata", and "IPR". A "Select version" table is visible, showing a grid of version numbers from 00 to 24, with "RFC 4253" selected. Below the table, there is a "Compare versions" section with dropdown menus for selecting versions to compare.

SSH key algorithms

The most common key algorithms are:

- RSA
- DSA
- ECDSA
- Ed25519

Ed25519 and ECDSA are the recommended algorithms.



SSH key fingerprint

- Short version of a public key
- Used to verify the identity of a public key
- Can help detect man-in-the-middle attacks
- Stored in the `~/.ssh/known_hosts` file

```
File: /Users/lude1afo/.ssh/known_hosts
1 gitlab.com ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIAfuCHKVTjquxvt6CM6tdG4S
  Lp1Bbn/nOeHHE5U0zRdf
2 github.com ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIOMqknkVzrm0SdG6U0oqKLsa
  bgH5C9okWi0dh2l9GKJl
3 github.com ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlz
  dHAyNTYAAABBBEmKSEnjQEez0mxkZMy7opKgwFB9nkt5YRrYmJNuG5N87uRgg6CLrbo5wAd
  T/y6v0mKV0U2w0WZ2YB/++Tpockg=
4 10.11.12.2 ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIPWNyMrBqFwK8ZLKHPXFuMHZ
  8YE6dusguYZrLFbkN/5J
5 10.11.12.2 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlz
  dHAyNTYAAABBBI7fotRNXr8yxKs0lQq1iaYDnS/3dQ8YIM/Pnu7BKVUJBic8Ea1iNd7Hp4
  sqLL2ISde/XuWM2EkFiIU1VmWLak=
6 193.134.218.28 ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAICa1+G0AgQj6mPy3jexw
  gOL1qDl6YIIG5/VamkAQdR6h
7 193.134.218.28 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAI
  bmlzdHAyNTYAAABBBBCUzu5yeNm/mXMGJpFUq5b9BmT29NG1KXyWloYIhhZugDV0dKaPmo9q
  ylCoXZvXPxGs2J4vuNk77WnsqI9WozGE=
8 iict-mv298-aii40.iict.ch ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAICa1+G0AgQ
  j6mPy3jexwgOL1qDl6YIIG5/VamkAQdR6h
9 194.182.160.98 ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIPTmVF0fhRC75SwSm2Y
  BE48fXDflhG6ueATXN7p22EI
10 194.182.160.98 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAI
  bmlzdHAyNTYAAABBBMVWT0Z+OiUe1ft4f31H/Onud/DLnS/oAOKW0EMtVewZvWoRpPUY4P8
  HRN7ewt4U5W1KrMN+ZmM2L/nRr22L4MQ=
11 github.com ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGCj7ndNxQowgcQnjshcLrqP
  Eiiphnt+VTTvDP6mHBL9j1aNUkY4Ue1gvwnGLV10hGeYrnZaMgRK6+PKCUXaDbC7qtbW8gI
  khL7aGCs0r/C56SJMy/BCZfxd1nWzAOxSDPgVsmer0BYfnq1tV9/hWCqBywINIR+5dI6gJT
  J72pcEpEjcYgXkE2YEFXV1JHnsKgbLWNlhScqb2UmyRkQyytRtL+38TGxkxCflmO+5Z8CS
  SNY7GidjMIZ7Q4zMjA2n1nGr1TDkzwDCsw+wqFPGQA179cnfGW0WRVruj16z6XyvXvjJwbz
  0wQZ75XK5tKSb7FNyeIEs4TT4jk+S4dhPeAUC5y+bDYirYgM4GC7uEnznZyaVWQ7B381AK
  4Qdrwt51ZqExKbQpTUNn+EjqoTwvqNj4kqx5QUCI0ThS/YkOxJCXmPUWZbhjpcG56i+2aB6
  CmK2JGhn57K5mj0MNdBXA4/WnwH6XoPWJzK5Nyu2zB3nAZp+S5hpQs+p1vN1/wsjk=
12 iict-mv276-beescreens.iict.ch ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAICa1+
  G0AgQj6mPy3jexwgOL1qDl6YIIG5/VamkAQdR6h
13 192.168.1.203 ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIGMv8gcQhQUH00c4Jvby4
  3FzF1rZtYyRXeg2SWB1BS05
14 192.168.1.203 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIb
  mlzdHAyNTYAAABBB0IEIs0Y8dI7Ls+U0tP/sTVZUu/UnYMeo/j8gTWdK7Yb4g/s0esPSDF
```


SSH key generation

- Use the `ssh-keygen` command
- Choose the key algorithm
- Generate a private key and a public key
- Can be done with or without a passphrase

```
11-smtp-and-telnet -- ~/8/8/8/8/11-smtp-and-telnet -- -fish -- 80x42
[> ssh-keygen -t ed25519 -f /tmp/demo_ed25519 -C "Demo key!"
Generating public/private ed25519 key pair.
[Enter passphrase (empty for no passphrase):
[Enter same passphrase again:
Your identification has been saved in /tmp/demo_ed25519
Your public key has been saved in /tmp/demo_ed25519.pub
The key fingerprint is:
SHA256:WKLtgZemp9HIU3ZYUmNNAOgBCNM/JjvCvdIkGgDI3ik Demo key!
The key's randomart image is:
+--[ED25519 256]--+
|*0.. ...=+.
|+O. o o ..
|o .o..o o
| E.o+= O
|. ++o.@ S
|..o+ % o
|. .# +
| + *
|.
+-----[SHA256]-----+

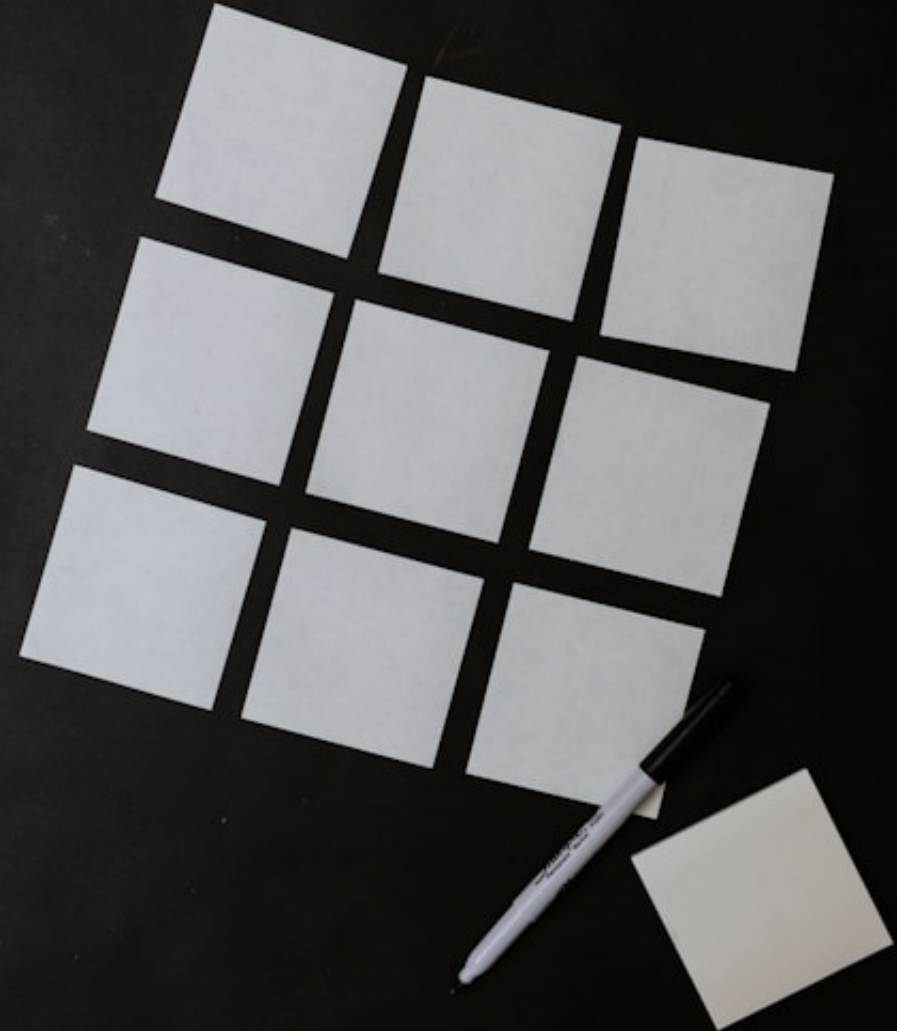
...[?]main [!?] via desktop-linux on ludovic.delafontaine@gmail.com took 3s
[> ssh-keygen -t ed25519 -f /tmp/revoke_ed25519 -C "Revoke key!"
Generating public/private ed25519 key pair.
[Enter passphrase (empty for no passphrase):
[Enter same passphrase again:
Your identification has been saved in /tmp/revoke_ed25519
Your public key has been saved in /tmp/revoke_ed25519.pub
The key fingerprint is:
SHA256:w1CzGztuYyU+SEjLrtvIUa3vas1/xigwzAhMPGrfSg4 Revoke key!
The key's randomart image is:
+--[ED25519 256]--+
|. o
|+ . o
|+ o . o
|o= o.o +
|o O.o.B S
| E.O.= =
|.=.B B o
|.+.+. * + +
|+oo.ooo.o
+-----[SHA256]-----+
```

SCP

More details for this section in the [course material](#). You can find other resources and alternatives as well.

SCP

- Secure Copy
- Uses TCP port 22
- A protocol to transfer files to/from a remote server
- Can be used to transfer files between two remote servers as well



Practical content

What will you do?

- Install and configure SSH and SCP
- Start a SSH server with Docker Compose
- Connect to the SSH server with SSH
- Transfer a file to the SSH server with SCP

```
12-ssh-and-scp -- ~/8/8/8/8/12-ssh-and-scp -- fish -- 80x42

...-scp on [?]main [!?] via desktop-linux on ludovic.delafontaine@gmail.com
[> docker compose up -d
[+] Running 1/0
✓ Container 12-ssh-and-scp-openssh-server-1 Running 0.0s

...-scp on [?]main [!?] via desktop-linux on ludovic.delafontaine@gmail.com
[> ssh daistudent@localhost -p 2222
[daistudent@localhost's password:
Welcome to OpenSSH Server
[0ddb4ef63ada:~$ exit
logout
Connection to localhost closed.

...[?]main [!?] via desktop-linux on ludovic.delafontaine@gmail.com took 2s
[> ssh -i demo_ed25519 -p 2222 daistudent@localhost
[Enter passphrase for key 'demo_ed25519':
Welcome to OpenSSH Server
[0ddb4ef63ada:~$ exit
logout
Connection to localhost closed.

...[?]main [!?] via desktop-linux on ludovic.delafontaine@gmail.com took 4s
[> vim openssh-server.env

...[?]main [!?] via desktop-linux on ludovic.delafontaine@gmail.com took 8s
[> docker compose up -d
[+] Running 1/1
✓ Container 12-ssh-and-scp-openssh-server-1 Started 4.2s

...[?]main [!?] via desktop-linux on ludovic.delafontaine@gmail.com took 4s
[> ssh -i revoke_ed25519 -p 2222 daistudent@localhost
daistudent@localhost: Permission denied (publickey,keyboard-interactive).

...-scp on [?]main [!?] via desktop-linux on ludovic.delafontaine@gmail.com
[> docker compose down
[+] Running 2/2
✓ Container 12-ssh-and-scp-openssh-server-1 Removed 4.2s
✓ Network 12-ssh-and-scp_default Removed 0.1s

...[?]main [!?] via desktop-linux on ludovic.delafontaine@gmail.com took 4s
>
```

Find the practical content

You can find the practical content for this chapter on [GitHub](#).



Finished? Was it easy? Was it hard?

Can you let us know what was easy and what was difficult for you during this chapter?

This will help us to improve the course and adapt the content to your needs. If we notice some difficulties, we will come back to you to help you.

 [GitHub Discussions](#)

You can use reactions to express your opinion on a comment!

What will you do next?

In the next chapter, you will learn the following topics:

- Java TCP programming
 - How to send an email with Java
 - How to create a TCP server
 - How to create a TCP client
 - How to handle multiple clients with concurrency



Sources

- Main illustration by [Mathew Schwartz](#) on [Unsplash](#)
- Illustration by [Aline de Nadai](#) on [Unsplash](#)
- Illustration by [Kelly Sikkema](#) on [Unsplash](#)
- Illustration by [Carl Nenzen Loven](#) on [Unsplash](#)