INTRODUCTION

A

LA LIGNE DE COMMANDE

Septembre 2025

OUTOFPLUTO

Introduction à la ligne de commande

1. Introduction

Interpréteur de commandes

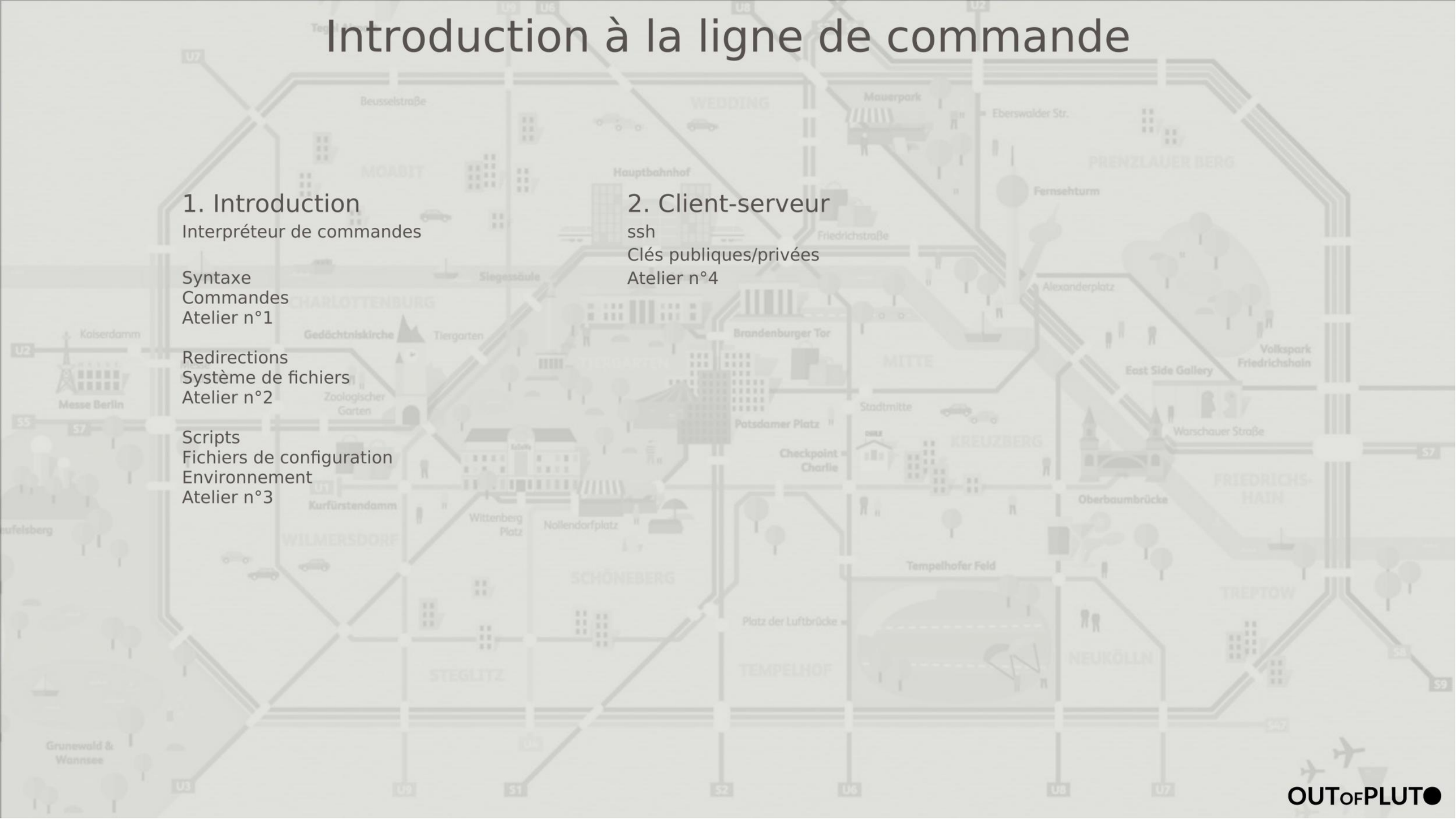
Syntaxe
Commandes
Atelier n°1

Redirections
Système de fichiers
Atelier n°2

Scripts
Fichiers de configuration
Environnement
Atelier n°3

2. Client-serveur

ssh
Clés publiques/privées
Atelier n°4



OUTOFPLUTO

INTRODUCTION

Septembre 2025

OUTOFPLUTO

Les bases

Définition courte

Interpréteur de commandes : Simple macro processeur qui exécute des commandes.
Macro processeur : technique où du texte et des symboles sont interprétés pour créer des expressions plus larges.

Quelques informations

- Accès au **système de fichier**
- Gestion de **droits/privilèges**
- “Équivalent” d’un serveur graphique (Windows, Gnome)
- Historique des commandes
- Particulièrement utile sur des serveurs distants (pas d’interface graphique) et dans des environnements disparates.
- **Interactif ou non**
- Langage de programmation
- Écriture de **scripts**
- File descriptor et redirections

Quelques exemples

Windows : cmd ou PowerShell

Linux et mac OS : sh, bash, zsh...

Syntaxe

Commande simple

```
$ ls
```

Mots réservés

Ensemble de mots qui “ont un sens” et seront interprétés par le shell. Ils ne peuvent pas être utilisés autrement.

```
if then  elif  else  fi time
for   in until while do done
case  esac  coproc   select   function
{  }  [[ ]] !
```

Quoting

Un ensemble de caractères entre guillemets n’est pas interprété par le shell.

```
$ echo "ls"
```

Échappement et caractères spéciaux

Les caractères spéciaux peuvent être **échappés** afin de ne pas être interprétés, par exemple : `\"`

A l’opposé, certaines séquences échappées avec un antislash ont une interprétation particulière : `\n`

Atelier n°1

Commandes simples

Trouver et exécuter les commandes permettant de réaliser les actions suivantes

1) Obtenir la documentation de la commande **`ls`**
2) Afficher le chemin du répertoire en cours
3) Afficher la liste des fichiers dans le répertoire en cours
4) Changer de répertoire pour la racine du système de fichier (ou du disque principal)
5) Afficher les fichiers et le détail des droits
6) Afficher son nom d'utilisateur
7) Afficher l'historique des commandes récentes
8) Réexécuter la commande 5) sans la taper
9) Afficher le contenu d'un fichier texte
10) Lancer firefox (ou n'importe quel navigateur)
11) Créer un répertoire puis créer un fichier dans ce répertoire (sans utiliser **`cd`**)
12) Supprimer le répertoire créé en 11)
13) Explorer l'utilisation de la touche **`Tab`** et des flèches directionnelles
14) Explorer les raccourcis comme **`Ctrl+A`**, **`Ctrl+L`** ou **`Ctrl+E`**

Redirections

Chevrons et *pipe*
Sorties et entrée standard

```
$ echo "Hello world" > hello.txt
$ cat hello.txt
$ echo -n "Hello you" > hello.txt
$ cat hello.txt
$ echo "!" >> hello.txt
$ cat hello.txt

$ cat hello.txt | grep "Hello"
$ cat hello.txt | less

$ python < hello.txt

$ python < hello.txt 2> /dev/null
$ python -c "i = input(); print(i); print(int(i))" 1> result.log 2> error.log
```

Système de fichiers

Les commandes `find`, `sed`, `sort`, `grep`, `head`, `awk` (entre autre)

```
$ find . -type f -iname hello

$ sed 's/you/everyboby' hello.txt

$ grep -n def command.py

$ head -n 10 command.py

$ awk --field-separator ';' '{print $6,$160}' data.csv
```

Atelier n°2

Réaliser sur la ligne de commande les manipulations suivantes

1) Récupérer l'archive suivante https://www.outofpluto.com/media/uploads/formation/shell/intro-shell.zip
2) Désarchiver le fichier
3) Changer de répertoire pour le répertoire désarchivé
4) Concaténer tous les fichiers **`.txt`** dans **`all.txt`**
5) Afficher le nom de tous les fichiers qui contiennent le mot **`Hello`**
6) Afficher le nombre de fichiers contenant le mot **`Hello`**
7) Afficher le résultat de la commande python credits sans fichier
8) Trouver le nombre total de lignes de code dans tous les .py d'un répertoire (sans compter les lignes vides)
9) Afficher toutes les lignes d'un fichier qui apparaissent en double
10) Afficher uniquement les permissions et noms des fichiers d'un dossier
11) Afficher le chemin absolu des 10 fichiers les plus gros dans un dossier et ses sous-dossiers
12) Afficher les lignes communes à deux fichiers
13) Afficher tous les fichiers exécutables dans le répertoire courant
14) Compter combien de fichiers sont vides dans un dossier et ses sous-dossiers
15) Lister tous les fichiers qui ont été modifiés dans les 24 dernières heures

Scripts

Un ensemble de commandes peut être regroupé dans un script plus complexe

```
$ cat  wcl.sh
#!/bin/bash

count=0

function count_words() {
 printf "%d words in line %d\n" $(echo $1 | wc -w) $2
}

while IFS='' read -r line || [[ -n "${line}" ]]; do
 ((count++))
 count_words "$line" $count
done < "$1"

$ ./wcl.sh lorem.txt
```

Configuration / environnement

Des comportements peuvent être automatisés et/ou sécurisés grâce à des fichiers de configuration et/ou des variables d’environnement

```
$ env

$ echo $PATH

$ export PATH=$PATH:`pwd`
$ wcl.sh lorem.txt

$ echo "UNIV_SERVER=pccop1b110-04.univ-eiffel.fr" > .env
$ source .env
$ echo $UNIV_SERVER
```

Atelier n°3

Scripts

1) Tester wcl.sh

2) Résoudre ce problème :
   https://www.hackerrank.com/challenges/lonely-integer-2/problem

3) Ecrire un script qui génère la sortie suivante :

```
# SYSTEM
OS:                     Debian GNU/Linux 11 (bullseye) (x86_64)

# HARDWARE
CPU:                    Intel(R) Xeon(R) CPU E3-1231 v3 @ 3.40GHz
CPU(s):                 8
Cores:                  4
Total Memory:           31 GB
Total Disk Space:       1.9T

# CURRENT STATE
Used Memory:            48% (15 GB)
Swap:                   45% (3809 MB)
Used Disk Space:        44%
```

4) Résoudre d'autres problèmes ici :
   https://www.hackerrank.com/domains/shell

CLIENT-SERVEUR

Septembre 2025

OUTOFPLUTO

SSH

Définition courte

Le protocole **Secure Shell** (protocole **SSH**) est un protocole réseau cryptographique permettant d'utiliser des services réseau en toute sécurité sur un réseau non sécurisé. Ses applications les plus notables sont la **connexion à distance** et l'exécution de lignes de commande.

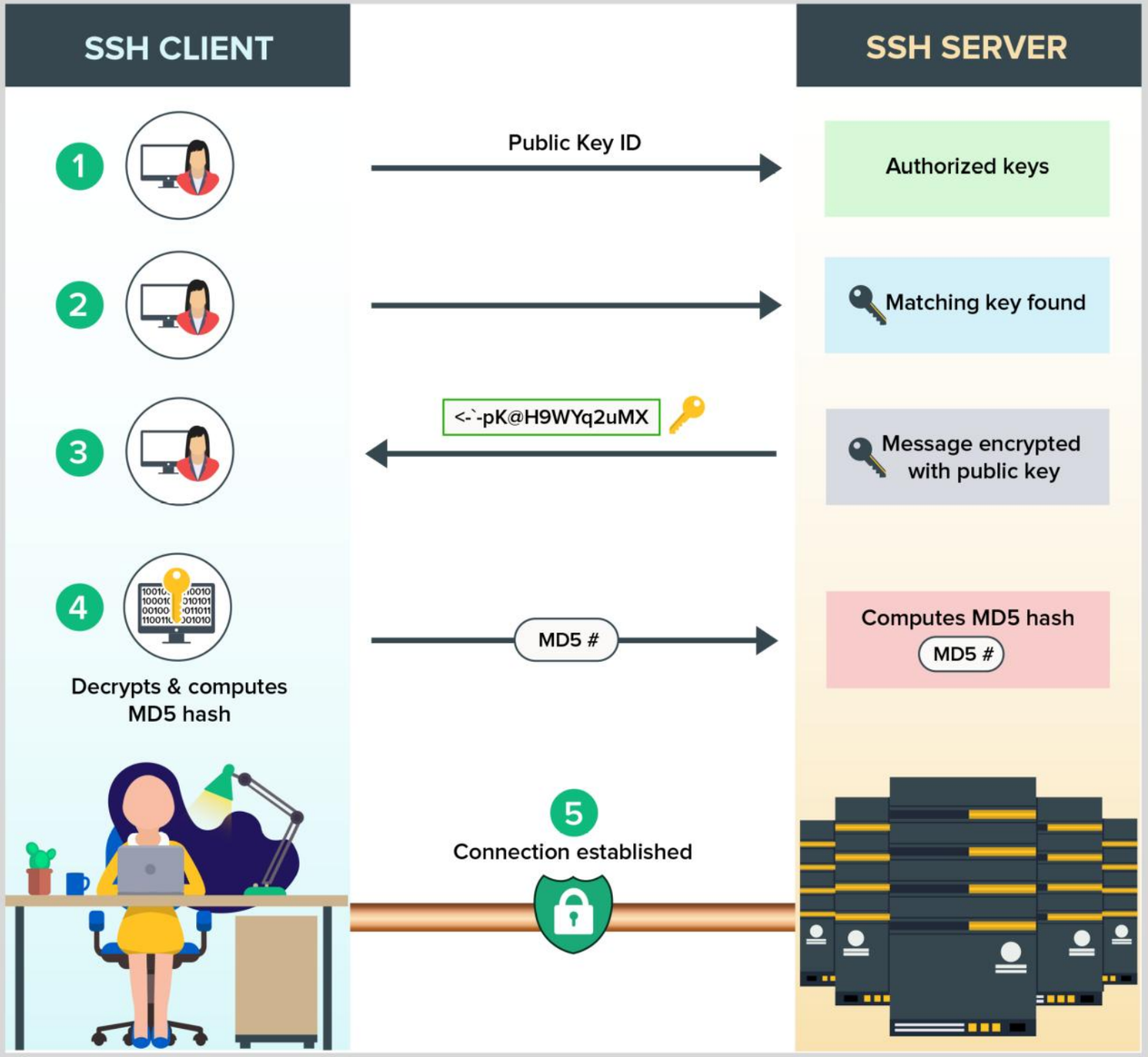
```
$ ssh anthony.baillard2@pccop1b110-04.univ-eiffel.fr

$ ssh -L 8188:localhost:8188 anthony.baillard2@pccop1b110-04.univ-eiffel.fr

$ echo "Host eiffel
  User anthony.baillard2
  HostName pccop1b110-04.univ-eiffel.fr
  LocalForward 8188 127.0.0.1:8188" >> ~/ssh/config

$ ssh eiffel
```

Clé SSH

```
$ ssh-keygen
```

Côté client :
- .ssh/id_ed25519_eiffel (clé privée)
- .ssh/id_ed25519_eiffel.pub (clé publique)
- .ssh/config

Côté serveur :
- .ssh/authorized_keys (clés publiques)



Atelier n°4

Connexion à l’université

1) Se connecter en SSH sur une machine Linux de l’université

   http://sassafras-edu.univ-eiffel.fr/maps/sceco

2) Configurer une paire de clé ed25519 pour se connecter de votre machine perso aux machines Linux de l’université.

3) Se connecter avec la clé ed25519

4) Copier votre script “infos système” sur une machine de l’université et l’exécuter.