COURS #9

Protocoles L7, deuxième partie

Introduction aux réseaux 2025 (Bloc 2)

Corentin Badot-Bertrand

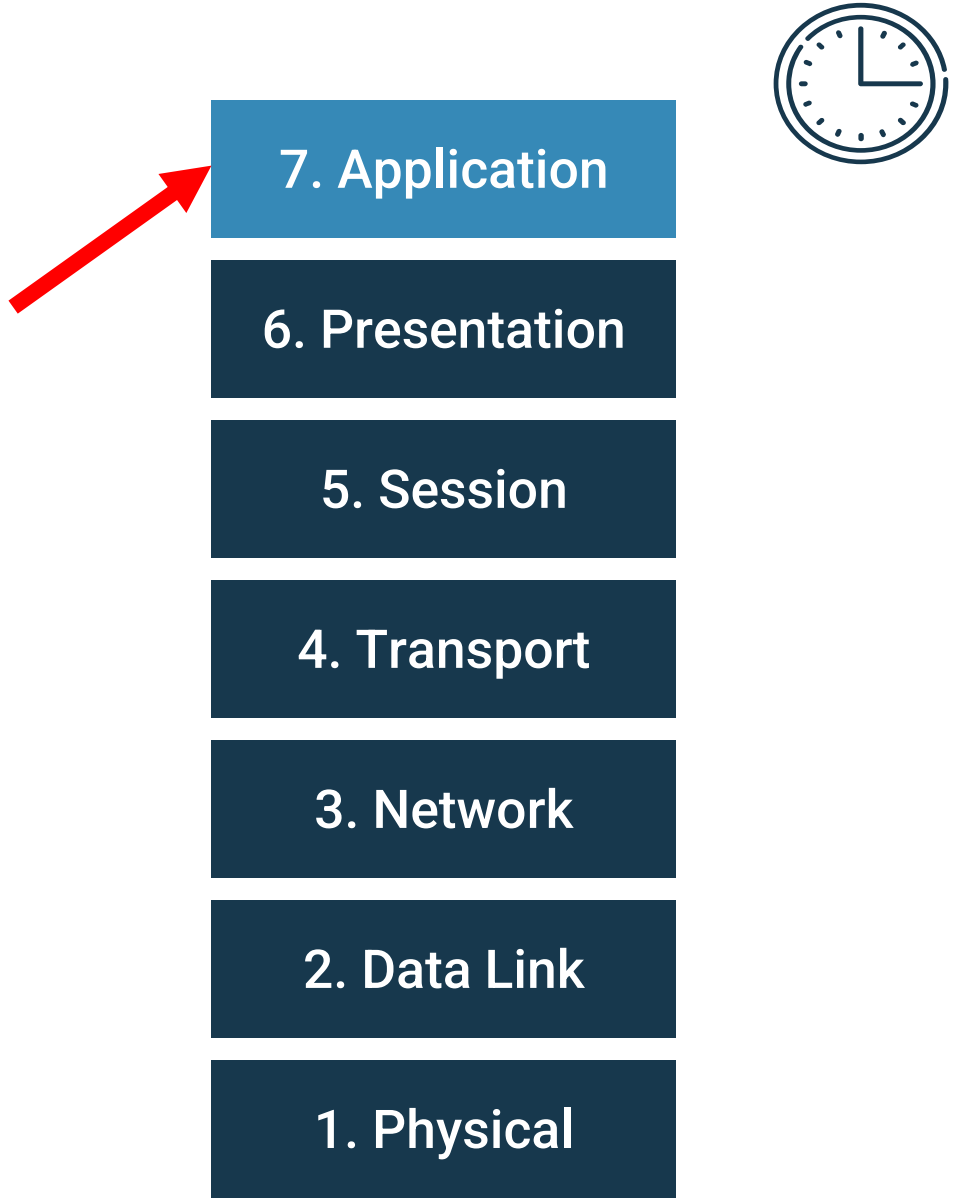
PREAMBULE

Rappels & mise en contexte

Quelques rappels sur le cours précédent avant de commencer

Dans l’épisode précédent

- Couche Application (L7)
- NAT
- DHCP
- DNS

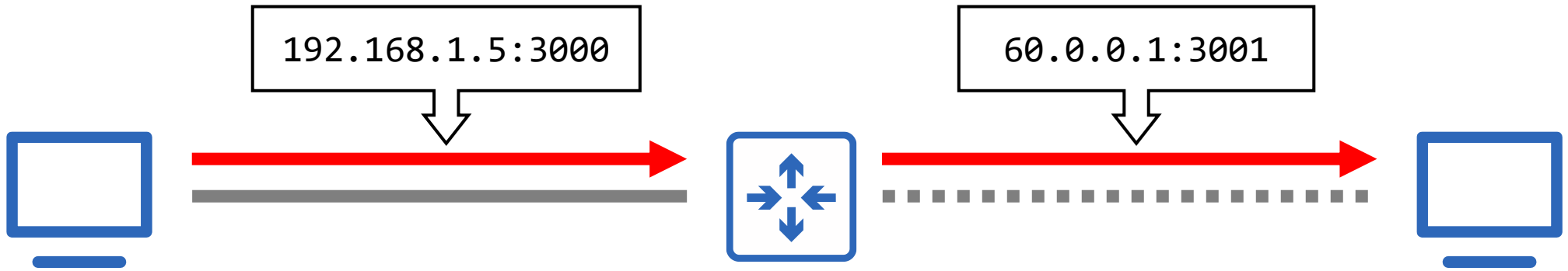



Questions
pour un
Champion

Comment peut-on passer d’un réseau privé vers un réseau public ?

NAT overlay (Port Address Translation)

Utilise les ports (TCP/UDP) pour partager une même adresse IPv4



Noeud interne

192.168.1.5

Noeud externe

50.0.0.1

IP interne	Port interne	IP externe	Port externe
192.168.1.5	3000	60.0.0.1	3001
192.168.1.6	3000	60.0.0.1	3002

Du port forwarding sur votre réseau domestique. Y a-t-il des risques ?

Access Control

Parental Control | Portmapping | Firewall | Remote Access

Port Map Rules

#	Enable	Service	Protocol	External start	External end	Lan port	Internal host	Remote host	Description
1	On Off	UPNP	UDP	57669	57669	57669	192.168.1.3	0.0.0.0	Teredo

#	Enable	Service	Protocol	External start	External end	Lan port	Internal host	Remote host	Description	

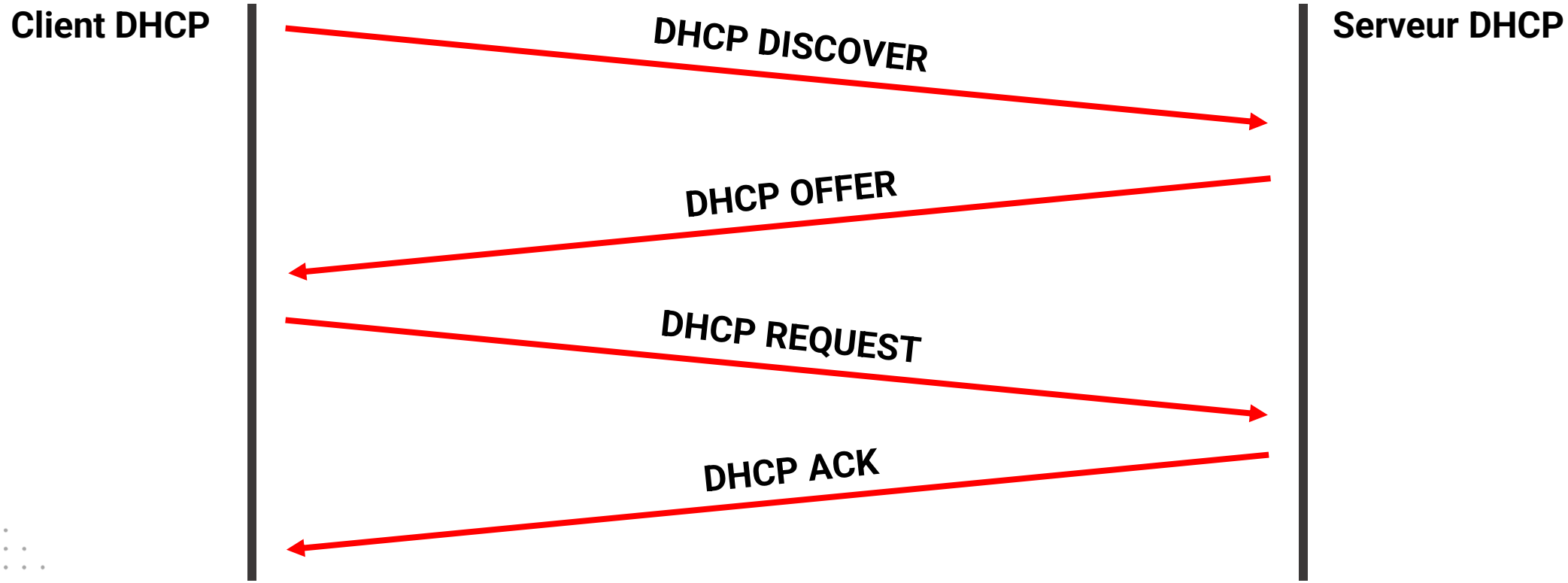
Create new portmap

Cancel OK

Quelles sont les 4 phases du protocole DHCP ?

DHCP, le déroulement

L’obtention d’une adresse IP se déroule en 4 phases



A quoi servent les ports 80 et 443 ?

Liste de ports TCP & UDP (bases)

Port	Protocole	TCP	UDP	Description
21	FTP			Transfert de fichiers (non-sécurisé)
22	SSH			Secure Shell & transfert de fichiers (sécurisé)
23	Telnet			Communications textuelles (non-sécurisé)
25	SMTP			Protocole d'envoi email (non-sécurisé)
53	DNS			Domain Name System
67/68	DHCP			Configuration de réseau dynamique
80	HTTP			Hypertext Transfer Protocol (non-sécurisé)
110	POP3			Protocole de réception email (non-sécurisé)

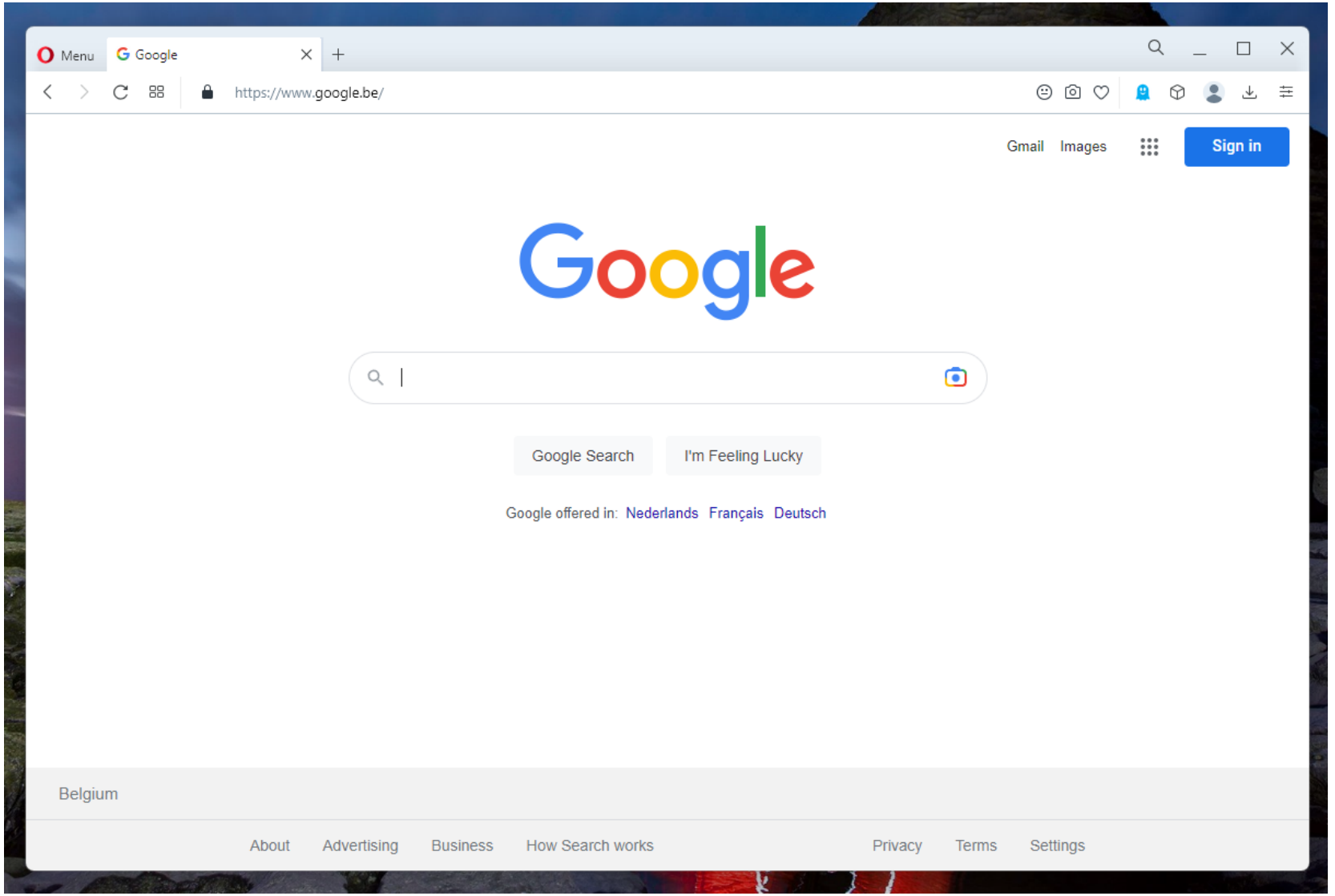
Liste de ports TCP & UDP (bases)

Port	Protocole	TCP	UDP	Description
123	NTP			Network Time Protocol, synchronisation du temps
143	IMAP			Protocole de réception email (non-sécurisé)
443	HTTPS			Hypertext Transfer Protocol (sécurisé, TLS/SSL)
465	SMTP (s)			Protocole d’envoi email (sécurisé, TLS/SSL)
993	IMAP (s)			Protocole de réception email (sécurisé, TLS/SSL)
995	POP3 (s)			Protocole de réception email (sécurisé, TLS/SSL)
3306	MySQL			Base de données MySQL
5432	PostgreSQL			Base de données PostgreSQL

PARTIE #1

HTTP, le protocole web

Hypertext Transfer Protocol,
Indispensable pour Internet


Menu
Google
https://www.google.be/
Gmail
Images
Sign in
Google
Google Search
I'm Feeling Lucky
Google offered in: Nederlands Français Deutsch
Belgium
About
Advertising
Business
How Search works
Privacy
Terms
Settings

Le protocole HTTP

Hypertext Transfert Protocol

- Développé par Tim Berners-Lee
- A servi aux fondations du World Wide Web avec HTML & URL
- Protocole de communication pour pages web
- Utilise le port 80 en TCP
- Protocole non-sécurisé
- Un client HTTP peut demander une page web (document)
- … et recevoir une réponse d'un serveur HTTP

Le protocole HTTP

Quelques spécificités techniques

- La requête HTTP est envoyée après l'établissement d'une connexion
- Les navigateurs modernes réutilisent des connexions TCP
- HTTP est **stateless** : il ne garde aucun état ni historique

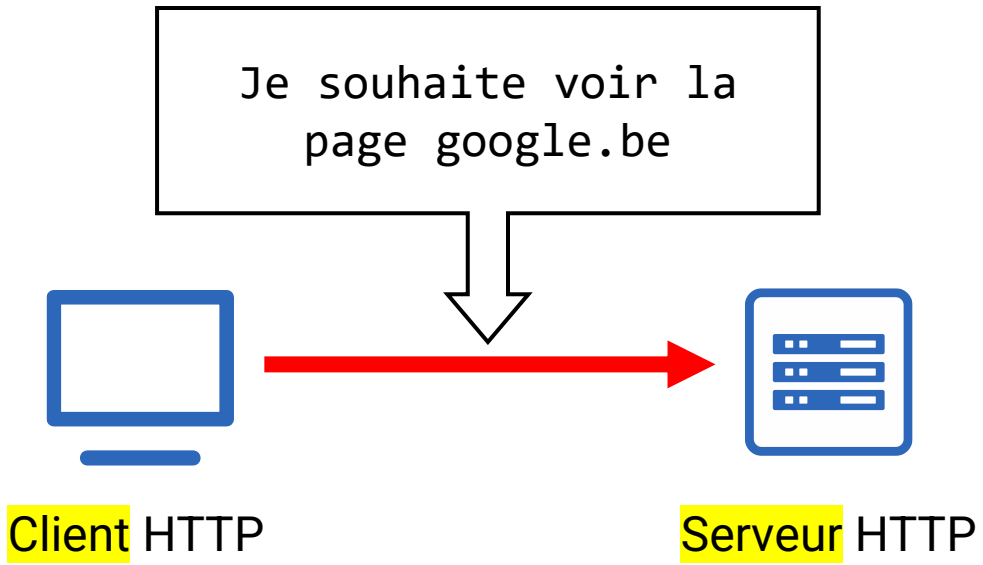
Connexions TCP & HTTP

Chaque version de HTTP diffère dans la gestion des connexions

- HTTP/1.0
  - Protocole historique
  - Ouverture d'une connexion TCP pour chaque requête réponse
- HTTP/1.1
  - Protocole courant
  - Les connexions deviennent « persistantes » (réutilisation)
  - Contrôlé via le header « Connection »
- HTTP/2
  - En cours de déploiement – tous les sites ne le proposent pas par défaut
  - Multiplexage des ressources au sein d'une même connexion

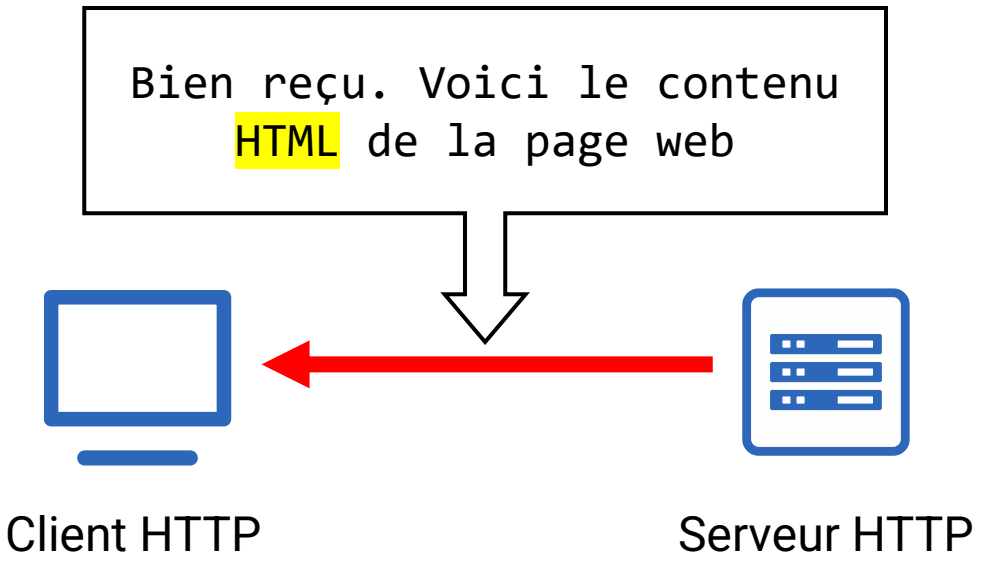
Le protocole HTTP : requête

Permet au client HTTP de demander une ressource (ou une action)



Le protocole HTTP : réponse

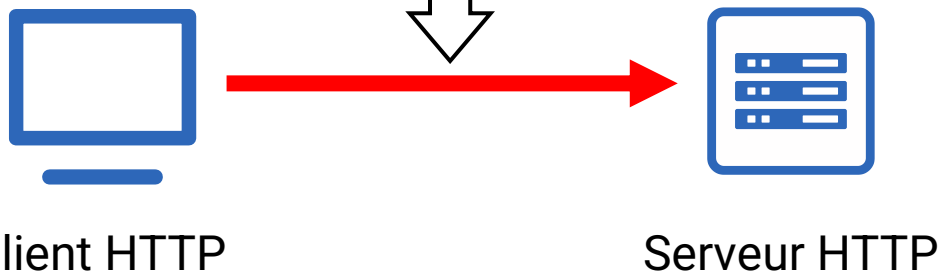
Fait suite à une requête et est envoyée par le serveur HTTP en réponse au client



Les URL, une structure pour les requêtes

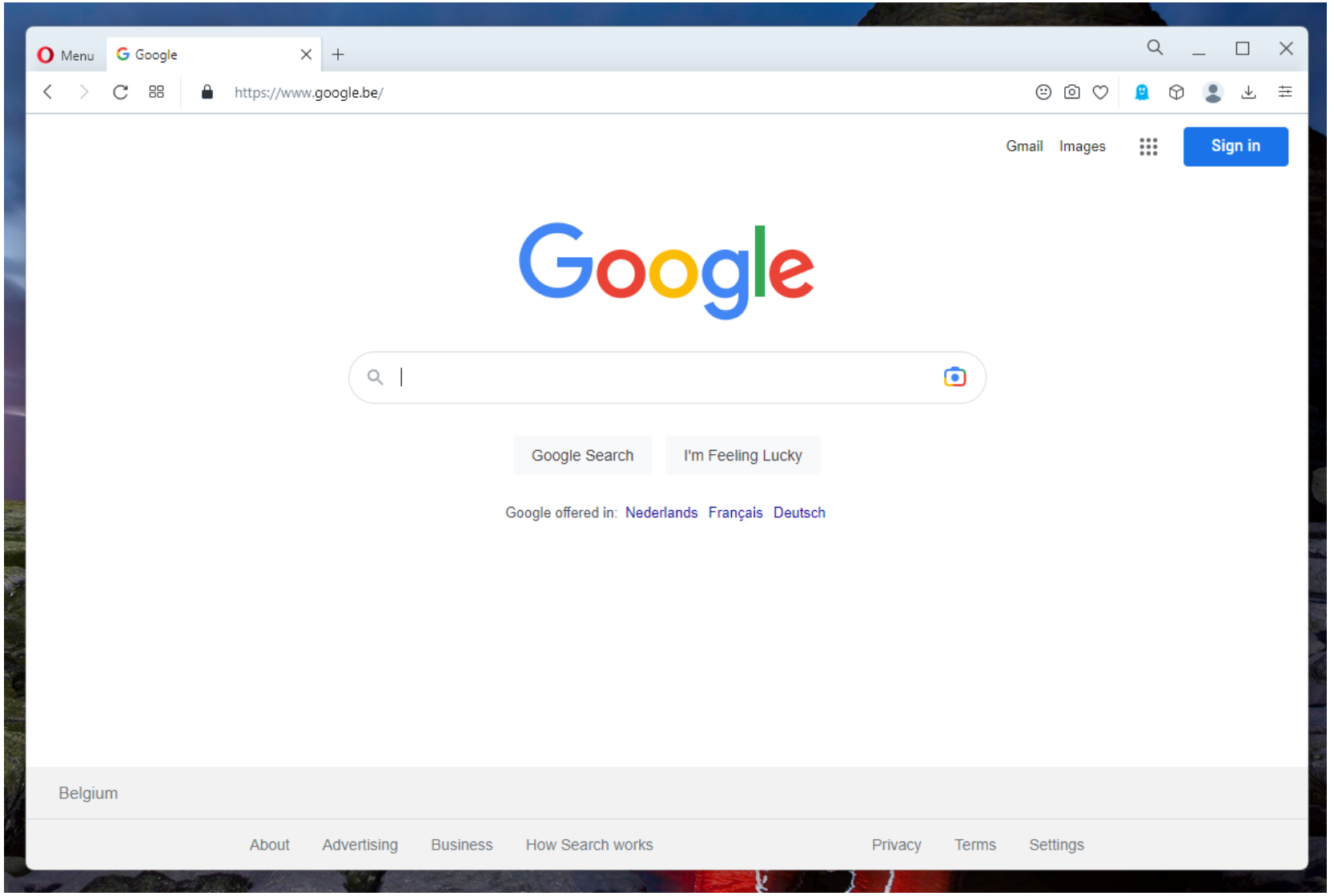
Uniform Resource Locator – permet de demander une ressource (document) de façon structurée

```
protocol://user:password@host:port/path?query
http://john:secret123@www.google.be:80/index.html?lang=fr
```



Client HTTP

Serveur HTTP


Menu
Google
https://www.google.be/
Gmail
Images
Sign in
Google
Google Search
I'm Feeling Lucky
Google offered in: Nederlands Français Deutsch
Belgium
About
Advertising
Business
How Search works
Privacy
Terms
Settings

Les composants d'une URL

`http://john:secret123@www.google.be:80/index.html?lang=fr`

- Utilise le protocole HTTP
- Utilisateur « john » & mot de passe pour s'authentifier sur le serveur
- Se connecter sur le serveur google.be avec résolution DNS
- Utiliser le port TCP 80
- Demander le document « index.html » à la racine du site web
- Utiliser le paramètre « lang » avec la valeur « fr »

Les composants d'une URL

~~http://john:secret123@~~www.google.be~~:80/index.html?lang=fr~~

- Protocole HTTP - complété par la navigateur
- Utilisateur - rarement utilisé sur les sites publics
- Serveur google.be – le seul paramètre indispensable
- Port TCP 80- complété par la navigateur
- Document – par défaut, « index.html » est demandé
- Query – optionnel, souvent défini par les applications web

Les actions (méthodes) HTTP

Méthode	Signification
GET (défaut)	Récupérer une ressource (*download*)
POST	Envoyer une ressource (*upload*)
PUT	Remplacer intégralement une ressource
PATCH	Modifier partiellement une ressource
DELETE	Supprimer une ressource

Contenu d'une requête HTTP

```
GET /fr/formations HTTP/1.1
Accept: text/html,application/xhtml+xml,application/xml
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,fr;q=0.8
Connection: keep-alive
Cookie: _ga=GA1.1.
Host: www.vinci.be
Referer: https://www.vinci.be/fr
User-Agent: Mozilla/5.0
```

Une requête HTTP est composé de texte – dont le premier composant est la méthode

(quelle actions voulons-nous effectuer ?)

Contenu d'une requête HTTP

```
GET /fr/formations HTTP/1.1
Accept: text/html,application/xhtml+xml,application/xml
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,fr;q=0.8
Connection: keep-alive
Cookie: _ga=GA1.1.
Host: www.vinci.be
Referer: https://www.vinci.be/fr
User-Agent: Mozilla/5.0
```

Nous demandons ensuite la ressource HTTP (URL, à partir du chemin) avec laquelle nous souhaitons interagir

Je veux voir la ressource /fr/formations

Contenu d'une requête HTTP

```
GET /fr/formations HTTP/1.1
Accept: text/html,application/xhtml+xml,application/xml
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,fr;q=0.8
Connection: keep-alive
Cookie: _ga=GA1.1.
Host: www.vinci.be
Referer: https://www.vinci.be/fr
User-Agent: Mozilla/5.0
```

Nous spécifions la version du protocole HTTP (1.1)

Contenu d'une requête HTTP

```
GET /fr/formations HTTP/1.1
Accept: text/html,application/xhtml+xml,application/xml
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,fr;q=0.8
Connection: keep-alive
Cookie: _ga=GA1.1.
Host: www.vinci.be
Referer: https://www.vinci.be/fr
User-Agent: Mozilla/5.0
```

Nous passons des **headers** (options supplémentaires) dans la requête HTTP

Contenu d'une réponse HTTP

```
HTTP/1.1 200 OK
Server: nginx/1.14.2
Date: Tue, 02 May 2023 15:52:01 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
Vary: Accept-Encoding
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Encoding: gzip
```

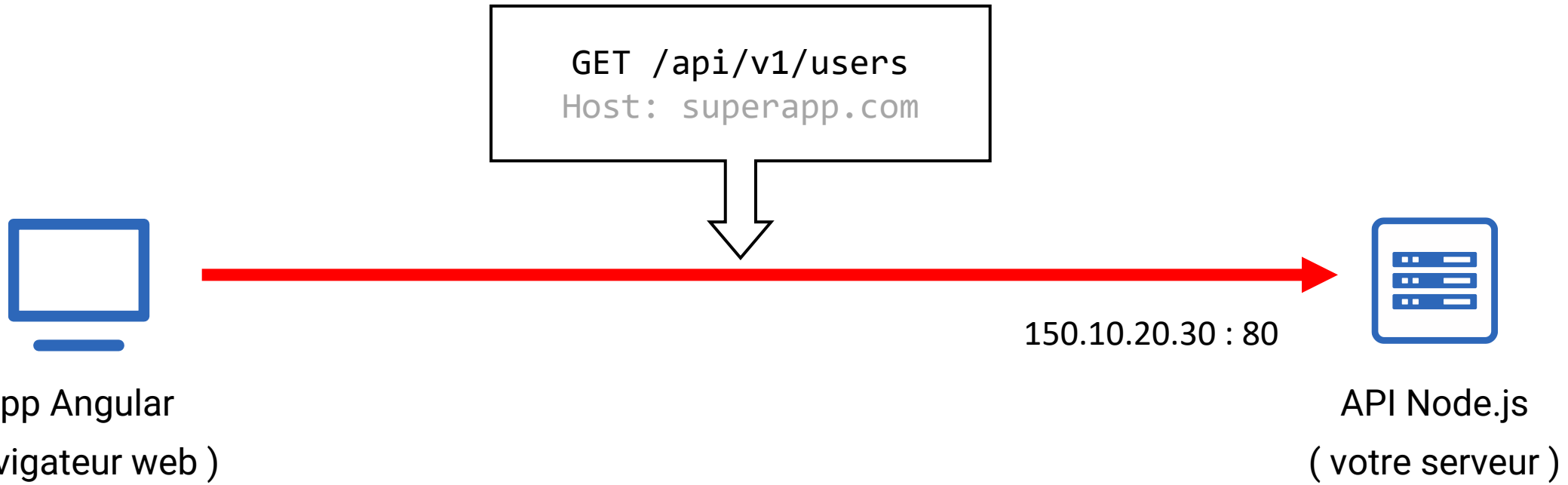
Le serveur renvoie la ressource précédée d'un **code de réponse HTTP** et de headers

Les codes de réponse HTTP (status codes)

Code	Signification
200	OK, tout s’est bien déroulé
201	La ressource a été créé
301	URL changée de façon permanente
400	*Bad request*, requête malformée par le client
401	*Unauthorized*, vous n’êtes pas authentifié
403	*Forbidden*, vous n’avez pas les droits nécessaires
404	*Not found*, ressource introuvable
500	Erreur côté serveur (crash système, …)

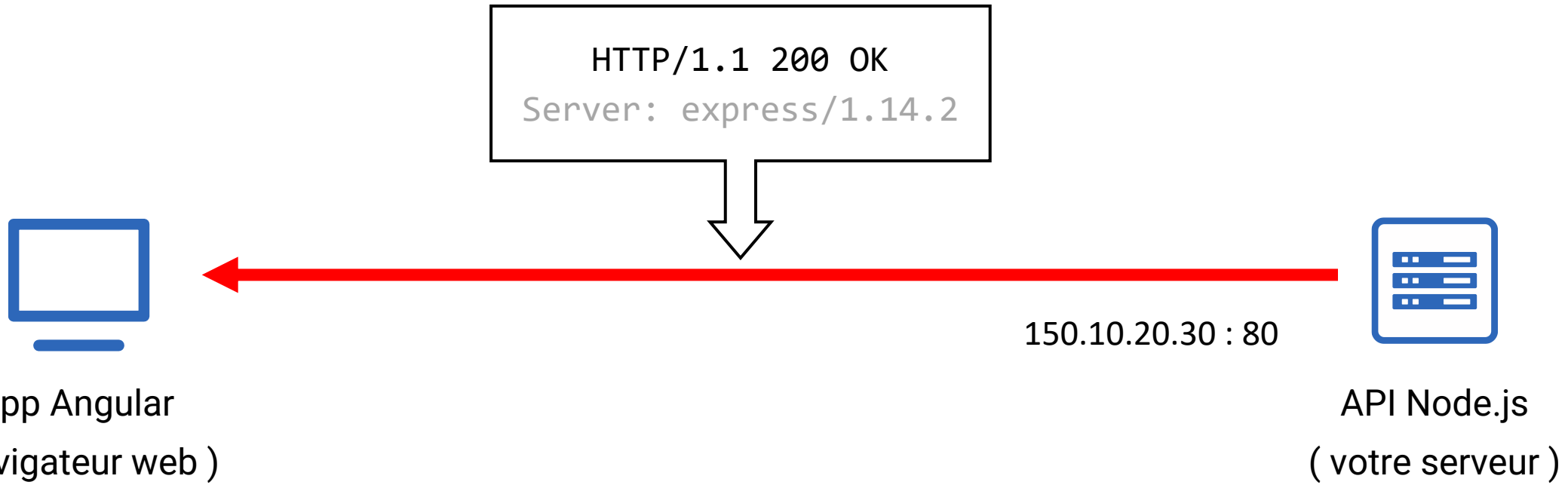
Un exemple pratique

Une API écrite en Node.js et une application web Angular.



Un exemple pratique

Une API écrite en Node.js et une application web Angular.



Est-ce une bonne approche pour déployer votre API Node.js ?

Problèmes potentiels

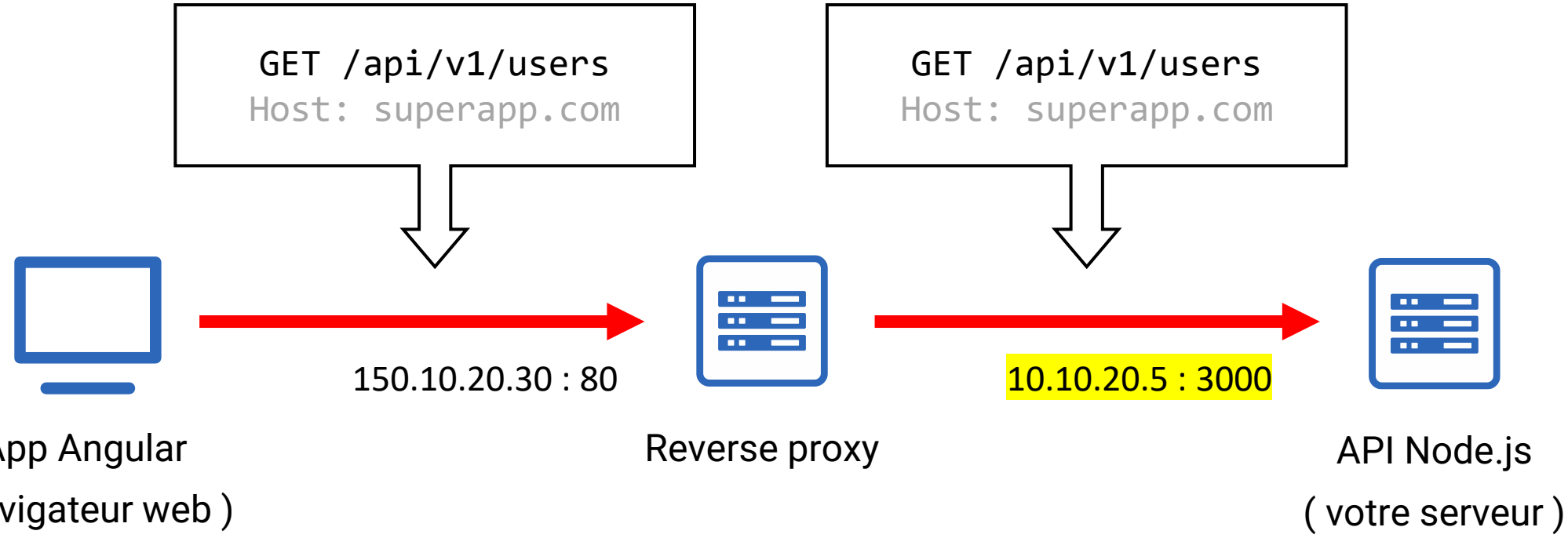
Quelques problèmes avec cette approche minimaliste

- Le process utilise un port privilégié (80) – tourne donc en privilégié
- Pas de *scaling* horizontal possible – on rajoute de la capacité à la machine
- Le serveur Express est exposé – vulnérable en cas de CVE
- Pas de redondance en cas de problèmes sur le serveur
- ….

Cette approche n’est pas privilégiée pour une production stable

Reverse proxy

Serveur exposé sur Internet permettant d'accéder à des serveurs internes



Reverse proxy

Technique courante pour optimiser les applications web

- Le reverse proxy est souvent un logiciel éprouvé (NGINX, Apache, Caddy, …)
- Permet de rediriger le trafic web sur base de règles
  - Machines disponibles
  - Round robin
  - Header « Host »
  - …
- Effectue de la compression et de la mise en cache
- Porte souvent le chiffrement des flux (HTTPS)
- Et éventuellement : authentification, filtre de sécurité, journalisation (!), …

Reverse proxy – projets de référence

PARTIE #2

FTP

Un protocole spécialisé dans
le transfert des fichiers

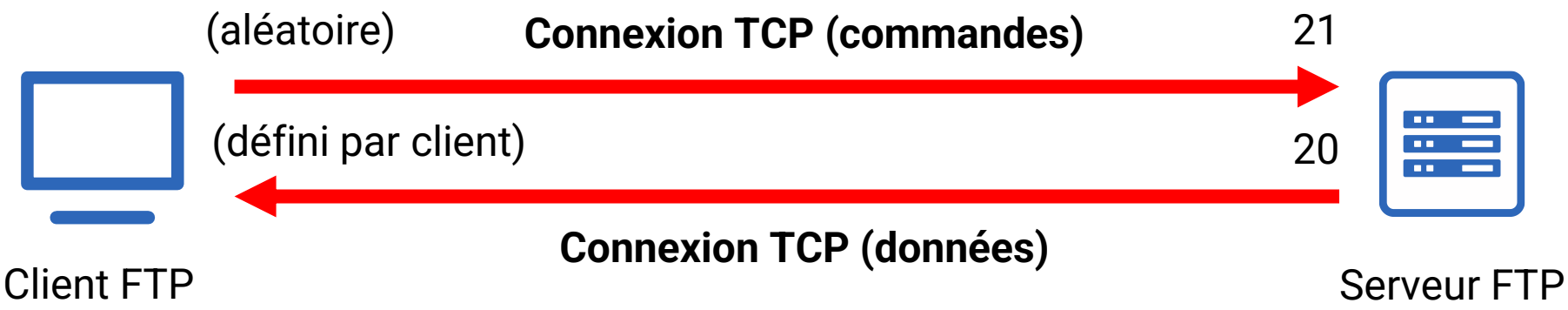
Le protocole FTP

File Transfer Protocol

- Permet le partage de fichiers
- Utilise une connexion de « contrôle » (TCP, port 21)
- Utilise une connexion de « données » (TCP, port 20)
- Protocole non-sécurisé – préférez le SFTP
- Fonctionne avec un système de
  - Commandes (HELP, STATUS, …)
  - Réponses numériques

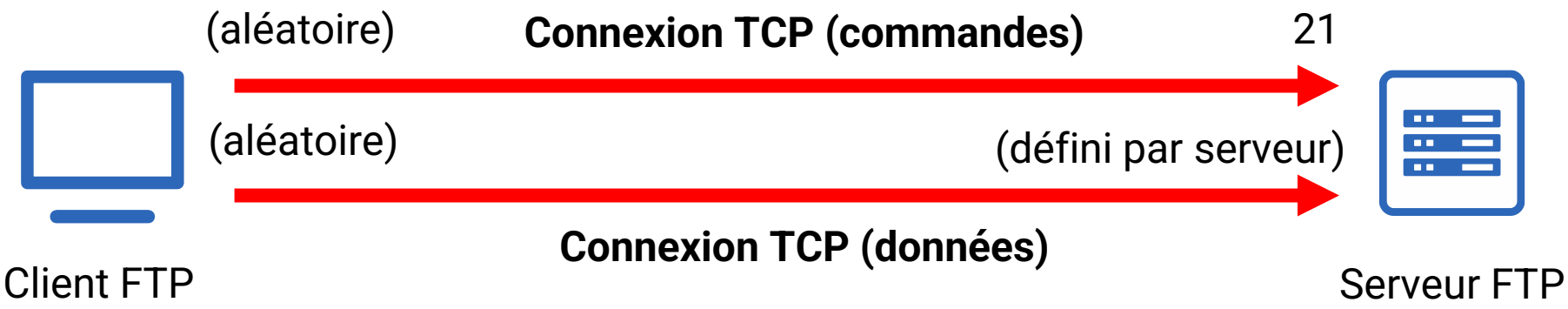
FTP, connexion en mode « active »

Le client détermine un port de son côté pour échanger les données, le serveur FTP s'y connecte pour le transfert



FTP, connexion en mode « passive »

Le serveur FTP détermine un port de son côté pour échanger les données, le client s'y connecte pour le transfert



Commandes FTP

Commande	Signification
HELP	Affiche les commandes disponibles
STATUS	Donne l’état de la connexion
USER	Spécifie l’utilisateur pour une connexion
PASS	Spécifie le mot de passe
STOR	Envoyer des données vers le serveur
PORT	Spécifie une adresse & port de connexion
QUIT	Fermeture de connexion

Le protocole FTP, une bonne cible pour les attaquants