# Curl

Afin d'éviter les écueils dus aux éventuels problèmes de redirection réseau des ports des machines virtuelles, nous allons utiliser la commande `curl` pour tester les sites web que nous mettrons en place au cours des TP.

Ceci constitue un petit inventaire des commandes les plus utiles de `curl` pour notre cas.

# Utilisation de base

```
curl http://example.org
```

La commande `curl` télécharge la ressource demandée (qui n'est pas nécessairement une adresse web, car `curl` est capable de télécharger des ressources d'autres protocoles, comme `ftp` par exemple) et en affiche le contenu sur la sortie standard, si ce contenu n'est pas un contenu binaire.

# Rediriger la ressource vers un fichier

```
curl http://example.org > fichier.html
curl http://example.org --output fichier.html
curl http://example.org -o fichier.html
```

# Réduire la verbosité de curl

Lorsque la sortie est redirigée vers un fichier, `curl` affiche une barre de progression donnant certaines indications sur le téléchargement de la ressource (le temps restant, la taille…).

Pour ne pas afficher ces informations, on utilise l'option `--silent` ou son abbréviation `-s`.

```
curl --silent http://example.org -o fichier.html
curl -s http://example.org -o fichier.html
```

# Faire autre chose qu'un GET

Pour utiliser une autre méthode HTTP que `GET`, on utilise l'option `--request` ou son abbréviation `-X`.

```
curl --request POST http://example.org
curl -X POST http://example.org
```

# Faire une requête HEAD

Si on tente de faire une requête `HEAD` avec l'option `--request`, `curl` affichera un message d'erreur :

```
Warning: Setting custom HTTP method to HEAD with -X/--request may not work the
Warning: way you want. Consider using -I/--head instead.
```

Il convient d'utiliser l'option `--head` ou son abbréviation `-I` :

```
curl --head http://example.org
curl -I http://example.org
```

# Pour faire une requête avec authentification HTTP

On spécifie l'identifiant et le mot de passe avec l'option `--user` ou son abbréviation `-u`, en les séparant par un caractère `:`.

```
curl --user login:password http://example.org
curl -u login:password http://example.org
```

# Forcer la connexion en IPv6 ou en IPv4

On utilise pour cela les options `-6` et `-4`.

```
curl -6 http://example.org
curl -4 http://example.org
```

# Utilisation avancée

## Forcer la connexion sur une autre adresse IP

Il est possible de dire à `curl` d'utiliser une adresse IP particulière au lieu de la véritable adresse IP d'un domaine. Il faut voir cela comme une alternative à la manipulation du fichier `/etc/hosts`.

```
curl --resolve example.org:80:127.0.0.1 http://example.org
curl --resolve "example.org:80:[::1]" http://example.org
```

La syntaxe de l'option est `host:port:addr`. Le port est celui qui sera utilisé par le protocole. Spécifier le port `80` pour le protocole `https` serait inutile : il faut dans ce cas utiliser le port `443`.

# Forcer la connexion sur une autre adresse IP et un autre port

On utilise pour cela l'option `--connect-to`, relativement similaire à l'option `--resolve`. La syntaxe de l'option est `host1:port1:host2:port2`

```
curl --connect-to example.org:80:127.0.0.1:8080 http://example.org
curl --connect-to "example.org:80:[::1]:8080" http://example.org
```

# Forcer la connexion depuis une certaine interface réseau

On utilise pour cela l'option `--interface` suivi du nom d'une interface, d'une adresse IP ou d'un nom d'hôte.

```
curl --interface wlo1 http://example.org
curl --interface 203.0.113.42 http://example.org
curl --interface example.com http://example.org
```

# Ne pas vérifier la sécurité du certificat du site

Que ce soit parce qu'un certificat est expiré ou parce qu'on utilise un certificat autosigné, on peut avoir besoin que `curl` effectue bien la requête sans se préoccuper de la validité du certificat utilisé par le site. On utilise alors l'option `--insecure` ou son abbréviation `-k`.

```
curl --insecure https://example.org
curl -k https://example.org
```

# Utiliser un fichier d’autorités de certification spécifique

Si on utilise, par exemple, un certificat autosigné, ou signé par une autorité de certification (AC) personnelle, et qu’on souhaite s’assurer que le certtificat utilisé par le site est bien valide, on peut donner à `curl` un fichier contenant le certificat public de l’AC (il est possible d’y mettre plusieurs certificats) au format PEM. On utilise l’option `--cacert`.

```
curl --cacert fichier_AC.pem https://example.org
```

On peut aussi utiliser l’option `--capath` dont l’argument est un dossier contenant des fichiers de certificats d’AC.

```
curl --capath ~/ACs https://example.org
```

---

Révision #1
Créé 19 janvier 2023 10:11:50 par Luc
Mis à jour 21 août 2023 09:10:56 par Luc