**TP - Maîtrise du CMD Windows : Administration système et réseau**

**Objectifs pédagogiques**

- Maîtriser les commandes essentielles de l'invite de commandes Windows
- Comprendre la gestion des fichiers, processus et réseaux
- Développer des compétences en troubleshooting système

## Exercice 1 : Prise en main de l'environnement CMD

### 1.1 Lancement et configuration

- Ouvrez CMD en tant qu'administrateur :
  - Win + X → Invite de commandes (admin) **OU**
  - Win + R → cmd → Ctrl + Shift + Entrée

**Tâches :**

1. Vérifiez votre dossier courant avec ***echo %CD%***
2. Affichez la version de Windows avec ver
3. Capture d'écran montrant ces informations

### 1.2 Personnalisation de l'environnement

- Tapez ***color 0A***
- Tapez ***prompt $T [$P]$G***
- Tapez ***title Session_Admin_CMD***

**Questions :**

- Que fait la commande color ?

  **Change la couleur du CMDt**
- Comment le prompt a-t-il changé ?

  $T **: Affiche l'heure actuelle.**

  [$P] **: Affiche le répertoire actuel entre crochets.**

  $G **: Affiche le caractère > comme séparateur.**
- À quoi sert la commande title ?

  **La commande « title » dans l'invite de commande Windows sert à changer le titre de la fenêtre du terminal.**

## Exercice 2 : Exploration avancée du système de fichiers

### 2.1 Navigation et arborescence

**Tâches :**

1. Créez la structure suivante :

   ***TP_CMD***

   ***Documents/***

   ***Textes/***

*Images/*

*Archives/*

*Temp/*

**2.2 Attributs de fichiers avancés**

***echo CONTENU_SECRET > Documents\fichier_cache.txt***

***dir***

***attrib +H Documents\fichier_cache.txt***

***dir Documents***

***dir Documents /A***

**Questions :**

- Que signifient les attributs +H ?

  **`attrib` : commande Windows qui permet d'afficher ou de modifier les attributs d'un fichier ou dossier.**

  **`+H` : ajoute l'attribut** "Hidden" **(caché).**

  **`Documents\fichier_cache.txt` : chemin relatif vers le fichier ciblé (ici, un fichier `fichier_cache.txt` dans le dossier `Documents`).**

- Comment afficher les fichiers cachés avec dir ?

  **`dir` : liste le contenu d'un dossier.**

  **`/A` : affiche** tous les fichiers**, y compris ceux avec des attributs spéciaux (**cachés**,** système**,** lecture seule**, etc.).**

## Exercice 3 : Gestion des processus et services

**3.1 Analyse des processus système**

***tasklist /FO TABLE /V***

**Tâches :**

1. Lancez le Bloc-notes et Paint
2. Identifiez leurs PID avec :

   ***tasklist | findstr "notepad mspaint"***

   ***Notepad : 1660 Console –1—13932ko***

**3.2 Gestion fine des processus**

***:: Tuer un processus par PID***

***taskkill /PID [PID_NUMBER] /F***

***taskkill /PID 1660 /F***

***:: Tuer par nom d'image***

*taskkill /IM notepad.exe /T*

**Questions :**

- **Quelle est la différence entre /F et /T ?**

  **/F : Force ou détails selon la commande**

  **taskkill /F /PID 1234 (Force le processus à sa fermeture)**

  **/T : tue aussi tous les processus enfants (avec /T).**

  **taskkill /F /T /PID 1234**

- **Pourquoi faut-il être prudent avec taskkill ?**

  **Taskkill ferme des processus et selon le processus fermé cela peut endommager le fonctionnement de l'OS.**

## Exercice 4 : Diagnostic réseau avancé

### 4.1 Analyse complète de la configuration

*ipconfig /all*

*ou*

*netsh interface ip show config*

**Questions :**

- Quelle est votre adresse MAC ?

  **MAC / Adresse physique : 08-00-27-F3-80-A8**

- Le DHCP est-il activé ?

  **Oui**

- C'est quoi un DHCP ?

  Un DHCP : **est un protocole réseau qui sert à attribuer automatiquement des adresses IP et d'autres paramètres réseau aux appareils d'un réseau (comme ton PC, ton smartphone, une imprimante, etc.).**

  **En simple : Le DHCP permet à un appareil de se connecter facilement à un réseau sans qu'on ait à configurer manuellement son adresse IP.**

- Quel serveur DNS utilisez-vous ?

  **Serveur DNS : 193.49.251.6**

### 4.2 Tests de connectivité

*ping 8.8.8.8 -n 4*

*ping google.com*

*tracert google.com*

*pathping google.com*

**Tâches :**

1. Explique le résultat du premier ping
   - Réponse de 8.8.8.8 : octets=32 temps=12 ms TTL=115

**8.8.8.8 : adresse IP cible pingée.**

**octets=32 : taille des paquets envoyés (32 octets par défaut).**

**temps=12 ms : temps que met le paquet aller-retour (latence).**

**TTL=115 : "Time To Live", nombre de sauts restants (indique combien de routeurs le paquet peut traverser avant d'être supprimé).**

**En résumé : Cette ligne te dit que le ping vers 8.8.8.8 a réussi.**

**La connexion est rapide (12 millisecondes).**

**Le TTL indique que le paquet a traversé quelques routeurs (valeur descendante).**

2. Comparez les résultats de tracert et pathping

| Caractéristiques | tracert | pathping |
|---|---|---|
| Affiche la route | Oui | Oui |
| Analyse la perte | Non | Oui |
| Temps d'exécution | Rapide | Plus long (20-30 secondes) |
| Type de paquets envoyés | ICMP | ICMP |
| Usage typique | Diagnostic rapide de route | Diagnostic approfondi de réseau |

3. Testez la résolution DNS avec nslookup google.com

**nslookup signifie Name Server Lookup.**

**Cette commande demande au serveur DNS (par défaut celui configuré sur ta machine ou réseau) de résoudre le nom de domaine `google.com` en adresse IP.**

**Elle peut aussi fournir d'autres infos DNS comme les serveurs autoritaires, en fonction des options.**

### 4.3 Statistiques réseau

***netstat -ano***

***netstat -an | find ":80"***

***netstat -e 5***

**Questions :**

- Que montre netstat -ano ?

**`netstat` : affiche les connexions réseau TCP/IP, les ports ouverts, les statistiques réseau, etc.**

**`-a` : affiche** toutes les connexions et ports d'écoute **(écoute en attente de connexion).**

**`-n` : affiche les adresses et numéros de ports** en format numérique **(évite la résolution des noms).**

**`-o` : affiche l'**ID du processus (PID) **lié à chaque connexion.**

- Comment identifier les connexions établies sur le port 80 ?

  **netstat -ano | findstr ":80"**

## Exercice 5 : Scripting basique et automatisation

### 5.1 Création d'un script de sauvegarde

Créez un fichier sauvegarde.bat :

```
@echo off
echo === SAUVEGARDE DOSSIERS IMPORTANTS ===
set BACKUP_DIR=%USERPROFILE%\Backup_%DATE%
mkdir "%BACKUP_DIR%"

xcopy "%USERPROFILE%\Documents\*.txt" "%BACKUP_DIR%\Textes\" /S /I /Y
xcopy "%USERPROFILE%\Desktop\*.pdf" "%BACKUP_DIR%\PDF\" /S /I /Y

echo Sauvegarde terminee: %BACKUP_DIR%
dir "%BACKUP_DIR%" /S
pause
```

1. Que fait la commande @echo off et pourquoi est-elle utile ici ?

   **Désactive l'affichage des commandes elles-mêmes dans la console pour que la sortie soit plus propre (on ne voit que les résultats, pas les commandes).**

2. Que fait la commande set BACKUP_DIR=%USERPROFILE%\Backup_%DATE% ?

**Définit une variable d'environnement `BACKUP_DIR` qui contiendra le chemin du dossier de sauvegarde.**

**`%USERPROFILE%` = chemin du profil utilisateur (ex : `C:\Users\TonNom`)**

**`%DATE%` = date actuelle (format dépend de ta configuration régionale Windows)**

**Résultat : un dossier comme `C:\Users\TonNom\Backup_07/10/2025`**

- Que contient la variable %USERPROFILE% ?

3. **`%USERPROFILE%` = chemin du profil utilisateur (ex : `C:\Users\TonNom`)**
4.

- Que contient la variable %BACKUP_DIR% après son exécution ?

  **elle correspond au chemin du dossier de sauvegarde construit à partir du dossier utilisateur (%USERPROFILE%) suivi de Backup_ et de la date actuelle (%DATE%).**

5. Que fait mkdir "%BACKUP_DIR%" ?

   **Mkdir crée un dossier dans le %BACKUP_DIR%**

   - Que se passerait si le dossier existait déjà ?

     **Un message dans le cmd dirait : fichier déjà existant.**

6. Que fait xcopy "%USERPROFILE%\Documents*.txt" "%BACKUP_DIR%\Textes\" /S /I /Y ?

   - Que signifient les options /S /I /Y ?

**`/S`**
**Copie** tous les fichiers **correspondant au filtre (ici `*.txt`)** dans les sous-dossiers **du dossier source, sauf les dossiers vides.**
**→ Cela permet de copier aussi les fichiers `.txt` dans tous les sous-dossiers de `Documents`.**

**`/I`**
**Si la destination n'existe pas, cette option indique à `xcopy` de la traiter comme un** dossier **(et non un fichier).**
**→ Utile pour éviter qu'il demande confirmation quand il doit créer un nouveau dossier.**

**`/Y`**
**Supprime la demande de confirmation avant d'**écraser un fichier existant **dans la destination.**
**→ La copie se fait donc automatiquement sans interruption.**

7. Pourquoi crée-t-on deux dossiers différents (Textes et PDF) ?

   **Dans le script, on crée deux dossiers différents — Textes et PDF — pour organiser la sauvegarde des fichiers selon leur type :**

- **Le dossier Textes va contenir tous les fichiers .txt provenant du dossier Documents.**
- **Le dossier PDF va contenir tous les fichiers .pdf provenant du Bureau (Desktop).**

8. Que fait la commande dir "%BACKUP_DIR%" /S ?

**`dir` : commande qui liste le contenu d'un dossier.**

**`"%BACKUP_DIR%"` : chemin du dossier dont on veut afficher le contenu (ici la variable `%BACKUP_DIR%`).**

**`/s` : option qui indique à `dir` d'afficher** le contenu du dossier spécifié ET de tous ses sous-dossiers **(récursivement).**

9. Quel est le rôle de pause ? Que se passerait si on le supprimait ?

**`pause` est là pour te laisser le temps de voir les résultats avant de fermer la fenêtre.**

Sans `pause`, la console se ferme directement à la fin du script, ce qui peut être frustrant si tu veux vérifier ce qu'il s'est passé.

**Améliorations**

1. Comment pourriez-vous **sauvegarder d'autres types de fichiers** (images, vidéos) ?

   **Pour ajouter la sauvegarde d'autres types de fichiers, il suffit de rajouter des lignes xcopy supplémentaires dans le script, ciblant les extensions souhaitées. Par exemple :**

   **xcopy "%USERPROFILE%\Pictures*.jpg" "%BACKUP_DIR%\Images\" /S /I /Y**

   **xcopy "%USERPROFILE%\Pictures*.png" "%BACKUP_DIR%\Images\" /S /I /Y**

   **xcopy "%USERPROFILE%\Videos*.mp4" "%BACKUP_DIR%\Videos\" /S /I /Y**

   **xcopy "%USERPROFILE%\Videos*.avi" "%BACKUP_DIR%\Videos\" /S /I /Y**

2. Comment rendre le script **plus robuste** si un fichier est en cours d'utilisation ou s'il y a un problème de droits ?

   **Utiliser robocopy à la place de xcopy :**

   **robocopy est plus puissant, résilient, et gère mieux les erreurs**

   **robocopy "%USERPROFILE%\Documents" "%BACKUP_DIR%\Documents" *.txt /S /R:3 /W:5 /COPY:DAT**

3. Comment automatiser ce script pour qu'il **s'exécute tous les jours** sans intervention manuelle ?

   **Utiliser le Planificateur de tâches (Task Scheduler) :**

1. **Ouvre le Planificateur de tâches (taskschd.msc).**
2. **Crée une tâche basique.**
3. **Choisis un déclencheur : tous les jours à une heure définie.**
4. **Comme action, sélectionne Démarrer un programme.**
5. **Indique le chemin vers ton script .bat.**
6. **Configure les options pour exécuter même si l'utilisateur n'est pas connecté, avec les droits nécessaires.**
7. **Enregistre la tâche.**

   Le script se lancera alors automatiquement chaque jour à l'heure choisie, sans intervention.

**5.2 Surveillance système**

Créez monitor.bat :

```
@echo off
:loop
cls
echo === MONITORING SYSTEME ===
echo Date: %DATE% - Heure: %TIME%
echo.
echo === PROCESSUS ===
tasklist /FO TABLE | findstr /I "chrome firefox"
echo.
echo === CONNEXIONS RESEAU ===
netstat -an | find ":80"
timeout /t 10 /nobreak
goto loop
```

1. Que fait la commande @echo off ? Pourquoi est-elle utile dans ce script ?

   Elle désactive l'affichage des **commandes elles-mêmes** dans la console pendant l'exécution.
   Sans echo off, à chaque ligne du script, la commande serait affichée avant son résultat, ce qui rend la sortie moins lisible.

2. Quelle est la fonction de l'étiquette :loop et du goto loop ?

: loop : loop est une **étiquette** (ou un point de repère) dans le script batch.

Elle sert à **marquer un emplacement précis** dans le script vers lequel on peut revenir ou sauter.

goto loop demande au script de **revenir à l'étiquette :loop** et de continuer l'exécution à partir de là.

Cela crée une **boucle infinie**, car après avoir exécuté toutes les commandes sous :loop, le script revient au début de cette section.

3. Que se passe-t-il quand on exécute cls ? Pourquoi est-ce important dans ce script ?

**Comme le script est dans une** boucle infinie (`:loop` **avec** `goto loop`**), sans** `cls`**, chaque cycle afficherait les nouvelles informations** en dessous des anciennes**.**

**Cela rendrait l'affichage** illisible et encombré **très rapidement.**

`cls` **: permet donc d'avoir un** affichage propre et clair à chaque actualisation**, en remplaçant l'ancienne sortie par la nouvelle.**

**Cela donne l'impression d'un tableau de bord mis à jour en temps réel.**

**cls sert à nettoyer la console avant d'afficher les nouvelles infos, ce qui rend le monitoring facile à lire et professionnel.**

4. Que vont afficher %DATE% et %TIME% ? Que se passerait si on supprimait cette ligne ?

5. À quoi sert tasklist /FO TABLE | findstr /I "chrome firefox" ?

`tasklist /FO TABLE`
**Liste tous les processus en cours d'exécution sur le système,**
**avec la sortie formatée sous forme de** tableau lisible**.**

`|` (pipe)
**Envoie la sortie de `tasklist` comme entrée de la commande suivante.**

`findstr /I "chrome firefox"`
**Filtre la sortie pour ne garder que les lignes contenant** "chrome" **ou** "firefox"**,**
**grâce à `/I` qui rend la recherche** insensible à la casse**.**

**Cette commande affiche uniquement les processus actifs qui correspondent aux noms "chrome" ou "firefox", par exemple les navigateurs Google Chrome et Mozilla Firefox.**

**Cela permet de surveiller si ces applications sont ouvertes et éventuellement connaître leur état (PID, utilisation mémoire, etc.).**

   - Que se passerait si on remplaçait "chrome firefox" par "notepad" ?

     **Afficherait les processus notepad en cours d'éxécution.**

6. Que fait la commande netstat -an | find ":80" ? Que signifie le :80 ?

`netstat -an`
**Affiche toutes les connexions réseau** actives **et les** ports d'écoute**,**
**avec les adresses et ports en** format numérique **(`-n`),**
**et sans tenter de résoudre les noms d'hôtes (`-a` montre toutes les connexions et ports).**

`|` (pipe)
**Envoie la sortie de `netstat` à la commande suivante.**

`find ":80"`
**Filtre cette sortie pour ne garder que les lignes contenant `:80`.**

7. Quel est le rôle de timeout /t 10 /nobreak ? Que se passerait si on mettait /t 0 ?

la commande `timeout /t 10 /nobreak` met en pause l'exécution du script ou de la commande pendant 10 secondes et empêche l'utilisateur de l'interrompre en appuyant sur une touche.

-Le programme continue **immédiatement**.

-Aucun effet de pause.

-Aucun message ne s'affiche à l'écran (contrairement à /t avec une valeur > 0).

8. Pourquoi le script utilise-t-il une boucle infinie ? Quels sont les avantages et inconvénients d'un tel fonctionnement ?

**Avantages d'une boucle infinie dans ce contexte**

1. **Monitoring en temps réel**
   Le script fournit une mise à jour automatique et constante, ce qui est utile pour surveiller l'état du système en quasi temps réel.
2. **Automatisation simple**
   Pas besoin d'interventions manuelles pour relancer la commande, ce qui simplifie le suivi.
3. **Contrôle facile de la fréquence**
   Grâce à timeout /t 10 /nobreak, on contrôle la fréquence d'actualisation (toutes les 10 secondes ici).
4. **Ressources relativement faibles**
   Le script utilise peu de ressources et ne surcharge pas le système (10 secondes entre chaque boucle).

**Inconvénients d'une boucle infinie**

1. **Risque d'exécution illimitée**
   Le script ne s'arrête jamais seul, ce qui peut poser problème s'il tourne trop longtemps sans surveillance.
2. **Utilisation de ressources sur le long terme**
   Même s'il est léger, le script consomme un peu de CPU et mémoire en permanence, ce qui peut s'accumuler.
3. **Pas de gestion d'erreurs ni de sortie propre**
   En cas de problème (ex : une commande échoue), le script continue sans rien signaler.
4. **Interruption manuelle nécessaire**
   Il faut un CTRL+C ou fermer la fenêtre pour arrêter le script.
5. **Pas de journalisation intégrée**
   Les infos sont affichées à l'écran mais ne sont pas sauvegardées automatiquement.

**Améliorations**

1. Comment pourriez-vous modifier le script pour surveiller **plusieurs ports réseau** en même temps, par exemple 80 et 443 ?

Le find ou findstr accepte une expression régulière simple. Pour chercher plusieurs ports dans netstat, tu peux utiliser :

```
netstat -an | findstr /R /C:":80" /C:":443"
```

**Explication :**

- /R active l'usage d'expressions régulières.
- /C:"texte" indique la chaîne à chercher (ici ":80" et ":443").

- Le pipe affiche les connexions sur les ports 80 et 443.

2. Comment pourriez-vous faire en sorte que le script **enregistre l'historique** des processus et connexions dans un fichier texte pour consultation ultérieure ?

Pour sauvegarder les sorties de commandes dans un fichier et garder un historique

Exemple à placer dans la boucle :

```
3. (
4. echo === MONITORING SYSTEME ===
5. echo Date: %DATE% - Heure: %TIME%
6. echo.
7. echo === PROCESSUS ===
8. tasklist /FO TABLE | findstr /I "chrome firefox"
9. echo.
10.echo === CONNEXIONS RESEAU ===
11.netstat -an | findstr /R /C:":80" /C:":443"
12.echo.
13.echo ----------------------------------------------
14.) >> monitoring_log.txt
```

15. Chaque cycle ajoute un bloc dans `monitoring_log.txt`.

3.Comment pourriez-vous modifier le script pour qu'il **s'arrête automatiquement** après un certain nombre de cycles ou après un certain temps ?

```
@echo off
setlocal enabledelayedexpansion

rem Initialise la variable compteur à 0
set count=0

rem Définit le nombre maximum de cycles avant arrêt
set max_cycles=5
```

```
:loop
cls

rem Incrémente le compteur de 1 à chaque passage dans la boucle
set /a count+=1

rem Affiche le numéro du cycle actuel sur le total
echo Cycle !count! sur %max_cycles%

echo === MONITORING SYSTEME ===
echo Date: %DATE% - Heure: %TIME%
echo.

echo === PROCESSUS ===
tasklist /FO TABLE | findstr /I "chrome firefox"
echo.

echo === CONNEXIONS RESEAU ===
netstat -an | findstr /R /C:":80" /C:":443"
echo.

rem Pause de 10 secondes avant la prochaine itération, sans possibilité d'interruption
timeout /t 10 /nobreak

rem Condition : si le compteur atteint ou dépasse le maximum, quitter la boucle
if !count! GEQ %max_cycles% goto end

rem Sinon, retourner au début de la boucle
goto loop

:end
echo Fin du monitoring après %max_cycles% cycles.
pause
```

4.Que pourriez-vous faire pour rendre l'affichage **plus lisible ou esthétique**, par exemple avec des couleurs différentes pour les sections ?

```
@echo off
cls
rem Couleur verte pour le titre
color 0A
echo === MONITORING SYSTEME ===

rem Revenir à blanc pour la suite
color 07
echo Date: %DATE% - Heure: %TIME%
```

**Conclusion Programme :**

**Explications détaillées :**

- **@echo off : désactive l'affichage des commandes dans la console pour rendre la sortie plus propre.**
- **setlocal enabledelayedexpansion : active l'expansion différée des variables, nécessaire pour utiliser !count! à l'intérieur de la boucle.**
- **set count=0 : initialise la variable count qui va compter les cycles.**
- **set max_cycles=5 : fixe le nombre maximal d'itérations avant arrêt.**
- **:loop : label de début de la boucle.**
- **cls : efface l'écran à chaque début de cycle pour un affichage propre.**
- **set /a count+=1 : incrémente count de 1.**
- **echo Cycle !count! sur %max_cycles% : affiche à l'utilisateur le numéro du cycle en cours.**
- **Ensuite les commandes d'affichage des infos système : (tasklist, netstat).**
- **timeout /t 10 /nobreak : pause de 10 secondes sans interruption possible.**
- **if !count! GEQ %max_cycles% goto end : si le compteur est supérieur ou égal au max, on sort de la boucle.**
- **goto loop : sinon on recommence la boucle.**
- **:end : label de fin.**
- **echo Fin du monitoring... : message final.**
- **pause : attend que l'utilisateur appuie sur une touche avant de fermer la fenêtre.**
- **/I : dans ce contexte permet d'ignorer la casse (majuscules/minuscules) lors de la recherche. -à  find /I "bonjour" fichier.txt**