Gestion des packages Administration d'un serveur web

Lancement des services

- /etc/init.d/xxxxxx start/restart/reload/stop/etc.
- service xxxxxx start/restart/reload/stop/etc

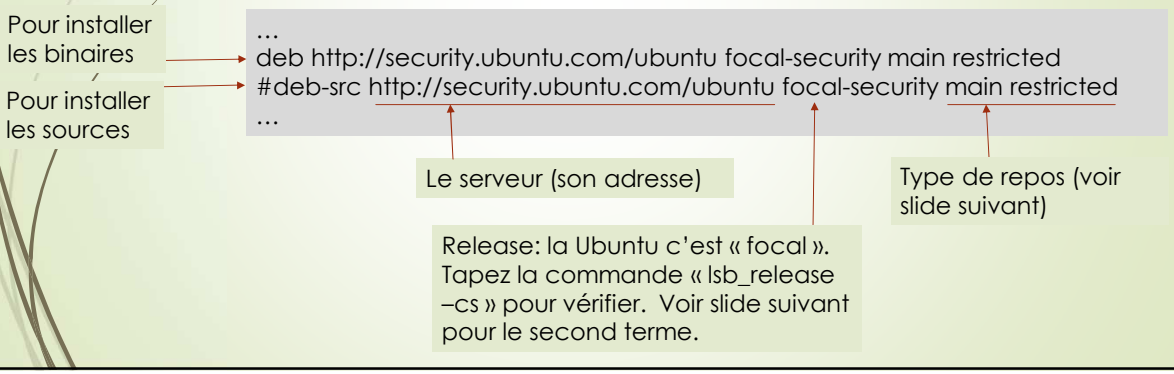
- Scripts se trouvant dans le dossier /etc/init.d
- Ils servent à gérer proprement l'exécution des services
- Les services doivent être gérés via ces scripts
  - Variable d'environnement qui peuvent être fixées au démarrage
  - Fermeture/écriture des fichiers ou autres lors de l'arrêt du service
  - Options passées en paramètres
  - Etc.

Gestion des packages

- Debian/Ubuntu: dpkg / apt-get (fichiers .deb)
- RedHat/Centos: dnf / yum (.rpm)

Repository list

- List des « repository »
  - Serveur stockant les packages
- Se trouve dans /etc/apt-get/sources.list



Repos

- Repos officiels
  - main: logiciels maintenu (par Canonical), et open-source. Contient les packages qui viennent avec l'installation.
  - restricted: logiciel non open source (closed source) et propriétaires. Contient les pilotes par exemple (nvidia, etc.).
  - universe: logiciels open-source maintenu par la communauté. Pas de support de la part de « Canonical ».
  - multiverse: logiciels closed-source, ayant des copiryght, etc.
- Il existe aussi des repos non officiels.
  - Qui peuvent être ajouté dans sources.list par exemple

Type de mis à jour

- Sans type: les logiciels courants.
- Les repos contenant les mis à jour:
  - Update: contient les logiciels qui devraient être mis à jour
  - Security: logiciels qui devraient mis à jour pour des raisons de sécurité
  - Backports: logiciels stables fournit à des release Ubuntu plus ancienne
  - Proposed: logiciels non encore stable (ne devraient pas faire partie de votre liste).
  - Il peut y en avoir d'autres.
  - Comparaison des logiciels installés et ceux de ces repos (par votre gestionnaire de packages).

Les commandes

- apt-get update: met à jour la liste des repos.
- apt-get upgrade: met à jour les logiciels déjà installé
- apt-get dist-upgrade: met à jour la distribution Linux
- apt-get install xxxx: installe le logiciel xxxxx
- apt-get remove xxxx: désinstalle le logiciel xxxx
  - L'option –purge supprime aussi les fichiers de config.
- apt-cache search gvim : cherche dans la description des packages disponibles ceux ou le mot clé gvim est indiqué (il eut y avoir plusieurs mots clés).
- apt-cache show gvim: affiche les propriétés du package gvim (si il existe).
- apt list –installed: affiche la liste des packages installés.

Qu'est ce qu'un package

- Un package est une archive qui contient:
  - Une arborescence de dossiers (identiques à celles du système) indiquant les différentes destinations des fichiers.
  - Des fichiers binaires, des fichiers de configurations, un man, etc.
  - Un fichier « control » qui contient les informations sur le package:
    - Description
    - L'architecture (i386 par exemple)
    - Les dépendances
    - Les conflits: ne peut pas être installé avec tel logiciel
    - Replaces: le logiciel qu'il remplace
  - Un fichier permettant de vérifier l'intégrité des fichiers (optionnel)
  - Un copyright (optionnel)

Serveurs web

Serveur apache

- Apache est un serveur web open-source
- 40% des serveurs web dans le monde[1]
- Pour rappel:
  - Transfert de fichier régit par le protocole http
  - Code pouvant être exécuté côté serveur ou côté client

Ouverture de connexion
TCP sur le port 80

Requête http:
get http1.1 /path/file.html

Réponse http:
http 1.1 200 OK
+ le fichier

Serveur apache

L'en-tête de réponse peut contenir beaucoup d'autres informations (cache, cookie, etc.).

[1] https://www.hostinger.com/tutorials/what-is-apache

Lancez le serveur

- Sous Ubuntu
  - /etc/init.d/apache2 start/stop/restart/reload/fullstatus/graceful/config-test/etc.
  - service apache2 start/etc.
  - config-test: vérifie la syntaxe des fichiers de configuration
  - graceful et graceful-stop: redémarre et arrête apache mais en ne coupant pas les connexions en cours
- Doit être lancé à partir des scripts
  - À cause des variables d'environnement entre autres

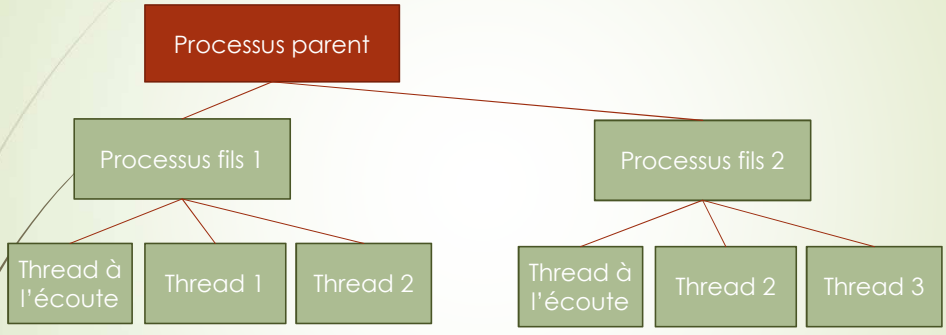
Les fichiers de configuration

- Peuvent être légèrement différent d'une distribution à l'autre (Ubuntu ici)
  - /etc/apache2.conf (conf générale)
    - Format des logs par exemple
  - /etc/ports.conf: ports à l'écoute (défaut : 80)
  - Dossiers mods-available: configuration des modules
    - *.load : directive LoadModule vers un module/librairie
    - *.conf: fichier de conf correspondant
  - Dossiers sites-available: vhost (voir plus loin)
  - Dossiers confs-enables: autres éléments de configuration
- Les modules/vhost/ou les autres éléments de configuration sont activés en plaçant un lien symbolique dans le dossier *-enabled qui pointe dans *-available.

MPM: Multi Processes Module

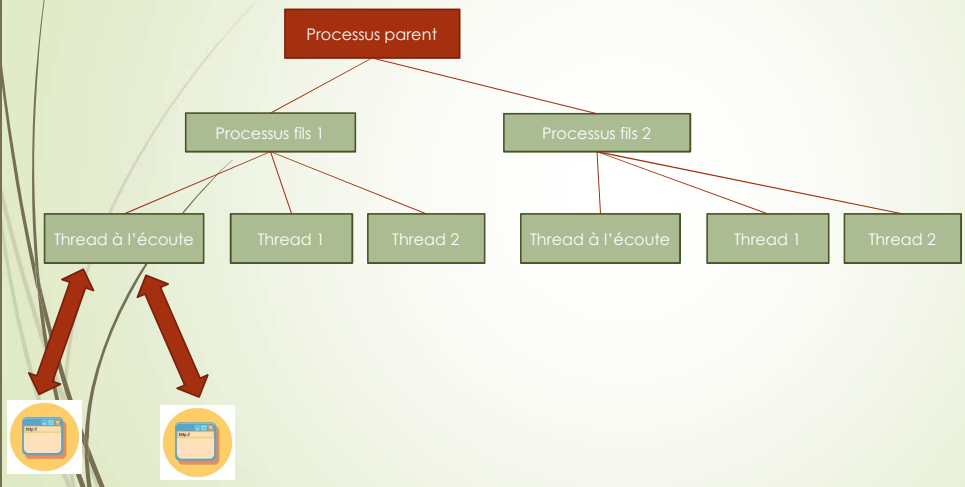
- Plusieurs processus écoutent sur le port 80
  - Traitement des requêtes et des clients en parallèles
  - Gain en efficacité (en particulier dans un environnement multi processeurs/cœurs
  - Les connexions sur le port 80 sont réparties
- Les variantes (suivant le système et la configuration apache)
  - MPM Pre-fork
    - Un processus fils est créer pour chaque nouvelle connexion
  - MPM Worker
    - Ancienne version de MPM event
    - Un thread pouvait rester assigné une connexion TCP persistante
  - MPM Event: multi processus et multi thread
    - Un processus parent gère un ensemble de fils (fork)
    - Il y a « ThreadsPerChild » threads pour chaque processus fils
    - Traitement des requêtes très parallélisé
      - Un thread par fils est à l'écoute de la socket
      - Une connexion entrante est associée à un autre thread (du même fils bien sûr)
      - Les connexions TCP persistantes peuvent être gérées par des threads différents pour optimiser les ressources (minimiser les temps d'attentes)

MPM event



Beaucoup plus de threads en pratique. Le thread à l'écoute attend les connexions entrantes. Il les donne aux autres threads pour traitement des requêtes.

MPM event



VHOST

- Un serveur web peut hebergé plusieurs site web
  - www.site1.com ; www.site2.fr ; etc.
  - Directive HOST obligatoire dans les champs de la requête http 1.1
- Configuration pouvant être différente au travers des vhost apache
  - Serveur virtuel /Virtual Host qui héberge un des sites
  - En fonction du host (son nom)
  - En fonction de l'adresse IP (si correspondance entre site et adresse IP dans le DNS)
  - Un fichier de configuration par vhost dans sites-available
  - Un lien dans sites-enable pour les vhost actifs
  - Voir exercice du TD pour les détails.

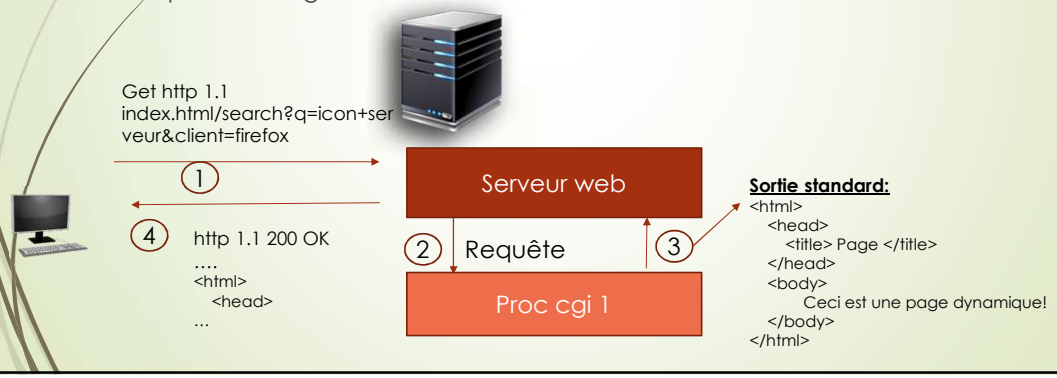
CGI: common gateway interface

- CGI: interface générique qui traite du contenu dynamique
  - A chaque requête http nécessitant l'exécution du programme un processus est lancé (binaire C/C++, perl, java, php, etc.)
  - Le serveur web récupère la sortie standard du processus et l'envoie au client (une page web typiquement)
  - Problèmes de performance:
    - Ceci impose la création des processus pour chaque requête
    - Autant de processus CGI en cours d'exécution que de requête web en cours

Requête http Sollicitant du contenu dynamique

CGI (2)

- La requête est passé au travers de variables d'environnement
  - PATH_INFO: contient le chemin du ficihier (http://www.sie.com/**chemin/file.php**?...)
  - Si GET: la variable REQUEST contient la requête
  - Si POST: la requête est passé sur l'entrée standard
- Le processus cgi renvoie le contenu sur sa sortie standard



Fast-cgi

- Un ou des processus actifs gèrent les requêtes dynamiques
  - Pas de création de processus pour chaque requête
- Nombre de processus plus ou moins indépendants du nombre de requêtes
- Les requêtes et des variables d'environnement sont envoyées par (au choix):
  - Tube nommé
  - Connexion TCP
  - Connexion Unix
- Intérêt:
  - Performance: pas besoin de créer un processus à chaque fois (lourd)
  - Gestion des services séparés du serveur web (plus flexible)
  - Implémentation non dépendantes du serveur web (on choisit ce que l'on veut)
  - Sécurité qui peut être propre.

Module php dans apache

- Deux approches
- Module apache2 (intégré à apache)
- Processus indépendant d'apache
  - Plus performant (code/processus indépendant d'apache)
  - Plus sécurisé (droits et configuration différente d'apache)
  - Peut être utilisé par d'autres serveur web
  - Exemple: php-fpm

Nginx (engine ex)

- Créer en 2004 (opensource)
- Concurrent de apache
  - Apache retse le plus populaire (de l'ordre de 50% des sites)
  - Les sites les plus performants/populaires utilisent Nginx
- Même principe que apache MPM-event:
  - Plusieurs processus / plusieurs threads
  - Distribution des requêtes http cliente sur les threads par un processus maître
  - Une même connexion persistante peut être servi par plusieurs threads
- Moins de module que apache allégeant le serveur
- Pas de module php par exemple

Nginx: fonctionnalités et configuration

- Serveur web
- Reverse proxy
  - Reçoit les requêtes http des clients
  - Les redistribue sur d'autres serveurs
  - Partage de charge
- Connexion sécurisé (https)
- Streaming vidéo

- Configuration:
  - Même principe que apache: dossier *-enable et *-available