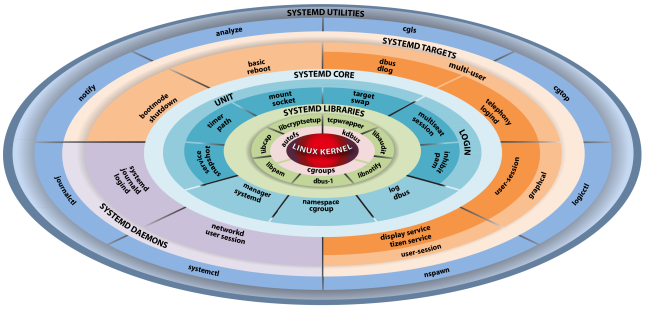
UNIVERSITÉ CAEN NORMANDIE

# Les Services et Systemd, upstart et sysvinit

20.01.2023

## systemd : l'init martyrisé, l'init bafoué, mais l'init libéré !.☺

Auteur : Pascal Fougeray




**Source** : https://linoxide.com/linux-how-to/linux-systemd/

Nous avons vu dans le chapitre précédent les ports et les services.

**Je vous propose dans ce cours de voir comment on lance, on arrête et on configure un service**

# 1 Préambule

**Systemd** je ne suis ni pour ni contre bien au contraire, mais il fait que le cours n'est pas toujours bon ☹
Non non je ne saborde pas mon CM juste que je suis un peu inquiet pour les années à venir ?
Va-t'on revenir en arrière... ?
Le monde de Linux ne tourne pas rond ou plutôt ça tourne en rond

| HIER | AUJOURD'HUI | DEMAIN |
|---|---|---|
| Pas Systemd | Systemd | Plus Systemd ? |

Bon on est aujourd'hui, je continue ☺

Avant il y avait ***sysinit***, ça marchait plutôt bien... du moins j'aimais bien... puis il y a eu ***upstart,*** surtout utilisé par **Ubuntu** jusqu'à la version 16.04 et... https://doc.ubuntu-fr.org/systemd qui écrit :
***Systemd*** *est le gestionnaire de système qui remplace* ***upstart*** *et son prédécesseur*
**Debian** l'a adopté depuis la version 8 (***jessie***) ce qui n'a pas plu à tous les développeurs de la communauté **Debian** et ils ont créé une nouvelle distribution nommée **Devuan** [1] https://www.devuan.org/
*So,* ***upstart in 2006*** *and* ***systemd in 2010*** *were proposed to replace the existing and widely used init system.*
*Both the systems had their own supporters, and after a* ***long conflict****, systemd was chosen as the new system to replace init.*

Donc il ne reste donc plus que ***systemd*** [2] que nous allons étudier plus en détails dans ce cours

Voici ce que **Linus Torvalds** disait au sujet de **systemd** "***I don't actually have any particularly strong opinions on systemd itself. I've had issues with some of the core developers that I think are much too cavalier about bugs and compatibility, and I think some of the design details are insane (I dislike the binary logs, for example), but those are details, not big issues.***"
Voici ce que ***Google translate*** en pense " Je n'ai pas d'opinion particulièrement forte sur ***systemd*** lui-même. J'ai eu des problèmes avec certains des développeurs de base qui sont trop cavaliers sur les bogues et la compatibilité, et je pense

1. ***Debian Veteran Unix Admins****, le premier qui dit que le prof est un vétéran prend ma ... sur la...*
2. Quand je vous dis qu'on est "démerde" même Linux l'est devenu avec son "**système D**". j'ai toujours été un grand visionnaire ☺

que certains détails de conception sont fous (je n'aime pas les logs binaires, par exemple), mais ce sont des détails, pas des gros problèmes."

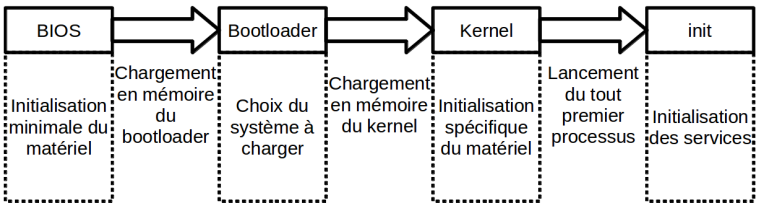
Voici ce que j'en pense " *je ne suis ni pour ni contre bien au contraire... j'ai juste dû m'y mettre... et écrire ce cours* " allez plus sérieux... beaucoup de "professionnels" du domaine disent que cela permet d'aller plus vite... je ne suis pas persuadé... mon portable avec un core-i7 et 32G de ram démarre moins vite que mon core2duo et ses 4G de RAM... pourquoi et bien parce que l'un à une interface graphique et l'autre pas, parce que l'un est un ordinateur avec un SE à la Fenêtre et l'autre est un ordinateur serveur en CLI.

Voilà c'est écrit ! Bien passons à la suite.

— http://linuxfr.org/news/évolutions-techniques-de-systemd
— https://linuxfr.org/users/ghusson/journaux/mes-notes-de-synthese-sur-systemd
— https://linuxfr.org/news/systemd-l-init-martyrise-l-init-bafoue-mais-l-init-libere

# 2 Le démarrage d'un ordinateur

En une image voici comment démarre un ordinateur




source : http://www.linuxembedded.fr/2014/09/maitriser-les-services-gnulinux-a-laide-de-systemd/

En plus avec systemd , **Init** n'est plus le processus père des processus Orphelins des utilisateurs alors que je l'écris dans mon cours de L2 et c'est ce que j'ai appris au siècle dernier ... ☹

# 3 *Systemd*

## 3.1 Introduction

**systemd** est LE ? remplaçant du démon ***init system V*** pour *Linux*.

Son site : https://systemd-free.org/index.php

Il est écrit en C normal pour le noyau et en ... python ☺

**systemd** est le premier programme lancé par le noyau (Pas toujours vrai...), il a donc le PID 1 et il se charge de lancer tous les programmes suivants en ordre jusqu'à obtenir un système opérationnel pour l'utilisateur

Il y a 3 modes principaux ou 3 niveaux d'exécution ou ***runlevel*** de *Gnu/Linux* contrôlant le choix des **processus** et/ou **services** démarrés automatiquement par le système

1. ***single user***, qui correspond à l'ancien ***runlevels* 1**
2. ***multi-user***, qui correspond à l'ancien ***runlevels* 3**
3. ***graphique***.qui correspond à l'ancien ***runlevels* 5**

En réalité il y a 8 niveaux !

| Cibles systemd | runlevels | Fait quoi... |
|---|---|---|
| **poweroff.target - runlevel0.target** | 0 | Arrêt de l'OS |
| **rescue.target - runlevel1.target** | 1, s, single | Mode utilisateur unique, mode maintenance. |
| **multi-user.target - runlevel3.target** | 3 | Mode multi-utilisateur sans GUI |
| **multi-user.target - runlevel3,4.target** | 2, 4 | Modes multi-utilisateur sans GUI |
| **graphical.target - runlevel5.target** | 5 | Mode multi-utilisateur avec GUI |
| **reboot.target - runlevel6.target** | 6 | Redémarrage de l'OS |
| **emergency.target** | emergency | Shell d'urgence, avec système de fichiers "***lié***" en RO. |

C'est également à **systemd** qu'incombe la tâche de **redémarrer** et d'**arrêter** le système proprement.

**systemd** a aussi la **gestion des dépendances entre services**, et de permettre le chargement en **parallèle** des services au démarrage. Cela permet (MDR) de **réduire le temps de démarrage** du système et de lancer moins de processus [3].

Pourquoi le MDR, tout simplement parce qu'on ne redémarre pas un serveur en permanence donc qu'il mettent 3mn33s ou 3mn64s [4] importe peu... ☺

3. Ah bon, des services auraient disparus et donc moins de processus ?
4. ce qui fait je crois 4mn04

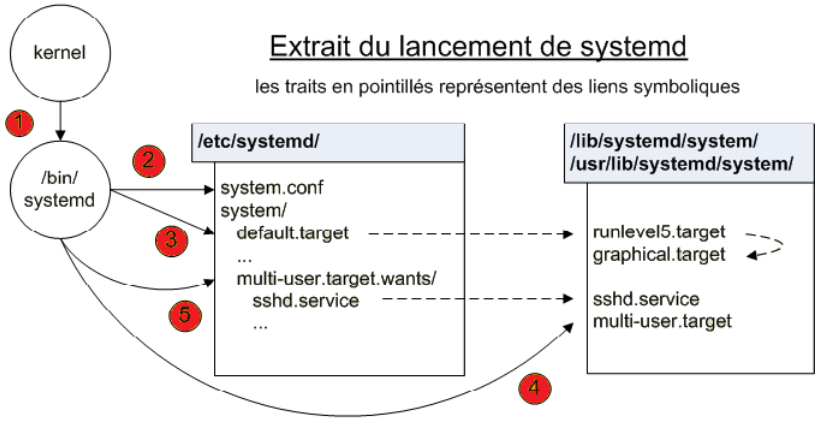
Avant le noyau lançait le processus **INIT** et bien maintenant il lance **systemd** en lui passant comme paramètre **INIT=/bin/systemd** à la place de **/sbin/init**

**Remarque** : on peut passer de l'un à l'autre... à essayer sur une VM, moi je ne m'y risque pas... ! ! !

Voici une belle image récupéré ici

https://connect.ed-diamond.com/GNU-Linux-Magazine/GLMF-153/Systemd-vainqueur-de-Upstart-et-des-scripts-System-V

Excellent article de **linux magazine** datant de 2012 ! ! !

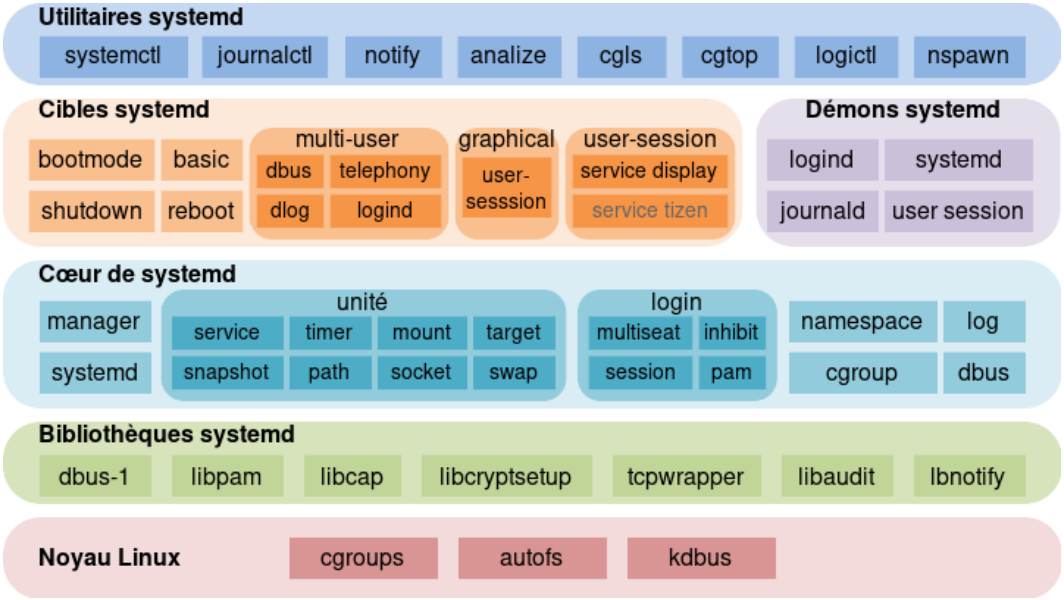



## 3.2 Les entités de systemd

Attention, on ne va pas tout étudier dans ce cours, non non..., pas le temps...

Alors une belle image (encore une...) vaut mieux qu'un grand texte indigeste.

Image prise sur Wikipédia : https://fr.wikipedia.org/wiki/Systemd




Quel beau modèle en couches que vous devez comprendre et que vous utilisez souvent sans le savoir.

La commande ***pstree*** renvoie quelque chose comme

```
systemd─┬─ModemManager─┬─{gdbus}
        │              └─{gmain}
        ├─NetworkManager─┬─dhclient
        │                ├─{gdbus}
        │                └─{gmain}
        ├─systemd─┬─(sd-pam)
        │         ├─dbus-daemon
        │         └─gvfsd─┬─{gdbus}
        │                 └─{gmain}
        ├─systemd───(sd-pam)
        ├─systemd-journal
        ├─systemd-logind
        ├─systemd-timesyn───{sd-resol
        └─systemd-udevd
```

**service** : pour un service système ;

**socket** : pour une socket de communication entre processus ;

**busname** : pour les logiciels de communication inter-processus exemple dbus ;

**target** : macro-unité qui permet de grouper plusieurs unités (exemple : multi-user.target pour définir une cible) ;

**path** : pour l'activation d'un service basée sur la modification de fichiers ou de répertoires ;

**mount** : pour un SGF, exemple : home.mount, et utilise ***/etc/fstab*** ;

**automount** : pour un système de fichiers "lié" à la demande ;

**swap** : pour les partitions de swap ;

**device** : pour un périphérique ;

**timer** : pour l'activation basée sur une date ;

**slice** : la gestion des **cgroups**, fonctionnalité du kernel pour **limiter**, **compter** et **isoler** l'utilisation des ressources ;

**scope** : utilisé par systemd lui-même pour gérer des groupes de processus, typiquement, par session utilisateurs ;

**snapshot** : unités utilisées pour sauvegarder l'état actuel des services et les restaurer ensuite, mode veille.

## 3.3 Les services propres à systemd

**systemd** utilise en interne un certain nombre de services pour la gestion du système, exemples :

**systemd-journald** : les **logs** du système et des programmes ;

**systemd-logind** : la connexion des utilisateurs ;

**systemd-vconsole-setup** : la configuration des consoles virtuelles ;

**systemd-remount-api-vfs** : le "**reliage**" du SGF, en tenant compte des options de ***/etc/fstab*** ;

**systemd-sysctl** : L'application des paramètres système de ***/etc/sysctl.conf*** ;

**systemd-stdout-syslog-bridge** : utilisation de **syslog** pour les processus de systemd ;

**systemd-tmpfiles-setup** : Pour la création de fichiers et répertoires temporaires ;

## 3.4 Configurer systemd

Comme tout, ***systemd*** est à configurer et ses fichiers et répertoires de configuration se trouvent ici ***/etc/systemd/***
Il contient 6 fichiers et 3 répertoires

1. ***journald.conf,***
2. ***logind.conf,***
3. ***resolved.conf,***
4. ***system.conf,***
5. ***timesyncd.conf*** ,
6. ***user.conf.***

1. ***network,***
2. ***system,***
3. ***user.***

**Expliquer en détails est trop long...**

Les paramètres principaux de *systemd* sont dans le fichier ***/etc/systemd/system.conf.***
Parmi ceux-ci, il y a **LogTarget= journal**.
Cela indique à ***systemd*** d'envoyer tous les messages de log à un nouveau démon nommé ***systemd-journald*** lui-même activé par ***systemd*** car décrit par l'unité ***systemd-journald*.service**.

Nous verrons cela plus en détails lors du cours sur les journaux, les **Logs**

### 3.5 Configurer ses services

La commande ***systemctl*** permet de configurer les services qui sont lancés au démarrage.
Voici quelques exemples de commandes à connaître !

| Quoi | Commande |
|---|---|
| **Activer** un service au démarrage | systemctl **enable** nom_du_service.service |
| # ***systemctl enable sshd.service*** | |
| ***ln -s '/usr/lib/systemd/system/sshd.service' '/etc/systemd/system/multi-user.target.wants/sshd.service'*** | |
| **Désactiver** un service au démarrage | systemctl **disable** nom_du_service.service |
| Statut d'un service (actif ou pas) | systemctl **is-active** nom_du_service.service |
| Statut **détaillé** d'un service | systemctl **status** nom_du_service.service |
| **Lister** tous les services actifs | systemctl **list-units --type=service** ou systemctl **list-units -a** |
| **Démarrer** un service | systemctl **start** nom_du_service.service |
| **Arrêter** un service | systemctl **stop** nom_du_service.service |
| **Redémarrer** un service | systemctl **restart** nom_du_service.service |
| **Recharger** la configuration un service | systemctl **reload** nom_du_service.service |
| ! ! ! **sans interrompre le service** ! ! ! | |
| **Gérer** les niveaux d'exécution (*runlevels*) | 2 niveaux mutli-user console et GUI |
| **Connaitre** son ***runlevel*** | systemctl **get-default** |
| **Basculer** temporairement de *runlevel* | systemctl **isolate** ***multi-user.target*** ou ***graphical.target*** |
| **Changer** de *runlevel* par défaut | systemctl **set-default** ***multi-user.target*** ou ***graphical.target*** |

## 4 Gestion des processus avec *systemd*

Par défaut, ***systemd*** maintient automatiquement la relation entre PID et processus activés.
Ainsi, le PID d'un démon tel **sshd** est stocké par systemd dans le fichier **/var/run/sshd.pid**.
root@debian /etc/systemd #> ***ps aux | grep $(cat /var/run/sshd.pid)*** renvoie la ligne correspondant à *sshd*
*root **495** 0.0 0.0 69944 5724 ? Ss 13 :06 0 :00 /usr/sbin/sshd -D*
Mais pour les services dont la valeur de Type= est ***forked*** (alors que le type est simple pour sshd.service), *systemd* peut se tromper dans son analyse du processus principal. En effet, quand un service est de type ***forked***, cela signifie que le processus lancé par **systemd** va lancer un ou plusieurs processus fils puis va se terminer.
S'il a lancé plusieurs processus fils, le stockage de PID du processus fils « principal » est à la charge du processus parent et il faut indiquer à *systemd* dans quel fichier est stocké ce PID. On fait pour cela un usage de la directive PIDFile= dans le fichier de configuration de l'unité.

## 5 Visualisation des logs avec systemd

Nous n'avons pas encore fait le cours sur les LOGS mais juste un extrait...
***systemd*** possède son propre système de logs au format binaire qui est ***journald***[5]. Binaire veut dire que le commun des mortels ne peut le lire...
Vous pouvez toujours essayer en faisant un ***cat /run/log/journal/xxxx/system.journal*** ☺
Ces logs sont stockés non pas dans le répertoire **/var/log** mais dans le répertoire ***/run/log/journal/xxx***
Le ***xxx*** représente 1 **boot** c'est à dire qu'il est supprimé à chaque **boot**, remarque on peut le modifier mais cela dépasse le cadre du cours ...
Ce processus est configuré par le fichier ***/etc/systemd/systemd-journald.conf***

Pour le consulter, rien de plus simple, la commande ***journalctl***. avec ses nombreuses options passées en paramètre apportant ainsi beaucoup de puissance...

| | |
|---|---|
| **journalctl** | fait |
| **-b ou -k** | affiche tout comme la commande **dmesg** |
| **-f (follow)** | affiche les logs en continue ! ! ! |
| **-n valeur** | affiche les x valeurs dernières lignes |
| **-xe** | affiche les 50 dernières TB et souvent utilisés |
| **-r (reverse)** | commence par les derniers |

**Exemple d'utilisation** :

1. *root@debian-11-GNS3 :~#* **journalctl /usr/sbin/sshd**
2. *-- Journal begins at Tue 2023-01-17 17 :47 :29 CET, ends at Wed 2023-01-18 09 :30 :45 CET. --*

5. J'adore quand les anglo-saxons nous empreinte des mots

3. *janv. 17 17 :48 :04 debian-11-GNS3 sshd[715] : Server listening on 0.0.0.0 port 22.*
4. *janv. 17 17 :48 :04 debian-11-GNS3 sshd[715] : Server listening on : : port 22.*
5. *janv. 17 17 :53 :55 debian-11-GNS3 sshd[1703] : Accepted password for etudiant from 192.168.56.1 port 40378 ssh2*
6. ...
7. **journalctl** permet de filtrer par le niveau de log, comme défini par syslog. Pour n'afficher que les erreurs :
   (a) de tous les services : root@debian ~ #> ***journalctl -p err***
   (b) du service **sshd** : root@debian ~ #> ***journalctl -p err /usr/sbin/sshd***

# 6 Autres outils de systemd

Avec systemd sont apparus d'autres outils comme vus à l'image du début tels

1. **hostnamectl** qui permet de changer le nom de votre machine et modifier le fichier /etc/hostname
   root@debian ~ #> ***hostnamectl set-hostname debian***
   root@debian ~ #> ***hostnamectl status***
   ***Static hostname :*** ***debian-11-GNS3***
   ***Icon name :*** ***computer-vm***
   ***Chassis :*** ***vm***
   ***Machine ID :*** ***4630e43e4e35449e83b14f140d40c127***
   ***Boot ID :*** ***9482fc7eb0654d0ea09866044bf75625***
   ***Virtualization :*** ***oracle***
   ***Operating System :*** ***Debian GNU/Linux 11 (bullseye)***
   ***Kernel :*** ***Linux 6.0.0-6-amd64***
   ***Architecture :*** ***x86-64***
2. **loginctl** permet de connaître qui est logué, il appartient à ***systemd-logind*** et plus d'informations ici https://wiki.archlinux.fr/Systemd/logind
3. **systemd-sysctl** permet de configurer les paramètres du noyau au boot... à manipuler avec une grande rigueur, car si le noyau plante... et bien nous sommes mal...
4. **bootctl** status : Contrôler les paramètres de démarrage du firmware EFI et gère le chargeur de démarrage, Grub !

# 7 La pratique sous Linux

Sur Linux, il existe 3 méthodes pour **démarrer**, **arrêter**, **redémarrer** un service ... ☹
Et oui c'est un beau "bordel"...
**ATTENTION : Il faut être root ! ! !**

1. La commande **service**,
2. La commande **systemctl**,
3. Les scripts dans le répertoire **/etc/init.d**

**Question** : Laquelle est la meilleure ?
**Réponse** : alors là, aucune idée.
J'ai testé les 3 sur la VM...
Pour chaque commande, il y a 5 options principales du moins ces 5 là suffisent dans ce cours ☺ :

1. **stop** arrête le service
2. **start** démarre le service
3. **restart** le stop et redémarre
4. **reload** ne fait que demander au service de lire son fichier de configuration et donc de modifier son état !
5. **status** Permet de voir le statut d'un service, c'est à dire s'il fonctionne bien ou pas, sur quelle interface il rend le service qu'on attende de lui etc ...

Voici un exemple avec le **serveur Web Apache2,** que nous n'avons pas encore installé mais ça va venir dans le **TP Intitulé LAMP**

1. root@debian-11-GNS3 :~# ***service apache2 stop***
   renvoie rien ...

2. root@debian-11-GNS3 :~# ***systemctl start apache2.service***
   renvoie rien ...
3. root@debian-11-GNS3 :~# ***/etc/init.d/apache2 stop***
   renvoie : Stopping apache2 (via systemctl) : apache2.service.
4. root@debian-11-GNS3 :~# ***/etc/init.d/apache2 start***
   renvoie : start Starting apache2 (via systemctl) : apache2.service.
5. root@debian-11-GNS3 :~# ***service apache2 status***
   renvoie :

```
root@debian-11-GNS3:/etc/init.d# service apache2 status
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
     Active: inactive (dead) since Wed 2023-01-18 12:58:39 CET; 5h 1min ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 22883 (code=exited, status=0/SUCCESS)
        CPU: 585ms

janv. 18 11:42:07 debian-11-GNS3 systemd[1]: Starting The Apache HTTP Server...
janv. 18 11:42:07 debian-11-GNS3 apachectl[22882]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set the 'ServerName'
janv. 18 11:42:07 debian-11-GNS3 systemd[1]: Started The Apache HTTP Server.
janv. 18 12:58:39 debian-11-GNS3 systemd[1]: Stopping The Apache HTTP Server...
janv. 18 12:58:39 debian-11-GNS3 apachectl[23519]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set the 'ServerName'
janv. 18 12:58:39 debian-11-GNS3 systemd[1]: apache2.service: Succeeded.
janv. 18 12:58:39 debian-11-GNS3 systemd[1]: Stopped The Apache HTTP Server.
root@debian-11-GNS3:/etc/init.d# ^C
root@debian-11-GNS3:/etc/init.d#
```

## 7.1 Lister et afficher le statut des services

Avant de modifier le statut d'un service, il faut être capable d'en afficher le statut.
C'est à dire la liste des services et leurs états, s'ils sont **démarrés** ou **arrêtés**.
Pour lister et afficher les services avec leur statut c'est la commande : ***service --status-all***

root@debian-11-GNS3 :~# ***service --status-all***

[ + ] apache-htcacheclean
**[ + ] apache2**
[ + ] apparmor
[ + ] avahi-daemon
[ - ] cgroupfs-mount
[ - ] console-setup.sh
[ + ] dbus
[ + ] docker
**[ - ] dovecot**
[ + ] hddtemp
[ - ] hwclock.sh
[ - ] keyboard-setup.sh
[ + ] kmod
[ + ] lightdm
[ + ] lm-sensors
[ - ] lvm2
[ - ] lvm2-lvmpolld
**[ + ] networking**
[ + ] openbsd-inetd
[ - ] plymouth
[ + ] plymouth-log
**[ - ] popa3d**
**[ - ] postfix**
[ + ] procps
[ + ] rsyslog
[ + ] saned
[ - ] speech-dispatcher
**[ + ] ssh**
[ - ] sudo
[ + ] udev
[ - ] x11-common

root@debian-11-GNS3 :~# ***service --status-all*** après un ***service apache2 stop***

[ + ] apache-htcacheclean
**[ - ] apache2**
[ + ] apparmor
[ + ] avahi-daemon
[ - ] cgroupfs-mount
[ - ] console-setup.sh
[ + ] dbus
[ + ] docker
**[ - ] dovecot**
[ + ] hddtemp
[ - ] hwclock.sh
[ - ] keyboard-setup.sh
[ + ] kmod
[ + ] lightdm
[ + ] lm-sensors
[ - ] lvm2
[ - ] lvm2-lvmpolld
**[ + ] networking**
[ + ] openbsd-inetd
[ - ] plymouth
[ + ] plymouth-log
**[ - ] popa3d**
**[ - ] postfix**
[ + ] procps
[ + ] rsyslog
[ + ] saned
[ - ] speech-dispatcher
**[ + ] ssh**
[ - ] sudo
[ + ] udev
[ - ] x11-common

# 8 Conclusion

**Ce support de cours est bien trop exhaustif et je ne vous demande pas de tout savoir, mais de savoir que ça existe et de vous y référer au TP sur LAMP !**

Plus qu'à s'y mettre à **systemd**... et en TP sur les **logs, DHCP, DNS, et LAMP**... avec ***debian***

| Avant | devient | Après... |
|---|---|---|
| **ifconfig** , **route** | devient | **ip addr** , **ip route** |
| **dev** | devient | **udev** |
| **init** | devient | **systemd** |
| **eth0** | devient | **enp0s3** |
| etc... | deviendra | encore plus... ☺ |

Si **systemd** ne vous convient pas et bien il y a d'autres possibilités comme OpenRC :
https://systemd-free.org/index.php avec **openrc** (**rc** comme *Run Commands* du vieux SE DEX)
...

**Linux c'est libre et donc tout le monde fait ce qu'il veut ou presque ...**

Moi, je fais comme avant...