Les journaux

> Did I mention ? It also travels in time.
>
> Le neuvième docteur

L'état et la configuration du serveur web sont des éléments importants, mais ne sont cependant pas suffisants lors de l'administration au quotidien d'un serveur web. Il est aussi nécessaire de conserver une trace de l'historique de l'activité du serveur, ainsi que de tous les problèmes rencontrés.

Il existe deux types de journaux importants :

— les journaux de requêtes, qui contiennent les informations concernant les requêtes traitées, ainsi que les logiciels clients : le journal d'accès contient ainsi une ligne pour chaque requête traitée par le serveur ;
— le journal d'erreurs, qui trace les anomalies observées pendant le fonctionnement.

Lorsque le comportement de votre serveur web n'est pas celui souhaité, qu'il refuse de démarrer ou qu'il plante inopinément, jetez toujours un œil aux journaux, on y trouvera généralement des éléments expliquant le problème.

N'hésitez pas, lors des exercices, à avoir un `multitail` lancé sur les fichiers journaux pour observer le comportement de votre serveur web et valider vos tests.

1 Apache

1.1 Journal de requêtes

La journalisation des requêtes est assurée par le module `mod_log_config`[1].

La directive fournie par ce module permettant d'activer l'écriture du journal est `TransferLog` suivie par un nom de fichier. Cette directive ne permet pas de choisir un format de journal ou la journalisation conditionnelle. Ce manque de flexibilité fait qu'on lui préfère généralement l'utilisation combinée de `LogFormat` et de `CustomLog`.

Le format par défaut des journaux est le CLF (Common Log Format), défini de la manière suivante : *hôte_distant identd utilisateur [date] "URL demandée" état taille_en_octets*

Détails concernant différents champs :

hôte_distant	Adresse IP du client qui a émis la requête
identd	Nom d'utilisateur dans le cas de l'utilisation du protocole *identd* – inactif habituellement, un tiret est inscrit dans ce cas
utilisateur	Nom de l'utilisateur dans le cas d'une identification HTTP – un tiret sinon
état	code d'état HTTP à 3 chiffres

La directive `LogFormat`[2] permet de définir le format de chaque ligne inscrite dans un journal. La directive se décompose en deux parties : une chaîne décrivant le format, et un nom (optionnel) servant à la désigner (voir la documentation[3] pour la syntaxe de description du format).

```
LogFormat "%h %l %u %t \"%r\" %>s %b" common
```

1. http://httpd.apache.org/docs/2.4/fr/mod/mod_log_config.html
2. À mettre dans la configuration globale ou dans un serveur virtuel
3. https://httpd.apache.org/docs/2.4/fr/mod/mod_log_config.html#formats

La directive `CustomLog`[4] indique le nom et l'emplacement d'un fichier journal, en permettant de spécifier éventuellement un nom de format défini auparavant. Si aucun nom de format n'est indiqué, le format utilisé est le dernier format sans nom défini, ou, à défaut, le CLF.

```
CustomLog /var/log/apache2/logs/access.log common
```

On peut aussi définir un format de log directement avec `CustomLog` (mais ce n'est pas une pratique courante, on utilise généralement un nom de format précédemment défini avec `LogFormat`) :

```
CustomLog /var/log/apache2/logs/access.log "%h %l %u %t \"%r\" %>s %b"
```

Écriture conditionnelle dans un journal Un format conditionnel existe, permettant de consigner une requête en fonction de la présence ou de l'absence d'une variable d'environnement. Ici on souhaite par exemple consigner les évènements concernant les documents flash dans un journal différent (nécessite le module `mod_setenvif`[5]) :

```
SetEnvIf Request_URI \.swf$ flash
CustomLog /var/log/apache2/flash.log common env=flash
CustomLog /var/log/apache2/access.log common env=!flash
```

Pour aller plus loin :

- *Journaux pour les hôtes virtuels* : il est possible de définir un journal pour chaque hôte en redéfinissant la directive `CustomLog` dans le conteneur de l'hôte virtuel. Cette solution amène une multiplication des fichiers de journaux (ce qui peut être pratique : un site = un log, ou ennuyeux, selon ses goûts personnels). La spécification de format `%v` permet d'indiquer le serveur (c'est-à-dire l'hôte virtuel) qui traite la requête, indispensable si l'on souhaite n'utiliser qu'un seul fichier de log.
- *Suivi des sessions utilisateur* : l'adresse IP est de moins en moins suffisante pour identifier un · e utilisateur · ice (pare-feux, proxys, NAT...). Le module `mod_usertrack` attribue un identifiant unique aux utilisateurs au moyen d'un cookie. Il faut utiliser la directive `CookieTracking` positionnée à `on` pour suivre les utilisateur · ices. On pourra alors, dans le format du log de requête, ajouter `%{Apache}n`[6], ce qui permettra de pouvoir suivre un · e utilisateur · ice dans les fichiers journaux.

1.2 Journal d'erreur

Le journal d'erreur a un fonctionnement beaucoup plus simple ques les journaux de requêtes. La directive `ErrorLog`[7] permet de spécifier le fichier à utiliser, ou via le mot clé `syslog` d'utiliser le démon `syslogd` du système. Une directive appelée `LogLevel`[8] permet de faire varier la quantité d'informations consignées. Le niveau `error` est recommandé. D'autres utilisations peuvent nous amener à utiliser, du plus sélectif au plus verbeux : *emerg, alert, crit, error, warn, notice, info, debug.*

De même que `LogFormat` permet de spécifier le format des logs, il est possible de spécifier celui des logs d'erreur avec `ErrorLogFormat`[9].

4. À mettre dans la configuration globale ou dans un serveur virtuel
5. https://httpd.apache.org/docs/current/mod/mod_setenvif.html
6. non, le `n` final n'est pas une faute de frappe
7. À mettre dans la configuration globale ou dans un serveur virtuel
8. À mettre dans la configuration globale, dans un serveur virtuel ou dans un conteneur `Directory`
9. https://httpd.apache.org/docs/2.4/fr/mod/core.html#errorlogformat

2 Nginx

2.1 Journal de requêtes

La directive `log_format` [10] permettra, comme pour Apache, de spécifier un format de log particulier. Elle sera suivie du nom du format de log et de son formatage.

La configuration inclue toujours le format pré-défini `combined` qui contient les informations de l'exemple ci-dessous.

On définit un format de log ainsi :

```
log_format nom_format '$remote_addr - $remote_user [$time_local] '
                      '"$request" $status $body_bytes_sent '
                      '"$http_referer" "$http_user_agent"';
```

Les variables utilisées dans le formatage pourront être les variables communes de Nginx [11] et les variables n'existant qu'au moment de l'écriture d'un log (voir la doc de `log_format`).

On utilisera ensuite `access_log` [12] pour utiliser ce format :

```
access_log /var/log/nginx/access_log nom_format;
```

Le premier argument peut correspondre au nom d'un fichier de log, de l'adresse d'un serveur `syslog` (avec la syntaxe `syslog:server=address[,parameter=value]`) ou juste `off` pour désactiver le logging des requêtes.

À noter qu'il n'est pas nécessaire d'indiquer le nom du format si on se contente du format `combined`.

Écriture conditionnelle dans un journal

On utilisera une `map` [13] pour cela :

```
map $status $loggable {
    ~^[23]  0;
    default 1;
}
...
access_log /path/to/access.log combined if=$loggable;
```

Avec cette `map`, les requêtes avec un statut de réponse 2xx ou 3xx ne seront pas loguées.

Apparté : une `map` Nginx permet de définir une variable dont la valeur dépend de la valeur d'une ou plusieurs variables. Voyez ça comme une instruction `switch` en programmation [14]. Le premier argument d'une map est la ou les variables dont dépend la variable définie par la map (si l'on souhaite utiliser plusieurs variables, on les combinera par exemple comme ceci : `"$status:$request"`), et le deuxième argument est le nom de la variable à définir.

Une `map` ne peut s'écrire que dans le contexte `http`, contrairement à `access_log` que peut s'utiliser dans `http`, `server`, `location`, `if` in `location` et `limit_except`.

10. http://nginx.org/en/docs/http/ngx_http_log_module.html#log_format
11. http://nginx.org/en/docs/varindex.html
12. http://nginx.org/en/docs/http/ngx_http_log_module.html#access_log
13. http://nginx.org/en/docs/http/ngx_http_map_module.html#map
14. https://fr.wikipedia.org/wiki/Switch_(instruction)

2.2 Journal d'erreur

On utilisera la directive `error_log`[15]. Le niveau de log sera passé en argument après la destination du log (fichier, serveur `syslog` ou un tampon mémoire) : *debug, info, notice, warn, error, crit, alert, emerg.* Le niveau de log par défaut est `error`.

```
error_log logs/error.log;
```

15. http://nginx.org/en/docs/ngx_core_module.html#error_log