|  |
| --- |
|  |
| Squid - Proxy |
| Guide du débutant |

|  |
| --- |
| Adil BOUZIT  [Date] |

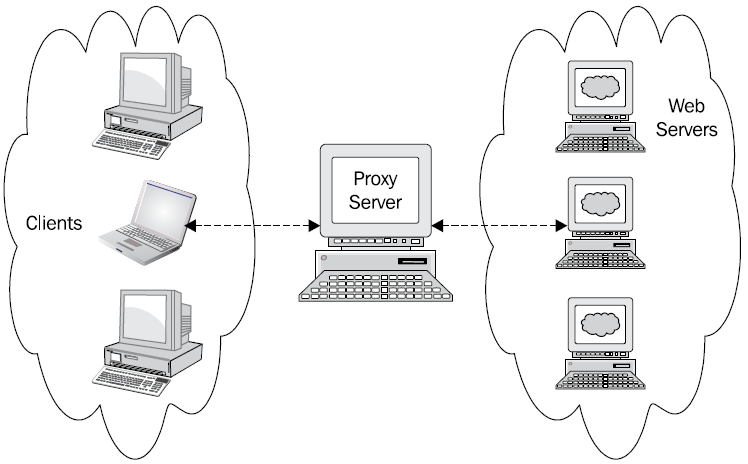
Proxy & Reverse Proxy

Qu’est-ce qu’un proxy ?

Un proxy est un système informatique, un intermédiaire situé entre un client et un serveur. Dans sa forme simple, le proxy facilite la communication entre un client et serveur cible sans modifier les requêtes cliente ou réponses serveur. La mise en place d’un serveur proxy empêche donc le client de communiquer directement avec le serveur cible.

Lorsque l’on initie une requête vers un serveur cible, le serveur proxy détourne la connexion et se présente au serveur cible comme étant lui-même le client, requêtant le serveur en son nom (celui du client), si une réponse est reçue, le serveur proxy nous la retourne, donnant le sentiment d’avoir communiqué directement avec le serveur cible.

Dans sa forme avancée, le serveur proxy peut filtrer les requêtes selon des règles définies permettant ou non la communication au moyen d’une validation. Les règles sont généralement basées selon les adresses ip des client ou d’un serveur, d’un protocole, type de document etc



Parfois, un serveur proxy peut modifier les requêtes ou réponses, voir même, conserver localement les réponses du serveur pour pouvoir les renvoyer à d’autres clients, ce procédé s’appelle la mise en cache ou « caching », c’est une technique populaire utilisé par les proxy pour sauvegarder de la bande passante, participer à la réduction de charge des serveurs et améliorer l’expérience utilisateur. Ils sont communément utilisés pour accomplir les tâches suivantes :

* Réduire la bande passante
* Améliorer l’expérience utilisateur en réduisant le temps de chargement de page
* Renforcer les règles de sécurité
* Monitorer le trafic utilisateur
* Anonymiser un utilisateur sur internet
* Filtre les requêtes ou réponse en utilisant un service de détection anti-virus
* Équilibrer les charges de trafic entre de multiples connexions internet

Qu’est-ce qu’un reverse proxy ?

Le reverse proxy ou « reverse proxying » est une technique de stockage local de réponse ou ressources d’un serveur web, ainsi, les requêtes peuvent être satisfaite depuis la copie locale sur le serveur proxy, parfois même sans avoir à contacter le serveur. Le proxy ou le web cache vérifie si les copies locales sont toujours valides avant de distribuer les copies

La durée de vie des copies stockées est calculée depuis une en-tête http additionnelle reçue par le serveur web. Utilisant les en-têtes HTTP, un serveur web peut contrôler ce qu’il distribue comme document/réponse pouvant être mis ou non en cache par le serveur proxy.

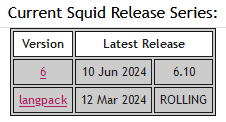
La mise en cache est communément utilisée :

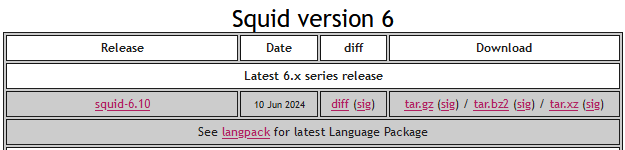
* Réduire l’utilisation de la bande passante : Un large nombre de page web statique comme les fichiers CSS et JavaScript, les images, vidéos etc peuvent être mise en cache du moment qu’ils ne changent pas fréquemment et constituent la majeure partie des réponse d’un serveur web.
* Par les FAI pour réduire le temps moyen de chargement d’une page pour améliorer l’expérience de leurs client sur leurs réseaux commutés ou à bande large.
* Pour réduire la charge de travail d’un serveur web occupé en distribuant les pages statiques etc depuis les contenu copié en cache du serveur proxy.

Squid comme proxy

Le package de binaires de Squid est pré-compilé et prêt à installer le bundle de logiciel. Squid est disponible sous plusieurs formes (Archive, code source, package binaire comme RPM, DEB etc) et téléchargeable depuis le site officiel : <http://www.squid-cache.org/Versions/>

Il est important pour nous d’avoir une version stable avec les dernières fonctionnalités, les versions stables sont généralement les testée avant d’être déployée et supposée fonctionner dans un environnement de production.





On prendra ici le package binaires de la version 6 disponible en tar.gz avec wget :

wget <http://www.squid-cache.org/Versions/v6/squid-6.10.tar.gz>

Installer Squid depuis son code source

L’installation de squid depuis son code source se fait en 3 étapes :

1. Sélectionner les fonctionnalités et les paramètres spécifiques à notre OS
2. Compiler le code source pour générer des fichiers exécutables
3. Placer les fichiers exécutables et autres fichiers dans les destinations désignée par Squid afin qu’il fonctionne correctement

Compiler Squid :

Compiler squid est un processus de compilation de fichiers critiques contenant du C/C++ dans son code source et génère des exécutables. Compiler Squid est simple et peut être fait en quelques étapes. Pour compiler Squid, nous avons besoin d’un compilateur ANSI C/C++. Si l’on a déjà un compilateur GNU C/C++ (GNU Compiler Collection (GCC) and g++, qui est disponible dans presque toutes les distributions Linux par défaut), nous sommes alors près à compiler.

Pourquoi compiler Squid

Compiler Squid peut être une tâche ardue, comparé à l’installer depuis le package de binaires, cependant, il est recommandé de compiler squid depuis son code source plutôt que d’utiliser des fichiers binaires précompilés, voici ici quelques avantages à la compilation :

* On peut activer des fonctionnalités supplémentaires qui peuvent ne pas être activée dans le package précompilé et/ou désactiver les fonctionnalités dont nous n’avons pas besoin pour un environnement particulier
* « ./configure » recherche des fonctionnalités sur le système et les active/désactive en conséquence, tandis que le package binaire précompilé disposent de fonctionnalités détectées pour le système sur lequel les sources ont été compilées
* En utilisant « ./configure » on peut personnalisé entièrement l’installation de Squid

Pour résumé, la compilation offre beaucoup d’avantages sur l’installation via les packages de binaires qui lui aussi possèdent ses propres avantages comme une installation rapide.

Configure ou system check

« Configure » ou « System Check » est la première étape du processus de compilation et s'effectue en exécutant « ./configure » à partir de la ligne de commande.

Ce programme sonde le système et s'assure que les paquets requis sont installés. Il vérifie aussi les capacités du système et collecte des informations sur son architecture et les paramètres par défaut, tels que les descripteurs de fichiers disponibles, etc.

Après le lancement de la commande « ./configure », le programme va générer les fichiers « Makefile » requis pour la compilation.

Une image contenant texte, Police, capture d’écran, nombre

Description générée automatiquement Une image contenant texte, Police, capture d’écran, nombre

Description générée automatiquement

L'exécution de « ./configure » sans paramètre utilisera les valeurs par défaut. Pour modifier les paramètres par défaut ou désactiver certaines fonctionnalité etc nous devons passer des options à « ./configure »

la commande suivante pour voir les options disponibles ainsi qu'une brève description :

« ./configure --help | less »

Les options :

--prefix : est utilisé pour identifier les différentes version et prévenir des interférences ou confusions entre celles-ci, il est conseillé de les installer dans des répertoires distincts .

Exemple : Pour installer la version 6.10 de Squid on peut utiliser le répertoire /opt/squid/6.10, la commande configure correspondante sera ./configure –prefix=/opt/squid/3.1.10/

Squid offre plus de contrôle sur l’emplacement des types de fichiers. Leurs emplacement peut être définit à l’aide d’options tels que --bindir , --sbindir , etc , pour plus de détails consulter la page d’aide de « configure » .

--enable-gnuregex : Des expressions régulières seront utilisées pour construire les ACL dans squid, si l’installation se fait sur Linux ou Unix, nous n’avons pas besoin d’activer l’option.

--disable-inline : Squid a beaucoup de code qui peut être intégré, ce qui est bon pour une utilisation en production. Mais le code en ligne prend plus de temps à compiler et est utile lorsque nous avons besoin de compiler une source une seule fois pour configurer Squid. Cette option est destinée à être utilisée lorsque nous avons besoin de compiler à plusieurs reprises.

--disable-optimizations : Par défaut Squid est compilé avec des optimisations du compilateur qui se traduisent par de meilleurs performances. Cette option doit être utilisée pour résoudre un problème/tester les versions car cela réduit le temps de compilation.

L’option --disable-inline est automatiquement utilisée.

--enable-storeio : Les performances de Squid dépendent des performances d'E/S (écriture par seconde) du disque lorsque la mise en cache du disque est activée. Plus Squid peut lire/écrire rapidement des fichiers à partir du cache , moins il faudra de temps pour satisfaire une requête.

Différentes techniques de stockage permettent d'optimiser les perf, en fonction du type de trafic/utilisation. On peut utiliser l’option pour construire Squid avec le support de différents modules d'entrées/sorties. Consultez le répertoire src/fs/ dans le code source de Squid pour connaître les modules d'entrées-sorties disponibles.

« ./configure --enable-storeio=ufs,aufs,coss,diskd,null »

--enable-removal-policies : En utilisant la mise en cache sur disque, on demande à Squid d'utiliser un espace disque spécifique pour la mise en cache. A un moment, Squid aura besoin de plus d'espace pour mettre en cache les documents, il doit alors décider quels anciens documents doivent être supprimés/purgés du cache pour libérer de l'espace. Il existe plusieurs politiques de purge de documents pour tirer le meilleur parti de la mise en cache.

Les politiques sont basées sur les structures de données de type "tas" et "liste". La structure

de données en liste est activée par défaut.

Veuillez consulter le répertoire src/repl/ dans le code source de Squid pour connaître les politiques de suppression disponibles.

« ./configure --enable-removal-policies=heap,lru »

--enable-icmp : Permet de déterminer la distance par rapport aux autres pairs de cache et aux serveurs distants afin d'estimer la latence approximative. L’option n'est utile que s'il existe d'autres pairs de cache dans le réseau.

--enable-delay-pools : Squid utilise des pools de retard pour limiter/contrôler la bande passante utilisée par un client ou groupe de clients. Ses pools sont comme des seaux qui laissent échapper des données (trafic web) vers les clients, remplis à un rythme contrôlé. Ils sont utiles lorsque nous devons contrôler la bande passante utilisée.

--enable-esi : Cette option permet à Squid d'utiliser les Edge Side Includes. Si l’option est activée, Squid ignore complètement les en-têtes cache-control des clients. L’option ne doit être utilisée que lorsque Squid est utilisé en mode accélérateur.

--enable-useragent-log : Permet d'enregistrer les en-têtes de l'agent utilisateur dans les requêtes HTTP des clients.

--enable-referer-log : Si nous activons cette option, Squid pourra écrire un champ d'en-tête referer à partir des requêtes HTTP.

--enable-cachemgr-hostname : Cache Manager (cachemgr.cgi) est un utilitaire CGI qui permet de gérer le cache de Squid et de visualiser les statistiques du cache à l'aide d'une interface web. Le nom d'hôte pour accéder au gestionnaire de cache peut être défini à l'aide de cette option. Par défaut, nous pouvons accéder à l'interface web du gestionnaire de cache en utilisant localhost ou l'adresse IP du serveur Squid.

« ./configure --enable-cachemgr-hostname=squidproxy.example.com »

--enable-arp-acl : Permet de créer des listes de contrôle d'accès basées sur les adresses MAC (ou Ethernet). Cette fonction est désactivée par défaut. Si l’on veut contrôler l'accès des clients sur la base de l'Ethernet nous devons activer cette fonctionnalité.

--enable-ssl : Lorsqu’il est configuré en proxy inverse, il peut mettre fin aux connexions SSL initiées par les clients et les gérer pour le compte du serveur web dorsal en amont signifiant qu’il n'aura pas à effectuer de travail sur SSL. Dans ce cas, la communication entre Squid et le serveur web dorsal sera purement HTTP, mais les clients y verront toujours une connexion sécurisée avec le serveur web.

--enable-cache-digests : Les résumés de cache sont un moyen pour Squid de partager des informations avec les serveurs Squid voisins.

--enable-default-err-language : Quand Squid rencontre une erreur qui doit être communiquée au client, Squid utilise des pages par défaut pour afficher ces erreurs. Les pages d'erreur sont disponibles dans les langues locales. L’option permet de spécifier la langue par défaut pour toutes les pages d'erreur. La langue par défaut des pages d'erreur est l'anglais.

« ./configure --enable-default-err-language=Spanish »

--enable-err-languages : Par défaut, Squid prend en charge toutes les langues disponibles. Si nous ne voulons construire Squid qu'avec les langues familières, nous pouvons utiliser cette option. Consultez le répertoire errors/ dans le code source de Squid pour connaître les langues disponibles.

« ./configure --enable-err-languages='English French German' »

--enable-ipfw-transparent : IPFIREWALL (IPFW) est un pare-feu pour système FreeBSD maintenu par sa communauté. Cette option est utile pour configurer le Serveur Proxy Transparent sur les systèmes avec IPFW. Si notre système n'a pas IPFW alors on évite d'utiliser cette option car Squid ne se compilera pas. Le comportement par défaut est l'auto-détection.

--enable-ipf-transparent : IPFilter (IPF) est un pare-feu pour de nombreux systèmes Unix. Il est fourni par NetBSD, Solaris, etc. Si notre système dispose d'IPF, nous devons activer cette option pour pouvoir configurer Squid en mode transparent. L'activation de cette option en l'absence d'IPF sur le système entraînera des erreurs de compilation.

--enable-pf-transparent : Packet Filter (PF) est un pare-feu développé à l'origine pour OpenBSD. L’option est utile sur les systèmes où PF est installé , pour obtenir le mode Proxy transparent. N'activez pas cette option si PF n'est pas installé.

--enable-linux-netfliter : Netfilter est le cadre de filtrage des paquets dans les noyaux Linux des séries 2.4.x et 2.6.x. Cette option est utile pour activer la prise en charge du proxy transparent sur les systèmes Linux.

--enable-follow-x-forwarded-for : Lorsqu'une requête HTTP est transmise par un proxy, celui-ci écrit des informations sur lui-même et sur le client destinataire de la requête, dans les entêtes HTTP. Cette option permet à Squid d'essayer de rechercher l'adresse IP du client d'origine pour lequel la requête a été transmise par un ou plusieurs serveurs proxy.

--enable-default-hostsfile Cette option permet de sélectionner l'emplacement par défaut du fichier hosts. Sur la plupart des OS il se trouve dans le répertoire /etc/hosts.

« ./configure --enable-default-hostsfile=/some/other/location/hosts »

--enable-auth : Active la prise en charge des schémas d'authentification. Cette option est utilisée que pour activer la prise en charge globale de l'authentification et la liste des schémas d'authentification n'est pas transmise.

Le schéma d'authentification est activé avec --enable-auth-AUTHENTICATION\_SCHEME où AUTHENTICATION\_SCHEME est le nom du schéma d'authentification.

Par défaut, tous les schémas d'authentification sont activés et les aides à l'authentification correspondantes sont construites lors de la compilation. Les aides à l'authentification sont des programmes externes qui peuvent authentifier les clients à l'aide de divers mécanismes l'authentification, par rapport à différentes bases de données d'utilisateurs.

--enable-auth-basic : Active la prise en charge d'un schéma d'authentification de base et établit la liste des aides spécifiées. Si la liste des aides n'est pas fournie, cette option activera toutes les aides possibles. Une liste des aides disponibles se situe dans le répertoire helpers/basic\_auth/ dans le code source de Squid.

Pour désactiver ce schéma d'authentification, nous utilisons --disable- auth-basic.

« ./configure --enable-auth-basic=PAM,NCSA,LDAP »

Si nous voulons activer cette option mais que nous ne voulons pas construire d'aides, nous devons utiliser "none" au lieu d'une liste d'aides.

« ./configure --enable-auth-basic=none »

Auparavant, cette option était connue sous le nom de --enable-basic-auth-helpers.

La liste des aides est transmise de la même manière.

« ./configure --enable-basic-auth-helpers=PAM,NCSA,LDAP »

--enable-auth-ntlm : Permet la prise en charge le schéma d'authentification NTLM. Les aides disponibles pour ce schéma se trouvent dans le répertoire helpers/ntlm\_auth/ du code source de Squid.

Pour désactiver la prise en charge de ce schéma, utilisez l'option --disable-auth-ntlm.

« ./configure --enable-auth-ntlm=smb\_lm,no\_check »

--enable-auth-negotiate : Active le schéma d'authentification négocié. Les détails et la syntaxe sont similaires à ceux de l'option de schéma d'authentification ci-dessus.

« ./configure --enable-auth-negotiate=kerberos »

--enable-ntlm-fail-open : Si cette option est activée et qu'une aide échoue lors de l'authentification d'un utilisateur, elle peut toujours permettre à Squid d'authentifier l'utilisateur. Cette option doit être utilisée avec précaution car elle peut conduire à des erreurs d'authentification. aux failles de sécurité.

--enable-external-acl-helpers : Squid supporte les ACLs externes en utilisant des aides. Si nous voulons utiliser des ACLs externes, nous devrions envisager d'utiliser cette option. Nous pouvons également utiliser cette option pendant l'apprentissage. Une liste d'ACLs externes. Les aides ACL doivent être passées pour construire des aides spécifiques.

Par défaut Squid construit toutes les aides disponibles. Une liste des aides ACL externes disponibles se trouve dans le répertoire helpers/external\_acl/ de son code source.

« ./configure --enable-external-acl-helpers=unix\_group,ldap\_group »

--disable-http-violations : Squid dispose d'options de configuration lui permettant la violation des normes du protocole HTTP, en remplaçant les champs d'en-tête dans les requêtes/réponses HTTP. La manipulation des en-têtes va à l'encontre des normes HTTP standard. L’option désactive la prise en charge de toutes violations du protocole HTTP.

--disable-wccp : Désactive la prise en charge du protocole de communication de cache Web (WCCP) de Cisco. Le WCCP permet la communication entre les caches, ce qui contribue à la localisation du trafic. Par défaut, la prise en charge du WCCP est activée.

--disable-wccpv2 : Désactive la prise en charge le protocole WCCP v2 de Cisco (activée par défaut) qui dispose d'un support intégré pour le loadbalancing, la scalabilité, des mécanismes de tolérance aux pannes et d'assurance de service.

--disable-snmp : Dans les versions 3.x de Squid, le protocole SNMP est activé par défaut. SNMP est très populaire parmi les administrateurs de systèmes pour la surveillance des serveurs et des périphériques de réseau.

--disable-htcp : Le protocole HTCP (Hypertext Caching Protocol) peut être utilisé par Squid pour envoyer et recevoir des résumés de cache aux caches voisins. Cette option désactive la prise en charge du protocole HTCP.

--disable-ident-lookups : Empêche Squid d'effectuer des recherches d'identité ou d'identifier un nom d'utilisateur pour chaque connexion. Désactiver cette fonction peut empêcher une attaque DDOS de la part d'un client malveillant.

--disable-internal-dns : Squid possède sa propre implémentation DNS et est capable de construire des requêtes DNS, pour l’utiliser nous ne devons pas le désactiver.

--disable-translation : Par défaut, Squid essaie de présenter les pages d'erreur et de manuel dans la langue locale. Pour que cela ne se produise pas, nous utilisons cette option.

--disable-auto-locale : Sur la base des en-têtes d’une requête cliente, Squid tente de fournir des pages d'erreur localisées. L’option permet de désactiver la localisation automatique. La balise error\_directory dans le fichier de configuration de Squid doit être configurée si nous utilisons cette option.

--disable-unlinkd : unlinkd est un processus externe utilisé pour effectuer des appels système de type "unlink". Cette option désactive le support de unlinkd dans Squid. Désactiver unlinkd n'est pas une bonne idée car l'appel système unlink peut bloquer un processus pendant un temps considérable, ce qui peut entraîner un retard dans les réponses.

--with-default-user : Normalement, nous ne voulons pas exécuter Squid en tant qu'utilisateur root afin d'éviter tout risque de sécurité. Par défaut, Squid s'exécute sous l'utilisateur « nobody ». Si nous avons installé Squid à partir d'un binaire pré-compilé, Squid peut s'exécuter en tant qu'utilisateur « squid »ou « proxy » en fonction de l’OS utilisé. Cette option permet de définir l'utilisateur par défaut pour l'exécution de Squid.

L'exemple suivant montre comment utiliser cette option :

« ./configure --with-default-user=squid »

--with-logdir : Par défaut, Squid écrit tous les journaux et rapports d'erreur dans des fichiers désignés dans ${prefix}/var/ logs/.

Cet emplacement est différent de celui utilisé par tous les autres processus et daemons pour écrire leurs journaux. Afin d'accéder rapidement aux journaux de Squid, nous pourrions vouloir les placer dans le répertoire de journaux système par défaut, qui est /var/log/ dans la plupart des systèmes Linux.

Voir Démarrer avec Squid l'exemple suivant de la syntaxe pour y parvenir :

« ./configure --with-logdir=/var/log/squid/ »

--with-pidfile : L'emplacement du PID de Squid est ${prefix}/var/run/squid.pid, ce qui n'est pas l'emplacement standard du système pour le stockage des fichiers PID.

Sur la plupart des systèmes Linux les fichiers PID sont stockés dans /var/run/. Il est donc possible de modifier l'emplacement par défaut du fichier PID à l'aide de l'option suivante :

« ./configure --with-pidfile=/var/run/squid.pid »

--with-aufs-threads : Permet de spécifier le nombre de threads à utiliser lorsque le système de stockage AUFS est utilisé pour gérer les répertoires du cache. Si cette option n'est pas utilisée, Squid calcule automatiquement le nombre de threads à utiliser :

« ./configure --with-aufs-threads=12 »

--without-pthreads : Les anciennes versions de Squid ne supportaient pas les threads POSIX. Maintenant, Squid est construit avec le support pthreads par défaut, donc, si nous ne voulons pas activer le support pthreads, nous devrons le désactiver explicitement.

--with-openssl : Utilisée pour construire Squid avec le support d’OpenSSL en spécifiant le chemin d’installation d’OpenSSL s’il n’est pas installé à l’emplacement par défaut

« ./configure --with-openssl=/opt/openssl/ »

--with-large-files : En cas de trafic important, les fichiers journaux de Squid (en particulier le journal d'accès) augmentent rapidement et, à long terme, la taille du fichier peut devenir assez importante. Nous pouvons utiliser cette option pour activer la prise en charge des fichiers journaux volumineux.

Pour de meilleures performance, il est recommandé de changer de fichier de log fréquemment plutôt de concentrer les logs dans un seul fichier plus large.

--with-filedescriptors : Les OS utilisent des descripteurs de fichiers (essentiellement des nombres entiers) pour suivre les fichiers et les sockets ouverts. Par défaut, le nombre de descripteurs de fichiers qu'un utilisateur peut utiliser est limité (normalement 1024). Une fois que Squid a accepté des connexions qui ont consommé tous les descripteurs de fichiers disponibles pour l'utilisateur Squid, il ne peut plus accepter d'autres connexions, sauf si certains des descripteurs de fichiers sont libérés.

En cas de forte charge, Squid se retrouve fréquemment à court de descripteurs de fichiers.

Nous pouvons utiliser l'option suivante pour résoudre le problème de pénurie de

descripteurs de fichiers :

« ./configure --with-filedescriptors=8192 »

Un héros en herbe – Le descripteur de fichiers !

Déterminez le nombre maximum de descripteurs de fichiers disponibles pour votre utilisateur. Notez également les commandes qui permettront de fixer la limite maximale de descripteurs de fichiers disponibles à 8192.

Solution : Pour vérifier le nombre de descripteurs de fichiers disponibles, utilisez la commande suivante : « ulimit -n »

Pour fixer la limite des descripteurs de fichiers à 8192, nous devons ajouter les lignes suivantes au fichier /etc/security/limits.conf :

username hard nofile 8192

username soft nofile 8192

L'action précédente ne peut être effectuée qu'avec les privilèges de root ou de super utilisateur.

Passer à l’action – Lancer notre première commande

Maintenant que nous avons jeté un bref coup d'œil à plusieurs des options disponibles, nous pouvons présenter les options pour l'environnement pour lequel nous construisons Squid. Nous sommes maintenant prêts à lancer la commande configure avec les options suivantes :

« ./configure --prefix=/opt/squid/ --with-logdir=/var/log/squid/ --with-pidfile=/var/run/squid.pid --enable-storeio=ufs,aufs --enable-removal-policies=lru,heap --enable-icmp --enable-useragent-log --enable-referer-log --enable-cache-digests --with-large-files »

Explorer Squid

Jetons un œil sur les fichiers et répertoires générés durant l’installation. Utilisons la commande tree pour afficher le répertoire de Squid en arborescence : « tree ${prefix} | less »

${prefix} est le répertoire utilisé pour l’installation avec l’option « --prefix » ajouté à « ./configure ». En pratique, notre commande sera : « tree /opt/squid | less » .

${prefix}/bin

Ce répertoire contient les programmes pouvant être exécutés ou lancés par les utilisateur.

${prefix}/bin/squidclient

squidclient est un client HTTP avec des fonctions avancées permettant la manipulation de requêtes HTTP à des fins de test du serveur Squid. Lancez le pour voir les options disponibles.

${prefix}/etc

**C’est le répertoire contenant tous les fichiers de configuration de Squid.**

${prefix}/etc/squid.conf

C’est le fichier de configuration de Squid, squid.conf est généré durant l’installation, il dispose de la configuration minimale requise pour permettre l’utilisation de Squid. Pour apporter une nouvelle configuration à Squid, c’est ce fichier que nous devons modifier.

${prefix}/etc/squid.conf.default

Squid génère ce fichier de configuration par défaut, ainsi, nous disposons de la configuration minimale par défaut si nous décidons un retour à zéro de la configuration.

${prefix}/etc/squid.conf.documented

Ce fichier est une version entièrement documentée de squid.conf contenant des centaines de commentaires, nous devons toujours nous référer à ce document lorsque l’on veut chercher et ajouter des tags de configuration correspondant à notre version de squid.

${prefix}/libexec

Répertoire contenant des « programmes aidants » construits durant la compilation de Squid.

${prefix}/libexec/cachemgr.cgi

Ce programme CGI fourni une interface web pour gérer le cache de Squid : Cache Manager.

${prefix}/sbin

Répertoire contenant des programmes pouvant être exécutés par un utilisateur root/sudo.

${prefix}/sbin/squid

C’est le programme Squid, qui est généralement exécuté comme un daemon (processus).

${prefix}/share

Répertoire contenant les modèles de page d’erreur, documentation et autres, utilisé par Squid.

${prefix}/share/errors

Répertoire

share

This is the location for error page templates, documentation, and other files used by Squid.

share/errors

This directory contains localized error page templates. The templates are HTML pages and we can customize the error messages displayed by Squid, by modifying these HTML templates.

share/icons

This directory contains a number of small images used for FTP or gopher directory listing.

share/man

This is the place where the man pages for squid, squidclient, and helpers are built during compilation. Man pages are manual or help pages which can be viewed using the command man (available on all Linux/Unix distributions). To view a man page located at /opt/squid/share/man/man8/squid.8, we can use the man command as follows:

**man /opt/squid/share/man/man8/squid.8**

For more details about man pages, please visit http://en.wikipedia.org/wiki/Man\_page.

var

A place for files that change frequently while Squid is running.

var/cache

This is the default directory for storing the cached web documents on a hard disk.

var/logs

This is the default home for all the log files (such as cache.log, access.log, and so on) used by Squid.