Avec les Nuls, tout devient facile !

EXCELLANCE

Le Load Balancing POUR LES NULS

Répartition de charge et haute disponibilité des services Web

A mettre dans toutes les poches !

Willy Tarreau
Concepteur d'HAProxy

Wilfried Train
Chef de produit ALOHA

Le Load Balancing

Répartition de charge et haute disponibilité des services Web

POUR LES NULS

Willy Tarreau Concepteur d'HAProxy

Wilfried Train Chef de produit ALOHA

FIRST Interactive

Le Load Balancing pour les Nuls

Edition française publiée en accord avec Wiley Publishing, Inc.
© 2010 Éditions First
60 rue Mazarine
75006 Paris - France
Tél. 01 45 49 60 00
Fax 01 45 49 60 01
E-mail : firstinfo@efirst.com
Web : www.editionsfirst.fr
Dépôt légal : 1er trimestre 2010

Collection dirigée par Jean-Pierre Cano
Edition : Pierre Chauvot

Imprimé en France

Sommaire

Introduction

Votre entreprise dispose d'un site Web ? D'un Intranet ou d'un Extranet ? D'applications ou services en ligne ? Alors cet ouvrage est fait pour vous. Bienvenue dans l'univers du Load Balancing !

Contexte, problématiques et enjeux

L'histoire commence par un site vitrine sur le Web. Une façon comme une autre, mais nouvelle, de présenter son entreprise, ses produits. Nous sommes à la fin du XXème siècle.

Dix ans, et même un peu plus, se sont écoulés depuis. Une éternité : l'accès à l'information distante est partout. Tout, ou presque, se fait aujourd'hui à partir d'un navigateur Web.

Un arbre de services Web qui cache la forêt de l'infrastructure indispensable à la disponibilité et aux performances de ces services distants. Car aussi pratique et efficace soit-il, le modèle "orienté services" comporte aussi ses risques : engorgement du trafic, baisse des performances, coupures de service, sécurité de l'information, etc. Avec des conséquences parfois lourdes en termes de productivité des équipes, d'image de marque voire même de baisse de chiffre d'affaires.

Objectifs de l'ouvrage

Du client au serveur, le parcours est parfois long et très souvent semé d'embuches. Tout l'enjeu de ce livre est de vous apporter les clés de compréhension d'une démarche de répartition de charge des flux d'information des services Web.

Il n'apporte pas de réponse universelle (chaque cas est unique) mais guidera les responsables d'applications, DSI et autres chefs d'entreprises dans leurs choix en matière d'architecture applicative et réseau.

Dans ce livre, vous retrouverez notamment les outils et les méthodes de répartition de charge mais aussi les bonnes pratiques d'une démarche de *load balancing*, en fonction de vos besoins et de vos contraintes.

Les icônes utilisées dans ce livre

Dans ces encadrés, stop aux idées préconçues. Après, vous ne pourrez pas dire que vous ne saviez pas !

Les limites, parfois mêmes les inconvénients, de telle ou telle technique.

Ce qu'il faut retenir…

Pour approfondir vos connaissances sur l'un des thèmes abordés.

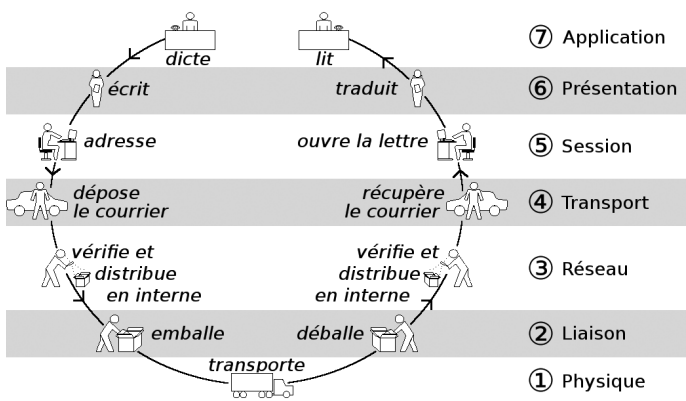
Chapitre 1

Architecture, définitions et concepts

L'architecture réseau type

D'un côté un serveur. De l'autre un poste client. Entre les deux, tout un ensemble d'équipements reliés entre eux pour permettre aux informations de circuler de l'un à l'autre : le réseau.

Simple dans son concept, le réseau est plus complexe qu'il n'y paraît. De nombreux éléments le composent, regroupés en "couches" dans le modèle OSI, sur lequel repose le fonctionnement d'un réseau informatique.



Chaque composant participe, à sa manière, au bon déroulement des opérations de circulation des informations. Une solution de *load balancing* peut intervenir à deux niveaux :

- Niveaux 3/4 : Réseau et transport (protocole TCP/IP) ;
- Niveau 7 : Applications (protocole HTTP, RDP…).

Nous détaillerons plus loin les leviers d'action à manœuvrer à chaque niveau pour améliorer les performances et assurer la disponibilité des services qui s'appuient sur cette architecture.

Définitions et concepts

Avant de plonger dans le vif du sujet, quelques rappels de base.

- ***Paquet*** : afin de circuler plus facilement, les informations échangées sont segmentées en "paquets". La couche 4-Transport s'occupe de cette segmentation et de leur transmission, en s'appuyant la couche 3-Réseau.
- ***Application*** : ce sont les logiciels utilisés pour l'échange des données, tels que les navigateurs, les clients de messagerie, les outils de CRM ou encore d'échange FTP.
- ***Load balancing*** : littéralement "répartition de charge" ou "équilibrage de charge".
- ***Haute disponibilité*** : fait de garantir à l'utilisateur une continuité de service, la bonne marche et/ou l'accessibilité aux applications.

Chapitre 2

La répartition de charge… sans répartiteur

La méthode la plus simple d'opérer une répartition de charge consiste à dédier des serveurs à des groupes d'utilisateurs prédéfinis. Si cette méthode est simple à mettre en œuvre pour un Intranet, elle est très complexe, voire impossible, pour des serveurs Internet.

Dans ce cas, un paramétrage spécifique du DNS (Domain Name Server) au niveau réseau et/ou l'optimisation applicative (niveau 7) sont un premier pas vers une démarche de répartition de charge sans répartiteur.

Le DNS (Domain Name Server)

Dans la majorité des cas, la méthode du **DNS en rotation** (*round robin*) est un bon début en matière de répartition de charge. Pour un même nom de serveur particulier, le serveur DNS dispose de plusieurs adresses IP de serveurs, qu'il présente à tour de rôle aux clients, dans un ordre cyclique.

Pour l'utilisateur, cette méthode est totalement transparente : il ne verra que l'adresse du site. On utilise géné-

ralement la répartition par DNS pour des moteurs de recherche, des serveurs POP (messagerie) ou des sites proposant du contenu statique.

La méthode du DNS souffre de deux limites majeures :

- Elle n'est pas adaptée à la gestion de contexte (cas d'une application nécessitant l'ouverture d'une session utilisateur) ;
- Elle ne fournit pas à elle seule des moyens de vérifier la disponibilité des serveurs, le cache DNS conservant des informations erronées en cas de panne d'un serveur ou d'inaccessibilité d'un site. Des moyens additionnels sont nécessaires pour tester l'état des serveurs et basculer de l'un à l'autre le cas échéant.

L'optimisation applicative

Il s'agit le plus souvent de recettes simples mais particulièrement efficaces, même en présence d'un répartiteur.

La séparation des contenus statiques et dynamiques

Dans la plupart des applications, on peut estimer à seulement un quart la part des contenus dynamiques. Pour les objets statiques, l'utilisation d'un cache *reverse proxy* en frontal de la ferme de serveurs permet aux serveurs de se concentrer uniquement sur les requêtes concernant les contenus dynamiques.

Une tâche qui incombe très simplement à un serveur HTTP léger (lighttpd ou thttpd par exemple), et qui ne nécessite pas forcément l'installation de serveurs dédiés.

L'optimisation du serveur HTTP

Quel que soit le serveur Web retenu (Apache, IIS, etc.), son paramétrage jouera un rôle essentiel quant à son niveau d'absorption de la charge.

Ne négligez pas cette étape d'optimisation du serveur HTTP !

Elle favorisera considérablementles performances de vos applications. Reportez-vous aux spécifications de l'éditeur pour une configuration optimale de votre serveur HTTP.

Sans répartiteur, il est donc possible d'engager une démarche de répartition de charge. Néanmoins, les méthodes décrites ci-dessus ne fournissent aucun contrôle de la disponibilité et exigent donc des moyens additionnels pour tester en permanence l'état des serveurs et basculer le trafic d'un serveur défectueux vers un autre le cas échéant.

Ces méthodes ne constituent donc pas une solution de répartition de charge principale mais conservent leur intérêt et restent complémentaires à la mise en place d'un répartiteur.

Chapitre 3
Le répartiteur de charge

Logiquement, la mise en œuvre d'un répartiteur de charge est la méthode la plus pertinente et efficace lorsqu'il s'agit d'engager une démarche de répartition de charge de la population des utilisateurs sur plusieurs serveurs, voire plusieurs sites.

Le répartiteur peut indifféremment prendre la forme d'un matériel spécifique, d'un logiciel installé sur un serveur dédié ou sur les serveurs d'application. Cette dernière solution étant la plus risquée, en raison notamment du risque de dysfonctionnent lors du déploiement de nouveaux composants.

En fonction des besoins, un répartiteur agira au niveau réseau (couche 3/4) et/ou au niveau applicatif (couche 7).

Les différents niveaux d'action

Matériel ou logiciel, le répartiteur de charge (ou *load balancer*) est la couche supplémentaire qui permet d'optimiser et de réguler le trafic, tout en soulageant les serveurs, en répartissant la charge selon des algorithmes prédéfinis (niveaux 3/4 et 7) et/ou selon des fonctions

intelligentes capables de tenir compte du contenu de chaque requête (niveau 7).

La répartition de charge de niveau 3/4 (Réseau)

La répartition de charge de niveau 3/4 consiste à travailler sur les paquets réseau en agissant sur leur routage (TCP/IP). Le répartiteur de niveau 3/4 intervient donc à l'ouverture de la connexion TCP puis aiguille les paquets en fonction des algorithmes retenus, selon 3 méthodes :

- **Le routage direct**

  Avec cette méthode, le répartiteur "distribue les cartes" : il se charge de répartir les requêtes sur une même adresse entre les serveurs locaux. Les serveurs répondent ensuite directement aux clients : on parle alors de "retour direct du serveur" (DSR : *Direct Server Return*).

  Simple à mettre en œuvre car ne nécessitant aucune modification au niveau IP, cette méthode requiert de solides compétences du modèle TCP/IP pour obtenir une configuration correcte et optimale. Elle implique en outre la proximité des serveurs, qui doivent se trouver sur le même segment réseau que le répartiteur.

- **Le tunneling**

  Le tunneling est une évolution du routage direct qui permet de s'amender de la problématique de proximité des serveurs grâce à la mise en place de tunnels entre des serveurs distants et le répartiteur.

- **La translation d'adresses IP (NAT)**

  Dans ce cas, le répartiteur centralise tous les flux : les requêtes (comme dans le cas du routage direct, il les répartit entre les serveurs), mais aussi les réponses. Il "masque" ainsi l'ensemble de la ferme de serveurs, qui ne nécessitent aucun paramétrage particulier.

  En revanche, cette méthode implique une configuration applicative très stricte afin notamment que les serveurs évitent de retourner leurs adresses internes dans les réponses. En outre, cette méthode augmente considérablement la charge du répartiteur qui doit convertir les adresses dans les deux sens, maintenir lui-même une table de sessions et supporter la charge de trafic retour.

La répartition de charge de niveau 7 (Applicatif)

En s'attaquant à la couche applicative, le répartiteur de charge ne se contente plus d'aiguiller "aveuglément" les requêtes. Il analyse le contenu de chaque requête HTTP, RDP ou autre pour décider du routage.

Le reverse proxy

Dans son rôle de transmission du trafic, le répartiteur de niveau 7 agit comme un *reverse proxy* et prétend être le serveur. Il a alors pour rôle d'accepter les connexions à destination du client et d'établir des connexions avec les serveurs pour faire transiter les données (requêtes et réponses).

Cette méthode implique que les serveurs ne puissent pas être joints directement par les utilisateurs et ne nécessite aucune configuration particulière côté serveur.

Dans ce cas, le répartiteur de niveau 7 nécessite plus de puissance que les solutions matérielles agissant au niveau du réseau. Ils fournissent cependant un premier niveau de sécurité en ne transmettant au serveur que ce qu'ils comprennent.

Le reverse proxy transparent

Parfois, la répartition de charge de niveau 7 peut se heurter à des contraintes d'intégration des serveurs (surveillance des logs ou pare-feu de niveau 3/4) ou de protocole (FTP). Auquel cas, le répartiteur peut simuler l'adresse IP du client en source des connexions qu'il établit vers les serveurs. On parle alors de *reverse proxy* transparent.

Nécessitant un mode coupure, le *reverse proxy* transparent implique que les serveurs doivent non seulement être configurés pour joindre les clients à travers le *load balancer*, mais aussi se trouver sur un segment réseau différent des clients.

En matière de répartition de charge, il n'y a pas de bons ou mauvais choix.

Arbitrer entre un répartiteur de niveau réseau et un répartiteur de niveau applicatif est une affaire de besoins distincts ou cumulatifs. Dans ce cas, deux répartiteurs de niveaux 3/4 et 7 peuvent cohabiter dans la même infrastructure.

Milliers ou millions de sessions concurrentes ?

Pourquoi, quand il s'agit de répartiteur de niveau 3/4, évoque-t-on un support de plusieurs millions de connexions quand les répartiteurs de niveau 7 n'en gèrent que quelques millers ? Simplement parce qu'au niveau TCP (Réseau), le répartiteur ne fait que distribuer à la ferme de serveurs les requêtes qui lui parviennent, en comptabilisant les sessions actives et inactives. Il peut donc utiliser toute sa puissance à ce travail. Tandis qu'un répartiteur applicatif (qui gère les connexions http) doit consulter chaque requête, l'analyser, afin de la transmettre au bon serveur qui, lui, ne comptabilise que les sessions actives. Ce travail, beaucoup plus long et gourmand en ressources, restreint la capacité de traitement du nombre de connexions concurrentes.

Les fonctions principales d'un répartiteur

Nous l'avons vu, un répartiteur peut revêtir plusieurs aspects : matériel et logiciel. Et travailler à plusieurs niveaux : réseau ou application. Pourtant, dans sa définition fonctionnelle, un répartiteur, quel qu'il soit, assumera au moins deux missions principales : l'équilibrage de charge entre les serveurs et la surveillance de ces derniers.

Equilibrer dynamiquement la charge sur les serveurs

En pratique, le répartiteur s'appuie sur des algorithmes de répartition. Il en existe de nombreux qui répondent à des critères bien différents : les durées de sessions, les variations de charge, les capacités de serveurs, etc. Parmi eux, on peut citer :

- **Round Robin**

  Dans ce cas, le répartiteur utilise chaque serveur à la suite, selon un ordre cyclique. Si les serveurs sont de capacités inégales, il est possible d'ajouter une pondération (*weighted round robin*). C'est la méthode la plus efficace pour des serveurs Web.

- **Least Conn**

  Cet algorithme consiste à envoyer la nouvelle requête sur le serveur le moins chargé. Idéal pour des sessions longues, cette méthode est inadaptée à des serveurs dont la charge varie d'une seconde à l'autre.

- **Le hachage d'url ou d'adresse IP**

  Plus déterministe, cette méthode permet de créer une affinité plus ou moins efficace (sous certaines conditions) entre un client et un serveur.

Priorité au serveur le plus rapide : faux !

Envoyer la requête au serveur qui répond le plus vite n'est pas une solution pertinente car cela peut rapidement entraîner un déséquilibre à l'intérieur de la ferme de serveur.

Priorité au serveur le moins chargé : faux !

Nous l'avons vu précédemment, l'algorithme Least Conn (*Least Connections* : le moins de connexion) permet d'envoyer systématiquement les requêtes au serveur le moins chargé. Cette méthode n'est pas du tout adaptée à des serveurs Web par exemple, dont la charge peut varier d'une seconde à l'autre.

Surveiller l'état des serveurs

C'est l'un des aspects les plus complexes de la répartition de charge. Mais néanmoins indispensable : en cas de défaillance de l'un des serveurs, toutes les requêtes seront réparties automatiquement entre les autres serveurs de la ferme. Et ceci, de façon totalement transparente pour l'utilisateur. On parle alors de haute disponibilité applicative.

Baptisées *health checks* (contrôles de vitalité), les méthodes de surveillance des serveurs prennent de nombreuses formes telles qu'un ping, une tentative de connexion TCP ou bien encore l'envoi d'une requête HTTP (niveau 7 uniquement). En effet, un serveur défaillant peut parfois répondre au ping mais pas aux connexions TCP ou accepter ces dernières mais refuser de répondre aux requêtes HTTP. Parfois même, lorsqu'il s'agit d'un serveur d'application Web multi-niveaux, certaines requêtes HTTP retourneront une réponse immédiate tandis que d'autres échoueront.

C'est la raison pour laquelle, afin de garantir la haute disponibilité des services Web, il s'agira avant tout de bâtir

la série de tests réguliers la plus pertinente possible, en fonction du type d'application.

Les contrôles de vitalité des serveurs nécessitent un paramétrage très fin, afin de trouver le meilleur équilibre entre performances dédiées aux applications et surveillance des serveurs.

En matière de surveillance des serveurs, les répartiteurs logiciels sont de loin les plus flexibles. Ils permettent en effet de créer des scénarios automatisés qui pourront être rapidement modifiés et corrigés en très peu de temps.

Le cas des flux chiffrés (https)

Face à l'utilisation croissante des flux chiffrés, de nombreux répartiteurs de niveau 7 fournissent le support du protocole SSL, agissant ainsi comme un *reverse proxy* et servant de terminaison SSL.

Attention : gourmand en ressources, le déchiffrement de requêtes SSL peut rapidement saturer le répartiteur

Dans ce cas, la bonne solution consiste à isoler les traitements SSL en disposant d'une ferme de *reverse proxies* SSL dédiés. Ce qui a pour conséquence de soulager à la fois le répartiteur et les serveurs Web du déchiffrement des requêtes.

Les fonctions "intelligentes" du répartiteur applicatif

Pour répondre à certains besoins particuliers, le répartiteur de niveau 7 dispose de fonctions supplémentaires, issues de sa capacité d'analyse des requêtes.

La persistance

La persistance répond au cas où toutes les requêtes d'un même utilisateur doivent impérativement être adressées à un même serveur sur une session donnée, afin de conserver en mémoire les informations relatives à cette session (le "contexte" de l'utilisateur), telles que le contenu du panier sur un site de e-commerce par exemple.

On parle alors de sessions persistantes ou "collantes" (*sticky sessions*) et de répartition de charge avec affinité de serveur.

Différentes méthodes permettent la mise en œuvre de la persistance :

- ***La redirection :*** c'est l'une des solutions les plus économiques. Elle consiste à faire en sorte que l'application redirige la demande vers l'adresse locale du serveur concerné (en cas de panne du serveur, l'utilisateur sera tout de même redirigé vers ce dernier).
- ***L'affinité de sessions :*** le répartiteur associe un utilisateur à un serveur. La manière la plus simple étant d'identifier le serveur sur lequel s'est connectée l'adresse IP de l'utilisateur lors de la dernière connexion.
- ***L'apprentissage de cookie*** : à l'arrivée d'une requête utilisateur, le répartiteur vérifie la présence du cookie de l'application dans l'en-tête de la requête et compare les valeurs de ce dernier à sa table de

sessions, pour rediriger ensuite l'utilisateur vers le bon serveur. S'il n'y a pas de correspondance, la requête sera dirigée vers n'importe quel serveur, via l'algorithme choisi. Lors de la réponse du serveur à la requête du client, le répartiteur collectera l'information d'identification du serveur (par apprentissage) pour ensuite l'enregistrer dans sa table de sessions. A la seconde requête du client, l'enregistrement alors effectué permettra de rediriger le client vers le bon serveur.

- ***L'insertion de cookie :*** sur le fond, le principe est le même que précédemment, à ceci près que le cookie est inséré directement sur la machine utilisateur

Limites de ces méthodes :

L'affinité de sessions sur l'adresse IP source n'est pas fiable car bon nombre d'utilisateurs surfe sur internet avec des adresses IP variables.

L'apprentissage de cookie implique l'usage d'une table de sessions. En cas de défaillance du répartiteur principal, son backup ne sera pas en mesure de recréer l'association client-serveur. Seule une synchronisation de la table de sessions permettrait de résoudre cette difficulté. Or, il est très difficile de maintenir une table de sessions à jour entre deux répartiteurs.

Si ***l'insertion de cookies*** gomme les limites liées à l'apprentissage de cookie, elle est conditionnée à l'acceptation des cookies par l'utilisateur ? L'usage de terminaux mobiles, parfois limités à un seule cookie, rend cette méthode inexploitable. Une solution de contournement consiste par exemple à décliner cette méthode en modifiant le cookie exixtant et en le préfixant avec l'identifiant du serveur.

par le répartiteur (qui n'a plus alors à maintenir une table de sessions), lors de la réponse du serveur. A partir de la seconde requête, le cookie est lu par le répartiteur et les requêtes sont alors immédiatement redirigées vers le même serveur tout au long de la session.

L'amélioration de la qualité de service

- ***Limites de connexion :*** afin d'éviter toute saturation des applications et des serveurs, le répartiteur limite le nombre de connexions simultanées.
- ***Gestion de la file d'attente et des tampons :*** placé entre les postes clients et les serveurs, le répartiteur régule et optimise le trafic, absorbant ainsi les pics de charge et soulageant les serveurs en privilégiant ceux qui ne sont pas saturés.
- ***Commutation dynamique :*** en fonction du contenu, des dossiers, du nom d'hôte, de l'IP ou du port, le répartiteur dirige la requête vers le serveur adéquat.

La protection contre les attaques

- ***Filtrage des en-têtes HTTP et validation protocolaire :*** sa capacité à analyser les requêtes permet au répartiteur de vérifier la validité de la requête, ainsi que la création de règles sur tous les champs de l'en-tête HTTP.
- ***Protection contre les DoS et DDoS :*** en s'interposant entre les responsables de l'attaque par déni de service (DoS) ou déni de service distribué (DDoS), le répartiteur va non seulement maintenir et ralentir les requêtes incriminées mais aussi simuler une réussite en leur renvoyant un message d'erreur serveur.

- ***Liste des contrôles d'accès IP :*** réactif, un répartiteur de niveau 7 peut faciliter le repérage d'adresses IP responsables d'une attaque, permettant ainsi la création et/ou la modification automatique de règles, de façon à bloquer l'attaque.

Chapitre 4

Comment choisir son load balancer ?

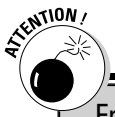
Nous l'avons constaté, il existe de nombreuses solutions de répartition de charge. Toutes ne s'appuient pas sur les mêmes outils et ne répondent pas aux mêmes besoins.

Se poser les bonnes questions

Avant de choisir la ou les bonnes solutions, il convient d'analyser ses besoins et ses attentes d'une solution de *load balancing*.

- **Performances** : combien de connexions simultanées l'application concernée nécessite-t-elle ? C'est un point crucial afin d'éviter que le répartiteur ne devienne le point de congestion du réseau (une situation difficile à remédier).
- **Fiabilité** : Matériel ? Logiciel ? Sur les serveurs Web ? Sur un serveur dédié ? Dans tous les cas, la fiabilité de la solution est à prioriser, l'ensemble du trafic étant alors supporté par le répartiteur.
- **Fonctionnalités** : Simple routeur ? Provenance des requêtes ? Besoin d'analyse des requêtes ? Multi-sites ? Déterminantes certes, les fonctionnalités ne

sont en revanche pas critiques dans le choix d'une solution.



Fréquemment les éditeurs annoncent pouvoir tout faire d'un point de vue fonctionnel grâce à l'usage de scripts, qui s'apparentent plus à du développement spécifique.

La meilleure solution : la combinaison

Indépendamment, chacune des solutions de *load balancing* répond à un besoin particulier. Pour une répartition globale, la meilleure solution est donc la combinaison multi-niveaux.

Un premier niveau de répartiteurs réseau (matériel ou logiciel selon les performances attendues) assurera :

- La répartition de charge de niveau 3/4 entre les reverse proxies SSL et les répartiteurs de niveau 7 ;
- La surveillance des répartiteurs de niveau 7.

Le second niveau de répartiteurs applicatifs permettra d'offrir le niveau "d'intelligence" adéquate, notamment quant au maintien du contexte utilisateurs.

La combinaison est aussi la meilleure solution pour garantir l'évolutivité de l'ensemble. En effet, lorsque les *reverse proxies* satureront, il sera facile d'en ajouter jusqu'à saturation des répartiteurs de niveau 3/4, ce qui correspond déjà à des réseaux multi-gigabits.

Enfin, et si cela était nécessaire, il serait possible d'aller encore plus loin en utilisant la méthode du DNS cyclique (*round robin*) pour adresser de multiples répartiteurs de niveau 3/4, de préférence répartis sur plusieurs sites.

Il est à noter que dans la majorité des cas, les architectures et plateformes Web restent peu sollicitées par rapport aux capacités techniques des solutions actuelles. Dans ce cas, la gestion combinée des niveaux 3/4 et 7 peut parfaitement être assurée par le même produit, réduisant d'autant le coût global de la solution.

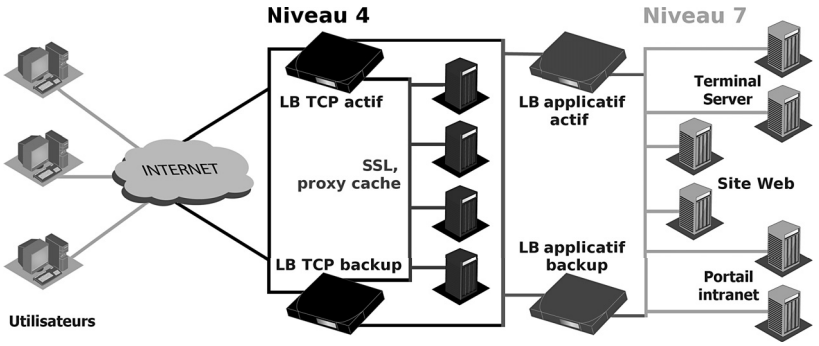
Conclusion

Une dégradation des temps de réponse ou une indisponibilité du service, liée à un serveur saturé (en cas de pic de charge) ou momentanément indisponible (en cas de panne ou de maintenance) ? L'impossibilité d'arrêter la production en pleine journée ? Avec la mise en place d'une solution de répartition de charge, ces problématiques ne sont plus que de lointains souvenirs, le répartiteur redirigeant automatiquement les requêtes sur d'autres serveurs pour absorber la hausse de trafic ou procéder à la maintenance.

Le déploiement de *load balancer* permet en outre un pilotage plus sain et plus serein du système d'informations (connaissance et maîtrise du SI, anticipation des besoins, etc.) et une optimisation des ressources en place.

Néanmoins, la répartition de charge n'accomplira pas de miracle et ne comblera pas les carences et les points de blocage d'une application mal écrite (lente, coûteuse en ressources systèmes, etc.) ni d'une architecture mal pensée, qui créera invariablement des problèmes dans les flux réseau.

En résumé, l'architecture idéale peut ressembler au schéma de la page suivante.



Ainsi "segmentée", l'architecture gagne en agilité et en scalabilité. Elle évolue alors facilement au gré des besoins additifs de l'entreprise.

L'architecture idéale est forcément onéreuse : faux !

L'optimisation des performances des serveurs obtenue grâce aux outils de répartition de charge permet de s'équiper de machines plus légères. Même en plus grand nombre, elles restent néanmoins administrables, plus flexibles et moins gourmandes en énergie que des serveurs traditionnels.

A propos d'Exceliance

Exceliance propose une gamme d'appliances de répartition de charge et de haute disponibilité pour améliorer les performances, assurer la disponibilité et optimiser l'infrastructure des plateformes applicatives critiques d'entreprise (Web, SGBD, messagerie, Terminal Server, ERP...).

Initialement développées à partir du logiciel de *load balancing* open source HAProxy, les solutions ALOHA permettent d'optimiser les flux réseaux et applicatifs.

Reconnues pour leurs performances de traitement, leur fiabilité et leur richesse fonctionnelle, elles sont proposées sous trois formats (logiciel embarqué sur une plateforme matérielle dédiée, sur un disque SSD ou dans une machine virtuelle) à des prix plus abordables que les autres solutions du marché, pour mettre la répartition de charge et la haute disponibilité des applications et services Web à la portée de toutes les entreprises.

Basée à Jouy en Josas (78), Exceliance est labellisée jeune entreprise innovante depuis 2005.

www.exceliance.fr

Enfin des livres qui vous ressemblent !

Tout ce que vous avez toujours voulu savoir sur le *load balancing* !

Votre entreprise dispose d'un site Web ? D'un Intranet ou d'un Extranet ? D'applications ou services en ligne ? Alors cet ouvrage est fait pour vous !

Tout, ou presque, se fait aujourd'hui à partir d'un navigateur Internet. Un arbre de services Web qui cache la forêt de l'infrastructure indispensable pour absorber les pics de charge, garantir des performances optimales et éviter les coupures de service…

Les méthodes, les outils et les bonnes pratiques

Ce livre a pour objectif de vous apporter les clés de compréhension d'une démarche globale de répartition de charge et de haute disponibilité des services Web, parfois même de chasser quelques idées reçues.

Vous y (re)découvrirez -de manière synthétique- les différentes méthodes, les différents niveaux d'action et les outils de load balancing, ainsi que leurs avantages, leurs limites et leurs inconvénients… selon les contextes.

Cet ouvrage n'apporte pas de réponse universelle (chaque cas est unique), mais il vous guidera dans vos choix en matière d'architecture de plateformes applicatives.

Découvrez

Les définitions et concepts

La répartition de charge… sans répartiteur

Les niveaux d'action d'un répartiteur

Les fonctions principales d'un répartiteur

Comment choisir son load balancer ?

Suivez le guide

Cette icône prévient d'un danger et vous indique généralement ce qu'il ne faut *pas* faire.

Dans ces encadrés, stop aux idées préconçues. Après, vous ne pourrez pas dire que vous ne saviez pas !

Pour approfondir vos connaissances sur l'un des thèmes abordés.

Ce qu'il faut retenir…

FIRST Interactive