reseaux peer to peer

Table of contents

Ce module explore une partie de l'ensemble des technologies peer to peer qui existent aujourd'hui. On y donne aussi un tableau comparatif des deux modèles en concurrence le P2P et « client/serveur ».

1. Le modèle client-serveur

1.1. Introduction à la notion client/serveur

Dans les entreprises, le nombre élevé des machines peut devenir parfois problématique, même le réseau internet est basé sur ce type d'architecture appelée centralisée, par conséquent la maintenance et la gestion deviennent des enjeux capitaux, les répercussions peuvent être catastrophique.

Comme son nom l'indique cette architecture est composée d'un client et d'un serveur qui vont collaborer afin d'effectuer une tâche particulière.

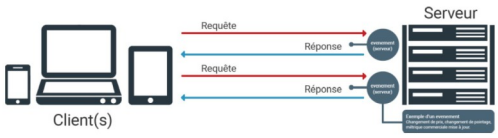
Définition *Definition*

Le modèle client-serveur se traduit autour d'un réseau dont deux types d'ordinateurs y sont connectés, « client et serveur ». La communication client-serveur est rétablie par des protocoles. Les données et les applications sont réparties entre client et serveur de tel sorte à réduire le cout (latence, fraies matérielles, etc.). Le client-serveur fait référence au dialogue entre processus via des messages. Le processus serveur réalise ce dont le processus client sous-traite comme service. Généralement les processus s'exécutent dans des machines, des OS et des réseaux hétérogènes. Et c'est grâce au client-serveur que les environnements hétérogènes permettent de tirer profit du meilleur des deux monde (interface des Pc, puissance et sécurité des serveurs).

Notion client / serveur *Definition*

Un environnement client-serveur est un mode de communication entre plusieurs programmes ou logiciels, celui qui envoie des requêtes dit Client, l'autre en revanche attend les requêtes des clients et y répond, dit serveur, autrement dit dans des ordinateurs Client le logiciel client est exécuté, ainsi dans des ordinateurs serveurs le logiciel serveur s'y exécute.

Les serveurs sont généralement des ordinateurs voués à abriter des logiciels serveurs, leurs aptitudes sont bien supérieures à celles des ordinateurs personnels de par leur puissance de calcul, les entrées-sorties et connexions réseau. Ils se distinguent par leur capacité de répondre aux requêtes d'un grand nombre de Clients. Quant aux clients, habituellement se sont des ordinateurs personnels ou des appareils individuels tels que le téléphone, la tablette... etc.



Le client *Definition*

Le logiciel Client permet à son utilisateur d'exploiter l'ensemble des services présents sur un réseau donné, d'ailleurs l'appellation reste la même pour les ordinateurs connectés à un serveur. La couche applicative du modèle OSI dicte la conduite de la communication de ce dernier.

Caractéristique client

-Actif (maitre).

-Envoie des requêtes au serveur.

-Il attend et reçoit les réponses du serveur.

-Il existe de sorte de client-serveur, il est dit plat si les clients communiquent qu'avec un seul serveur a l'inverse de l'hiérarchique ou les clients contactent seulement les serveurs en dessus d'eux, par exemple les serveurs DNS.

Le serveur *Definition*

Les machines dotées d'un système d'exploitation adapté à la gestion du réseau sont appelé Serveur, ces stations de travail ayant ce rôle sont généralement de puissants ordinateurs. A son tour le serveur communique comme prévu avec le client par la couche application du modèle OSI.

Parmi les services offerts par le serveur on cite :

-**La gestion des données**, elle consiste à conserver les données enregistrées par les utilisateurs en leur attribuant un espace disque suffisant.

-**Le partage d'information**, est le fait de pouvoir stocker un ensemble de fichiers et de logiciels mise à disposition des clients.

-L'administration du réseau, permet la communication entre les clients.

-La sécurité, assure l'intégrité et la confidentialité des informations, ou encore l'intrusion d'un virus dans le système. Les parades contre les pirates sont multiples, par exemple les pare-feu, l'antivirus… etc.

Caractéristique serveur

-Le serveur sert à héberger des services, il est éventuellement possible qu'il soit spécialisé en serveur d'applications, de fichier, de terminaux ou de messagerie électronique.

-Si il est en attente d'une requête, il est passif (esclave).

-Il est disposé de répondre aux requêtes des clients lorsqu'il est à l'écoute.

-A la réception d'une requête il se met au traitement pour y répondre.

-Il peut répondre à plusieurs requêtes de divers clients simultanément.

-Il est contrôleur d'accès et responsable à l'intégrité globale.

-Aucun impact sur les clients lors de la mise à niveau des serveurs du moment où l'interface des messages reste la même.

Avantage et inconvénients

L'architecture centralisée présente avant tout un avantage sur deux plans simplicité et puissance de calcul, c'est-à-dire les interactions entre l'utilisateur et le serveur sont assurés par le client, il a pour tâche de s'en charger du traitement inhérent à ces interactions et envoie les requêtes au serveur. Le serveur, quant à lui, exécute les tâches correspondantes après avoir traité les requêtes de tous les clients.

Ses inconvénients se résument à son non-Scalabilité, le besoin du serveur à avoir assez de ressources peut être problématique notamment en terme de coût afin d'être capable de supporter un grand nombre de pairs. Sans négliger qu'il présente un point unique de défaillance, une panne du serveur et c'est tout le système qui se trouve en panne ce qui fait sa vulnérabilité face aux attaques tel que le déni de service.

Caractéristiques Client / Serveur *Note*

Les éléments qui caractérisent une architecture client-serveur sont

-Service, la relation entre l'ensemble des processus qui tourne sur de multiple machine sont dites modèle client-serveur, le client consomme ce que le serveur fournie comme services.

-Partage de ressources, Un serveur assure la gestion et le contrôle d'accès aux ressources de plusieurs clients.

-Protocole asymétrique, l'asymétrie du protocole de communication est due au partage de ressources, décrite par le scénario à sens unique qui réside entre le client et le serveur, le client débute l'interaction, le serveur attend les requêtes des clients.

Transparence de la localisation, peu importe que le service soit sur la même machine ou accessible par le réseau l'architecture client-serveur doit dissimuler au client la localisation du serveur.

-Message, les échanges entre client et serveur se font par messages.

-Evolution, est le fait que l'architecture client-serveur ait le pourvoir de garantir une éventuelle évolution aussi bien qu'horizontal (lorsque le nombre des clients augmente) que vertical (l'évolution d'à la fois le nombre et les caractéristiques des serveurs).

2. Le modèle pair-à-pair

pair-à-pair *Definition*

il y a plusieurs définitions pour les systèmes pair-à-pair, nous citons ici quelques-uns :

-Le pair-à-pair (P2P : Peer-to-Peer) est un réseau informatique dans lequel chaque noeud du réseau est un serveur et un client à la fois. L'information est répartie sur l'ensemble des machines qui composent le réseau pair à pair.

-Les systèmes pair-à-pair permettent à plusieurs machines de communiquer via un réseau, comme il permet aussi à tous les pairs de jouer les deux rôles client et serveur à la fois, de plus chacun de ces noeuds permet de télécharger des services ou ressources fourni par un autre noeud dans le réseau.

-Les réseaux pair-à-pair (égal à égal) offrent l'opportunité d'avoir une communication et une collaboration des noeuds de manière explicite, ces derniers permettent aussi la décentralisation du réseau, le partage de l'ensemble des ressources du réseau pair-à-pair sans transiter par un serveur ou un pair central.

Principe et fonctionnement *Method*

Le système pair-à-pair est composer par un ensemble de machine, ces machine ont besoin d'un logiciel particulier car ces derniers sont très importants pour sa mise en oeuvre, ces logiciels sont capable de fonctionner à la fois en mode client et en mode serveur. Cette particularité est l'une des avantages de ce system, à partir de là, chaque pair est capable d'envoyer et récupérer des données. De plus ce system comporte un nombre important de noeuds qui collaborent l'un avec l'autre, par conséquent plus le nombre de services disponibles dans ce système est grand plus la probabilité de trouver un service est élevé, ce system nous permet aussi de partager l'information facilement et empêcher plus au moins les attaques légales ou pirates. Le modèle pair-à-pair permet la décentralisation des services et mettre à disposition des ressources dans le réseau, appelées objets. Dont chaque noeud du réseau a la possibilité d'en publier et d'en obtenir sur le réseau.

Ce modèle a deux méthodes pour réaliser la tâche principale du réseau, qui est le partage et l'exploitation des ressources du réseau.

-La méthode centralisée est basée sur un serveur qui possède la liste des fichiers partagés et qui oriente les utilisateurs vers un utilisateur possédant le fichier souhaité.

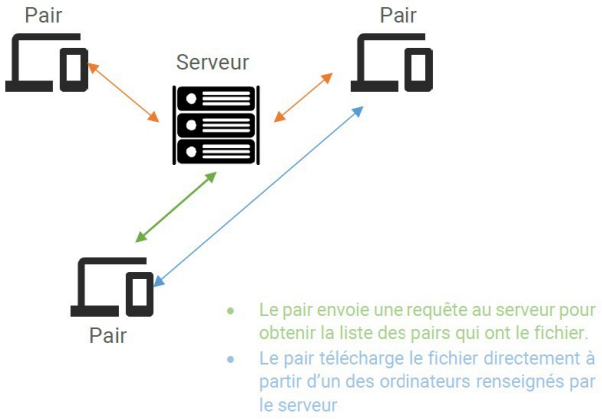
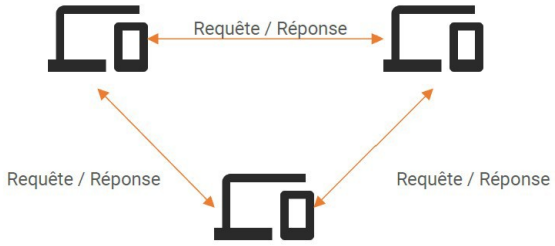


Image 1 pair-a-pair approche centralisée

-La méthode décentralisée utilise chaque utilisateur comme un mini-serveur et ne possède aucun serveur fixe. Cette méthode a l'avantage de répartir les responsabilités.



Pair-à-pair décentralisée

Avantages du P2P

- Décentralisation des ressources
- Les communications entre les paires sont directes
- Passage à l'échelle : c'est-à-dire servir un nombre important de noeuds (jusqu'à des milliers ou des millions) pour partager leurs ressources tout en maintenant une bonne performance du système.
- Tolérance aux fautes
- Possibilité de créer des groupes
- Si une machine tombe en panne, cela ne remet pas en cause l'ensemble du système.
- Le réseau est faiblement couplé.
- Connectivité occasionnelle.
- La réplication, redondance des données.

Inconvénient du p2p

- Pas de Qualité du Service (QoS)
- Problèmes de sécurité
- Les temps de localisation sont plus longs
- Les noeuds et connexions sont non fiables.

Caractéristique du réseau P2P

les principales caractéristiques du modèle pair-a-pair

Décentralisation, c'est-à-dire, chaque noeud gouverne ses propres ressources ce qui permet la décentralisation du contrôle, par la suite le système est capable de fonctionner sans la nécessité d'une administration centrale ce qui empêche les goulets d'étranglements et d'augmenter la tolérance du système aux pannes et aux défaillances.

Passage à l'échelle, elle consiste à la contribution de nombreux noeuds (jusqu'à des milliers voire des millions) afin de partager leurs ressources en garantissant la performance du système. De ce fait le système doit offrir des méthodes adéquates avec l'environnement où les données à partager sont volumineux, résultat d'un échange important de messages suite au partage des ressources d'un grand nombre de noeuds dans un réseau largement distribué.

L'auto-organisation, pour cela il suffit de rejoindre un point d'accès via un noeud déjà connecté afin de se connecter en retour dans le système, désormais avec le déploiement des systèmes P2P sur internet, cette démarche est plus facile par rapport au cout qui ne demande pas une infrastructure couteuse. Il se doit au système pair-à-pair d'être ouvert, autrement dit, il est primordiale qu'un utilisateur sur un noeud ait la capacité de connecter son noeud sans avoir à faire à l'intermédiaire d'une seconde personne ni par une autorité centrale.

Autonomie des noeuds, la gestion des ressources propres aux noeuds est autonome, Libre au noeud de décider quelle partie de ses données veut partager d'autant qu'il est libre de se connecter et se déconnecter par sa volonté. D'ailleurs son autonomie ne se limite pas qu'à ça mais il est également de sa portée de gérer sa puissance de calcul et sa capacité de stockage.

Hétérogénéité, le système pair-à-pair dispose de techniques adaptées afin de contrer les problèmes liés à l'hétérogénéité des ressources engendrés par l'autonomie des noeuds ayant des architectures aussi bien matérielles que logicielles hétérogènes.

Dynamisme, encore une fois l'autonomie pose problème, le fait que chaque noeud peut quitter le système à sa guise, ses ressources du système disparaissent avec lui, et vice versa lorsqu'un nouveau noeud se connecte, ses ressources s'ajoutent au système, donc l'instabilité des noeuds a fait que le système pair-à-pair soit dans l'obligation de pouvoir gérer cette forte variation du nombre de ressources. Comme elle se doit également de tolérer ou minimiser l'impact d'une éventuelle panne d'un noeud sur la performance de tout le système.

Comparaison des infrastructures client/serveur et P2P *Extra*

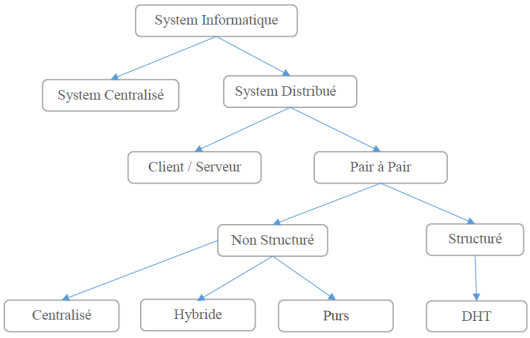
la technique client-serveur appuie l'échange de service entre ordinateurs, selon elle, il n'y a de place que pour une seule entité centrale très puissante, c'est le serveur, quant aux restes des entités, souvent de puissances inférieurs, c'est des clients. Seul le serveur fournit des services aux clients, le client n'a qu'à consommer les services exécutés par le serveur sans avoir à partager ses propres ressources. Tandis que l'architecture pair-à-pair advienne comme une solution de rechange de par ses multiples avantages par rapport au modèle client-serveur.

critère	Modèle client-serveur	Modèle pair-à-pair
Gestion	Supervisé	Auto-organisé
Présence	Permanente	Ad Hoc
Accès aux ressources	Recherche	Découverte
Organisation	Hiérarchique	Distribuée
Mobilité	Statique	Mobile
Disponibilité	Dépendante du serveur	Indépendante des pairs
Nommage	DNS	Indépendant

Table 1

Classification des architectures pair-à-pair

Les systèmes informatiques peuvent être classé en deux catégories différentes systèmes distribués et systèmes centralisés, ces derniers à leur tour sont divisés en deux modèles, le modèle client/serveur et le modèle pair à pair, le modèle pair à pair est aussi divisé en deux réseaux, les réseaux structurés et non structurés.



Classification de l'architecture P2P

Réseaux non structures

Dans un réseau pair à pair non structuré les liens entre les pairs sont basés sur des liens physiques, ce dernier a la possibilité de supporter des requêtes plus complexes avec des métas informations. La connexion d'un nouveau pair se fait par un pair (intermédiaire) déjà connecté dans le réseau, malgré cela ce system n'arrive pas à localiser les fichiers impopulaires.

Approche centralisée

La première génération du système pair à pair a gardé le concept de centralisation. La particularité de ce modèle par rapport au modèle client/serveur est dans la récupération d'objets qui est décentralisé et qui s'effectue directement entre les pairs sans passer par le serveur central. Ce dernier contient aucune ressources il possède juste un annuaire qui contient tous les informations concernant la description des ressources partagées (nom, taille, chemin…), ainsi que des informations sur les pairs qui les hébergent (nom du pair,adresse IP, nombre de fichiers partagés), l'exemple le plus courant reposant sur ce modèle est Napster (system de partage de fichiers MP3).

Napster

Definition

En 1999, Shawn Fanning un étudiant à l'université de boston aidé par Ritter et Sean Parker, créent et développent une application de partage de fichiers musicaux mp3 sur le réseau internet, Napster est le premier réseau pair-à-pair grand public qui adopte le modèle pair à pair centralisé, cela signifie que ça façon de fonctionner est la même que l'architecture pair-à-pair centralisée.

Comment fonctionne Napster

Method

Tout d'abord tous les utilisateurs doivent avoir à la fois le logiciel et une connexion internet pour exécuter ce dernier. Après avoir une connexion entre les clients et le serveur, les clients doivent annoncer au serveur leur ressources (fichiers, adresse IP, port) pour que les autres clients puissent les contacter, Par la suite le serveur maintient un annuaire qui contient la liste des clients connectés et aussi des informations sur ces clients en particulier les fichiers partagées, les clients doivent contacter le serveur napster en envoyant une requête. S'ils veulent chercher un fichier dans le réseau, quand le serveur reçoit une requête il renvoie aux clients une liste des réponses qui contient adresse IP, nom du client ,la taille du fichier.., après le client demandeur établit une connexion direct avec le client qui lui convient le mieux parmi la liste des clients possédant le fichier désiré, il envoie par la suite au pair choisi son adresse IP et le nom du fichier voulu, l'échange du fichier se fait directement entre eux.

Avantages d'un système centralisé

Extra

- Avantages habituels d'un serveur central : facile à administrer et à contrôler et aussi à fermer.
- Evite les recherches coûteuses sur le réseau : pas de routage ni de planification de la gestion des utilisateurs
- Tolérance aux fautes : par un sondage régulier des pairs connectés, état cohérent.

Inconvénient d'un système centralisé

Extra

- Les limites habituelles d'un serveur central : l'attaque ou la panne du serveur qui rend le serveur non disponible dans le réseau.
- Problème de passage à l'échèle : saturation de la bande passante et du nombre de processus.
- Pas d'anonymat car chaque pair est connu du serveur et des pairs sur lesquels il télécharge.

Approche décentralisée (Purs)

Dans cette architecture tous les noeuds sont égaux et jouent le même rôle et communiquent entre eux, c'est-à-dire, ils s'en passent du serveur, chaque élément du réseau est appelé servent qui est le rassemblement de deux mots client et serveur, la question qui se pose dans cette approche est comment un servent peut découvrir et accéder à une ressource dans ce type d'architecture ? La technique d'inondation en est la solution.

Recherche par inondation

Method

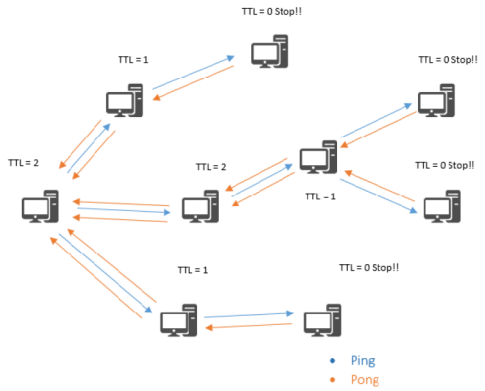
Cette solution a pour but de découvrir une ressource dans le réseau par la transmission d'une requête d'un servent à autre jusqu'à arriver au servent qui dispose de l'objet voulu. Pour éviter l'inondation du réseau durant un temps trop long, la requête est associé par un temporisateur TTL "Time To Live". Généralement la valeur du TTL est allouée à 7, si la valeur de ce dernier arrive à zéro la requête n'est plus renvoyée, mais l'inconvénient majeur de cette solution est lorsque le TTL expire c'est-à-dire il arrive à la valeur zéro sans être sûr de traverser tous les éléments du réseau, ce qui peut conclure une recherche par un échec malgré la disponibilité de l'objet dans le réseau. Cette technique est utilisée par le Protocol Gnutella.

Gnutella *Definition*

Gnutella est un protocole de partage de fichiers qui repose sur l'architecture pair à pair non structurée, succédant à Napster, crée en mars 2000 par Justin Fränkel et Tom Pepper, le réseau est composé par un ensemble de pairs connectant le réseau dont la transmission des informations entre les servents (ou noeuds) du réseau est basé en moyenne sur 5 descripteur (voir le tableau). Si un client veut rejoindre le réseau Gnutella,

premièrement il envoie une trame d'identification (PING) a ses voisins qui eux-mêmes la transmettront à leurs tours et ainsi de suite. Lorsque le TTL devient 0 la retransmission s'arrête, avec cette technique d'inondation des messages, on peut vite tomber dans le problème des boucles, pour éviter ce soucie, la trame va être stocké pendant un court instant lorsque le noeud reçoit la trame Ping. Si un noeud reçoit une trame identique pendant ce laps de temps il la rejette car elle est déjà reçue, par la suite chaque client qui reçoit une requête Ping répond avec une requête Pong qui contient l'adresse IP, le numéro de port, le nombre et la taille des fichiers partagés. Avec tout ça le client peut identifier les noeuds connectés dans le réseau.

Type	Description	Information
Ping	Annonce la disponibilité, et lance une recherche de pair	Vide
Pong	Réponse à une requête Ping	Adresse IP, N° port, Nombre et Taille de fichiers partagés
Query	Requête visant à trouver un ou plusieurs fichiers	Bande passante, adresse IP et numéro du port, nom du fichier
QueryHit	Une réponse à un Query si on possède le ressource	Adresse IP, numéro du port, bande passante, taille de fichier
Push	Demande de téléchargement pour les pairs derrière un firewall	IP, Index du fichier demandé numéro du port…

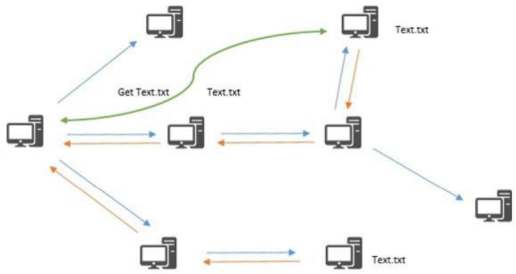


la découverte des pairs avec Ping et Pong

La recherche

Extra

La recherche des ressources dans le réseau Gnutella se lance quand l'un des servents envoie un descripteur Query en spécifiant la vitesse de transfert minimum et les critères de recherche. Si un servent reçoit une trame de type Query et s'il dispose de la ressource, il renvoie une requête de réponse QueryHit au voisin qui lui a retransmis la requête, spécifiant son adresse IP et son numéro de port TCP ou l'objet peut être téléchargé, la réponse va remonter jusqu'au servent qui a demandé la ressource. ce dernier va sélectionner les fichiers à télécharger et envoyer une requête de téléchargement au servent qui possède la ressource, Il existe un autre descripteur Push est utilisé si les ressources sont derrière un pare feu.



recherche des fichiers sur un réseau Gnutella

Avantages d'un système décentralisé

Extra

- Administration simple.
- Topologie évolutive.
- Disponibilité du réseau, on peut l'arrêter.
- Simplicité du mécanisme de routage et de recherche.
- Réseau tolérant aux fautes.
- S'adapte bien à la dynamique du réseau.

Inconvénient d'un système décentralisé

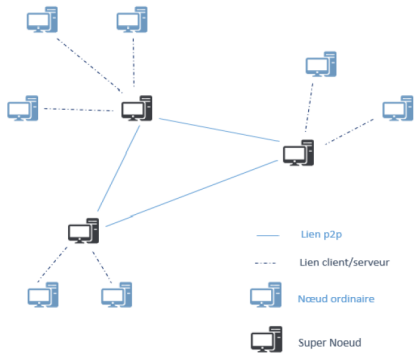
Extra

- Pas de sécurité.
- Problème du free-riding (personnes ne partageant pas de fichiers).
- La recherche non déterministe à cause de la limite du TTL.
- Gros consommateur de Bande passante.

- Ne supporte pas le passage à l'échelle parce que le réseau est rapidement inondé par des Ping et des Pong.

Approche Hybride

Ce modèle est une combinaison entre le modèle centralisé et le modèle décentralisé, autrement dit ce dernier est organisé d'une manière hiérarchique. On distingue deux types de noeuds, le noeud standard et le super noeud, la liaison entre eux se fait de manière centralisée, chaque noeud standard se connecte à un super noeud. L'interconnexion entre les supers noeuds est située en haut niveau de la hiérarchie selon le modèle décentralisé, FastTrack est un exemple typique de protocole P2P partiellement centralisé. Dans FastTrack les super noeuds ont un rôle particulier et sont caractérisés par une forte capacité de calcul et une large bande passante etc. Ces super-noeuds ont la possibilité de gérer des groupe de noeuds (noeuds standard moins puissants) ou ils servent de serveur local dont les objets partagés par les noeuds standards sont enregistrés sur ces derniers (voir la figure ci-dessous), de nombreuses applications utilisent ce modèle, par exemple Kazaa, Bit Torrent.



architecture hybride

Torrent *Definition*

Le protocole BitTorrent est un protocole de partage de fichiers peer-to-peer (P2P) qui permet aux utilisateurs de distribuer des données sur Internet de manière efficace. Plutôt que de télécharger un fichier à partir d'une seule source, BitTorrent permet aux utilisateurs de se joindre et de télécharger des morceaux de fichiers à partir de plusieurs sources simultanément.

Acteurs dans le Protocole Torrent *Fundamental*

- Le Seed : C'est l'utilisateur qui a un fichier complet et le met à disposition pour le téléchargement.
- Torrent : Extension du fichier contenant les informations sur les données à télécharger..
- Le Leecher : Client désirant télécharger des parties (ne possède pas l'intégralité des données)
- Le Tracker (Traqueur) : Il s'agit d'un serveur qui facilite la communication entre les peers. Il aide les utilisateurs à trouver d'autres utilisateurs qui partagent le fichier désiré.

Comment fonctionne le Protocole Torrent *Method*

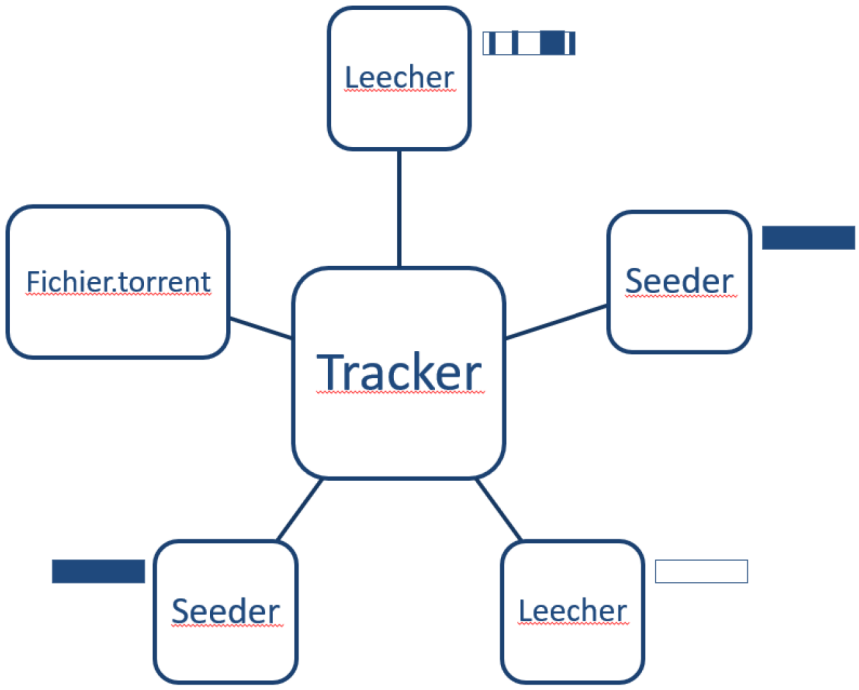
Quand un fichier est mis en partage, il est divisé en petites parties, appelées morceaux. Ces morceaux peuvent être téléchargés séparément, permettant aux utilisateurs de télécharger des morceaux de différents seeders.

Quand vous voulez télécharger un fichier en utilisant le protocole BitTorrent, vous commencez par télécharger un petit fichier .torrent. Ce fichier contient des informations sur le fichier que vous voulez télécharger, y compris son nom, sa taille et les morceaux dans lesquels il est divisé.

Votre client BitTorrent utilise ces informations pour se connecter au tracker, qui a une liste des seeders et des leechers pour ce fichier. Votre client BitTorrent se connecte ensuite à autant de seeders que possible et commence à télécharger des morceaux du fichier.

À mesure que vous téléchargez des morceaux, votre client BitTorrent commence également à les partager avec d'autres utilisateurs. Cela signifie que même si vous n'avez pas encore téléchargé le fichier complet, vous contribuez déjà à sa distribution.

C'est ce mécanisme qui fait de BitTorrent une méthode très efficace pour partager de grands fichiers avec un grand nombre d'utilisateurs.



Fonctionnement

L'Encodage dans BitTorrent ***Fundamental***

L'encodage dans BitTorrent fait référence à la façon dont les données sont structurées et organisées pour le partage. Les fichiers torrent utilisent un format d'encodage appelé "Bencoding".

Le Bencoding est un moyen d'organiser et de structurer les données pour qu'elles puissent être facilement transmises et comprises par différents clients BitTorrent. Il y a quatre types de données différentes dans le Bencoding: les chaînes de caractères, les entiers, les listes et les dictionnaires.

1. Les Chaînes de Caractères:

Dans le Bencoding, une chaîne est encodée en écrivant d'abord sa longueur, suivie de deux-points, puis la chaîne elle-même. Par exemple, la chaîne "torrent" serait encodée comme

"7:torrent".

2. Les Entiers:

Un entier est encodé en écrivant 'i', suivi de l'entier lui-même, suivi de 'e'. Par exemple, l'entier 123 serait encodé comme

"i123e".

3. Les Listes:

Une liste est une collection d'autres éléments, qu'il s'agisse de chaînes, d'entiers, de listes ou de dictionnaires. Une liste est encodée en écrivant 'l', suivie par l'encodage de chaque élément de la liste, puis 'e'. Par exemple, une liste contenant les entiers 123 et 456 serait encodée comme

"li123ei456ee".

4. Les Dictionnaires:

Un dictionnaire est une collection de paires clé-valeur. Il est encodé en écrivant 'd', suivi par l'encodage de chaque clé et de sa valeur correspondante, puis 'e'. Les clés doivent être des chaînes et elles doivent apparaître dans l'ordre lexicographique. Par exemple, un dictionnaire avec une seule clé "abc" et une valeur 123 serait encodé comme

"d3:abci123ee".

voici un fichier torrent pour ubuntu

d8:announce35:https://torrent.ubuntu.com/announce13:announce-listll35:https://
torrent.ubuntu.com/announceel40:https://ipv6.torrent.ubuntu.com/announceee7:comment29:Ubuntu
CD releases.ubuntu.com10:created by13:mktorrent 1.113:creation
datei1634219565e4:infod6:lengthi3116482560e4:name30:ubuntu-21.10-desktop-amd64.iso12:piece
lengthi262144e6:pieces237780:

announce	https://torrent.ubuntu.com/announce
announce-list	https://torrent.ubuntu.com/announce https://ipv6.torrent.ubuntu.com/announce
comment	Ubuntu CD releases.ubuntu.com
created by	mktorrent 1.1
creation date	1634219565
info	Length = 3116482560 Name = ubuntu-21.10-desktop-amd64.iso piece length = 262144 Pieces = 237780

Avantages d'un système hybride *Extra*

- Présence d'un serveur central : facile à administrer, et donc facile à contrôler
- Evite les recherches couteuses sur le réseau : pas de routage et planification de la gestion des utilisateurs
- Tolérance aux fautes en sondant régulièrement les pairs connectes et en maintenant un état cohérent

Inconvénient d'un système hybride *Extra*

- Pas d'anonymat partout car chaque pair est connu du serveur et des pairs sur lesquels il télécharge.
- Limites habituelles d'un serveur central : problème de disponibilité, de passage à l´échelle (saturation de la bande passante et du nombre de processus).
- Certains pairs peuvent mentir sur leur débit pour ne pas être sollicités.

Protocoles réseau structuré

Un réseau structuré repose sur la structure logique du réseau, les noeuds sont reliés par un réseau recouvrant construit sous certaines contraintes, répondant à plusieurs propriétés et connectant les pairs selon une structure particulière donnée. La topologie virtuelle est assurée par des méthodes de routage et de localisation, chaque topologie a ses propres méthodes de routage, de nommage et de recherche

Chord

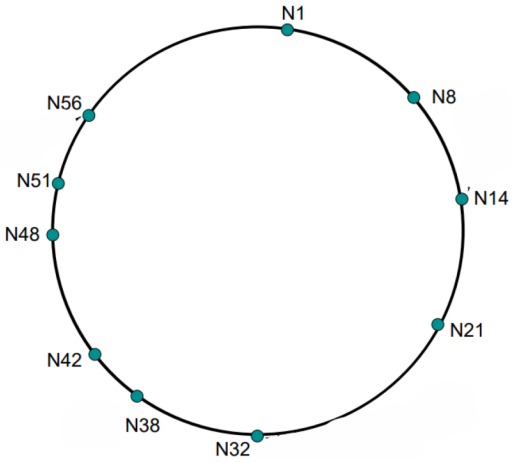
Definition

Chord utilise une topologie en anneau. Il dispose notamment d'un algorithme d'une complexité d'au plus O (log N) requêtes, il utilise le principe des tables de hachage distribuées afin de trouver une information dans un anneau de N éléments. Cette complexité est due à la liste des pairs successeurs en puissance de deux dont chaque pair maintient.

L'espace d'adressage

Method

Le hachage de Chord consistent à attribuer à chaque noeud et chaque donnée un identifiant ID à m bits, où m est un paramètre prédéfinie de système. Les ID se situent dans un intervalle de 0 à 2^m -1. Les noeuds sont ordonnés dans un cercle d'identifiant modulo 2^m, l'identifiant d'une donnée est caché dans le noeud qui la succède, c'est le prochain noeud dans le cercle dans le sens des aiguilles d'une montre.



m=6

$2^6 = 64$

Nodes =0,1,2…..61,62,63

Les noms de fichiers sont également mappés en utilisant la même fonction de

hachage cohérente

SHA-1(fichie)→160bit (clef k)

Le fichier est stocké sur le premier noeud dont l'ID est supérieur ou

égal à sa clef K mod 2^m

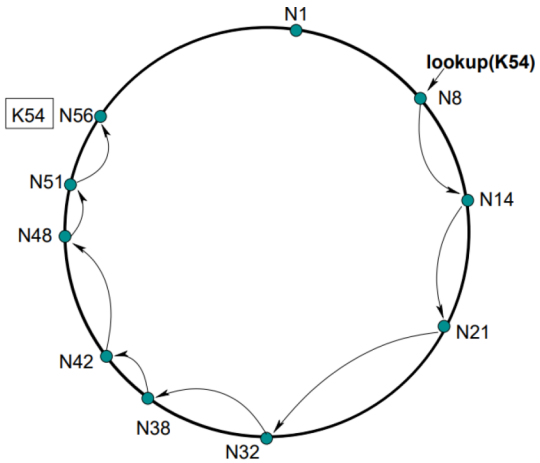
Recherche simple

Method

Pour localiser une clé dans le réseau Chord, un pair commence par vérifier si la clé se trouve entre son propre identifiant et l'identifiant de son successeur immédiat. Si c'est le cas, le pair sait que la clé est stockée sur son successeur. Sinon, il demande à un pair dans le réseau qui est plus proche . Ce processus se répète jusqu'à ce que la clé soit trouvée.

```
// ask node n to find the successor of id
n.find_successor(id)
  if (id ∈ (n, successor])
    return successor;
  else
    // forward the query around the circle
    return successor.find_successor(id);
```

(a)



(b)

Recherche simple

recherche avancée *Method*

La recherche de clés peut être accélérée en utilisant une structure appelée Finger Table. Chaque pair maintient uneFinger Table, qui est essentiellement une liste de raccourcis vers d'autres pairs dans l'anneau Chord.

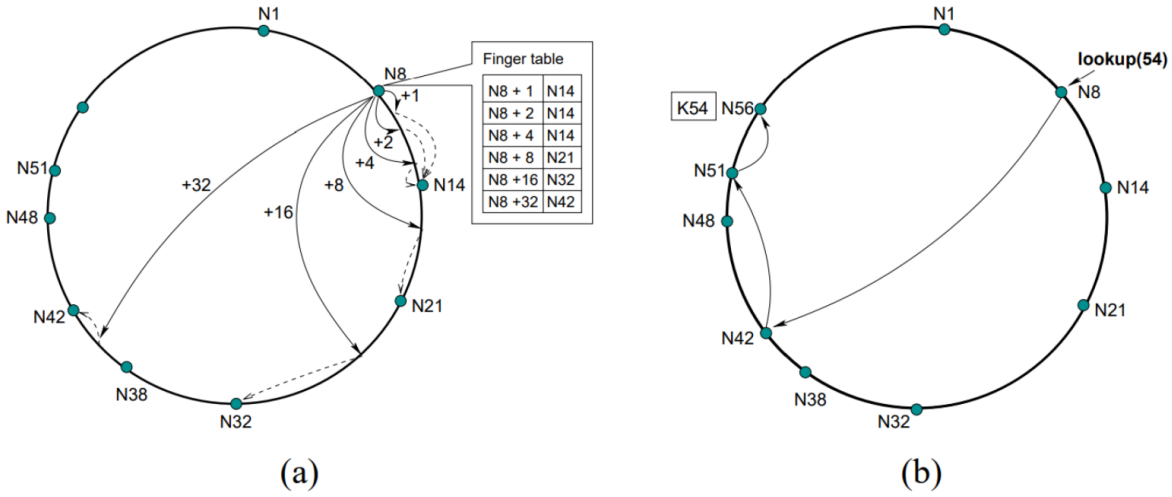
i	f(i)		succ(n)
1	$(N + 2^{i-1}) \bmod 2^m$	(8+1)mod64=9	N14
2	$(N + 2^{i-1}) \bmod 2^m$	(8+2)mod64=10	N14
3	$(N + 2^{i-1}) \bmod 2^m$	(8+4)mod64=12	N14
4	$(N + 2^{i-1}) \bmod 2^m$	(8+8)mod64=16	N21
5	$(N + 2^{i-1}) \bmod 2^m$	(8+16)mod64=24	N32
6	$(N + 2^{i-1}) \bmod 2^m$	(8+32)mod64=40	N42

Finget table pour N8

1- Si ID est entre N et succ(N), le succ(N) est retourné.

2- Sinon, la recherche est effectuée au noeud N*, qui est le noeud dans

Finger table de N qui précède l'ID .



Le protocole Kademlia *Definition*

Le protocole Kademlia est un algorithme de routage distribué qui permet de stocker et récupérer des données dans un réseau pair à pair (P2P). Il a été proposé en 2002 par Petar Maymounkov et David Mazières. Kademlia utilise une structure de données appelée table de routage (DHT, Distributed Hash Table) pour localiser les noeuds et les données dans le réseau.

Identifiant de noeud (Node ID) : *Definition*

Chaque noeud du réseau possède un identifiant unique, généralement un nombre de 160 bits. Les identifiants sont utilisés pour organiser et localiser les noeuds dans l'espace des clés.

Distance XOR :

Definition

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

La distance entre deux noeuds est calculée en utilisant la fonction XOR (ou

exclusif) sur leurs identifiants. Cette distance est utilisée pour déterminer les noeuds les plus

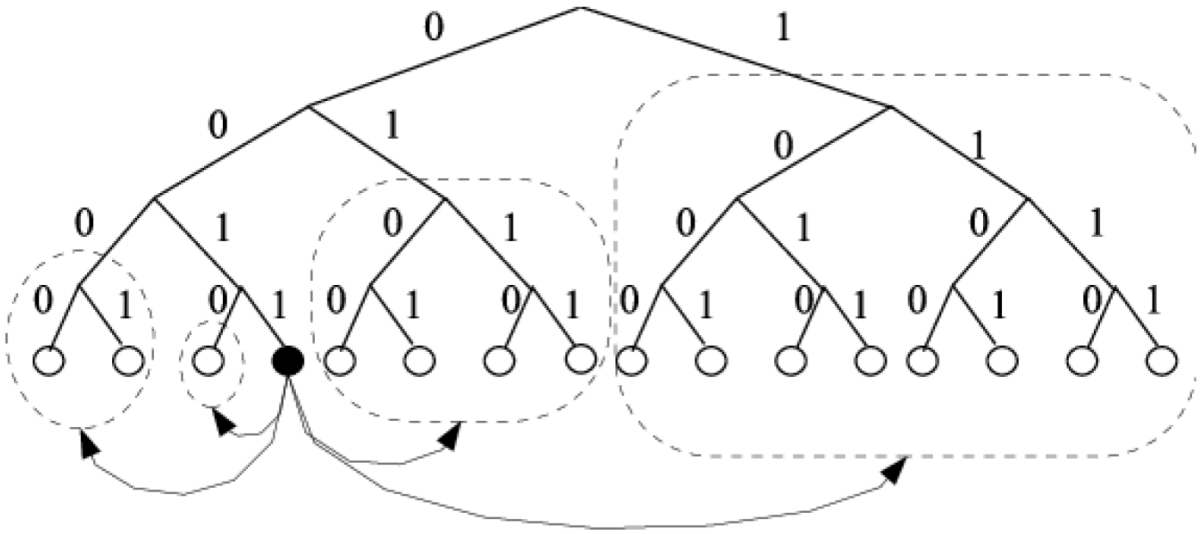
proches lors de la recherche d'informations.

Table de routage (DHT) :

Method

Chaque noeud maintient une table de routage contenant des informations sur d'autres noeuds du réseau. La table est organisée en "k buckets ", où "k" est un

paramètre (nombre des noeuds).Chaque k bucket contient des noeuds à une certaine distance de l'identifiant du noeud local



Dans Kademlia, il est important que chaque nœud ait une connaissance des nœuds dans d'autres sous-arbres. C'est-à-dire qu'il devrait y avoir au moins un nœud dans chaque Seau-K qui n'est pas dans le même sous-arbre que le nœud actuel. Cela assure que le réseau reste connecté et que les nœuds peuvent toujours atteindre d'autres parties du réseau, même si certains nœuds tombent en panne ou quittent le réseau comme le montre la figure

Cette connaissance des nœuds dans d'autres sous-arbres est également importante pour la recherche de clés. Lorsqu'un nœud recherche une clé, il peut avoir besoin de consulter des nœuds qui sont dans d'autres sous-arbres pour trouver la clé. Avoir des informations sur ces nœuds dans sa table de routage permet au nœud de trouver la clé plus efficacement.

Recherche et stockage de données : *Method*

Les données sont stockées dans le réseau sous forme de paires clé valeur, où la clé est un identifiant unique et la valeur est l'information associée. Pour stocker ou rechercher des données, un noeud interroge les noeuds les plus proches de l'identifiant de la clé.

Lorsqu'un nœud cherche une clé dans le réseau Kademlia, il commence par consulter son K-bucket pour lequel la distance à la clé est la plus courte. Il envoie ensuite une requête à ces nœuds pour la clé.

Si aucun de ces nœuds ne connaît la clé, le nœud demande alors à le plus proches noued de la clé. Ce processus est répété jusqu'à ce que la clé soit trouvée, ou jusqu'à ce qu'il n'y ait plus de nœuds plus proches à interroger.