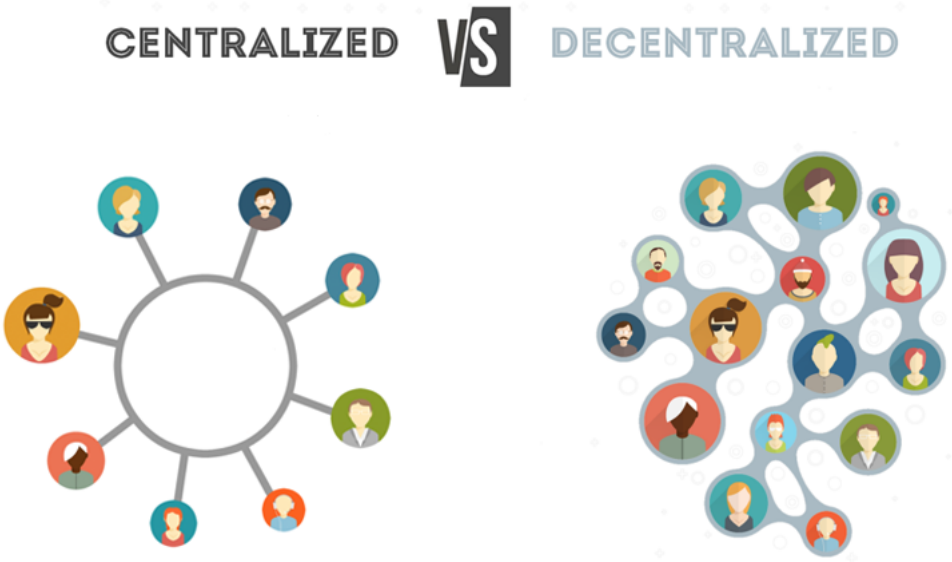
Title

Subtitle



Auteurs

Benjamin NIDDAM

Encadrants

Adrien LUXEY-BITRI

Table des matières

1 Introduction **2**
1.1 Contexte 2
1.2 Problématique 3
1.3 Objectifs 3
1.4 Plan 3

2 Solutions existantes **4**
2.1 Les technologies de communication Web 4
2.1.1 WebSockets 4
2.1.2 WebRTC 5

3 Comment mettre en place une connection paire à paire **6**
3.1 Avec WebRTC 6
3.1.1 Initialisation d'une paire et récupération de son ID 6
3.1.2 Partage des ID entre les deux pairs 6
3.1.3 Création d'une connexion entre les deux pairs 6
3.1.4 Echange de données entre les deux pairs 7
3.2 Avec WebSockets 8
3.2.1 Récupération de l'adresse IP publique du client 8
3.2.2 Partage des adresses IP entre les deux pairs 8
3.2.3 Création d'une connexion entre les deux pairs 8
3.2.4 Echange de données entre les deux pairs 8
3.3 Les outils tiers nécessaires pour la communication pair à pair 9
3.3.1 Les serveurs de signalisation 9
3.3.2 Les protocoles TURN et STUN 10
3.3.3 Les serveurs de relais 11

Chapitre 1

Introduction

1.1 Contexte

Dans le cadre du PJI (Projet Individuel) en fin de master 1, j'ai choisi de travailler sous la tutelle de monsieur LUXEY-BITRI sur le sujet des communications réseaux dites paire à paire. Avec peu de connaissances en réseaux, j'y ai vu l'occasion de découvrir un domaine qui m'était inconnu et de me confronter à un sujet de recherche pour l'instant peu exploré mais avec un potentiel d'application très important. En effet, si aujourd'hui les communications réseaux sont majoritairement centralisées, on voit émerger de plus en plus de réseaux dits décentralisés.

Les réseaux décentralisés, également connus sous le nom de réseaux pair-à-pair (P2P), offrent une approche alternative à la communication et à l'échange de données. Contrairement aux réseaux centralisés, où les données transitent par un point central tel qu'un serveur, les réseaux P2P permettent aux utilisateurs de communiquer directement entre eux, en utilisant leurs propres ressources (telles que la puissance de calcul et la bande passante) pour faciliter les échanges.

Ce modèle de communication présente plusieurs avantages, notamment une meilleure résilience face aux pannes, une répartition de la charge plus équilibrée et une réduction de la dépendance à l'égard d'entités centrales. Les réseaux P2P sont largement utilisés dans divers domaines, tels que le partage de fichiers (comme BitTorrent), la messagerie instantanée (comme Skype) et les systèmes de paiement (comme Bitcoin).

J'ai choisi ce sujet car la majorité de mon travail personnel consiste à développer des applications web basées sur des architectures centralisées. Je souhaitais donc découvrir un nouveau domaine et me confronter à de nouvelles problématiques. De plus, les réseaux pair à pair sont un sujet de recherche très actuel et qui a un fort potentiel d'application. En effet, les réseaux pair à pair sont de plus en plus utilisés dans le domaine des communications réseaux. On peut citer par exemple le protocole WebRTC qui permet la communication audio et vidéo entre deux clients sans passer par un serveur.

1.2 Problématique

Quelles sont les solutions possible pour mettre en place une communication textuelle temps réel entre deux clients dans un réseau pair à pair ?

1.3 Objectifs

L'objectif de ce projet est l'étude des mécanismes permettant le "paire à paire" dans le web moderne grace à l'implémentation de prototypes permettant la communication textuelle entre deux clients.

1.4 Plan

1. Dans un premier temps, afin de mieux comprendre le fonctionnement des réseaux pair à pair, nous allons étudier et présenter brièvement les différents les solutions existantes
2. Dans un second temps, nous étudierons les différentes façons de mettre en place une communication textuelle temps réel entre deux clients dans un réseau pair à pair. Nous verrons qu'il existe plusieurs solutions possibles telles que l'utilisation de sockets, l'utilisation de WebRTC ou encore l'utilisation de WebSockets.
3. Dans un troisième temps, présenterons la l'alternative avec WebRTC que nous avons implémenter. Nous expliquerons les raisons de ce choix et nous détaillerons l'implémentation de notre solution. Avant de tenter la réalisation de cette même solution via des sockets. Nous verrons dans cette partie les difficultés rencontrées, les solutions apportées et les limites de notre implémentation.
4. Enfin, nous conclurons sur les résultats obtenus et nous évoquerons les perspectives d'amélioration de notre solution.

Chapitre 2

Solutions existantes

Dans le domaine des communications réseaux pair à pair, plusieurs solutions ont déjà été développées pour établir une communication entre deux clients. Dans cette partie, nous présenterons brièvement quelques-unes de ces solutions.

2.1 Les technologies de communication Web

Avec l'avènement du Web, de nouvelles technologies de communication ont émergé pour permettre des interactions directes entre les utilisateurs. Parmi celles-ci, nous pouvons citer les WebSockets et WebRTC.

2.1.1 WebSockets

Les WebSockets sont une technologie de communication bidirectionnelle qui permet aux navigateurs Web d'établir une connexion persistante avec un serveur. Contrairement aux requêtes HTTP traditionnelles, qui suivent le modèle de requête-réponse, les WebSockets permettent une communication en temps réel, où les données peuvent être transmises dans les deux sens de manière asynchrone.

Les WebSockets sont largement utilisés dans les applications nécessitant une communication instantanée, telles que les chats en temps réel, les tableaux de bord de suivi des données en temps réel, les jeux en ligne, etc. Ils offrent une latence réduite et une meilleure efficacité en éliminant le besoin de requêtes HTTP fréquentes pour récupérer les mises à jour des données.

En utilisant les WebSockets, les clients peuvent établir une connexion directe avec un serveur WebSocket, et une fois la connexion établie, ils peuvent échanger des messages sous forme de flux de données. Cette technologie permet une communication bidirectionnelle et simultanée, ce qui en fait une solution efficace pour la communication textuelle temps réel entre deux clients dans un réseau pair à pair.

2.1.2 WebRTC

WebRTC (Web Real-Time Communication) est une technologie qui permet aux navigateurs Web d'établir des connexions peer-to-peer directes pour la communication audio, vidéo et de données en temps réel. WebRTC utilise une combinaison de protocoles et de codecs pour faciliter la communication directe entre les navigateurs, sans passer par un serveur intermédiaire.

WebRTC a ouvert de nouvelles possibilités pour des applications telles que les appels audio et vidéo en temps réel, les conférences Web, les partages d'écran et bien plus encore. Il permet aux utilisateurs d'interagir directement sans avoir à installer de plugins ou de logiciels tiers.

Dans le contexte de la communication textuelle temps réel dans un réseau pair à pair, WebRTC peut être utilisé pour établir une connexion directe entre deux clients, permettant ainsi une communication en temps réel. Les clients peuvent échanger des messages textuels en utilisant la fonctionnalité de transfert de données offerte par WebRTC.

L'avantage de WebRTC réside dans sa capacité à établir des connexions directes entre les clients, ce qui réduit la latence et offre une expérience de communication plus fluide. Cependant, il convient de noter que l'utilisation de WebRTC nécessite la mise en place de mécanismes de signalisation pour l'établissement de la connexion initiale entre les clients, ainsi que des aspects de sécurité pour protéger la confidentialité des données échangées.

Chapitre 3

Comment mettre en place une connection paire à paire

3.1 Avec WebRTC

L'utilisation de WebRTC est l'une des solutions possibles pour mettre en place une communication textuelle en temps réel entre deux clients dans un réseau pair à pair. Pour établir cette connexion, les étapes suivantes peuvent être suivies :

3.1.1 Initialisation d'une paire et récupération de son ID

Chaque client doit être initialisé en tant que paire et se voit attribuer un ID unique. Cet ID est utilisé pour identifier la paire et permettre à d'autres pairs de se connecter à elle.

3.1.2 Partage des ID entre les deux pairs

Une fois que chaque paire a obtenu son ID, ces identifiants doivent être partagés entre les deux pairs afin qu'ils puissent s'identifier mutuellement lors de l'établissement de la connexion. Cela peut être réalisé par l'intermédiaire d'un serveur de signalisation ou d'un canal de communication préalablement établi, tel qu'un serveur centralisé ou un autre moyen de communication sécurisé.

3.1.3 Création d'une connexion entre les deux pairs

Une fois que chaque pair dispose de l'ID de l'autre, ils peuvent utiliser ce dernier pour établir une connexion directe entre eux.

3.1.4 Echange de données entre les deux pairs

Une fois la connexion établie, les deux pairs peuvent s'échanger des données. Ces données peuvent être des messages textuels, des fichiers, des flux audio et vidéo, etc. (Dans notre cas, nous nous intéressons uniquement aux messages textuels.)

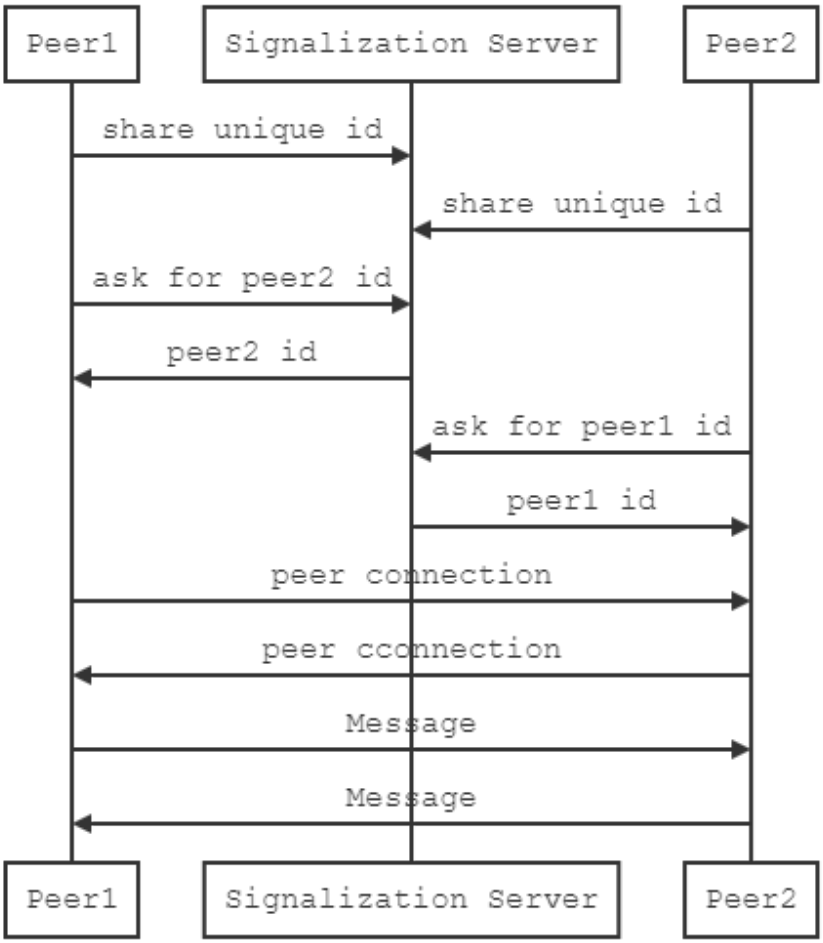


FIGURE 3.1 – Schéma de fonctionnement de WebRTC

3.2 Avec WebSockets

Une autre solution pour mettre en place une communication textuelle en temps réel entre deux clients dans un réseau pair à pair est d'utiliser les WebSockets. La principale difference avec WebRTC, c'est que les Sockets nécessitent la connaissance des adresses IP des deux pairs pour pouvoir établir une connexion entre eux. Pour établir cette connexion, les étapes suivantes peuvent être suivies :

3.2.1 Récupération de l'adresse IP publique du client

Pour pouvoir établir une connexion entre deux pairs, il faut que chaque pair connaisse l'adresse IP publique ainsi que le port utilisé par l'autre pair. Mais avant cela, il faut que chaque pair récupère son adresse IP publique afin de la partager avec l'autre pair. Pour cela, il existe une solution (non unique) qui consiste à utiliser un service STUN qui permet de récupérer l'adresse IP publique du client ainsi qu'un port ouvert sur le pare-feu du client.

3.2.2 Partage des adresses IP entre les deux pairs

Une fois ces informations récupérées, chaque pair doit partager son adresse IP publique ainsi que le port ouvert sur son pare-feu avec l'autre pair. Cela peut être réalisé par l'intermédiaire d'un serveur de signalisation ou d'un canal de communication préalablement établi, tel qu'un serveur centralisé ou un autre moyen de communication sécurisé.

3.2.3 Création d'une connexion entre les deux pairs

Une fois que chaque pair dispose de l'adresse IP publique et du port ouvert sur le pare-feu de l'autre pair, ils peuvent utiliser ces informations pour établir une connexion directe entre eux.

3.2.4 Echange de données entre les deux pairs

Une fois la connexion établie, les deux pairs peuvent s'échanger des données. Ces données peuvent être des messages textuels, des fichiers, des flux audio et vidéo, etc. (Dans notre cas, nous nous intéressons uniquement aux messages textuels.)

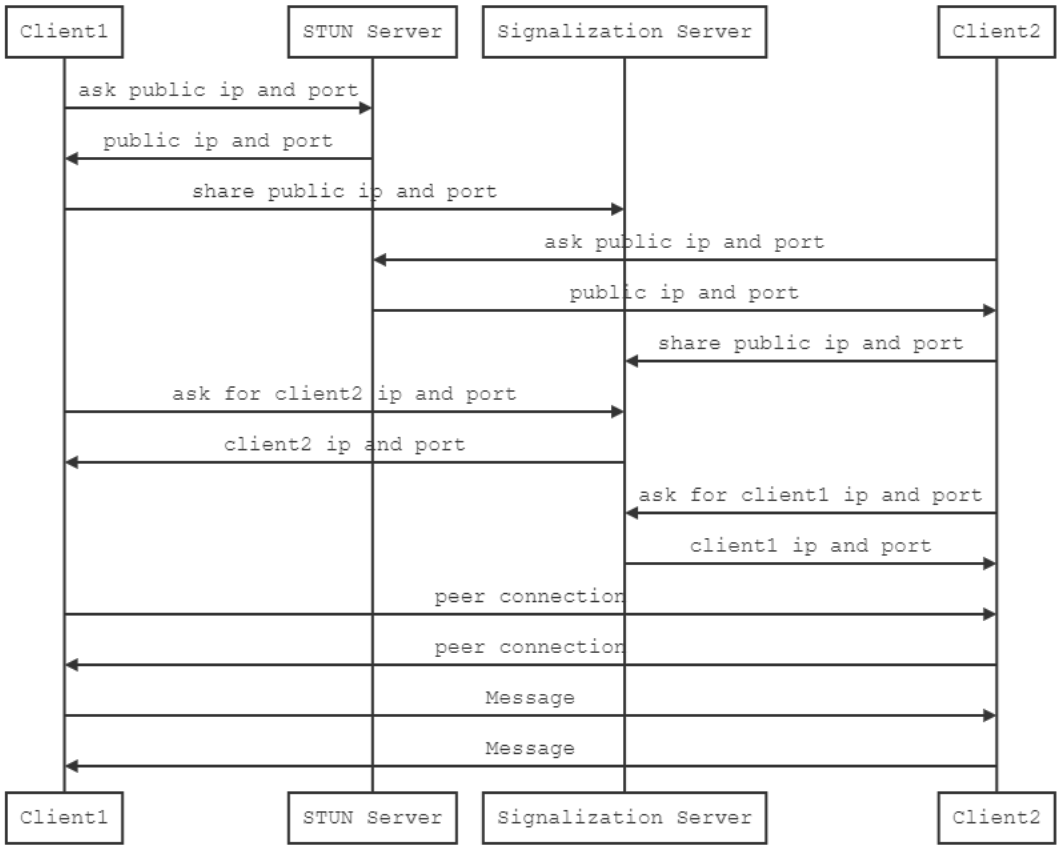


FIGURE 3.2 – Schéma de fonctionnement de WebSockets

3.3 Les outils tiers nécessaires pour la communication pair à pair

3.3.1 Les serveurs de signalisation

Pour établir une communication pair à pair entre deux clients, il est souvent nécessaire de mettre en place des serveurs de signalisation. Les serveurs de signalisation agissent comme des intermédiaires pour faciliter l'établissement de la connexion initiale entre les clients.

Lorsqu'un client souhaite établir une communication avec un autre client, il doit d'abord échanger des informations telles que les adresses IP et les ports. Les serveurs de signalisation permettent aux clients de partager ces informations entre eux. Ils peuvent également aider à coordonner les étapes de négociation et

d'établissement de la connexion.

Les serveurs de signalisation peuvent utiliser différents protocoles et techniques pour échanger les informations nécessaires. Par exemple, ils peuvent utiliser des protocoles basés sur le web tels que HTTP pour envoyer des messages de signalisation entre les clients. Ils peuvent également utiliser des protocoles de signalisation spécialisés tels que SIP (Session Initiation Protocol) ou XMPP (Extensible Messaging and Presence Protocol).

Il existe différentes options pour mettre en place des serveurs de signalisation, allant des solutions prêtes à l'emploi aux serveurs personnalisés développés en interne. Certains services cloud offrent également des fonctionnalités de signalisation pour faciliter l'établissement de connexions pair à pair.

Il est important de noter que les serveurs de signalisation ne sont généralement pas impliqués dans le transfert direct des données entre les clients. Ils jouent simplement un rôle de coordination pour permettre l'établissement de la connexion initiale.

3.3.2 Les protocoles TURN et STUN

Pour faciliter les communications pair à pair, il est souvent nécessaire de prendre en compte les contraintes de réseau telles que les pare-feu et les NAT (Network Address Translation). Deux protocoles couramment utilisés pour résoudre ces problèmes sont TURN (Traversal Using Relays around NAT) et STUN (Session Traversal Utilities for NAT).

Le protocole STUN permet à un client d'obtenir des informations sur son adresse IP publique et le type de NAT qu'il traverse. Cela permet au client de comprendre comment il est connecté au réseau et d'adapter ses stratégies de connexion en conséquence. STUN est généralement utilisé pour faciliter l'établissement de connexions pair à pair lorsque les clients sont derrière des NAT symétriques ou des pare-feu restrictifs.

Le protocole TURN est utilisé lorsque les connexions pair à pair directes ne sont pas possibles en raison de pare-feu ou de NAT restrictifs. TURN utilise un serveur intermédiaire (relais) auquel les clients se connectent pour acheminer leurs données. Ce serveur intermédiaire agit comme un proxy et transfère les données entre les clients. Cela permet aux clients de communiquer même s'ils ne peuvent pas établir une connexion directe.

Lorsqu'un client rencontre des difficultés pour établir une connexion pair à pair directe en raison des restrictions du réseau, il peut utiliser STUN pour obtenir son adresse IP publique, puis essayer d'établir une connexion directe. Si cela échoue, le client peut passer à l'utilisation de TURN pour acheminer les données via un serveur relais.

Il est important de noter que l'utilisation de TURN peut entraîner une latence supplémentaire car les données doivent être acheminées via le serveur relais. De plus, l'utilisation d'un serveur relais peut entraîner des coûts supplémentaires, notamment si vous utilisez un service cloud ou un fournisseur tiers pour héberger le serveur.

En résumé, les protocoles TURN et STUN sont des outils importants pour faciliter les communications pair à pair lorsque les clients sont derrière des NAT restrictifs ou des pare-feu. Ils permettent de contourner les contraintes réseau et de faciliter l'établissement de connexions directes ou le transfert des données via un serveur relais lorsque cela est nécessaire.

3.3.3 Les serveurs de relais

Dans certains cas, il peut être nécessaire d'utiliser des serveurs de relais pour faciliter la communication pair à pair. Les serveurs de relais agissent comme des intermédiaires pour acheminer les données entre les clients lorsque des connexions directes ne sont pas possibles en raison de restrictions de pare-feu, de problèmes de NAT (Network Address Translation) ou d'autres contraintes réseau.

Lorsqu'un client ne peut pas établir une connexion directe avec un autre client en raison de ces contraintes, il peut utiliser un serveur de relais pour envoyer ses données vers le serveur, qui les transférera ensuite au client destinataire.

Les serveurs de relais peuvent être mis en place de différentes manières. Certains services cloud proposent des serveurs de relais prêts à l'emploi, tandis que d'autres solutions nécessitent la configuration de serveurs dédiés pour le relais des données.

L'utilisation de serveurs de relais peut être bénéfique dans certains scénarios où les connexions directes entre les clients ne sont pas possibles. Cependant, cela introduit généralement une surcharge supplémentaire et peut augmenter la latence de la communication.

En étudiant ces outils tiers nécessaires pour la communication pair à pair, nous pourrons mieux comprendre comment ils peuvent être intégrés dans notre solution de communication textuelle temps réel. Cela nous permettra d'explorer les fonctionnalités, les avantages et les limites de chaque outil, et d'évaluer leur pertinence pour notre projet.