الجمـــهوريـــة الجـــزائريـــة الديـــمقراطيـــة الشعبيـــة
RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
وزارة التعـلـيـــم العـالـــي و البـــحث العلـــــمي
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE
جـــامعـة عمـــار ثليـجي بالأغـــواط
UNIVERSITÉ AMMAR TELIDJI LAGHOUAT

كـلـيـــة العلـــــوم و الـهـنـدسـة
FACULTÉ DES SCIENCES ET SCIENCES DE L'INGÉNIEUR
DÉPARTEMENT DE GÉNIE INFORMATIQUE

MÉMOIRE DE MASTER

DOMAINE : MATHÉMATIQUES ET INFORMATIQUE **(MI)**
FILIÈRE : INFORMATIQUE
OPTION. : RÉSEAUX, SYSTÈMES ET APPLICATIONS RÉPARTIS **(ReSar)**

Présenté par :
Sid Amina

Thème :

Les protocoles de traversée de NAT pour les communications Peer-to-Peer

Soutenu publiquement devant le jury composé de :

Mr. T.ALLAOUI	Maître-assistant (A)	U. Laghouat (Président)
Mr. Y.GUELLOUMA	Maître-assistant (B)	U. Laghouat (Examinateur)
Mr. L.K.OULED JDID	Maître-assistant (B)	U. Laghouat (Examinateur)
Mr. M.L.BENSAAD	Maître-assistant (A)	U. Laghouat (Rapporteur)

ANNÉE UNIVERSITAIRE 2010/2011

Dédicace

A mes très chers parents qui ont toujours été là pour moi et qui m'ont donné un magnifique modèle de labeur et de persévérance. J'espère qu'ils trouveront dans ce travail toute ma reconnaissance et tout mon amour.

A mes chers frères et sœurs : Mohamed, Asma, Ahmed, Fareh et le petit Mehdi.
A mes neveux Mustapha et Yacine.

A mes meilleures amies.
Je dédie ce mémoire

Remerciements

Je tiens à remercier Mr Lahcen BENSAAD, pour avoir accepté d'être rapporteur de mon travail et pour leurs commentaires et critiques constructives qui m'ont apporté un regard éclairant sur les contributions de ce mémoire.

Je remercie très sincèrement Mr DJOUDI, pour l'honneur qu'il m'a fait en présidant le jury de ma mémoire.

Je remercie également *l'ensemble des membres du jury* « Mr GUELLOUMA, et Mr OULED JDID » *qui me font le grand honneur d'avoir accepté de juger mon travail.*

Je voudrais remercier tous mes amies et collègues pour leur aide et leur amitié.

Résumé

Les protocoles de traversée le NAT sont des protocoles qui assurent la traversée des connexions de type Peer-to-Peer les routeurs NAT. La traverse de NAT pose un problème car il change l'adresse IP/port privée vers une différente adresse IP/port publique.

La connexion Peer-to-Peer ne peut pas traverser le NAT car ce dernier change l'adresse IP privée de clients vers une adresse IP publique et les clients ne savent pas les adresses IP publique. La solution est utiliser le protocole de traversée de NAT qui assure à cette connexion de traverse le NAT.

Dans ce mémoire, nous nous intéressons d'étudier trois protocoles de traversée le NAT(ICE, TURN et STUN) avec détail puis on fait une comparaison entre ces protocoles.

Mots clé: ICE, TURN, STUN.

Abstract

The NAT traversal protocols make sure that the Peer-to-Peer connection traverses the NAT. The problem is that the NAT changes the source IP address/port from private IP address/port to public IP address/port address.

The Connection Peer-to-Peer can not pass through the NAT because it changes the private IP address of clients to a public IP address and the clients can't know the IP public addresses the solution is to use protocol of NAT traversal .

In this work, we interested to study three traversal protocols ICE, TURN and STUN then we compare between them.

Keywords : ICE, TURN, STUN.

Sommaire

Liste des figures

Liste des tableaux

Glossaire des acronymes

D

DNS	Domain Name System
DoS	Deny of service

H

HTTP	Hypertext Transfer Protocol

I

ICE	Interactive Connectivity Establishment
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6

N

NAT	Network Address Translation

R

RTCP	Real-Time Control Protocol
RTP	Real-Time transport Protocol

S

SDP	Session Description Protocol
STUN	Simple Traversal of UDP Trough NAT

T

TCP	transmission control Protocol
TLS	Transport Layer Security
TURN	Traversal Using Relay NAT

U

UDP	User Datagram Protocol

Introduction générale

Introduction générale

Le nombre d'adresses dans IPV4 est de ~4 000 000 000 mais ce grand nombre est insuffisant pour toutes les machines dans le monde, le NAT est une solution qui permet à toutes les machines d'utiliser une adresse IP pour se connecter avec internet. L'idée du NAT est que chaque groupe de machines qui appartient à un réseau privé a une seule adresse IP globale unique et chaque machine a une adresse IP privée différente, mais la question qui se pose, est comment toutes les machines doivent se connecter à internet en utilisant la même adresse IP globale ?

La réponse est d'utiliser des ports différents. Le NAT travaille parfaitement avec tous les types de communication sauf les communications (Peer-to-Peer). Ces dernières posent un grand problème car il existe de multiples adresses (adresse privée et publique) de chaque machine et le client peut avoir juste l'adresse IP publique. Des protocoles sont créés pour résoudre ce problème, ces protocoles consistent à utiliser un serveur Rendez-vous connecter directement à internet, le serveur fournit toutes les informations nécessaires pour que deux clients puissent communiquer.

Ce mémoire est composé de trois chapitres : le premier chapitre est une introduction de NAT. Il explique l'IPv4 (Internet Protocol version 4) et l'IPv6 (Internet Protocol version 6), il contient aussi la définition du NAT, variation de NAT, le pare-feu, le problème de communication Peer-to-Peer de traversé de NAT et la technique UDP Hole Punching. Le deuxième chapitre détaille les protocoles de traversé de NAT (TURN, STUN, ICE), qui aident la connexion Peer-to-Peer de traversé le NAT. Enfin nous présentons dans le troisième chapitre une comparaison entre les trois protocoles en se basant sur les avantages et les inconvénients de chacun d'eux.

Chapitre 01

Introduction du NAT

1. Chapitre 1

1.1.Introduction

L'internet utilise une architecture d'adresse : chaque nœud a une adresse IP globale unique, les nœuds peuvent communiquer directement avec tous les autres nœuds. Mais avec la pénurie d'adresse IP, on ne peut pas appliquer cette architecture pour toutes les machines dans le monde, le NAT propose une solution pour que tous les ordinateurs peuvent utiliser des adresses IP globales pour communiquer avec internet.

L'idée de NAT, est de trouver toutes les machines derrière un routeur NAT avec une adresse IP globale unique et chacune de ces machines a une adresse IP privée.

Mais l'utilisation de NAT pose une difficulté dans les communications Peer-to-Peer, la méthode la plus effective pour établir des communications Peer-to-Peer entre deux réseaux privées est le Hole Punching.

La nouvelle méthode d'adressage IPv6 peut diminuer l'utilisation de NAT à long terme. Dans un court terme, IPv6 met la demande de NAT en croissance, car le NAT fournit une méthode simple pour les communications entre les deux domaines d'adresse IPv4 et IPv6.

Ce chapitre a pour but de présenter l'adressage IP, le NAT, les types de NAT, les variétés de NAT, le problème de traverser du routeur NAT, pare-feu et UDP Hole Punching.

1.2.IPv4

La plupart du trafic internet se fait au moyen du protocole IPv4 qui est la première version de protocole internet (IP), dans ce protocole il y a 4 milliards adresses, mais ce nombre ne suffit pas.

Il existe 5 classes d'adresses IP

Classe A : 1.xxx.xxx.xxx .. 126.xxx.xxx.xxx

Tableau 1 Classe A

0XXX XXXX	**XXXX XXXX**	**XXXX XXXX**	**XXXX XXXX**
Octet 1	Octet 2	Octet 3	Octet 4
Partie réseau			

Classe B : 128.xxx.xxx.xxx ..191.xxx.xxx.xxx

Tableau 2 Classe B

10XX XXXX	**XXXX XXXX**	**XXXX XXXX**	**XXXX XXXX**
Octet 1	Octet 2	Octet 3	Octet 4
Partie réseau			

Classe C : 192.xxx.xxx.xxx …223.xxx.xxx.xxx

Tableau 3 Classe C

110X XXXX	XXXX XXXX	XXXX XXXX	XXXX XXXX
Octet 1	Octet 2	Octet 3	Octet 4
Partie réseau			

Classe D : 224.xxx.xxx.xxx …. 239.xxx.xxx.xxx
Il s'agit d'une zone d'adresses dédiées aux services de multidiffusion vers des groupes d'hôtes (1)

Tableau 4 Classe D

1110 XXXX	XXXX XXXX	XXXX XXXX	XXXX XXXX
Octet 1	Octet 2	Octet 3	Octet 3
Adresse multidiffusion			

Classe E : 240.xxx.xxx.xxx 255.xxx.xxx.xxx
Il s'agit d'une zone d'adresses réservée aux expérimentations. (1)

Les solutions de pénurie d'adresse

Il existe deux solutions pour résoudre le problème de pénurie d'adresse : NAT et IPv6,

1- IPv6 est une solution en cours de développement. elle contient un grand nombre d'adresses IP.
2- NAT est une ancienne solution pour minimiser un nombre d'adresses IP publique alors qu'un ensemble de machines a une seule adresse publique.

1.3.IPv6

IPv6 est un système de numérotage qui assure un grand nombre d'adresse qu'IPv4, il répond au besoin d'adresse IP du monde.

Tableau 5 IPv4, IPv6

	Version 4 du protocole Internet (IPv4)	Version 6 du protocole Internet (IPv6)
Déploiement	1981	1999
Espace d'adresse	Numéros de 32 bits	Numéro de 128 bits
Format d'adresse	Notation décimale à point : 192.149.252.76	Notation hexadécimale : 3FFE :F200:0234:AB00:0123:4567:8901 :ABCD
Notation préfixée	192.149.252.76	3FFE :F200:0234:AB00:0123:4567 :8901 :ABCD
Nombre	2^{32} = ~4 000 000 000	2^{128} =~340 000 000 000 000 000 000 000 000 000 000 000 000

Les avantages d'IPv6 :

- Espace d'adressage plus grand (16 octets)
- Routage plus efficace (entête simplifiée, plus d'options)
- Marquage des flux particuliers (QOS, priorités, etc.)
- Sécurité (authentification et intégrité) (2)

1.4.NAT (Network Adresse Translation)

(Traduction d'adresse réseau)

Un problème major dans le protocole IPv4 est la pénurie d'adresses de l'internet, le NAT est une solution de cours terme jusqu'à l'implémentation du protocole IPv6 ; le principe général de NAT : un sous-ensemble de machines d'un réseau privée utilise une seule adresse IP globale pour connecter à l'internet.

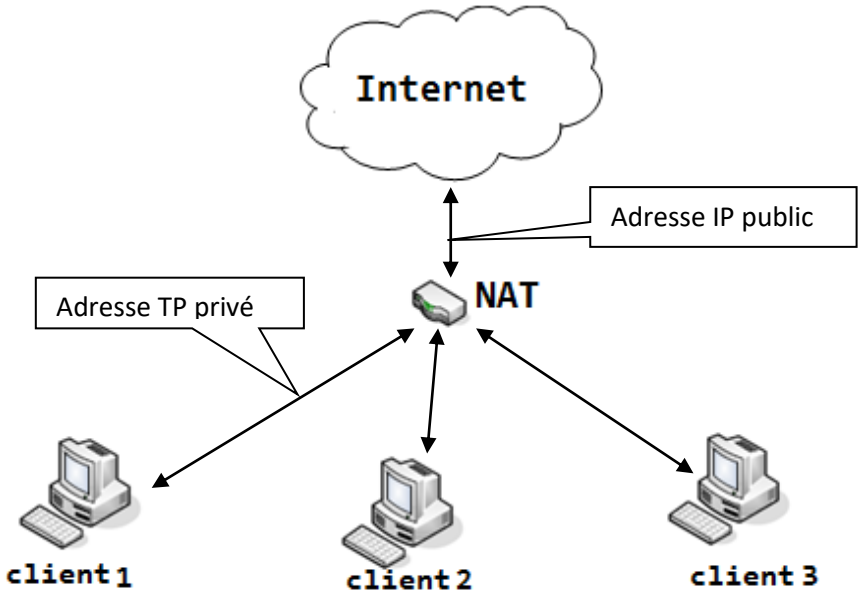


Figure 1 NAT Network Adresse Translation

Il existe deux types de NAT :

- NAT statique
- NAT dynamique

1.4.1. NAT statique :

Chaque adresse IP privée a une adresse IP publique ; ce NAT est dit statique car l'association d'une adresse interne vers une adresse externe est statique (la première adresse IP interne vers une première adresse IP externe), il permet seulement à un seul ordinateur d'accéder à internet.

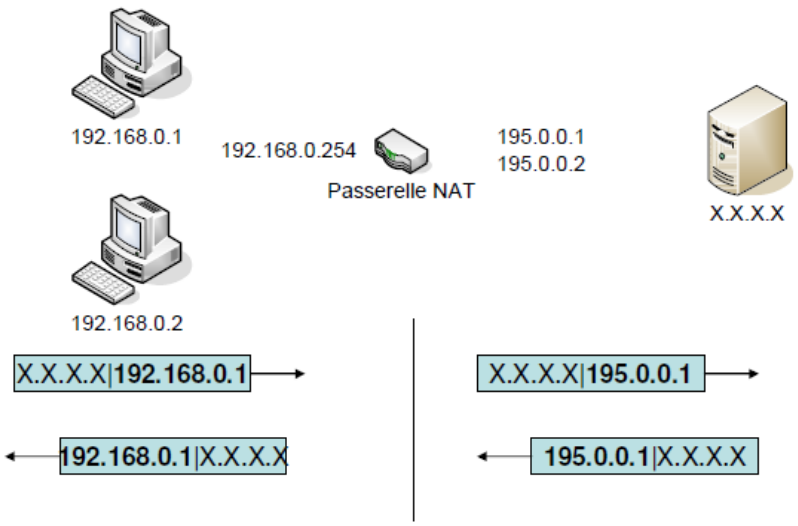


Figure 2 NAT statique

Avantage :

- Adresse IP privée peut accéder à l'internet
- Les adresses privées sont masquées

Inconvénient :

- Le problème de punir d'adresse IP n'est pas résolu (3)

1.4.2. NAT dynamique

Chaque sous réseau peut contenir des adresses privées et une seule adresse IP publique; le NAT dynamique permet à plusieurs ordinateurs d'un réseau d'accéder à l'internet via une seule adresse IP publique; NAT dynamique fait l'association entre adresse IP privée et le changement de port UDP, TCP.

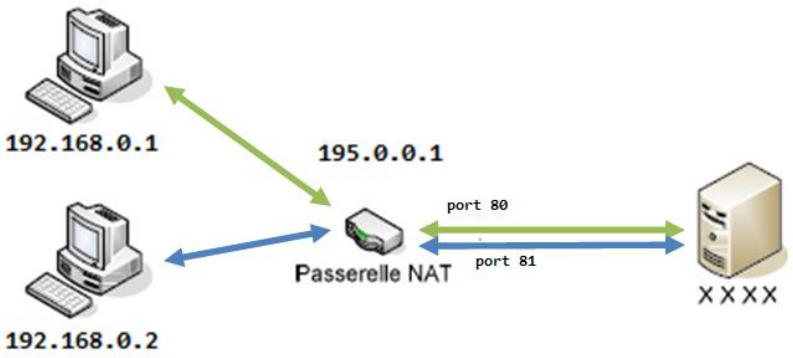


Figure 3 NAT dynamique

Avantages :

- Utilisation moins d'adresse IP.
- Sécurité

Inconvénients :

- Machines internes non joignable depuis internet.
- Problème de modification des adresses et des ports avec les protocoles :FTP , DNS, DHCP , IPSec, …) (3)

1.5.Variation de NAT :

1.5.1. Le NAT à Cône plein (Full-cone NAT)

Il consiste simplement à remplacer l'adresse IP privée de la source, par une adresse IP publique du routeur. Un hôte externe peut envoyer un paquet vers l'hôte interne, par l'envoi vers l'adresse externe (4)

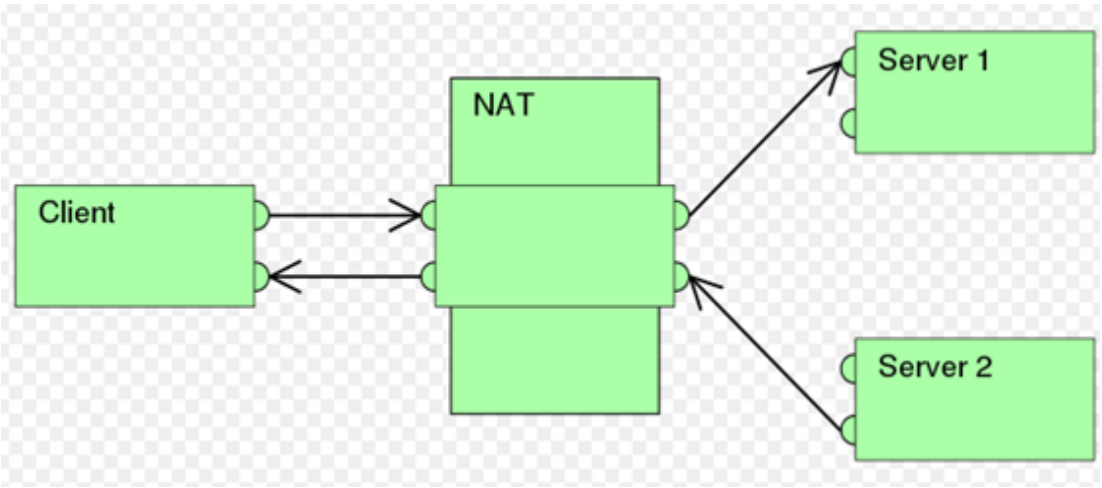


Figure 4 NAT à cône plein

1.5.2. Le NAT à Cône restrictif (restricted cone NAT)

Le principe est le même que le NAT à cône plein, la seul différence entre eux est : l'hôte externe ne peut envoyer un paquet vers l'hôte interne sauf si l'hôte interne a déjà envoyé un paquet vers cet hôte externe (4)

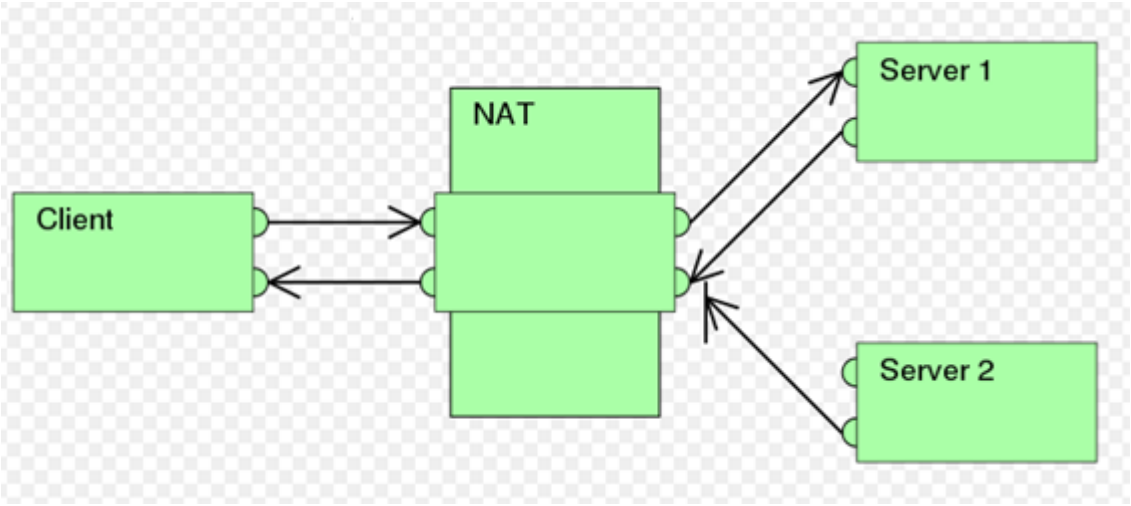


Figure 5 NAT à cône restrictif

1.5.3. NAT à Cône restrictif sur les ports (Port-restricted cone NAT)

Il a le même principe que le NAT à cône restrictif, mais la restriction inclut le nombre de ports. Un hôte externe peut envoyer un paquet avec l'adresse IP source X et port source P vers l'hôte interne si et seulement si l'hôte interne a déjà envoyé un paquet vers l'adresse X et port P (4)

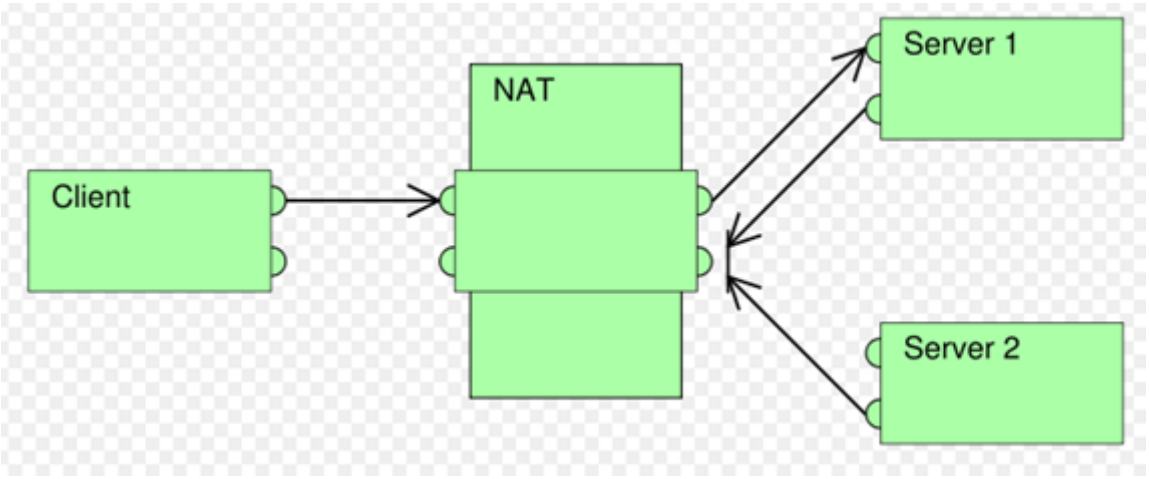


Figure 6 NAT à cône restrictif sur les ports

1.5.4. NAT Symétrique

Il consiste à remplacer l'adresse IP privée de la source, par une adresse IP publique du routeur. Si le même hôte envoie un paquet avec la même adresse et le même port source, vers une autre destination, différents mappages est utilisé. Juste l'hôte externe qui reçoit un paquet peut envoyer paquet UDP vers hôte interne. (4)

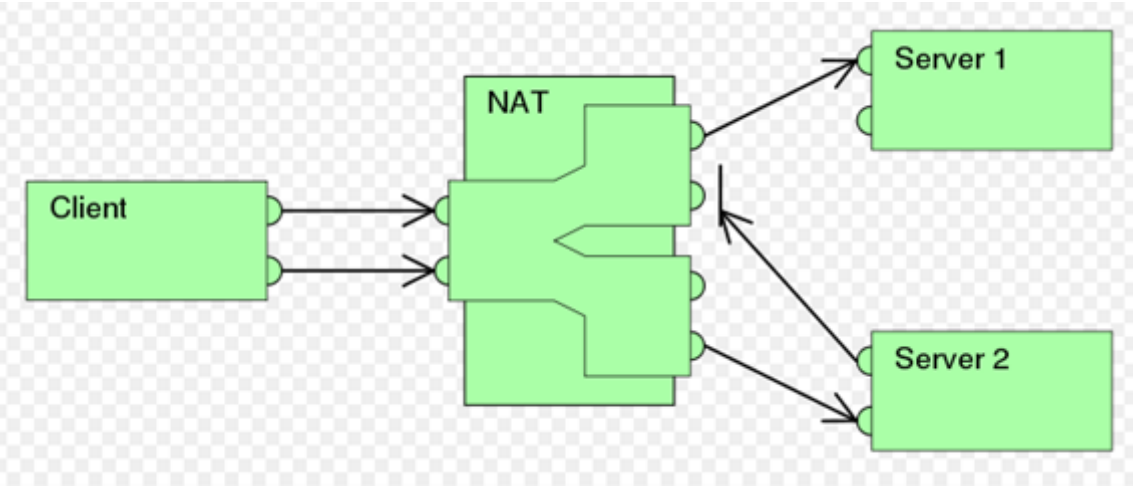


Figure 7 NAT Symétrique

1.6.Pare-feu (firewall)

Un Pare-feu peut être logiciel ou matériel il permet de contrôler et de gérer les paquets entrants et sortants et de sécuriser un réseau interne.

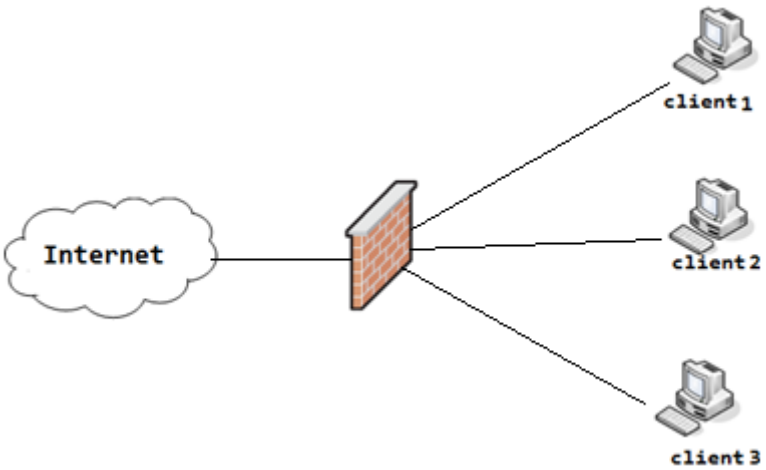


Figure 8 pare-feu

1.6.1. Catégories de pare-feu

Pare-feu sans état

C'est le plus ancien pare-feu, introduit sur les routeurs, Il compare chaque paquet indépendamment avec une liste de règles (5)

Pare-feu à états

Les pare-feux à états vérifient l'état de chaque paquet (la suite du précédent paquet) et ils vérifient aussi la réponse à un paquet dans l'autre sens (Ils vérifient si les paquets de connexion en cours sont conformes). (5)

Pare-feu applicatif

Les pare-feux applicatifs vérifient la conformité de tous les paquets à un protocole par exemple vérifier les paquets HTTP passe par le port TCP 80. Ce traitement prend grand temps de calcul et débit important. (5)

Pare-feu identifiant

L'administrateur définit des règles de filtrage, les pare-feu identifiants réalisent l'identification des connexions qui traversent le filtre IP. (5)

Pare-feu personnel

Un pare-feu personnel est installé sur une machine de travail. Il est contre les virus informatique et contre les logiciels espions. (5)

1.7.Communication Peer-to-Peer qui traverse le NAT

Le problème de la traverse du NAT pour les communications Peer-to-Peer est résolu par plusieurs techniques, les plus simples techniques sont relais, connexion inverse et Hole Punching. (6)

Relais :

La pluparts des méthodes de communication Peer-to-Peer de la traverse du NAT fait la communication comme client/serveur. Si deux clients veulent communiquer entre eux, ils utilisent un serveur pour relier les messages. Les relais travaillent toujours tant que les

clients peuvent connecter avec le serveur. C'est un inconvénient: consommation d'énergie et de bande passante. (6)

Connexion inverse :

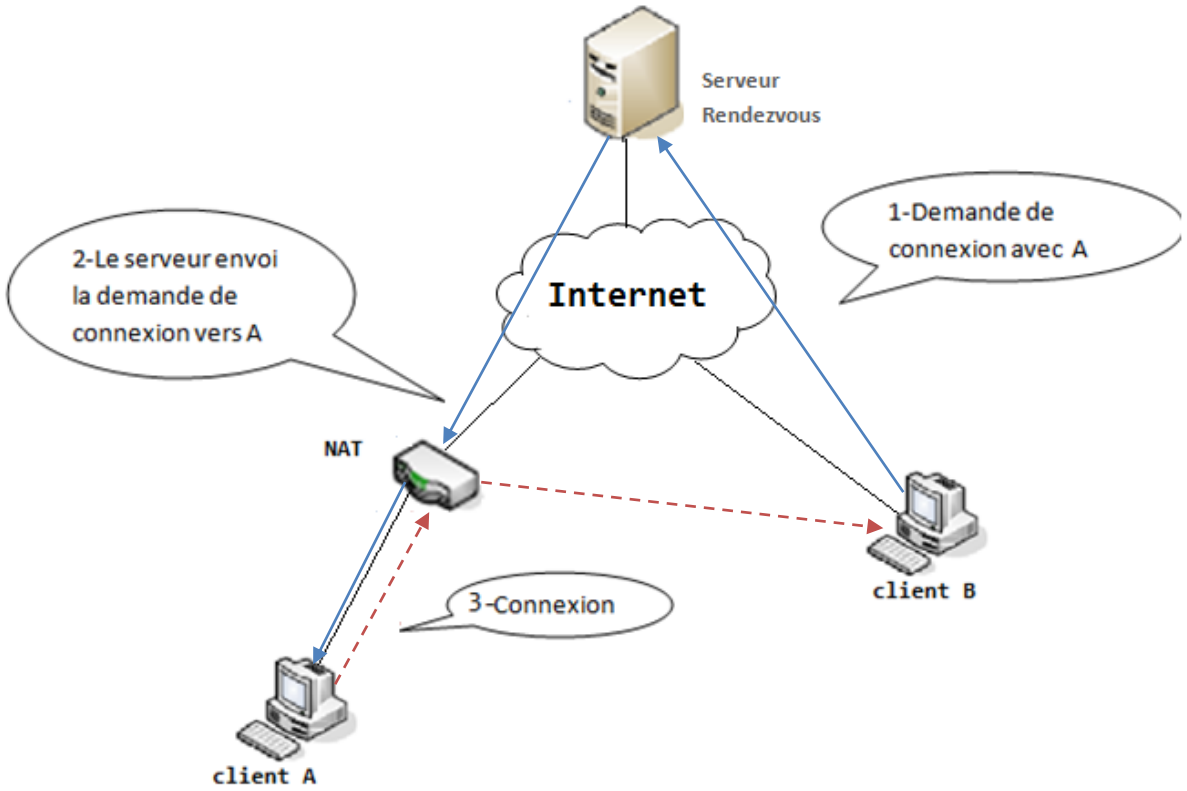


Figure 9 Communication Peer-to-Peer

Pour fournir une communication entre deux clients A et B et l'un de ces clients derrière un NAT(A) :

- Si le client A veut commencer la connexion avec le client B, la connexion démarre directement et automatiquement, parce que le client B est connecté directement à l'internet et le NAT interprète la connexion comme session sortante.
- Si le client B veut commencer la connexion avec le client A, le NAT bloque toutes les connexions arrivant vers le client A. Alors il faut que le client B envoie la demande de connexion vers le client A utilisant un serveur de Rendez-vous. (6)

1.8.UDP Hole Punching

UDP Hole Punching permet à deux clients derrière NAT de démarrer une connexion UDP Peer-to-Peer avec l'aide d'un serveur Rendez-vous.

UDP Idle timeout :

Des techniques utilisées pour déterminer la vie de session UDP qui traverse le NAT. La plupart de NAT associe un Idle timer avec les translations UDP pour fermer le Hole, s'il n'y a pas de trafic dans une période donnée. Pour ne pas arrêter la connexion, il faut envoyer périodiquement paquet ' keep-alive '. (6)

Serveur Rendez-vous :

Les informations de connexion (Endpoint) est l'adresse IP/port interne et l'adresse IP/port externe.

Hole Punching utilise un serveur Rendez-vous pour créer une connexion UDP Peer-to-Peer. si les deux clients ont déjà activés session UDP avec le serveur de Rendez-vous :

- ces clients sont enregistrés avec le serveur
- Le serveur enregistre les informations de connexion.
- La première paire (adresse IP, port) est l'information de connexion privée et la deuxième paire est l'information de connexion publique, le serveur peut obtenir l'information de connexion privée du client car il existe dans le message de registration, et peut obtenir l'information de connexion publique dans l'entête d'IP et UDP de ce message de registration.
- Si le client n'est pas derrière un NAT alors les deux informations de connexion sont les mêmes. (6)

1.8.1. Les clients derrière un seul NAT

On suppose qu'un client A veuille établir une directe session UDP avec le client B (les deux clients situés dans le même réseau privée):

1- Le client A ne connait pas comment arriver à client B, A demande de l'aide au serveur S pour établir une session UDP avec B. (envoi demande de connexion)
2- Serveur S répond à A avec un message contenant les informations de connexion de B privées et publiques. En le même temps S envoie un message vers B contenant une demande de connexion et les informations de connexion de A publiques et privées.
3- Quand le client A reçoit les informations de connexion de B, il commence l'envoi des paquets UDP vers les deux informations de connexion de B, et il observe qu'elle est la première information qui réalise une valide réponse.
4- Quand B reçoit les informations de A, B commence l'envoi des paquets UDP et observe quelle est la première information qui travaille.
5- Connexion directe plus rapide que la connexion qui traverse le NAT

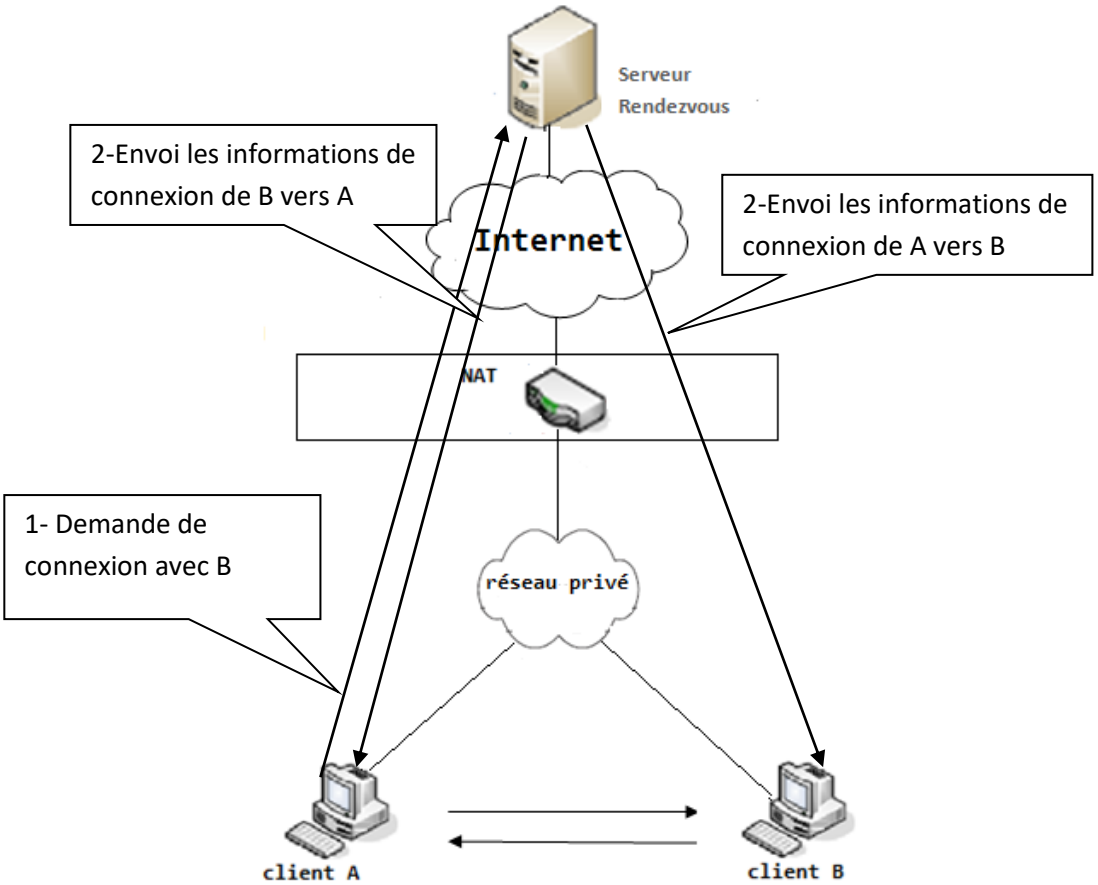


Figure 10 Hole punching

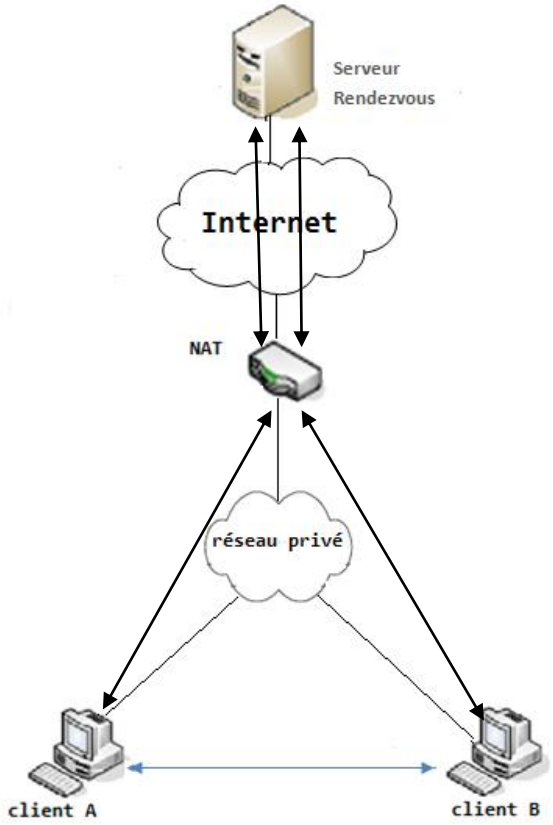


Figure 11 aprés Hole Punching

1.8.2. Les clients derrière multiple niveau de NAT

Dans ce cas chaque client (A ou B) a une adresse IP publique et adresse IP privée, dans le cas de communication Peer-to-Peer :

- A envoie un message de demande de connexion avec le client B à l'aide de serveur S qui contient l'adresse IP privée et adresse publique de A
- S envoie au même temps les informations de connexion publique et privée de B vers A et envoie les informations de connexion publique et privée de A vers B
- A et B commencent à envoyer du datagramme UDP vers ces adresses.
- A envoie le premier message qui traverse le NAT du réseau privée de A, NAT enregistre que ce paquet est le premier paquet UDP qui sort
- Si le message de A est reçu par le NAT de B avant le premier message de B vers A, alors NAT de B peut annuler le message A
- Le premier message de B vers A ouvre un' Hole' dans le NAT de B
- Quand les premiers messages entre A et B traversent leurs NAT, les Holes sont ouverts et la communication UDP peut démarrer normalement
- Quand le client vérifie que la publique information de connexion travaille, il arrête l'envoi des messages vers alternatif réseau privée (6)

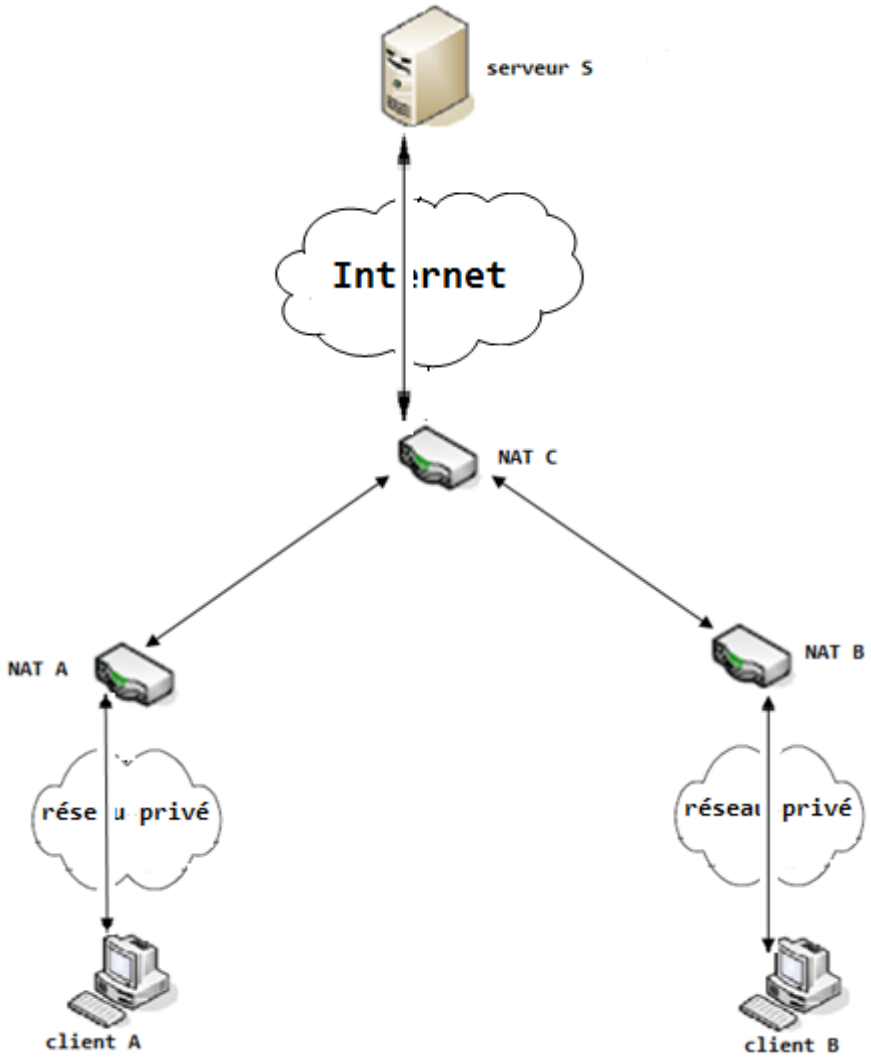


Figure 12 les clients derrière multiple niveau de NAT

Les adresses IP publique du NAT A et du NAT B sont des adresses privées de NAT C, quand un client veut contacter avec le serveur S, il crée une adresse IP privée/publique pour traverser NAT A et NAT B, NAT C établit une translation privée/publique pour chaque session.

Client A envoie un message vers l'adresse publique de NAT B. le client B envoie un message vers l'adresse publique de NAT A. Chaque client ne peut pas connaitre l'adresse de l'autre car le serveur S connait juste l'adresse IP globale de clients. Donc les clients n'ont aucun choix que l'utilisation des adresses publiques pour la communication Peer-to-Peer.

1.9.Conclusion

Comme nous avons vu dans ce chapitre, le problème de communication Peer-to-Peer qui traverse le NAT est résolu par des techniques comme UDP Hole Punching, cette technique utilise le modèle client /serveur pour traverser le NAT, nous avons vu aussi si le client est derrière un NAT ou les deux clients sont derrière multiple NAT.

Dans le chapitre suivant, nous allons étudier trois protocoles (TURN, STUN, ICE) qui résolvent le problème de Peer-to-Peer.

Chapitre 02 Les protocoles de traversée le NAT

Chapitre 02

Les protocoles de traversée le NAT

1.1. Introduction

Il est impossible que deux clients situés dans différents réseaux privés (situé derrière différent NAT) établissent une connexion Peer-to-Peer directe. Cette connexion ne peut pas traverser le NAT car il utilise deux adresses (adresse privée et adresse publique) et peut être un ou les deux clients derrière multiple NAT. Ceci rend la traversée du NAT plus difficile. La solution est l'utilisation du protocole de traversée le NAT.

Dans ce chapitre, nous allons détailler quelque protocoles de traversée le NAT qui sont TURN, STUN et ICE. Ces protocoles résolus le problème de connexion Peer-to-Peer qui traverse le NAT. Tous ces protocoles utilisent le modèle client/serveur, ils utilisent le serveur comme un relais entre les Peers.

Le protocole TURN permet aux Peers derrière des NAT de créer des connexions entre eux en passant par un serveur de relais.

Le protocole STUN est un protocole qui détermine l'adresse de transport (adresses et ports). Il permet de traverser les routeurs NAT et de connaitre l'adresse IP et port publique d'un Peer situé dans un réseau privée pour fournir une communication UDP. Le client envoie des messages à un serveur et la réponse de ce dernier est l'adresse IP et le Port de Peer à communiquer.

Le protocole ICE consiste à intégrer le protocole STUN et le protocole TURN pour déterminer toutes les connexions possibles entre deux postes.

1.2. TURN

Dans certaines situations, Il est impossible pour deux clients derrière des routeurs NAT de communiquer directement (Peer-to-Peer). Dans ce cas on utilise un protocole qui s'appelle TURN (Traversal Using Relay around NAT) qui permet aux clients de communiquer entre eux avec l'utilisation d'un relais (serveur) qui est connecté directement à l'internet.

Le client TURN est connecté avec un réseau privé et il traverse un ou plusieurs NAT vers l'internet. Il utilise un serveur comme un relais pour envoyer et recevoir les paquets de l'autre paire (Peer)

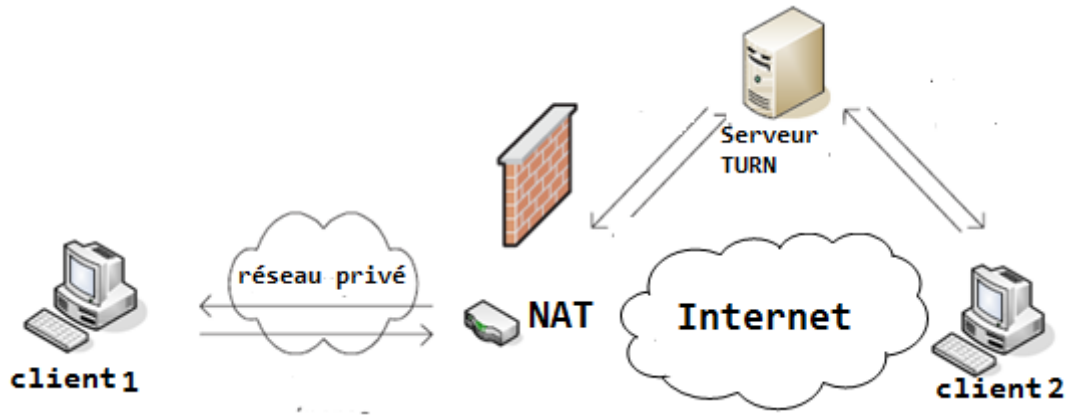


Figure 13 TURN Traversal Using Relay around NAT

1.2.1. Transport

Dans les applications Peer-to-Peer, on utilise des paquets UDP entre les paires (Peers), mais le protocole TURN permet l'utilisation d'UDP, TCP, TLS (Transport Layer Security) over TCP pour transmettre les messages entre le client et le serveur. (7)

Les clients utilisent des paquets UDP dans les communications Peer-to-Peer.

Pourquoi TURN supporte TCP et TLS-over-TCP ? La réponse c'est que :

- TURN supporte TCP parce qu'ils existent des pare-feu configurés pour bloquer les paquets UDP. Dans ce cas, on utilise TCP
- TURN supporte TLS-over-TCP entre le client et le serveur car TLS fournit des propriétés de sécurité supplémentaires (TLS fournit une méthode pour le client connecter avec le correct serveur) (7)

1.2.2. Allocation

L'allocation consiste de :

- Adresse de transport relayé
- 5-tuple (Adresse IP de client, port de client, adresse IP de serveur, port de serveur, protocole de transport)
- L'information d'authentification (nom d'utilisateur, mot de passe, domaine et nonce)

- Le temps d'expiration (c'est le temps restant en seconde jusqu'à l'expiration de l'allocation. Chaque allocation ou actualisation de transaction régle cette minuterie, elle descend vers 0, chaque allocation ou actualisation de transaction régle la minuterie par Lifetime par défaut 600s (10 minutes), le client peut demander une différente valeur)
- Une liste de permission
- Une liste de canaux pour les liaisons Peer (7)

1.2.2.1. Créer une allocation

Les étapes de créer une allocation sont :

Envoyer une demande d'allocation

Le client forme une demande d'allocation comme suivant :

- Il prend l'adresse de transport.
- Il prend un protocole de transport (UDP, TCP, TLS-over-TCP) pour l'utiliser entre le client et le serveur.
- Il prend l'adresse de transport du serveur.
- Il ajoute l'attribut de demande-de-transport (Requestes-Transport) dans la demande.
- Si le client veut initialiser le champ de temps d'expiration de l'allocation avec une valeur différente de la valeur par défaut, alors le serveur peut inclure un attribut Lifetime spécifiant sa valeur désirée.

Quand il termine, il envoie la demande vers l'adresse de transport de serveur (7)

Recevoir la demande d'allocation

Quand le serveur reçoit la demande il fait les vérifications suivantes :

- L'authentification avec le mécanisme Long-Term
- Il vérifie si le 5-tuple est le même de l'allocation existant. Si oui le serveur rejette la demande.
- il vérifie si la demande contenant l'attribut de demande-de-transport. Si l'attribut de demande-transport n'existe pas ou est mal formée, le serveur rejette la demande.

Si toutes les vérifications passent, le serveur crée l'allocation. (7)

Recevoir la réponse de succès d'allocation

Si le client reçoit une réponse succès, alors :

- il vérifie si l'adresse mappée et l'adresse de transport relayée dans les adresses que le client comprend, si le client ne comprend pas ces deux adresses, il doit supprimer l'allocation.
- il Crée leur copie de l'allocation pour suivre le serveur.
- il mémorise le Lifetime et le 5-tuple qui est utilisé dans la demande, le nom d'utilisateur et le mot de passe utilisé pour l'authentification de demande. le client a besoin de suivre les canaux et les permissions. (7)

Recevoir réponse d'erreur d'allocation

Si le client reçoit une réponse d'erreur d'allocation alors :

Request timed out: le problème existe dans le serveur ou avant de parvenir au serveur. Le client considère que la transaction ayant échoué mais peut-être qu'il choisit de renvoyer la demande d'allocation avec un différent protocole de transport (UDP, TCP). (7)

Les réponses d'erreur

400 (Bad Request) : le serveur suppose que la demande de client est malformée. Le client considère la transaction échouée. Le client peut informer l'utilisateur et ne doit pas renvoyer la demande vers ce serveur jusqu'à ce qu'il sache que ce problème est fixé. (7)

401 (Unauthorized) : si le client suit les procédures de mécanisme de Long-term et il reçoit cette erreur a chaque instant, alors le serveur n'accepte pas le client. Dans ce cas, le client considère que la transaction courante a échoué. Il doit informer l'utilisateur. Il faut que le client n'envoie pas de demandes à ce serveur, jusqu'à ce qu'il sache que ce problème est fixé. (7)

403(Forbidden) : la demande est valide, mais le serveur refuse de l'effectuer, de raison administrative. Le client considère que la transaction a échoué. Le client peut informer l'utilisateur, il faut que le client n'envoie pas de demandes à ce serveur, jusqu'à ce qu'il sache que ce problème est fixé. (7)

441(Wrong Credentials) : Il faut que le client ne reçoive pas cette erreur dans la réponse de demande de l'allocation. Le client peut informer l'utilisateur et ne renvoie pas la demande à ce serveur jusqu'à ce qu'il sache que ce problème est fixé. (7)

442(Unsupported Transport Address) : il faut que le client ne reçoive pas cette erreur dans la réponse de demande de l'allocation UDP. Le client peut informer l'utilisateur et ne renvoie pas la demande à ce serveur jusqu'à ce qu'il sache que ce problème est fixé. (7)

486(Allocation Quota Reached) : le serveur est incapable de créer des allocations en plus avec ce nom d'utilisateur. Le client considère la transaction a échoué. Il faut que le client atteigne une minute au minimum avant de créer des allocations dans ce serveur. (7)

508(Insufficient Capacity) : le serveur n'a pas d'adresses de transport relayé disponible, ou l'adresse qui a été réservée n'est plus disponible. Le client considère que l'opération a échoué. Il faut que le client atteint une minute au minimum avant de créer des allocations dans ce serveur. (7)

Actualiser une allocation

Actualisation de la transaction utilisée pour :

a- Actualiser une allocation déjà existante et mise-à-jour de leur temps-d'expiration.
b- Supprimer une allocation existante.

Si un client veut continuer d'utiliser une allocation, alors il faut l'actualiser avant qu'elle expire. Il faut que le client actualise l'allocation environ une minute avant qu'elle expire. Si le client ne veut pas continuer d'utiliser une allocation, alors il doit la supprimer. (7)

Les étapes d'actualiser une allocation :

Envoyer une demande d'actualisation

Si le client veut supprimer une allocation qui existe déja, il inclue un attribut Lifetime avec la valeur 0. L'actualisation de la transaction permet la mise-à-jour du temps-de-l'expiration dans une allocation. (7)

Recevoir une demande d'actualisation

Le serveur reçoit la demande d'actualisation et calcule si la demande contenant l'attribut Liftetime et la valeur de cet attribut :

- est 0, la demande est acceptée avec succès et l'allocation est supprimée.
- est plus de 0, la demande est acceptée avec succès et le temps-de-l' expiration de l'allocation est posé dans lifetime désiré. (7)

Recevoir réponse d'actualisation

Si le client reçoit une réponse de succès de demande d'actualisation avec Lifetime plus de 0, il fait la mise-à-jour de leur copie de l'allocation avec la valeur de temps-de-l' expiration existante dans la réponse. (7)

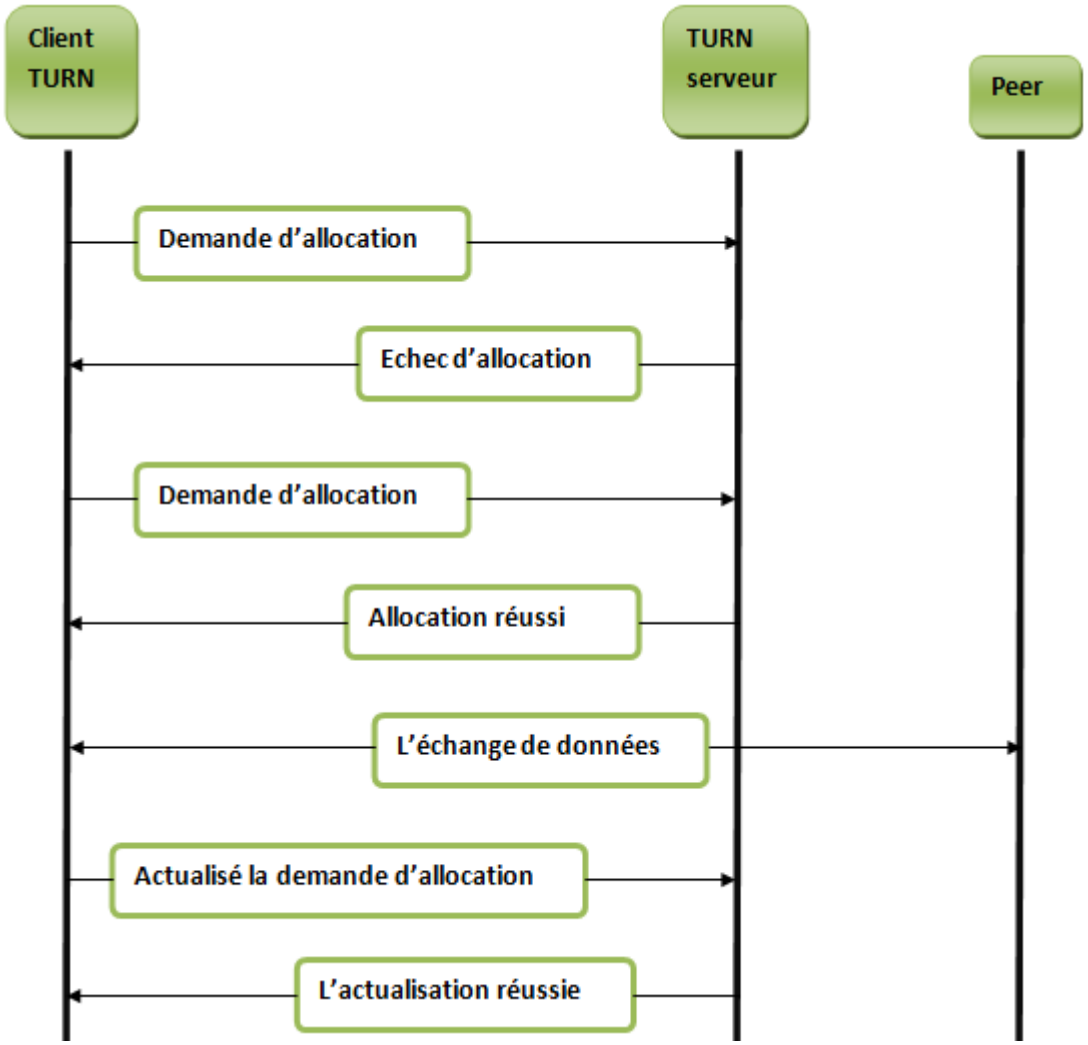


Figure 14 Allocation

1.2.3. Permission

Une allocation peut avoir 0 ou plus de permissions. Chaque permission utilise l'adresse IP et Life-time. Quand un serveur reçoit un datagramme UDP, il vérifie la liste de permissions. Si l'adresse IP source de datagramme a une permission alors, les données sont transmises au client, sinon le datagramme UDP est ignoré.

Une permission est expirée après 5 minutes (Lifetime) si elle n'est pas actualisée. Le client peut installer ou actualiser une permission avec la demande CreatPermission ou la demande ChanelBind. (7)

CreatePermission

Le client utilise la méthode de Createpermission pour installer ou actualiser les permissions dans le serveur.

Former une demande de CreatePermission

Quand le client forme la demande de Createpermission, il faut qu'il inclue au moins l'attribut de Xor-Peer-Address. Cet attribut contient l'adresse IP de la permission installée ou actualisée. Le port dans Xor-Peer-Address sera annulé.

Recevoir la demande de CreatePermission

Quand un serveur reçoit la demande de CreatePermission : Il vérifie la validité de massage. Il faut que la demande de CreatePermission contienne au moins un attribut Xor-Peer-Address. S'il n'existe aucun attribut, ou si aucun de ces attributs n'est valide, alors il envoie une réponse d'erreur de code 400 (Bad Request). Si la demande est valide mais, le serveur incapable de satisfaire la demande en raison de certaines limites, alors la réponse d'erreur de code 508 (Insufficient Capacity).

Le serveur peut imposer des restrictions sur l'adresse IP autorisé dans l'attribut Xor-Peer-Address. Si une valeur non autorisé, le serveur rejette la demande avec l'erreur 403(Forbidden).

Si le message est valide et le serveur est capable de réaliser cette demande, alors le serveur installe ou actualise la permission.

Le serveur répond avec une réponse de succès de CreatePermission.

Recevoir la réponse de demande de CreatePermission

Si le client reçoit une valide réponse de CreatePermission, le client fait la mise-à-jour de leur structure data pour indiquer que la permission a été installée ou actualisée. (7)

1.2.4. La méthode Send

Former une Send indication pour passer les données vers le serveur pour relayer avec le Peer. Il faut que le client assure qu'il existe une permission installée dans le serveur.

Quand un Send indication est formé, il faut que le client inclue un attribut Xor-Peer-Address et L'attribut Data. Xor-Peer-Address contenant l'adresse de transport de Peer, et l'attribut de Data contenant l'application actuelle de donnés. (7)

Recevoir Send Indication

Quand le serveur reçoit une Send Indication, il vérifie la validité de message. Il faut que la Send indication contenant deux attributs Xor-Peer-Address et Data. Si un de ces deux attributs n'existe pas ou n'est pas valide, alors le message est écarté.

Le serveur vérifie aussi l'existence d'une permission installée, s'il n'y a pas de permission, le message est écarté.

S'il n'y a pas d'erreur, le serveur forme Datagramme UDP comme suit :

- L'adresse de transport source est l'adresse de transport relayé de l'allocation.

- L'adresse de transport de destination existe dans l'attribut Xor-Peer-Address

Le résultat c'est le Datagramme UDP, le serveur envoie ce datagramme vers le Peer. (7)

Recevoir datagramme UDP

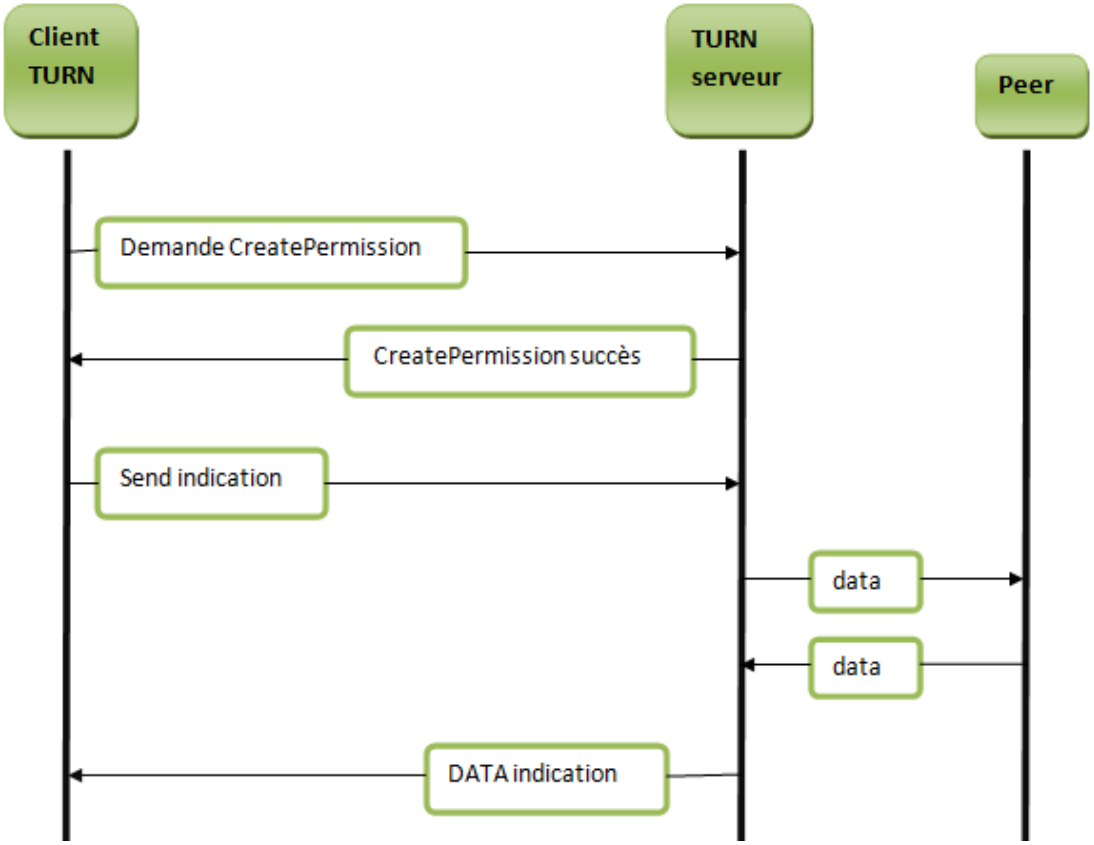
Quand le serveur reçoit le Datagramme UDP de Peer, le serveur vérifie l'allocation qui associe l'adresse de transport relayé, et il vérifie les permissions de cette allocation qui permit de relayer le datagramme UDP.

- Si ce relai est permis et le canal lié avec le Peer. Le serveur envoie le Datagramme vers le Peer.

- Si le relai est permis mais le canal n'est pas lié vers le Peer, alors le serveur forme et envoie une Data indication qui contient les attributs Xor-Peer-Address et Data.

Recevoir Data Indication

Quand le client reçoit le Data Indication, il vérifie les deux attributs (Xor-Peer-Address et Data) dans le Data indication. Le data indication sera rejetée si ces deux attributs sont absents. Si le Data indication termine la vérification, le client pose les données dans l'attribut Data. (7)



15 la méthode Send

1.2.5. Les canaux

Les canaux fournissent une méthode pour le client et le serveur pour envoyer les données en utilisant les messages ChannelData, l'entête de ce message et moins que Send et

Data indication. Le client peut lier un canal avec le Peer à tout moment pendant le Lifetime de l'allocation. Liaison d'un canal consiste à:

- Un numéro de canal.

- Adresse de transport (de Peer).

- Minuterie du temps-de-l'expiration. (7)

Envoyer la demande ChannelBind

La liaison de canal est créée ou rafraichie avec la transaction ChannelBind. Une transaction crée ou réfléchit une permission.

Pour initialiser la transaction ChannelBind, le client forme une demande de ChannelBind. Pour lier un canal spécifié dans l'attribut Channel-Number est une adresse de transport de Peer spécifiée dans l'attribut Xor-Peer-Address.

Relié un canal avec la même Adresse de transport qui est déjà lié fournit une méthode pour actualiser une liaison de canal et la permission sans l'envoi de données avec le Peer.

Recevoir la demande ChannelBind

Quand un serveur reçoit la demande de ChannelBind, il vérifie si :

- la demande contenant les deux attributs Channel-Number et Xor-Peer-Address.

- un canal n'est pas lié vers différentes adresses de transport

- L'adresse de transport n'est pas liée avec un différent numéro de canal.

Lorsqu'un de ces tests échoue, le serveur répond avec 400 (Bad Request)

Recevoir la réponse de demande ChannelBind

Quand le client reçoit une réponse succès de demande de ChannelBind, il fait la mise-à-jours de leurs data structure pour assurer que la liaison de canal est active et pour connaitre si la permission correspondante a été installée ou actualisée.

Si le client reçoit une réponse d'erreur, alors il supprime l'allocation et démarre de nouveau avec une nouvelle allocation.

Envoyer le message ChannelData

Quand le client est lié à un canal avec le Peer, il envoie les données avec le message ChannelData ou le message Send indication ; le serveur utilise le message ChannelData si le canal a été lié avec le Peer. (7)

Recevoir le message ChannelData

Si le message ChannelData est reçu dans un UDP datagramme, si le Datagramme UDP est trop court pour que la longueur de message ChannelData, alors le message est écarté

Si le message ChannelData est reçu dans le canal qui n'est pas lié avec aucun Peer, alors le message est écarté.

Il faut que le client écarte le message ChannelData s'il n'existe aucune permission vers le Peer. Il faut que le serveur n'actualise pas la liaison de canal ou la permission vers le Peer.

Dans le côté du serveur, s'il ne détecte aucune erreur, le serveur forme datagramme UDP comme suit :

- Adresse de transport source est l'adresse de transport relayé de l'allocation, l'allocation est déterminée par le 5-tuple dans le message ChannelData qui arrive.

- L'adresse de transport de destination est l'adresse de transport du canal lié.

- Les données suivant l'entête UDP sont le contenu de données de message ChannelData.

Alors le datagramme UDP est envoyé vers le Peer

L’attribut LifeTime

L’attribut LifeTime présente la duré pour que le serveur maintient une allocation dans l’absence d’une actualisation.

L’attribut Xor-Peer-Address

C'est un attribut qui spécifie l’adresse et le port de Peer

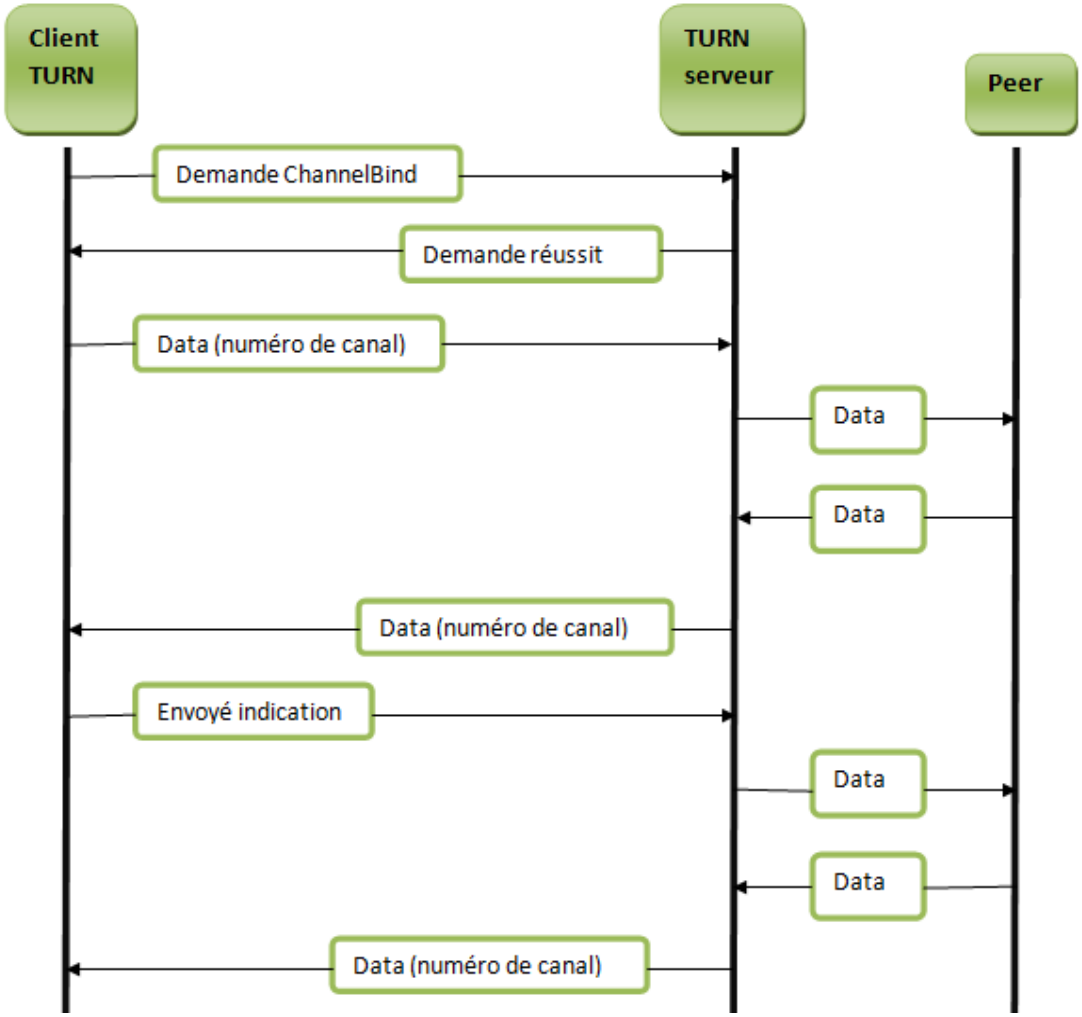


Figure 16 les canaux

Détail

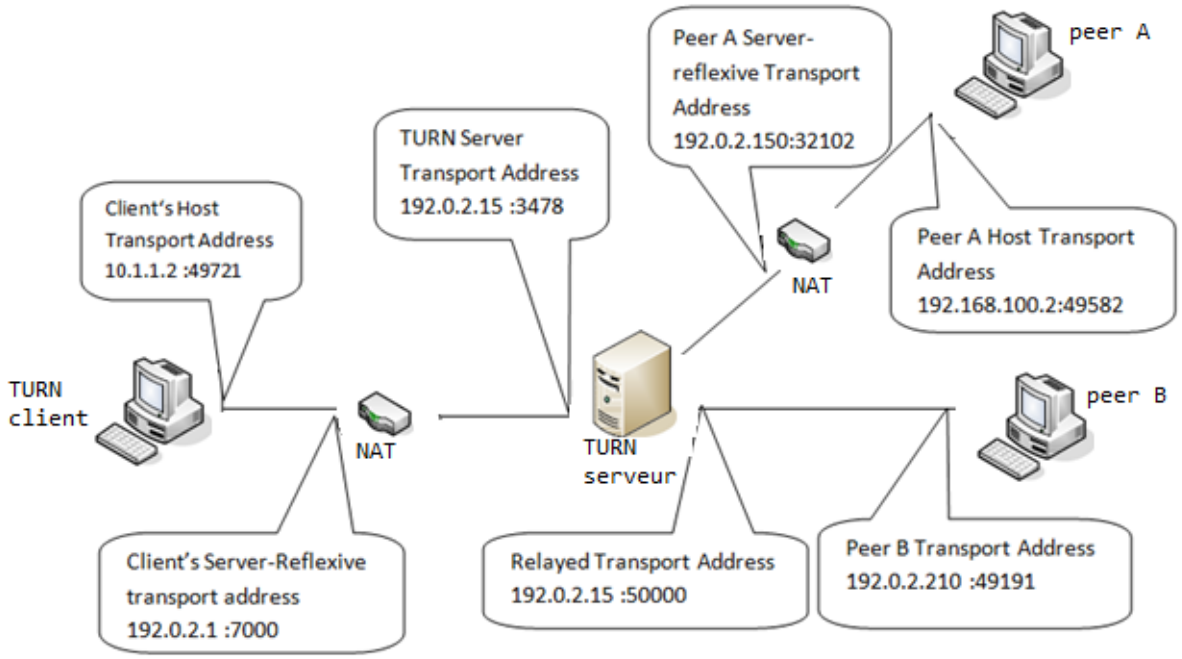


Figure 17 Les adresses

Peer A

- Le client selecte une adresse de transport pour l'hôte
- Il envoie une demande d'allocation vers le serveur dans l'adresse de transport du serveur
- IL inclue l'attribut LifeTime
- IL inclue l'attribut Requested-Transport dans la demande de l'allocation qui indique le Transport avec le protocole UDP entre le serveur et les Peers

- Le serveur demande que toutes les demandes authentifiées
- Le serveur rejette l'initiale demande car il n'est pas authentifié avec l'erreur 401(Unauthorized)
- Le client renvoie la demande de l'allocation avec l'attribut d'authentification
- Quand le serveur reçoit la demande d'allocation authentifié, il vérifie si le tout est bien formé, il crée l'allocation
- Le serveur envoie une réponse de succès allocation
- Le serveur inclut l'attribut LifeTime
- Le serveur inclut l'attribut Xor-Relayed-Address (Adresse de transport relayé de l'allocation)
- Le serveur inclut Xor-Mapped-Address (adresse réflexive de serveur sur client)

- Le client envoie la demande CreatePermission vers le serveur
- Le serveur crée une permission et envoie une réponse succès CreatePermission
- Le client envoie les données vers Peer avec Send indication
- Quand le serveur reçoit les données, il les envoie dans un datagramme UDP vers Peer
- Le Peer répond avec UDP datagramme (le Peer envoie le datagramme vers le serveur)
- Le serveur crée Data indication l'adresse source de datagramme UDP et les données de datagramme UDP
- Le serveur envoie data indication vers le client

Peer B

- Le client lié à un canal avec le Peer B et inclut l'attribut Channel-Number
- Le client utilise les informations d'authentification de la dernière demande
- Le serveur lié avec le Peer par le canal, il installe une permission pour l'adresse IP de Peer
- Le serveur envoie une réponse de succès liaison de canal
- Le client envoie le message de ChannelData vers le serveur avec des données destinées pour le Peer
- Quand le serveur reçoit le message, il vérifié si le canal est lié et envoie les données dans un datagramme UDP vers le Peer
- Le Peer envoie une Datagramme UDP vers le serveur
- Le serveur envoie un message channelData qui contient les données de datagramme UDP (7)

1.3. STUN

STUN est un protocole qui sert comme un outil pour les autres protocoles dans le NAT. un utilisateur peut utiliser STUN pour déterminer l'adresse IP et le port alloué par le NAT. On peut aussi utiliser STUN pour vérifier la connectivité entre deux utilisateurs.

STUN est utilisé pour une ou plusieurs solutions de traverser le NAT. Ces solutions appelées STUN usages (ICE, SIP Outbound).

STUN est un protocole client/serveur, le client envoie la demande vers le serveur, ce dernier envoie la réponse vers le client. Il y a deux types de demandes :

- Demande de liaison (binding request) : envoyé des paquets UDP
- Demande de secret partagé, envoi sur TLS over TCP

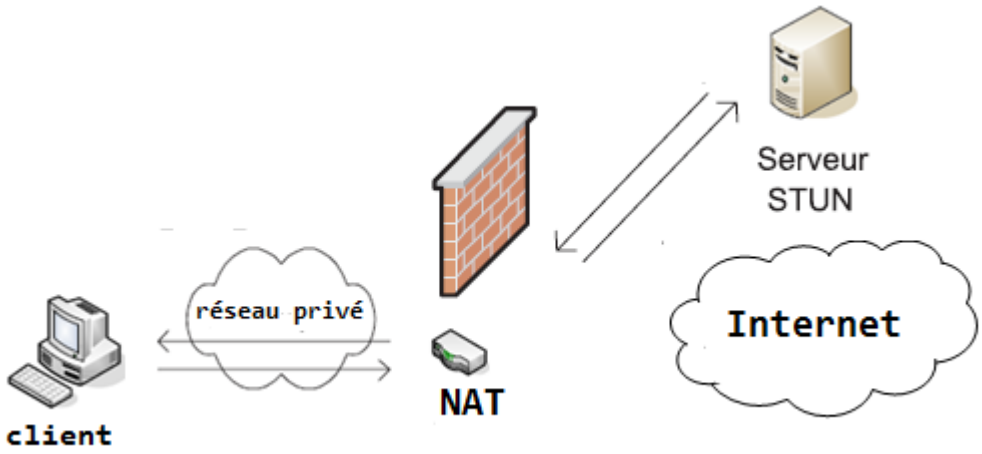


Figure 18 STUN session Traversal Utilitises for NAT

1.3.1. Méthode Binding

On peut utiliser cette méthode dans les transactions demande/réponse et l'indication. Dans la méthode Binding , le client STUN envoie la demande de liaison (binding request) vers le serveur STUN, la demande peut être passer par un ou plusieurs NAT. Quand la demande passe un NAT, ce dernier modifiera l'adresse source de transport (adresse IP, port) de paquet. Alors l'adresse IP sera l'adresse IP public et le port sera créé par le NAT plus proche de serveur, cette adresse de transport appelé « adresse de transport réflexive ». Le

serveur STUN copie cette adresse source de transport dans l'attribut XOR-MAPPED-ADDRESS dont la réponse STUN Binding et envoie la réponse vers le client STUN. Comme ce paquet traverse le NAT, le NAT modifiera l'adresse de destination dans l'en-tête IP, mais l'adresse de transport dans XOR-MAPPED-ADDRESS ne change jamais. Le client peut savoir leurs adresses de transport réflexives. (8)

Former une demande ou une indication

L'agent (client ou serveur) crée l'entête (La classe de message est une demande ou une indication). IL ajoute dans la demande l'attribut spécifie par la méthode.

Envoyer une demande ou une indication

L'agent envoie la demande ou l'indication. Peut être le message du protocole UDP perdu dans le réseau. Dans ce cas le client retransmit le message de demande et démarre avec un intervalle de temps RTO (retransmission TimeOut), cette intervalle est doublé dans chaque retransmission.

Recevoir le message STUN

Quand un agent reçoit un message, il vérifie l'entête de message et vérifie aussi que les deux premiers Octet sont 0, magic cookie a une valeur correct, la longueur de message, la classe de méthode (Succès réponse, réponse d'erreur), transaction ID. il vérifie l'utilisation de l'extension FingerPrint. Si l'agent détecte une erreur, le message sera ignoré. L'agent fait toutes les vérifications par le mécanisme d'authentification.

Après la vérification d'authentification, l'agent vérifie les attributs Unknown et Known-but-Unexpected dans le message. Les deux attributs sont ignorés par l'agent. (8)

Réponse de message de STUN

Si la réponse contenant un ou plusieurs attributs d'Unknown, le serveur répond avec l'erreur 420 (Unknown Attribute)

Former une réponse d'erreur ou succès

Quand le serveur forme une réponse (succès ou erreur), la méthode de réponse est la même comme la demande, et la classe de message est un Succes Response ou Error Response.

Dans la réponse d'erreur, le serveur ajoute l'attribut Error-Code contenant le code d'erreur.

Si le serveur authentifie la demande, alors il faut que le serveur ajoute l'approprié attribut d'authentification dans la réponse. Pour la methode Binding , il n' y a pas de vérification en plus . Quand le serveur forme une réponse succès, il ajoute l'attribut Xor-Mapped-Address dans la réponse.

Envoyer la réponse de succès ou d'erreur

Le serveur envoie la réponse vers l'adresse de transport source de message de demande

Traitement de l'indication

Si l'indication contenant l'attribut Unknowncomprehension-required, l'indication est ignorée.

L'agent fait toutes les vérifications nécessaires. Si toutes les vérifications succès, l'agent traite l'indication. Pas de réponse générée pour une indication

Traitement de réponse de succès

Si la Réponse contenant l'attribut Unknown Comprehension-Required, la réponse est ignorée et la transaction a échoué.

Si toutes les vérifications sont réussies, alors le client traite la réponse de succès.

Pour la méthode Binding, le client vérifie l'existence de l'attribut Xor-Mapped-Address dans la réponse. Le client vérifie la famille de l'adresse : si la famille d'adresse est insupportable, dans ce cas le client ignore l'attribut.

Traitement de réponse d'erreur

Si la réponse d'erreur contenant l'attribut Unknown Comprehension-Required, ou si la réponse d'erreur ne contenant pas l'attribut Error-Code, alors la transaction échoue.

Le client alors fait tous les traitements par le mécanisme d'authentification.

Le traitement dans ce point dépend sur le code d'erreur, la méthode et l'usage.

les règles par défaut :

- Si le code d'erreur est 300 vers 399, le client considère la transaction a échoué.
- Si le code d'erreur est 400 vers 499, le client déclare que la transaction a échoué
- Si le code d'erreur est 500 vers 599, le client renvoie la demande, mais il faut que le client limite le nombre de renvois. (8)

1.3.2. Le mécanisme FingerPrint

Ce mécanisme aide pour distinguer les messages STUN et les autres protocoles.

Dans chaque usage, les messages sont multiplexé dans la même adresse de transport avec des autres protocoles, comme RTP.il faut que les messages STUN soient séparés sur les applications de paquets.

Quand l'extension FingerPrint est utilisée, l'agent inclut l'attribut FingerPrint dans le message pour l'envoyer vers un autre agent. Quand l'agent reçoit le message, il vérifie l'attribut FingerPrint, cette vérification aide l'agent à détecter les messages de l'autre protocole. (8)

1.3.3. Les mécanismes d'authentification et l'intégrité de message

Il y a deux mécanismes pour fournir l'authentification et l'intégrité des messages Short-Term et Long-Term.

Mécanisme Short-Term

L'explication de mécanisme Short-Term :

Former une demande ou une indication

Il faut que l'agent inclut les attributs Username et Message-Integrity dans le message.

Recevoir la demande ou l'indication

Il faut que l'agent vérifié la liste dans l'ordre suivant :

- Si le message ne contenant pas les deux attributs Message-Integrity et Username :
    - Si le message est une demande, le serveur rejette la demande avec réponse d'erreur de code 400 (Bad Request)
    - Si le message est une indication, l'agent ignore l'indication
- Si l'attribut ne contenant pas un valide nom d'utilisateur :
    - Si le message est une demande, le serveur rejette la demande avec réponse d'erreur de code 401(Unauthorized)
    - Si le message est une indication, l'agent ignore l'indication

Chaque réponse générée par le serveur inclut l'attribut Message-Integrity

Si une vérification échoue, le serveur n'inclut pas l'attribut Message-Integrety ou Username dans la réponse d'erreur. (8)

Recevoir la réponse

Le client vérifie si l'attribut Message-Integrety existe dans le Réponse. Le client calcule l'intégrité de message de réponse avec l'utilisation de même mot de passe de demande. Si la valeur résultat est égale au contenu de l'attribut Message-Integrity, il considère que la réponse est authentifiée. Si la valeur n'est pas égale au contenu ou l'attribut Message-Integrity est absent, la réponse est ignorée.

Mécanisme de Long-Term

Long-Term est visible pour l'utilisateur, ce dernier reste en connexion jusqu'au moment où il veut s'arrêter, ou l'utilisateur a été changé (log-in Username, Password). Le client envoie une demande sans vérification d'intégrité.

Le serveur rejette cette demande, pour fournir à l'utilisateur un Realm (utiliser pour guider l'utilisateur de Username et Passeword) et le nonce (fournir la protection).

Le client renvoie la demande et il ajoute leur nom d'utilisateur Username, et Message-Integrity dans le message, et le nonce. Le serveur valide le nonce et les vérifications de l'intégrité de message. Si toutes les vérifications passent, la demande est authentifiée. Si le nonce n'est pas valide, il est considéré comme périmé, le serveur rejette la demande et il fournit un nouveau nonce. Le client revoie la demande vers le même serveur et il utilise le nonce, nom d'utilisateur, et le mot de passe. Les demandes ne sont pas rejetées jusqu'à ce que le nonce soit invalide pour le serveur, le refus fournit un nouveau nonce de client. (8)

Former une demande

- Première demande :

Le premier message est envoyé pour l'authentification et l'intégrité de message.

- Deuxième demande :

Quand la transaction demande/réponse est terminée avec succès, le client représente un Realm et nonce pour le serveur, et il sélecte le nom d'utilisateur et le mot de passe d'authentification. (8)

Recevoir la demande

Le serveur fait les vérifications suivantes en ordre:

- Si le message ne contenant pas l'attribut Message-Integrity, le serveur génère une réponse d'erreur avec le code 401(Unauthorized). La réponse inclut le

nonce sélecte par le serveur et Realm(domaine du nom qui fournit le serveur STUN)

- Si le message contenant l'attribut Message-Integrity, les attributs Username,Releam, ou nonce sont absent, le serveur génère un réponse d'erreur avec le code 400 (Bad Request)
- Si le nonce n'est pas valide, le serveur génère une réponse d'erreur avec le code 438 (Stale Nonce). Cette réponse inclut les attributs Nonce, Releam
- Si le nom d'utilisateur dans l'attribut Username n'est pas valide, le serveur génère une réponse d'erreur avec le code 401 (Unauthorized). La réponse inclut Realm, Nonce

Si toutes ces vérifications passent, la réponse générée par le serveur inclut l'attribut Message-Integrety (nom d'utilisateur et le mot de passe qui sont utilisés pour authentifier la demande). (8)

Recevoir la réponse

Si la réponse est une réponse d'erreur avec le code 401 (Unauthorized), le client renvoie une nouvelle transaction. Cette demande contenant l'attribut Username,qui est déterminée par le client (nom d'utilisateur de Realm existe dans la réponse d'erreur) et contenant les attributs Realm et Nonce et Message-Integrity

Si la réponse est une réponse d'erreur avec le code 438 (Stale Nonce), le client renvoie la demande, et il utilise le nouveau nonce qui existe dans la réponse d'erreur, la demande inclut les attributs Username, Realm et Message-Integrity.

1.3.4. Mécanisme de Alternate-Server

STUN permet au serveur de réorienter le client vers un autre serveur.

Le serveur réoriente le client vers un autre serveur d'après un message de demande avec une réponse d'erreur avec le code 300 (Try Alternate. Le serveur inclut l'attribut Alternate-Server dans la réponse d'erreur

Si le client trouve l'attribut Alternate-Server dans la réponse d'erreur, alors il considère que la transaction a échoué et il renvoie la demande vers le serveur qui est spécifié dans l'attribut. (8)

1.4. ICE

L'envoi de l'offre initiale

Le protocole ICE est une technique pour traverser le NAT.il est utilisé pour le flux média (UDP), qui est une extension de modèle offre/réponse, il fonctionne avec les adresses IP et les ports de la connexion Peer-to-Peer.

Le protocole ICE permet aux agents qui sont peut être derrière un NAT ou plusieurs NAT de découvrir les informations nécessaires de leurs topologies pour trouver un ou plusieurs chemins par lesquelles ils peuvent communiquer.

L'idée d'ICE c'est que chaque agent à plusieurs adresses de transports :

- Une adresse de transport est liée directement avec une interface réseau
- Une adresse de transport est traduite du coté publique de NAT
- Une adresse de transport est allouée pour un serveur TURN

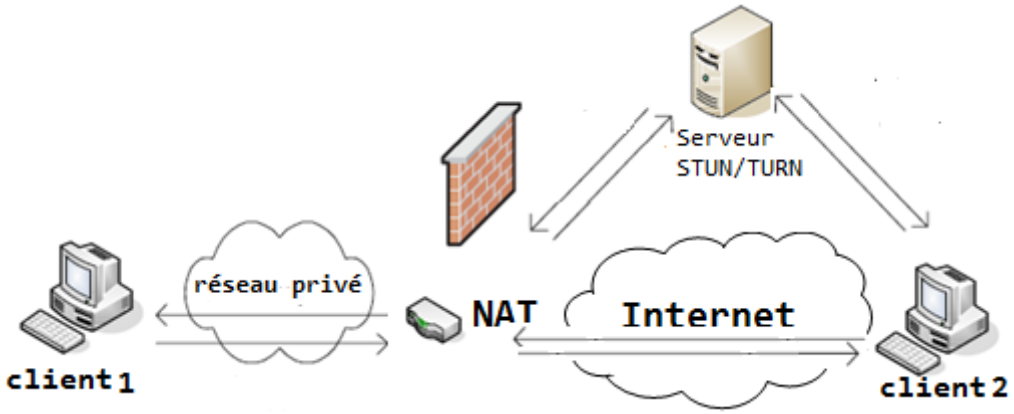


Figure 19 ICE Interactive Connectivity Etablishment

1.4.1. Collecter les candidats

Un candidat est une combinaison d'adresse IP, de port et de protocole de transport. Il associe leur ID, leur mot de passe et la priorité entre 1 et 0 (1 la plus haute priorité). Il y a 3 types de candidats : interface réseau physique, interface réseau logique, et les candidats découverts par STUN et TURN.

Host Candidat : est un candidat d'adresse de transport qui assure la communication Peer-to-Peer obtenu directement depuis une interface locale. L'interface locale peut être Ethernet ou WiFi, ou peut être obtenue d'après un mécanisme de tunnel.

Un agent Collecte les candidats (fait la liaison des ports et attache les adresses IP avec une interface). Il obtient un candidat pour chaque composante de flux média dans chaque adresse IP. Chaque composante a une ID.

Il faut que l'agent ramène les candidats relayés et les candidats réflexifs du serveur. Si l'agent (que ce soit client ou serveur) collecte les candidats réflexifs du serveur et les candidats relayés dans ce cas il utilise le serveur TURN. S'il collecte seulement les candidats du serveur dans ce deuxième cas il utilise le serveur STUN.

Quand les candidats réflexifs du serveur et les candidats relayés sont alloués. Ces candidats restent en vie jusqu'à la fin à leur traitement (binding request, refresh allocation, refresh trasaction).

Un client peut obtenir les candidats UDP local par liaison des ports dans toutes les interfaces d'un réseau valable. (9)

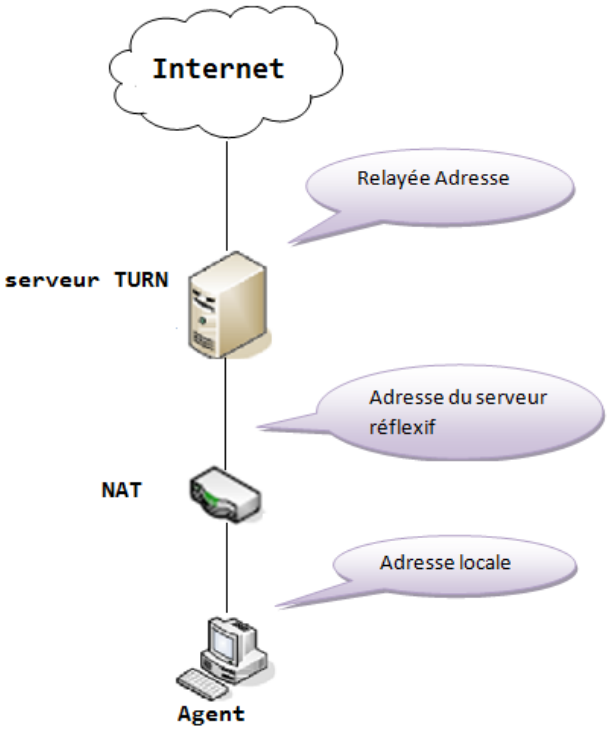


Figure 20 Collecter les candidats

1.4.2. Trier les candidats

L'algorithme recherche dans toutes les paires de candidats, s'il existe une paire qui travaille . Pour un rapide résultat, les candidats sont triés dans un ordre précis. Le résultat est une liste de paires des candidats triés appelé Check list.

Algorithme :

Chaque agent donne à ses candidats une priorité numérique, qui est envoyé avec le candidat vers le Peer.

Les priorités locales et distantes sont combinées de façon que chaque agent ait le même ordre pour les paires candidats.

ICE peut vérifier les deux directions, il envoie Check vers le NAT. (9)

1.4.3. Bloquer les candidats

Chaque candidat associe une propriété appelé Foundation. Deux candidats ont la même Foundation quand ils sont similaires (même type, même hôte candidat, même protocole de STUN). Juste les paires de candidats de même Foundation sont testées, les autres paires sont marquées 'Frozen' bloqué. (9)

1.4.4. Utilisation de serveur TURN

Quand un agent envoie la demande d'allocation, le NAT crée l'adresse du serveur réflexif, les paquets qui arrivent seront renvoyés vers le serveur TURN.

L'allocation arrive vers le serveur TURN, ce dernier alloue le port de l'adresse IP locale, et envoie la réponse d'allocation, serveur TURN travaille comme relais pour échanger le trafic entre les deux paires. (9)

1.4.5. Utilisation de STUN

Un agent envoie Binding-request vers le serveur STUN. Le serveur STUN informe l'agent de candidat réflexif par copie d'adresse de transport source de Binding-Request dans la réponse Binding. (9)

1.4.6. Vérification de connectivité

Quand un agent rassemble tous les candidats, il classe ces candidats par priorité et les envois vers l'autre pair (Peer) sur le canal de signalisation. Quand l'autre pair reçoit ces offres, il effectue le même processus de rassemblement et répond avec leur liste de candidats. Chaque agent a une liste complète qui rassemble : Ces candidats et les candidats de Peer. L'agent rassemble ces candidats paire à paire, le résultat Candidate Pairs. Pour savoir qu'elle paire de candidat travaille, chaque agent classe une série de CHECKS (Vérification de connectivité). Chaque CHECK est une transaction offre/réponse, le client

produit une paire de candidat par l'envoi de demande STUN de candidat local vers un candidat distant.

1- Classer les paires de candidats par ordre de priorité
2- Envoyer CHECKS sur chaque paire de candidats par ordre de priorité.
3- Connaître si CHECKS ont été reçus par l'autre agent.
4- Demande de liaison (STUN Binding request) est utilisé pour vérifier la connectivité
5- Réponse de STUN (Binding response) contenant adresse IP de transport de coté public de NAT qui existe entre l'agent et leur Peer
6- Si cette adresse de transport est différente d'autre candidat, Cette adresse représente un nouveau candidat (Peer reflexive candidate)
7- ICE teste cette adresse

À la fin, les deux Peers peuvent envoyer un message (et recevoir) dans les deux directions. (9)

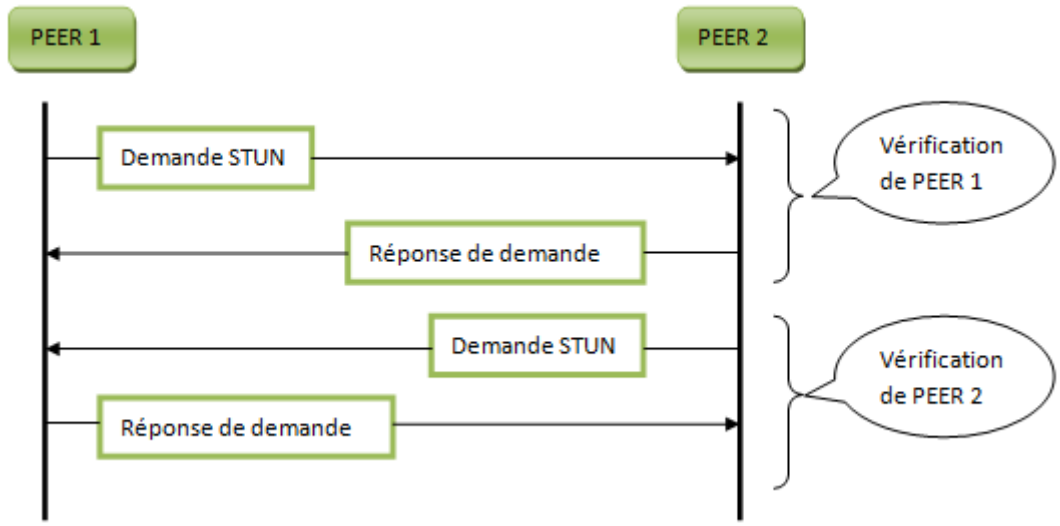


Figure 21 Vérification de connectivité

L'envoi de l'offre initiale

Si un client veut établir une session média avec un Peer, il collecte ces candidats locaux, il les code avec un protocole comme SDP. Le client envoie ces candidats collectés vers l'information de connexion de Peer par le canal de signalisation préétablie. Il désigne un candidat local comme un candidat par défaut dans l'offre initiale. Le candidat par défaut est un candidat UDP. S'il n'existe aucun candidat UDP collecté, la communication échoue.

Recevoir l'offre initiale

Le récepteur d'offre initiale collecte ces candidats locaux, il les code avec un protocole comme SDP. Le client envoie ces candidats collectés vers l'information de connexion de Peer par le canal de signalisation préétabli. Il désigne un candidat UDP local comme un candidat par défaut dans la réponse de l'offre initiale. S'il n'existe aucun candidat UDP collecté, la communication échoue.

Quand le récepteur collecte ces locaux candidats, il démarre la phase de connectivité (les vérifications). Le récepteur peut coder les candidats collectés, les envoie dans une réponse provisoire vers l'émetteur avant l'envoi de la réponse d'offre initiale pour réduire la latence de l'établissement de connectivité. Si une information de connexion envoie une réponse provisionnel, la réponse suivante (réponse de l'offre initiale) ait le même ensemble de candidats et le même candidat par défaut qu'il existe dans la réponse provisoire. (9)

Traitement de réponse provisoire de l'offre initiale

Après que l'émetteur reçoit la réponse provisoire, il démarre les vérifications de connectivité :

Les messages de demande de liaison STUN ont été envoyés par l'émetteur pour les paires des candidats qui sont TURN-dérivés. Il faut que ces messages ne contiennent pas l'attribut Username, ils servent juste de permissions ouvertes dans le serveur TURN pour les vérifications de connectivité de Peer.

Traitement de la réponse d'offre initiale

L'émetteur commence les vérifications de connectivité.

Générer l'offre finale

Quand toutes les vérifications sont terminées sans erreur, l'information de connexion qui a envoyé l'offre initiale, envoie l'offre finale. L'offre finale contient le candidat local et le candidat qui est sélecté par ce protocole, et code ces candidats avec SDP, et les envoie vers le Peer.

Recevoir l'offre finale et générer la réponse

Une information de connexion reçoit l'offre finale, elle coupe le flux média à l'aide de candidats locaux et distants. La réponse de l'offre finale contient les candidats locaux et distants.

Traitement de réponse de l'offre finale

Une information de connexion qui reçoit la réponse de l'offre finale, coupe le flux média à l'aide des candidats locaux et distants (existant dans la réponse), il libère tous les candidats locaux. (9)

1.5.Conclusion

Dans ce chapitre on a vu les protocoles de traversée le NAT. Ces protocoles aident la communication Peer-to-Peer de traverser le NAT.

Dans le chapitre suivant nous allons voir une comparaison entre les protocoles TURN, STUN, ICE.

Chapitre 03

Comparaison entre TURN, STUN et ICE

2. Chapitre 03

Comparaison

2.1.Introduction

Si un client veut établir une communication Peer-to-Peer. Il doit choisir un protocole qui traverse le NAT, mais cela dépend de la situation des clients et des types de NAT.

Dans le protocole TURN tous les paquets qui transitent vers le Peer passent par le serveur.

Le protocole STUN fournit les adresses et les ports de Peers.

Le protocole ICE travaille avec les protocoles STUN et TURN pour fournir toutes les possibilités de communication aux clients

2.2.TURN

TURN est un protocole permettant aux Peers derrière des NAT d'effectuer des connexions entre eux en passant par un serveur de relai.

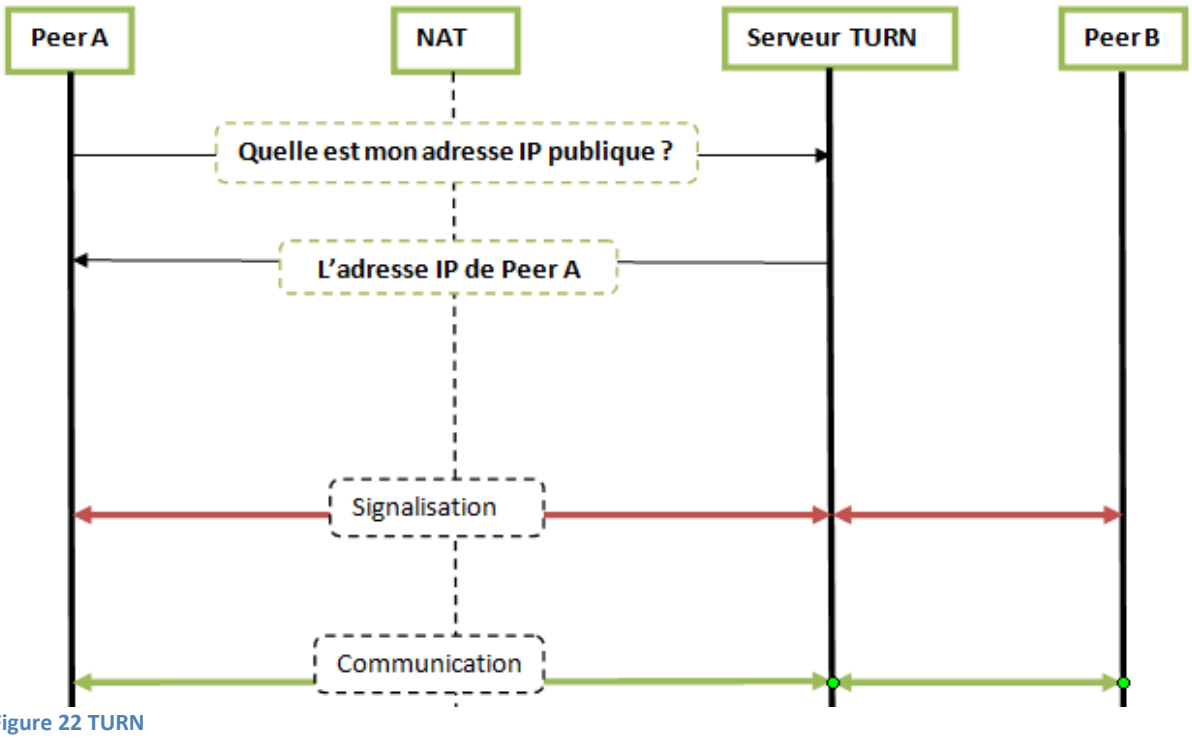


Figure 22 TURN

Avantages

- Une technique simple à comprendre et à déployer
- Le protocole STUN permet aux clients d'obtenir leurs adresses IP
- TURN est une extension de STUN
- TURN ne lit aucun protocole de communication
- Ce protocole en instance de standardisation offre un niveau de standardisation dans les serveurs de relais
- Le protocole utilise code d'erreur pour connaitre type d'erreur.
- Il utilise l'authentification de message, alors il est plus sécurisé.

Inconvénients

- afin d'utiliser le serveur TURN en tant que relais, une modification des programmes est nécessaire afin que ceux-ci intègre le protocole TURN
- Les liaisons entre les clients et le serveur est fragile à cause de LifeTime de l'allocation. Le client perde leur allocation s'il n'utilise pas l'attribut Keep-Alive.
- Le protocole TURN utilisé juste dans les NAT symétrique
- Finalement, étant donné que cette solution réside en un relais entre les postes, la latence est donc accrue puisque les données ne transigent plus directement.

La sécurité de protocole TURN

- Les attaques par dictionnaire hor-ligne : l'attaqueur capable d'écouter sur les messages qui transmit entre le client et le serveur, donc il peut connaitre le mot de passe (quand le mot de passe faible ou il existe dans le dictionnaire) la solution est :
    - L'utilisation d'un fort mot de passe est contre ce type d'attaque

- Faux actualisations et faux permissions : un attaqueur envoi demande d'actualisation avec expiration immédiate pour supprimer le service de client. Il envoi les demandes de CreatePermission pour créer des permissions vers indésirable destination
    - Ceci est empêché par l'authentification des actualisations.

- Faux Data : un attaqueur envoi des données vers le client et le Peer par l'envoi d'un faux message de Send Indication ou ChannelData vers le serveur TURN.
    - Le protocole TURN n'est pas protégé de ce type d'attaque (il permet de connaitre les adresses IP de Peers)

- Usurper l'identité d'un serveur : l'attaqueur intercepte ou redirige le trafic et fournit au client un faux relayé adresse
    - Le mécanisme Long-term est contre ce type d'attaque (message d'intégrité).

2.3.STUN

STUN est un protocole qui détermine l'adresse de transport (adresses et ports). Il permet de traverser les routeurs NAT, d'effectuer l'adresse IP et port publique d'un Peer situé dans un réseau privé pour fournir une communication UDP. Le client envoie des messages à un serveur et la réponse de ce dernier est l'adresse IP et le Port de Peer à communiquer.

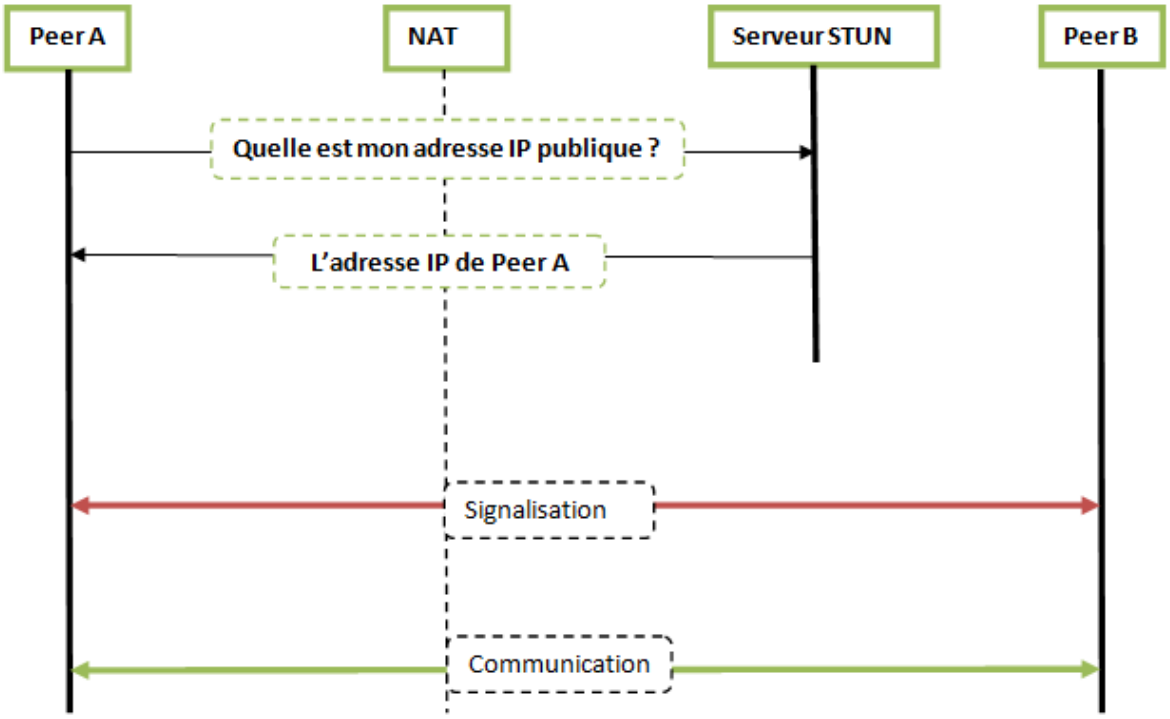


Figure 23 STUN

Avantages

- cette technique est une technique standardisée.
- cette technique ne requiert aucune modification de l'architecture du réseau présent.

- Utilise code d'erreur pour connaitre type d'erreur
- Il fonctionne avec tous les types de NAT sauf NAT symétrique.

Inconvénients

- les requêtes et les réponses pour caractériser la communication peuvent être interprétées comme une attaque par les dispositifs de sécurité.
- elles ne permettent pas la traversée de tous les pare-feu et les routeurs NAT.
- Il ne fonctionne pas avec le NAT symétrique.

La sécurité de protocole STUN

Les attaques extérieures

- Un attaqueur modifie les messages STUN pour insuccès opération de STUN
    - Ce type d'attaque est détecté par le message d'intégrité

Les attaques intérieures

- Un client (attaqueur) essaie de démarrer l'attaque DoS (envoyer un grand nombre de demande vers le serveur)
- Un client (attaqueur) envoie les demandes vers le serveur avec un falsifié adresse/port source, les réponses de serveur est envoyé vers cette adresse/port source (un grand nombre de messages de réponse).

Attaquer les utilisateurs de STUN

- L'attaqueur modifie l'adresse réflexive de client avant que le message de demande Binding arrive au serveur. Il change l'adresse IP source, le serveur répond le client falsifié avec l'attribut Xor-Mapped-Address pour recevoir des données destiné au client.

Ecoute électronique (Eavesdropping)

- L'attaqueur force le client d'utiliser adresse réflexive qui route vers l'attaqueur. recevoir tout les paquets destiné au client, après il envoi ces paquets vers le client (il permet d'observer tous les paquets qui destiné au serveur)

2.4.ICE

Le protocole ICE consiste a intégré STUN et TURN pour déterminer toutes les connexions possibles entre deux postes.

Avantages

- cette technique intègre STUN et TURN qui sont standardisés et non liés à un protocole de communication.
- elle permet la traversée de toutes les topologies et de tous les périphériques sauf dans le cas où des pare-feu sont utilisés.
- elle ne nécessite aucune modification dans les dispositifs de sécurité présents sur un réseau.
- Elle nécessite seulement le déploiement d'un serveur combiné STUN/TURN dans un réseau
- elle traverse tous les types de NAT.

Inconvénients

- cette technique consiste à caractériser toutes les communications possibles entre un poste, un réseau publique et entres deux postes. elle augmente le temps d'établissement d'un appel

- tout comme dans le cas de STUN, la caractérisation de la communication peut être interpréter par les dispositifs de sécurité comme étant une attaque mené d'un serveur à un poste situé dans un réseau privée

- cette technique force un serveur à utiliser quelques ports seulement pour les diverses transmissions de média

- une modification des serveurs et des clients est nécessaire afin de déployer ICE.

- Le temps d'établissement d'une communication est long.

La sécurité de protocole ICE

Attaquer les vérifications de connectivité

- Faux invalide : un attaqueur pose un valide candidat invalide d'une paire des agents, alors l'agent utilise différent candidat.
- Faux valide : un attaqueur pose un invalide candidat valide
- Faux candidat réflexive Peer : pour orienter l'agent vers un autre destinataire

2.5.Conclusion

Les protocoles de traversée le NAT (TUR, STUN, ICE) fournissent toutes les possibilités de communication aux clients. Par exemple si un client utilise le NAT symétrique, il ne peut pas utiliser le protocole STUN.

Dans ce chapitre, on a fait la comparaison entre les trois protocoles ICE, STUN et TURN.

Conclusion générale

3. Conclusion générale

Un client dans un réseau privé ne peut pas communiquer à l'internet car le nombre d'adresses IP d'IPv4 ne satisfait pas tout le monde. Comme il peut communiquer quand il utilise le NAT. Le NAT résout le problème de pénurie d'adresses. Un réseau privé a plusieurs adresses privées et une adresse publique.

Deux clients veulent établir une communication Peer-to-Peer, cette communication traverse un ou plusieurs NAT, donc le problème est comment trouver l'adresse de destination ? Comment faire pour que l'adresse source change à cause du NAT ?

L'utilisation du protocole qui traverse le NAT résout ce problème car il utilise un serveur Rendez-vous connecté à l'internet pour fournir les adresses et les ports de Peers et assure la communication jusqu'à la fin.

Dans ce travail, on a détaillé les protocoles TURN, STUN, ICE qui sont des protocoles de traverse le NAT et assurent la communication Peer-to-Peer.

En long terme l'utilisation d'IPv6 rend l'utilisation du NAT en diminution.

En cours terme, l'utilisation de NAT sera en augmentation car il fait la combinaison entre l'IPv4 et l'IPv6.

Bibliographie

1. **Latu, Philippe.** Adressage IPv4. *inetdoc.net.* [En ligne] avril 2011. http://www.inetdoc.net/articles/adressage.ipv4/.

2. TPv4 et IPv6. *ARIN American Registry For Internet Number .* [En ligne] https://www.arin.net/.

3. **Berdjugin, Jean-François.** Network Adress Translation. [En ligne] http://berdjugin.com/.

4. **J. Rosenberg, J. Weinberger,C. Huitema,R. Mahy.** STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs). 2003.

5. **Lahmadi, Abdelkader.** *Sécurité réseau : pare-feu.*

6. **Bryan Ford, Pyda Srisuresh,Dan Kegel.** *Peer-to-Peer Communication Across Network Address Translators.*

7. **R. Mahy, P. Matthews, J. Rosenberg.** rfc5766. *Traversal Using Relays around NAT (TURN) : Relay Extensions to Session Traversal Utilities for NAT (STUN).* 2010.

8. **J. Rosenberg, R. Mahy, P. Matthews, D. Wing.** rfc5389. *Session Traversal Utilities for NAT (STUN).* 2008.

9. **Rosenberg, J.** rfc5245. *Interactive Connectivity Establishment (ICE) : A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols.* 2010.

10. **BOUMEZZOUGH, Mr Mohamed El Mahdi.** *Etude et mise en oeuvre du service pilote ToIP de.*

11. **Perreault, Simon.** NAT and Firewall Traversal. [En ligne] http://www.viagenie.ca.

12. **Strowes, Stephen.** ICE, TURN and STUN. [Online] avril 2011.

13. **Dan Wing, Technical Leader.** Overview of ICE. [En ligne] Juillet 2011.

14. **Strowes, Stephen.** ICE, TURN and STUN for NAT Traversal. [Online] juillet 2011.

15. **Pain-Barre, Cyril.** Réseaux - Cours 4. [En ligne]

16. **Guermouche, A.** Administration réseau. [En ligne]

17. **GUERRIER, Xavier BUREAU & Emilien.** IPv6 IP Next Generation. [En ligne]

18. **Feamster, Nick.** Addressing: IPv4, IPv6, and Beyond. [En ligne]

19. *UDP* hole punching. [En ligne]http://en.wikipedia.org/wiki/UDP_hole_punching. (avril 2011)

20. Network address translation. [En ligne] http://en.wikipedia.org/wiki/Network_address_translation (avril 2011)