Protocoles IP

Différents standards sont utilisés pour établir un dialogue entre plusieurs hôtes réseau. Certains s'assurent de la livraison fiable d'une donnée tels que TCP pour la livraison fiable et complète de fichiers, d'autres sont optimisés pour une latence plus faible tels que UDP pour une diffusion en direct, d'autres pour une plus petite quantitée de données requises pour établir un dialogue entre deux appareils comme par exemple une requête ICMP qui sert seulement à déterminer la latence ou le chemin emprunté lors d'un échange entre deux hôtes.

ICMP

Internet Control Message Protocol

Le protocole ICMP (protocole IP #1) est généralement utilisé pour surveiller l'état d'un système, pour diagnostiquer la connexion entre deux systèmes ou pour stresser un système généralement dans l'objectif de nuire à son fonctionnement (ex. attaque de déni de service).

Les applications communes de ICMP sont "ping" et "traceroute". Ping permet de tester la connexion et la latence entre deux systèmes et traceroute permet de vérifier le chemin qu'emprunte une transmission entre deux systèmes (les routeurs qui acheminent les données entre les interlocuteurs).

Ping

Ping est une commande généralement utilisée pour tester la connectivité entre deux appareils mis en réseau ainsi que de tester la latence de communication entre ces appareils. Cette commande est présente dans la majorité des systèmes pouvant être mis en réseau et utilise le protocole standardisé ICMP.

ICMP est un protocole sans état (sans établissement de session) permettant l'envoi de messages simples entre deux appareils. Ping utilise principalement les messages de type 8 ou "echo" et de type 0 ou "echo reply". Par le nom de ces types de messages, on peut en déduire que "ping" fonctionne en envoyant un message et en attendant une réponse avec le même contenu. Si un appareil envoie une requête "echo" à un autre appareil contenant le message "12345", il s'attendra à recevoir une réponse contenant "12345". Toute autre réponse indiquerait que la donnée aurait été déformée en chemin soit vers où depuis la destination du ping.

Par défaut, la plupart des systèmes enverront une partie contigue (qui se suit) de la table ASCII à une longueur déterminée (Windows envoie par défaut 32 octets/32 caractères ASCII, Linux en envoie 64). Il est possible de personnaliser la longueur ainsi que le contenu de ce message des outils le permettant (ex. nping).

Exemple d'une requête echo telle que présentée par Wireshark

```
Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Data (32 bytes)
        Text: abcdefghijklmnopqrstuvwabcdefghi
        [Length: 32]
```

Exemple d'une réponse echo telle que présentée par Wireshark

```
Internet Control Message Protocol
    Type: 0 (Echo (ping) reply)
    [Response time: 40.602 ms]
    Data (32 bytes)
        Text: abcdefghijklmnopqrstuvwabcdefghi
        [Length: 32]
```

Cette application est parfois utilisée à des fins malveillantes pour "noyer" (flood) un système et surcharger sa connexion réseau. Ce type d'attaque se nomme une attaque de déni de service et pourrait être utilisée pour figer un serveur web ou pour augmenter la latence d'un compétiteur

dans un jeu en ligne.

Traceroute

Traceroute permet de déterminer l'itinéraire de la transmission d'une donnée (les routeurs responsables de l'acheminement de la donnée) d'un appareil à un autre. Cette application utilise aussi le protocole ICMP.

Ce protocole utilisera le type de réponse 11 du protocole ICMP (time exceeded) en déterminant un TTL (time to life) en fonction du nombre de sauts effectués pour rejoindre sa destination. Le premier ping envoyé aura un TTL de 1. De cette façon le premier routeur qui traitera la donnée indiquera que le délai est dépassé et répondra par sa propre adresse IP ainsi indiquant à la source du trafic le routeur à travers lequel la communication devra passer. La source enverra ensuite un ping avec un TTL de 2 pour que le deuxième routeur réponde avec un message de type 11/délai expiré pour que la source soit informée de l'adresse du deuxième routeur et ainsi de suite.

Exemple d'une première requête de traceroute avec un TTL de 1

```
Internet Protocol Version 4, Src: 192.168.1.2, Dst: 192.168.0.2
    Time to Live: 1
        [Expert Info (Note/Sequence): "Time To Live" only 1]
            ["Time To Live" only 1]
    Protocol: ICMP (1)
    Source Address: 192.168.1.2
    Destination Address: 192.168.0.2
Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
```

Exemple d'une réponse du premier saut d'une requête traceroute

```
Internet Control Message Protocol
    Type: 11 (Time-to-live exceeded)
    Code: 0 (Time to live exceeded in transit)
    Internet Protocol Version 4, Src: 192.168.1.2, Dst: 192.168.0.2
        Time to Live: 1
            [Expert Info (Note/Sequence): "Time To Live" only 1]
                ["Time To Live" only 1]
                [Severity level: Note]
                [Group: Sequence]
```

```
        Protocol: ICMP (1)
        Source Address: 192.168.1.2
        Destination Address: 192.168.0.2
```

Exemple d'une deuxième requête de traceroute avec un TTL de 2

```
Internet Protocol Version 4, Src: 192.168.1.2, Dst: 192.168.0.2
    Time to Live: 2
        [Expert Info (Note/Sequence): "Time To Live" only 2]
            ["Time To Live" only 2]
    Protocol: ICMP (1)
    Source Address: 192.168.1.2
    Destination Address: 192.168.0.2
Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
```

Exemple d'une réponse du deuxième saut d'une requête traceroute provenant de la destination finale de la requête

```
Internet Protocol Version 4, Src: 192.168.0.2, Dst: 192.168.1.2
    Protocol: ICMP (1)
    Source Address: 192.168.0.2
    Destination Address: 192.168.1.2
Internet Control Message Protocol
    Type: 0 (Echo (ping) reply)
    [Request frame: 3]
    [Response time: 1.846 ms]
```

UDP

UDP ou "User Datagram Protocol" est un protocole de communication IP sans établissement de session ou accusés de réception. Il permet le transfert de données à une très faible latence mais sans garantie de livraison et donc une moindre fiabilité que certains autres protocoles.

Cas d'utilisation

Applications sensibles à la latence

Certains mécanismes de communication réseau sont plus sensibles à la latence. On peut penser ici à des appels téléphoniques ou des jeux vidéos en ligne. L'utilisation de UDP ne nécessitant pas d'établissement de session avant un transfert de données permet donc l'établissement d'une connexion et le transfert d'une donnée avec un délai de livraison minimal.

Application en direct

Si une connexion est un flux en direct, il est parfois préférable d'avoir une latence réduite en prenant le risque qu'une donnée ne soit pas livrée. On peut ici penser à des caméras de surveillance ou de téléopération. L'utilisation de UDP permet de minimiser la latence et si une image du flux vidéo est perdue en chemin, l'image suivante mettera à jour l'affichage de toute façon.

Dans la transmission d'un flux vidéo, seulement les modifications sont transférées d'une image à une autre mais suite à un certain nombre de modifications d'image, une image clé contenant l'ensemble de l'image est transférée. Lorsque cette image clé est perdue en chemin, il y a de fortes chances que l'image soit distortionnée comme ceci :

Dans le contexte d'un jeu vidéo, une donnée qui serait perdue/abandonnée entre deux joueurs (si le jeu est en pair-à-pair) ou entre un joueur et le serveur pourrait avoir comme impact que lorsque la donnée suivante soit acheminée, le joueur risque de "rubber band" (retourner à l'arrière) puisque les autres joueurs ou le serveur n'auront pas été informés des derniers mouvements du joueur.

Applications de "broadcast"

Puisqu'un broadcast n'est pas destiné à une seule adresse IP, il n'est généralement pas possible d'établir une session entre deux appareils. Il est seulement possible de diffuser une donnée et laisser le domaine de diffusion acheminer la donnée. Certaines applications telles que DHCP ne nécessitent pas d'adresse IP pour fonctionner, ceux-ci sont généralement des protocoles UDP.

Petits transferts de données

On peut penser ici à une requête DNS ou une synchronisation d'horloge réseau par NTP. Ces transferts de données sont minimes et en cas de donnée perdue, effectuer une nouvelle requête devrait prendre un temps négligeable.

Fonctionnement

Chaque protocole IP a un numéro lui étant associé, dans le cas de UDP, il s'agit du protocole IP #17.

UDP est un des protocoles IP les plus simples dans son fonctionnement. L'en-tête d'un paquet UDP contient généralement seulement le port de source (16 bits), de destination (16 bits), la longueur de la donnée livrée (16 bits) et une donnée permettant de valider la donnée livrée à l'intérieur du paquet (aussi 16 bits). Cette en-tête a donc une longueur de seulement 64 bits.

Puisque UDP ne garanti pas la livraison d'une donnée, il n'est pas non plus garanti que les données seront livrées dans le bon ordre. Si une donnée doit être segmentée et réassemblée du côté du client, c'est au niveau de la couche d'application que les données devront être étiquettées pour être interprétées dans l'ordre par le client. La somme de contrôle permettant de confirmer la validité de la donnée livrée est le seul mécanisme pour s'assurer que la donnée acheminée ne soit pas erronnée.

Dans l'exemple d'en-tête suivante, une requête UDP est effectuée vers le port UDP 443 (QUIC) d'un serveur web et le message aura une longueur de 31 octets.

```
User Datagram Protocol, Src Port: 63857, Dst Port: 443
    Source Port: 63857
    Destination Port: 443
    Length: 39
    Checksum: 0x875d [unverified]
    UDP payload (31 bytes)
```

Le serveur nous répond ensuite au port source de notre requête initiale avec un message composé dans ce cas-ci de 24 octets.

```
User Datagram Protocol, Src Port: 443, Dst Port: 63857
    Source Port: 443
    Destination Port: 63857
    Length: 32
    Checksum: 0x46a0 [unverified]
    UDP payload (24 bytes)
```

TCP

TCP est le protocole de transmission IP le plus répandu nécessitant l'établissement d'une session de communication entre deux hôtes ou un hôte et un serveur et utilisant des accusés de réception pour s'assurer de la réussite de l'acheminement de données entre les interlocuteurs.

Cas d'utilisation

Contrairement à UDP, TCP permet l'établissement d'une session entre un client et un serveur. Ceci permet entre autres la retransmission partielle de données en cas de perte de données puisque chaque bloc de donnée est identifié par un numéro de séquence et que le client fait parvenir un accusé de réception pour chaque bloc de données transmis assurant par conséquent une connexion plus fiable mais nécessitant plus d'étapes avant le transfert d'une donnée.

Ce protocole est donc mieux adapté qu'UDP au transfert d'un large volume de données lorsque la latence n'est pas un enjeu pour la connexion à établir.

On pourrait penser ici au téléchargement d'une page web ou d'un fichier par HTTP(S) ou (S)FTP, à l'envoi d'un courriel par SMTP ou même à l'écoute d'un flux audio (ex. Spotify) / vidéo (ex. YouTube/Netflix) si celui-ci n'est pas en direct puisque la latence n'y est pas un enjeu mais la qualité de livraison le devient. Dans le cas de la transmission de ces flux audio/vidéo, une partie des données est téléchargée à l'avance dans une mémoire tampon (buffer) et sera téléchargée en "chunks" ou blocs au fur et à mesure que la lecture du média se poursuivra.

Fonctionnement

Le numéro de protocole Internet de TCP est #6. Avant de transférer des données entre deux hôtes ou un hôte et un serveur, il est nécessaire pour les deux partis d'établir une session pour maintenir un suivi des données transmises, à transmettre, reçues et à retransmettre.

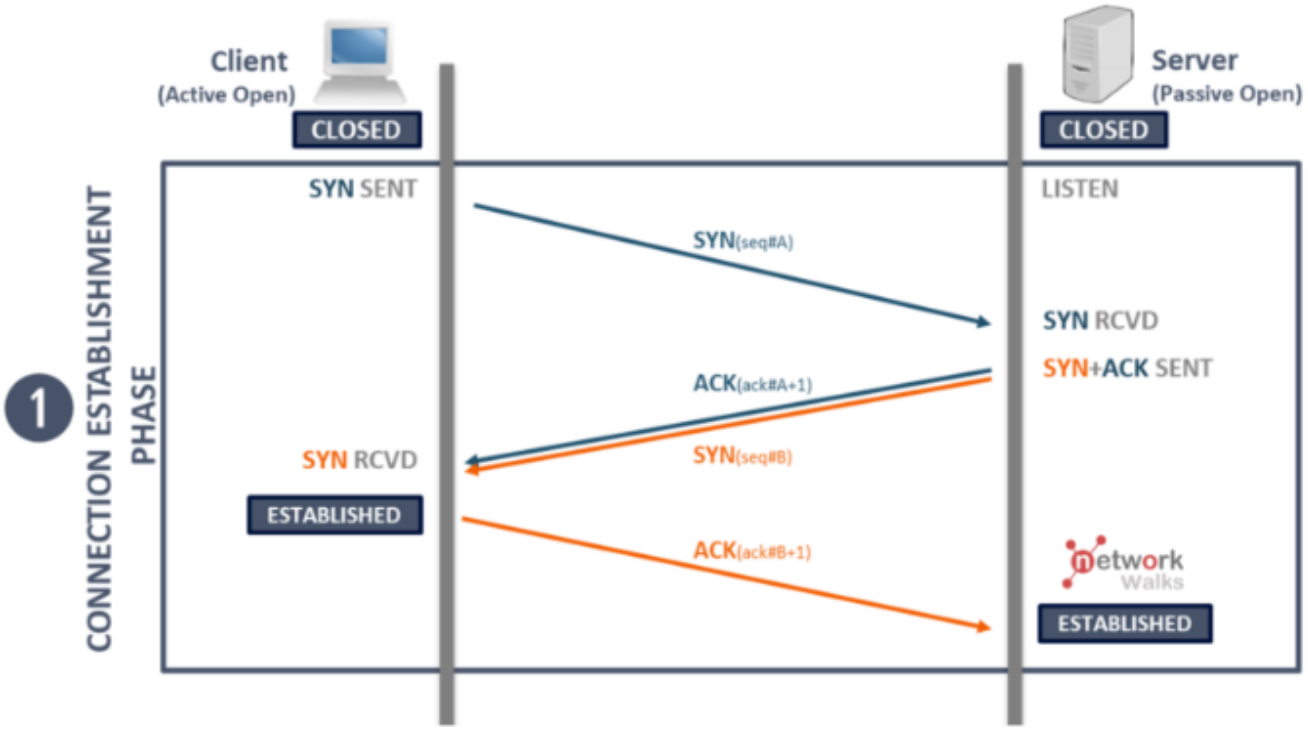
Structure d'en-tête

Une en-tête TCP est considérablement plus longue et plus complexe qu'une en-tête UDP. En plus des ports de source et destination, elle contient, entre autres, un numéro de séquence, du numéro de séquence attendu ensuite, et d'un identifiant de type de paquet (flag, ex. SYN, ACK, FIN, RST, etc.).

Dans l'exemple suivant, la connexion est déjà établie et il s' agit du 37 paquet acheminé du client vers le serveur confirmant la réception de la 85e partie de la donnée à transférer. On y voit aussi dans les "Flags" la mention "FIN". Cette mention indique au serveur que le client désire clore sa connexion auprès du serveur.

```
Transmission Control Protocol, Src Port: 62928, Dst Port: 53, Seq: 37, Ack: 85, Len: 0
    Source Port: 62928
    Destination Port: 53
    Sequence Number: 37    (relative sequence number)
    Sequence Number (raw): 743751263
    Acknowledgment Number: 85    (relative ack number)
    Acknowledgment number (raw): 287489219
    0101 .... = Header Length: 20 bytes (5)
    Flags: 0x011 (FIN, ACK)
    Window: 65434
    Checksum: 0x3188 [unverified]
    Urgent Pointer: 0
```

Échange de poignée de main



Pour établir la connexion entre un hôte et un serveur, le client doit tout d'abord envoyer une demande de connexion au serveur. Ce paquet sera identifié par la mention "SYN" pour "synchronisation". Le serveur répondra ensuite avec une validation de la synchronisation ou un "SYN-ACK" pour "Synchronization Acknowledgement". Le client répondra ensuite par un "ACK" ou "Acknowledgement" pour indiquer au serveur que la session est bien établie.

Transfert de données

Une fois la connexion établie, le transfert de données peut être entâmé. À chaque donnée reçue, le client fera parvenir au serveur un accusé de réception. Plusieurs données peuvent être transférées au sein d'une unique session.

Fermeture de la session

À la fin du transfert de la donnée, le client pourra faire parvenir un "FIN" ou "Final" au serveur pour lui indiquer la fin de la session. Si le serveur désire rompre la connexion prépaturément avec le client, il fera parvenir un "RST" ou "Reset" pour couper la connexion.