

Le Protocole DCC

par [Mathieu Lemoine](#)

Date de publication : 05/09/2006

Dernière mise à jour : 09/10/2006

Le protocole DCC est un un protocole semi-encapsulé dans le protocole CTCP. Il permet à deux clients IRC de communiquer directement sans passer par le serveur.

Notes

Remerciements

Introduction

I - Présentation

II - Initialisation de la connexion

III - Les détails des types de connexions

III-A - Connexion CHAT

III-B - Connexion SEND

Voir aussi

Notes

Remerciements

Merci à [Eusebius](#) pour sa relecture.

Introduction

Cet article présente le protocole DCC (Direct Client-to-Client ou Direct Client(-to-client) Connection) d'une manière simple et, normalement, compréhensible pour n'importe qui ayant un minimum de connaissances sur les protocoles TCP/IP, IRC et CTCP, ainsi qu'en théorie des Réseaux.

Au besoin, vous trouverez un lien vers des tutoriels sur les protocoles IRC et CTCP en fin d'article.

Le protocole DCC est un un protocole semi-encapsulé dans le protocole CTCP. Il permet à deux clients IRC de communiquer directement sans passer par le serveur.

I - Présentation

Le protocole DCC est à moitié encapsulé dans le protocole CTCP, à moitié Stand-Alone (sans encapsulation autre que TCP/IP).

C'est l'initialisation de la connexion qui consiste en l'envoi d'un message CTCP adéquat. La réception et le traitement du message CTCP aboutit à la création d'une connexion directe entre les deux clients et par la suite, le protocole ne dépend plus d'une encapsulation quelconque.

II - Initialisation de la connexion

La commande CTCP permettant d'initialiser une connexion DCC est la commande DCC. Elle ne peut être envoyée qu'en tant que requête CTCP. Sa syntaxe est :

```
DCC <type> <argument> <adresse> <port> [<size>]
```

- **type** : *CHAT* pour ouvrir une connexion destinée à la discussion, ou *SEND* pour transférer un fichier.
- **argument** : *chat* dans le cas d'une connexion CHAT, le nom du fichier dans le cas d'un transfert de fichier.
- **adresse** : représentation ASCII de l'adresse IP du client proposant la connexion DCC, envoyée en tant qu'entier sur quatre octets.
- **port** : représentation ASCII du numéro de port sur lequel le client attend la connexion DCC venant du client à qui la connexion a été proposée.
- **size** : dans le cas d'une connexion SEND, représentation ASCII de la taille du fichier envoyé (en octets), en tant qu'entier sur quatre octets.

À noter : pour adresse, port et size, il vous faut utiliser l'encodage *little endian* avant de convertir le nombre en sa représentation ASCII (c'est à dire : les octets de poids faibles en premier). Si ça n'a pas vraiment d'importance pour le numéro de port et la taille qui sont utilisés en tant que nombres, l'adresse IP, elle, doit être inversée avant d'être transformée en chiffres ASCII.

Avant d'envoyer cette commande, le client proposant la connexion IRC doit ouvrir le port adéquat en écoute afin de pouvoir accueillir la connexion.

Ceci peut poser quelques problèmes si le client en question est derrière un pare-feu.

III - Les détails des types de connexions

III-A - Connexion CHAT

La connexion CHAT consiste simplement en l'envoi du texte par les deux parties.

Elle prend fin dès qu'un des deux clients la coupe.

III-B - Connexion SEND

Pour envoyer un fichier, c'est la méthode des paquets qui est utilisée.

C'est à dire que le client ayant proposé la connexion (et donc envoyant le fichier), va envoyer un bout du fichier, de la taille de son choix, puis va attendre que le client receveur lui envoie la taille du packet reçu (représentation hexadécimale, et non ASCII de la taille en octets comme un entier sur 4octets, en *big endian* [octets de poids fort en premier] cette fois).

Tant que le receveur n'a pas signalé la réception intégrale du paquet (qui peut être faite en plusieurs fois), l'envoyeur ne continue pas l'émission.

De plus, l'envoyeur ne ferme pas la connexion jusqu'à ce que le receveur lui ait affirmé avoir reçu le dernier octet du dernier paquet.

De ce fait, on est toujours assuré que quand la connexion se ferme normalement (i.e. est fermée par l'envoyeur) que le fichier a bien été intégralement transmis.

Voir aussi

[Le Protocole CTCP](#) : Ce qu'est le protocole CTCP, comment il fonctionne, etc. Bref, tout ce qu'il est nécessaire de savoir pour comprendre la phase d'initialisation du protocole DCC, et même plus...

[Le Protocole IRC](#) : Ce qu'est le protocole IRC, comment il fonctionne, etc. Bref, tout ce qu'il est nécessaire de savoir pour comprendre le protocole CTCP, et même plus...

[La RFC du protocole CTCP](#) : Il s'agit en fait d'une "note" de doc, mais c'est la seule norme que j'ai réellement trouvée... La référence du protocole DCC est située dans l'Annexe A.

[L'ancienne RFC du protocole DCC](#) : Il s'agit également d'une "note" de doc, la version située dans l'annexe du document ci-dessus est plus récente, et un peu plus précise.