

Le Protocole IRC

par [Mathieu Lemoine](#)

Date de publication : 19/07/2006

Dernière mise à jour : 27/10/2006

Protocole de chat massif multi-chans et multi-serveurs parmi les plus courants, le protocole IRC est très utilisé pour des conférences, et des discussions massives sur Internet. Ce tutoriel vous aidera à acquérir les connaissances de base concernant ce protocole.

Notes

- Remerciements
- Introduction
- I - Généralités
 - I-A - Présentation
 - I-B - Structure sous-jacente au protocole
 - I-C - Syntaxe du protocole
- II - Initialisation et gestion de la connexion
- III - Commandes générales du protocole
 - III-A - Gestion des chans (et des utilisateurs dans ceux-ci)
 - III-B - Gestion des utilisateurs
 - III-C - Les commandes de discussion
- IV - Les réponses numériques
 - IV-A - Rappels/Informations
 - IV-B - Erreurs
- V - Divers
 - V-A - Voir le protocole IRC
 - V-B - Voir aussi

Notes

Remerciements

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

Introduction

Cet article présente le protocole IRC (Internet Relay Chat) d'une manière simple et, normalement, compréhensible pour n'importe qui ayant un minimum de connaissances en matière de Théorie des Réseaux Informatiques (notamment la structure client/serveur, échanges binaires/échanges textes, ...) et de Théorie des Graphes.

Sur l'Internet, de nombreux protocoles sont utilisés. Le protocole IRC est l'un des plus courants dans son domaine. Il permet de gérer des réseaux de chat massif multi-chans.

I - Généralités

I-A - Présentation

Il s'agit d'un protocole reposant sur de multiples échanges continus. Il permet deux types de connexions, les connexions classiques client/serveur qui permettent à un client de se connecter au réseau. Mais également des connexions de type serveur/serveur qui permettent de gérer de grands réseaux supportant de grandes montées en charges. Il est asynchrone : les serveurs et les clients envoient leurs données sans véritablement se soucier de ce que répond l'autre...

Les échanges entre le client et le serveur se font en mode texte. Le charset généralement utilisé est l'US-ASCII sur 8 bits. Il est cependant possible que cet encoding soit modifié selon le client ou le réseau.

I-B - Structure sous-jacente au protocole

Il est important de comprendre la structure sous-jacente au protocole IRC. Un réseau IRC est composé de serveurs et de clients.

Les serveurs sont les seuls composants d'un réseau IRC qui permettent d'interconnecter tous les autres.

Chaque serveur est identifié par un nom unique. On peut symboliser l'ensemble des serveurs composant un réseau IRC par un graphe non-orienté dont les sommets sont les serveurs et une arrête entre deux sommets symbolise la relation "Les deux serveurs sont connectés directement entre eux". Dans ce cas, le graphe doit être un arbre, c'est-à-dire qu'il doit être connexe (on peut toujours trouver un ensemble ordonné d'arrêtes reliant deux sommets quelconques du graphe) et acyclique (pour deux sommets donnés, il existe au maximum un seul et unique ensemble ordonné d'arrête permettant de les relier). Autrement dit, si un serveur quelconque envoie un message à tous les serveurs auquel il est connecté, puis que tous les autres serveurs envoient le message reçu à tous les serveurs auxquels ils sont connectés sauf à celui dont ils ont reçu le message, alors tous les serveurs recevront le message une et une seule fois. Ceci assure l'intégrité au niveau de la propagation des messages, et prévient les problèmes de boucles infinies sur un message.

Les clients sont tout ce qui se connecte à un serveur sans être soi-même un autre serveur.

- Les clients utilisateurs qui fournissent généralement une interface basée sur le texte et qui est utilisé pour communiquer interactivement via IRC.
- Les clients de service qui sont généralement des **bots** utilisés pour fournir une fonctionnalité supplémentaire sur le réseau.

Ces fonctionnalités ne sont généralement pas reliées directement au protocole IRC en lui-même.

Chaque client est ensuite connecté à un seul et unique serveur et est ainsi en liaison avec tous les autres clients et serveurs. Chaque client est identifié par un pseudonyme (nick) unique sur le réseau, composé de 9 caractères au plus. De plus, chaque client doit fournir aux serveurs un nom et un domaine de connexion (host). Chaque serveur est tenu de savoir à quel serveur est connecté directement un client. Pour communiquer entre eux, les clients ont deux possibilités : soit désigner explicitement les destinataires du message, soit joindre un chan (salle de discussion) et envoyer les messages à celui-ci (et par propagation à tous les clients qui y sont présents).

Un chan est créé dès qu'un utilisateur y est présent. Le chan peut avoir différentes options (modes), que nous détailleront plus tard... Le nom d'un chan doit commencer par un dièse (#) ou un esperluette (&).

- Les opérateurs IRC (IRCOPs) qui ont des fonctions d'administrateur au niveau des serveurs, leur permettant de faire redémarrer un serveur, de déconnecter un serveur, de déconnecter ou de bannir un client du réseau, etc...
- Les opérateurs de chan (chanops ou ops) qui ont des fonctions d'administrateur sur un ou plusieurs chans, leur permettant de changer les modes de ces chans, d'en éjecter (kicker) ou bannir un client. On peut les reconnaître sur un chan car leur nick est précédé d'un arobase (@) dans la liste des nick présents.

I-C - Syntaxe du protocole

Les serveurs et les clients IRC génèrent chacun des messages pouvant ou non générer une réponse. Ces messages ont toujours le même format (les crochets indiquent une partie optionnelle dans le message).

```
[{:}{prefix}{SP}]{COMMANDE}[{SP}{PARAMETRES}]{CRLF}
```

Le séparateur *SP* est un espace (code ascii : 0x20). Le séparateur *CRLF* est le duo "Retour Chariot, Fin de Ligne" (codes ascii 0x0D et 0x0A).

À noter que les caractères CR (0x0D), LF (0x0A) et NUL (0x00) sont interdits dans les messages IRC. De plus, la longueur d'un message IRC est de 512 caractères au maximum, y compris le *CRLF* final.

Le préfix n'est utilisé que pour un message provenant d'un serveur, il doit être ignoré s'il provient d'un client. Si le message est destiné à un serveur ou que le message a été émis sur le réseau par un serveur, le préfixe est simplement le nick du client, ou le nom du serveur ayant envoyé le message sur le réseau. Si le message est destiné à un client et que c'est un client qui a envoyé le message sur le réseau, alors le préfix suit cette syntaxe :

```
{NICK}{!}{NOM DU CLIENT}{@}{DOMAINE DU CLIENT}
```

La commande est soit un mot (une suite de une ou plusieurs lettres (A-Z) insensibles à la casse), soit un code de trois chiffres. Les codes de trois chiffres correspondent à des réponses de serveur, chacun ayant un nom, pour chaque commande, les réponses possibles seront listées. Les paramètres sont des chaînes de caractères sans espace, séparées par des espaces. À noter que le dernier paramètre peut contenir des espaces, dans ce cas, son premier caractère doit alors être les deux-points (:). Chaque commande a ses paramètres.

II - Initialisation et gestion de la connexion

Quand une connexion est ouverte sur un serveur, celui-ci ne peut déterminer de but-en-blanc si la connexion vient de la part d'un serveur ou d'un client. Ceci dépend de la séquence de commandes envoyée, il y en a deux possibles. Toute autre séquence donne lieu à un message d'erreur ou, plus simplement, à une rupture de la connexion. Les messages doivent être envoyés dans l'ordre de leur présentation pour ceux faisant partie de la séquence d'initialisation.

- **Message PASS**

Paramètre : *<le mot de passe>*

Réponses possibles : ERR_NEEDMOREPARAMS (461), ERR_ALREADYREGISTRED (462)

Ce message est optionnel, certains réseaux exigent que toutes les connexions soient identifiées par un mot de passe, chaque serveur contenant une liste de mots de passe valides pour les clients et une pour les serveurs. Il est très vivement recommandé que ce soit mis en place pour les connexions de serveurs. Ce message doit être le premier envoyé, il peut être envoyé plusieurs fois, seul le dernier est alors pris en compte. Il est invalide s'il est envoyé dans les autres cas.

- **Message NICK**

Paramètres : *<nouveau NICK> [<Compteur>]*

Réponses possibles : ERR_NONICKNAMEGIVEN (431), ERR_ERRONEUSNICKNAME (432), ERR_NICKNAMEINUSE (433), ERR_NICKCOLLISION (436)

Ce message fait partie de la séquence d'initialisation de connexion client/serveur, il permet à un client de spécifier le nick sous lequel il sera identifié. Il permet également à un client de changer de nick une fois la connexion initialisée, et peut donc être envoyé n'importe quand. Le *compteur* permet simplement d'indiquer à un serveur combien de serveur il y a entre le serveur qui lui a fait passer l'info et le client. Si un client fournit un compteur, alors il doit être ignoré. Si un serveur reçoit une information de changement de nick (de la part d'un autre serveur) et que le nouveau nick est déjà enregistré, alors il doit immédiatement arrêter de transmettre l'info et envoyer une commande KILL ainsi qu'un ERR_NICKCOLLISION à l'attention du client pour déconnecter les deux clients. Si un client tente de changer de nick pour un nick déjà utilisé, le serveur auquel il est directement connecté peut générer un ERR_NICKNAMEINUSE et éviter ainsi l'envoi de KILL (et donc les déconnexions).

- **Message USER**

Paramètres : *<nom d'utilisateur> <domaine> <nom de serveur> <nom réel>*

Réponses possibles : ERR_NEEDMOREPARAMS (461), ERR_ALREADYREGISTRED (462)

Ce message est la seconde et dernière étape d'initialisation de connexion client/serveur, il permet à un client de s'identifier clairement en indiquant ses nom, domaine de connexion, serveur de connexion et nom réel. En général, les paramètres domaine et serveur sont ignorés quand ils viennent directement du client, et sont renseignés par le serveur lui-même. Comme il est très facile de mentir sur ce type d'informations, il est conseillé d'avoir recours à un serveur d'identité s'il y en a un présent sur le client.

- **Message SERVER**

Paramètres : *<nom de serveur> <Compteur> <infos>*

Réponses possibles : ERR_ALREADYREGISTRED (462)

Ce message est la séquence d'initialisation d'une connexion serveur/serveur. Le compteur doit être à 1 quand c'est le nouveau serveur qui l'envoie. Ensuite le serveur qui reçoit la connexion doit envoyer le message en le préfixant de son nom, puis chaque serveur doit incrémenter le compteur. Une erreur à la suite de ce message réside le plus souvent en la coupure de la connexion et l'envoi d'un message ERROR.

- **Message QUIT**

Paramètres : [*<Message de départ>*]

Réponses possibles : Aucune

Cette commande permet d'indiquer la déconnexion d'un client, si un serveur la reçoit d'un client, il est ensuite censé couper la connexion. Le message est facultatif. En cas de split (coupure d'une connexion serveur/serveur), alors un message QUIT doit être émis pour chaque client derrière le serveur. De plus, le message doit être composé des noms des deux serveurs de part et d'autre de la connexion, d'abord celui qui est toujours accessible, et ensuite celui qui ne l'est plus.

- **Message SQUIT**

Paramètres : *<serveur> [<commentaire>]*

Réponses possibles : ERR_NOPRIVILEGES (481), ERR_NOSUCHSERVER (402)

Ce message est l'équivalent du message QUIT pour les serveurs, il peut également être utilisé par un IRCOP pour déconnecter un serveur.

- **Messages PING/PONG**

Paramètres : [*<param1>* [*<param2>*]]

Réponses possibles : ERR_NOORIGIN (409), ERR_NOSUCHSERVER (402)

Le classique PING/PONG, il permet de tester la validité d'une connection, tout client recevant un message PING doit renvoyer un message PONG avec les mêmes arguments, sous peine de voir sa connection éventuellement coupée.

- **Message OPER**

Paramètres : *<utilisateur>* *<mot de passe>*

Réponses possibles : ERR_NEEDMOREPARAMS (461), RPL_YOUREOPER (381), ERR_NOOPERHOST (491), ERR_PASSWDMISMATCH (464)

Ce message permet à un client de s'identifier en tant qu'IRCOP. Si l'authentification est validée, alors un message "MODE +o" est retransmis pour le pseudo identifié.

- **Message KILL**

Paramètres : *<nick>* *<commentaire>*

Réponses possibles : ERR_NEEDMOREPARAMS (461), ERR_NOPRIVILEGES (481), ERR_NOSUCHNICK (401), ERR_CANTKILLSERVER (483)

Ce message est utilisé par les serveurs et les IRCOPs et permet de déconnecter un utilisateur de force. Il est utilisé dans des cas tels qu'une atteinte grâce aux règles du réseau, ou une collision de nick.

- **Message ERROR**

Paramètres : *<message d'erreur>*

Réponses possibles : Aucune

Ce message est utilisé par les serveurs pour signaler une erreur grave ou fatale. Il peut également être envoyé à un utilisateur pour signaler l'erreur sans que ce client en soit la cause (très utile pour les IRCOPs par exemple), en l'encapsulant dans une NOTICE.

III - Commandes générales du protocole

- Les commandes de gestion des chans (et des utilisateurs dans les chans)
- Les commandes de gestion des utilisateurs
- Les commandes de discussion

III-A - Gestion des chans (et des utilisateurs dans ceux-ci)

- **Message JOIN**

Paramètres : *<nom du chan> [<clé>]*

Réponses possibles : ERR_NEEDMOREPARAMS (461), ERR_BANNEDFROMCHAN (474), ERR_INVITEONLYCHAN (473), ERR_BADCHANNELKEY (475), ERR_CHANNELISFULL (471), ERR_NOSUCHCHANNEL (403), RPL_TOPIC (332)

Ce message permet d'entrer sur un chan. Une fois qu'un utilisateur est sur un chan, il reçoit tous les messages (discussion et gestion) concernant ce chan. Si l'entrée est validée, le serveur transmet le message aux autres serveurs et autres utilisateurs présents sur le chan, puis le message RPL_TOPIC rappelle le topic du chan, et est suivi de messages RPL_NAMREPLY (353) avec la liste des nicks présents sur le chan, et enfin par un RPL_ENDOFNAMES (366).

- **Message PART**

Paramètres : *<nom du chan> [<commentaire>]*

Réponses possibles : ERR_NEEDMOREPARAMS (461), ERR_NOSUCHCHANNEL (403), ERR_NOTONCHANNEL (442)

Ce message permet de sortir d'un chan.

- **Message MODE**

Paramètres : *<nom du chan> <+|-|o|p|s|i|t|n|b|v|m>+ <autres paramètres>*

Réponses possibles : ERR_NEEDMOREPARAMS (461), ERR_NOSUCHCHANNEL (403), ERR_NOTONCHANNEL (442), ERR_NOSUCHNICK (401), ERR_CHANOPRIVSNEEDED (482), RPL_CHANNELMODEIS (324), ERR_KEYSET (467), RPL_BANLIST (367), RPL_ENDOFBANLIST (368), ERR_UNKNOWNMODE (472)

Ce message permet aux ops de modifier la configuration du chan.

o : ajoute ou enlève le status d'opérateur à un membre du chan

v : ajoute ou enlève le voice à un membre du chan (autorisation d'envoyer des messages au chan en état de modération)

b : ajoute ou enlève un ban (interdiction d'entrer sur le chan)

l : ajoute (éventuellement modifie) ou enlève le nombre maximum d'utilisateurs pouvant entrer sur le chan

k : ajoute (éventuellement modifie) ou enlève la clé du chan

t : interdit aux non-chanops de modifier le topic du chan

i : interdit l'entrée sur le chan aux non-invités

m : interdit aux non-voices de parler sur le chan (modération du chan)

n : interdit les messages venant de personnes n'étant pas sur le chan

s : Chan secret (n'apparaît pas dans les listing à moins que l'utilisateur ne soit présent sur le chan)

p : Chan privé (ne divulgue pas d'informations autres que l'existence du chan dans les listings à moins que l'utilisateur ne soit présent sur le chan)

- **Message TOPIC**

Paramètres : *<nom du chan> [<nouveau topic>]*

Réponses possibles : ERR_NEEDMOREPARAMS (461), ERR_NOTONCHANNEL (442), ERR_CHANOPRIVSNEEDED (482), RPL_TOPIC (332), RPL_NOTOPIC (331)

Si un nouveau topic est spécifié et si l'utilisateur en a le droit, alors ce message permet de modifier le topic (sujet) du chan. S'il n'est pas présent, alors il sert uniquement à redemander le topic du chan en question.

- **Message NAMES**

Paramètres : *<nom du chan>*

Réponses possibles : RPL_NAMREPLY (353), RPL_ENDOFNAMES (366)

Permet de redemander la liste des personnes présentes sur le chan et leur status (op, voice, simple membre).

- **Message LIST**

Paramètres : [*<nom du chan>*]

Réponses possibles : RPL_LISTSTART (321), RPL_LIST (322), RPL_LISTEND (323)

Permet de demander la liste des chans, ou les détails sur un chan en particulier.

- **Message INVITE**

Paramètres : *<nick> <nom du chan>*

Réponses possibles : ERR_NEEDMOREPARAMS (461), ERR_NOSUCHNICK (401), ERR_NOTONCHANNEL (442), ERR_USERONCHANNEL (443), ERR_CHANOPRIVSNEEDED (482), RPL_INVITING(341)

Permet d'inviter un utilisateur sur un chan.

- **Message KICK**

Paramètres : *<nom du chan> <nick>*

Réponses possibles : ERR_NEEDMOREPARAMS (461), ERR_NOSUCHCHANNEL (403), ERR_NOTONCHANNEL (442), ERR_CHANOPRIVSNEEDED (482)

Ce message sert aux ops à éjecter un utilisateur de force du chan.

III-B - Gestion des utilisateurs

- **Message WHOIS**

Paramètres : *<nick>*

Réponses possibles : ERR_NONICKNAMEGIVEN (431), RPL_WHOSUSER (311), RPL_WHOISSERVER (312), RPL_WHOSOPERATOR (313), RPL_WHOISIDLE (317), RPL_ENDOFWHOIS (318), RPL_WHOISCHANNELS (319)

Ce message permet de demander des informations sur un utilisateur tel que sont masque (nick, nom et domaine de connexion), les chans où il est (et son status sur ces chans), s'il a le status d'IRCOP, etc...

- **Message WHOWAS**

Paramètres : <*nick*>

Réponses possibles : ERR_NONICKNAMEGIVEN (431), ERR_WASNOSUCHNICK (406), RPL_WHOWASUSER (314), RPL_ENDOFWHOWAS (369), RPL_WHOISERVER (312)

Ce message est identique à WHOIS, mais pour des nicks qui ne sont plus utilisés. Bien sûr, ce n'est pas valide pour tous les nicks, uniquement ceux qui étaient utilisés il y a un court laps de temps.

- **Message MODE**

Paramètres : <*nick*> <+|-/o|i>+

Réponses possibles : ERR_NEEDMOREPARAMS (461), ERR_USERSDONTMATCH (502), ERR_UMODEUNKNOWNFLAG (501), RPL_UMODEIS (221)

Ce message permet d'attribuer certains pouvoir à certains utilisateurs.

o : ajoute ou enlève le status d'IRCOP

i : ajoute ou enlève le status invisible (informations réduites dans le WHOIS)

III-C - Les commandes de discussion

Elles sont au nombre de 2 (NOTICE et PRIVMSG) et suivent la même syntaxe : <*nick ou chan*> <*message*>

La NOTICE se distingue du message classique (PRIVMSG) par une simple signification sémantique : il n'y a pas de réelle réponse attendue, c'est uniquement un envoi d'information. De plus, aucune réponse d'erreur (réponse numérique) ne peut être envoyée à une notice. Pour les PRIVMSG, les réponses possibles sont : ERR_NORECIPIENT (411), ERR_NOTEXTTOSEND (412), ERR_CANNOTSENDTOCHAN (404), ERR_NOTOPLEVEL (413), ERR_WILDTOPLEVEL (414), ERR_NOSUCHNICK (401)

IV - Les réponses numériques

Voici les diverses réponses numériques énoncées dans les parties précédentes et leur significations.

IV-A - Rappels/Informations

Voici les réponses de rappel ou d'informations, leurs nom commencent par *RPL_* et leur numéro par un 2 ou un 3.

- **221 RPL_UMODEIS**

Paramètres : *<modes utilisateurs>*

Réponse à : MODE

Cette réponse indique les modes possibles pour les utilisateurs.

- **311 RPL_WHOISUSER**

Paramètres : *<nick> <nom> <domaine> <vrai nom>*

Réponse à : WHOIS

Cette réponse indique les informations disponibles pour le nick indiqué dans le message WHOIS.

- **312 RPL_WHOISSERVER**

Paramètres : *<nick> <serveur> [<infos serveur>]*

Réponse à : WHOIS, WHOWAS

Cette réponse indique le serveur auquel est ou était connecté l'utilisateur.

- **313 RPL_WHOISOPERATOR**

Paramètres : *<nick>*

Réponse à : WHOIS

Cette réponse indique que l'utilisateur est un IRCOP.

- **314 RPL_WHOWASUSER**

Paramètres : *<nick> <nom> <domaine> <vrai nom>*

Réponse à : WHOWAS

Cette réponse indique les (anciennes) informations disponibles pour le nick indiqué dans le message WHOWAS.

- **317 RPL_WHOSIDLE**

Paramètres : *<nick> <temps d'idle en secondes>*

Réponse à : WHOIS

Cette réponse indique le temps d'idle en secondes (temps depuis l'envoi du dernier message au serveur) de l'utilisateur.

- **318 RPL_ENDOFWHOIS**

Paramètres : *<nick>*

Réponse à : WHOIS

Cette réponse indique qu'on a envoyé toutes les informations sur l'utilisateur.

- **319 RPL_WHOSCHANNELS**

Paramètres : *<nick> <liste de channels avec status>*

Réponse à : WHOIS

Cette réponse indique les chans où est présent l'utilisateur ainsi que son status sur ceux-ci (@ pour op, +

pour voice), le symbole du status précède le nom du chan où il s'applique.

- **322 RPL_LIST**

Paramètres : *<nom du chan> <nombre d'utilisateurs présents> <topic>*

Réponse à : LIST

Cette réponse indique le début de la liste des chans enregistrés sur le réseau.

- **323 RPL_LISTEND**

Paramètres : *<>*

Réponses à : LIST

Cette réponse indique la fin de la liste des chans enregistrés sur le réseau.

- **324 RPL_CHANNELMODEIS**

Paramètres : *<modes chan>*

Réponse à : MODE

Cette réponse indique les modes possibles pour les chans.

- **331 RPL_NOTOPIC**

Paramètres : *<chan>*

Réponse à : TOPIC

Cette réponse indique que le chan concerné n'a pas de topic.

- **332 RPL_TOPIC**

Paramètres : *<chan> <topic>*

Réponse à : JOIN, TOPIC

Cette réponse indique le topic du chan concerné.

- **341 RPL_INVITING**

Paramètres : *<chan> <nick>*

Réponse à : INVITE

Cette réponse confirme l'invitation d'un utilisateur sur un chan.

- **353 RPL_NAMREPLY**

Paramètres : *<chan> <liste des utilisateurs avec status>*

Réponse à : NAMES

Cette réponse indique les utilisateurs présents sur le chan ainsi que leur status, indiqué de la même façon que pour RPL_WHOSCHANNELS (319).

- **366 RPL_ENDOFNAMES**

Paramètres : *<chan>*

Réponse à : NAMES

Cette réponse indique que tous les utilisateurs présents sur le chan ont été indiqués.

- **367 RPL_BANLIST**

Paramètres : *<chan> <masque d'utilisateur banni>*

Réponse à : MODE

Cette réponse indique un masque d'utilisateur qui est banni du chan.

- **368 RPL_ENDOFBANLIST**

Paramètres : <*chan*>

Réponse à : MODE

Cette réponse indique que tous les masques d'utilisateurs bannis du chan ont été indiqués.

- **369 RPL_ENDOFWHOWAS**

Paramètres : <*nick*>

Réponse à : WHOWAS

Cette réponse indique que toutes anciennes informations concernant le nick ont été indiquées.

- **381 RPL_YOUREOPER**

Paramètres : <>

Réponse à : OPER

Cette réponse indique que le status d'IRCOP vous a bien été attribué.

IV-B - Erreurs

Voici les réponses d'erreurs, leurs nom commencent par *ERR_* et leur numéro par un 4 ou un 5.

- **401 ERR_NOSUCHNICK**

Paramètres : *<nick>*

Réponse à : KILL, MODE, INVITE, PRIVMSG

Cette erreur est renvoyée quand le nick n'a pas été trouvé sur le réseau.

- **404 ERR_NOSUCHSERVER**

Paramètres : *<serveur>*

Réponse à : SQUIT, PING, PONG

Cette erreur est renvoyée quand le serveur n'a pas été trouvé sur le réseau.

- **403 ERR_NOSUCHCHANNEL**

Paramètres : *<chan>*

Réponse à : JOIN, MODE, PART, KICK

Cette erreur est renvoyée quand le chan n'a pas été trouvé sur le réseau.

- **404 ERR_CANNOTSENDTOCHAN**

Paramètres : *<chan>*

Réponse à : PRIVMSG

Cette erreur est renvoyée quand vous ne pouvez envoyer de message au chan du fait des restrictions en cours sur ce chan.

- **406 ERR_WASNOSUCHNICK**

Paramètres : *<nick>*

Réponse à : WHOWAS

Cette erreur est renvoyée quand aucune information (ancienne) sur le nick n'a été trouvée sur le réseau.

- **409 ERR_NOORIGIN**

Paramètres : <>

Réponse à : PING, PONG

Cette erreur est renvoyée quand il n'y a pas eu d'origine (de paramètre) envoyée avec cette commande.

- **411 ERR_NORECIPIENT**

Paramètres : <>

Réponse à : PRIVMSG

Cette erreur est renvoyée quand vous n'avez pas spécifié de destinataire pour un message.

- **412 ERR_NOTEXTTOSEND**

Paramètres : <>

Réponse à : PRIVMSG

Cette erreur est renvoyée quand vous n'avez pas spécifié de texte à envoyer pour votre message.

- **413 ERR_NOTOPLEVEL**

Paramètres : <masque d'utilisateur>

Réponse à : PRIVMSG

Cette erreur est renvoyée quand vous ne spécifiez pas de domaine dans un masque de destinataire pour un message.

- **414 ERR_WILDTOPLEVEL**

Paramètres : <masque>

Réponse à : PRIVMSG

Cette erreur est renvoyée quand vous avez spécifié une étoile (i.e. un joker) comme dernier caractère d'un masque destinataire pour un message.

- **431 ERR_NONICKNAMEGIVEN**

Paramètres : <nick>

Réponse à : NICK, WHOIS, HOWAS

Cette erreur est renvoyée quand vous n'avez pas indiqué de nick alors qu'il y en avait un d'attendu.

- **432 ERR_ERRONEUSNICKNAME**

Paramètres : <nick>

Réponse à : NICK

Cette erreur est renvoyée quand le nick contient des caractères non autorisés par le réseau.

- **433 ERR_NICKNAMEINUSE**

Paramètres : <nick>

Réponse à : NICK

Cette erreur est renvoyée quand le nick est déjà utilisé sur le réseau.

- **436 ERR_NICKCOLLISION**

Paramètres : <nick>

Réponse à : NICK

Cette erreur est renvoyée quand le nick est déjà utilisé sur un autre serveur.

- **442 ERR_NOTONCHANNEL**

Paramètres : <*chan*>

Réponse à : PART, MODE, INVITE, TOPIC, KICK

Cette erreur est renvoyée quand vous n'êtes pas présent sur le chan.

- **443 ERR_USERONCHANNEL**

Paramètres : <*nick*>

Réponse à : INVITE

Cette erreur est renvoyée quand le nick est déjà sur le chan.

- **461 ERR_NEEDMOREPARAMS**

Paramètres : <*commande*>

Réponse à : PASS, USER, OPER, KILL, JOIN, PART, MODE, TOPIC, INVITE, KICK

Cette erreur est renvoyée quand il manque un paramètre à la commande.

- **462 ERR_ALREADYREGISTRED**

Paramètres : <>

Réponse à : PASS, USER, SERVER

Cette erreur est renvoyée quand la connection a déjà été identifiée.

- **464 ERR_PASSWDMISMATCH**

Paramètres : <>

Réponse à : OPER

Cette erreur est renvoyée quand le mot de passe pour devenir IRCOP est incorrect.

- **467 ERR_KEYSET**

Paramètres : <chan>

Réponse à : MODE

Cette erreur est renvoyée quand la clé du chan est déjà définie.

- **471 ERR_CHANNELISFULL**

Paramètres : <chan>

Réponse à : JOIN

Cette erreur est renvoyée quand la limite de nombres d'utilisateurs présents sur le chan a déjà été atteinte.

- **472 ERR_UNKNOWNMODE**

Paramètres : <caractère>

Réponse à : MODE

Cette erreur est renvoyée quand le un code de mode de chan inconnu est utilisé.

- **473 ERR_INVITEONLYCHAN**

Paramètres : <chan>

Réponse à : JOIN

Cette erreur est renvoyée quand le chan n'est joignable que sur invitation (et que vous n'êtes pas invité).

- **474 ERR_BANNEDFROMCHAN**

Paramètres : <chan>

Réponse à : JOIN

Cette erreur est renvoyée quand vous tentez d'entrer sur un chan dont vous avez été banni.

- **475 ERR_BADCHANNELKEY**

Paramètres : <chan>

Réponse à : JOIN

Cette erreur est renvoyée quand la clé spécifiée pour le chan est invalide.

- **481 ERR_NOPRIVILEGES**

Paramètres : <>

Réponse à : SQUIT, KILL

Cette erreur est renvoyée quand vous tentez d'exécuter des commandes réservées aux IRCOPs alors que vous n'en avez pas le status.

- **482 ERR_CHANOPRIVSNEEDED**

Paramètres : <chan>

Réponse à : MODE, TOPIC, INVITE, KICK

Cette erreur est renvoyée quand vous tentez d'effectuer une opération d'administration sur un chan où vous n'êtes pas op.

- **483 ERR_CANNTKILLSERVER**

Paramètres : <>

Réponse à : KILL

Cette erreur est renvoyée quand vous tentez d'exécuter une commande KILL sur un serveur au lieu d'un utilisateur.

- **491 ERR_NOOPERHOST**

Paramètres : <>

Réponse à : OPER

Cette erreur est renvoyée quand votre domaine n'est pas enregistré pour accéder au rang d'IRCOP malgré votre tentative d'identification en tant que tel.

- **501 ERR_UMODUUNKNOWNFLAG**

Paramètres : <>

Réponse à : MODE

Cette erreur est renvoyée quand vous envoyez une commande MODE sur un utilisateur avec un mode inconnu.

- **502 ERR_USERSDONTMATCH**

Paramètres : <>

Réponse à : MODE

Cette erreur est renvoyée quand vous tentez d'effectuer une commande MODE sur un nick autre que le votre.

V - Divers

V-A - Voir le protocole IRC

Si vous souhaitez tester le protocole IRC par vous-même, vous pouvez utiliser *telnet*. Ce programme est disponible sur la plupart des plate-formes. Pour cela faites (en général c'est le port 6667 qui est utilisé pour IRC) :

```
telnet {nom de domaine} 6667
```

V-B - Voir aussi

[La RFC du protocole IRC \(RFC 1459\) \(Version Française\)](#)

[La mise à jour concernant l'architecture du protocole \(RFC 2810\).](#)