

Internet Relay Chat Protocol

Status of This Memo

This memo defines an Experimental Protocol for the Internet community. Discussion and suggestions for improvement are requested. Please refer to the current edition of the "IAB Official Protocol Standards" for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Abstract

The IRC protocol was developed over the last 4 years since it was first implemented as a means for users on a BBS to chat amongst themselves. Now it supports a world-wide network of servers and clients, and is stringing to cope with growth. Over the past 2 years, the average number of users connected to the main IRC network has grown by a factor of 10.

The IRC protocol is a text-based protocol, with the simplest client being any socket program capable of connecting to the server.

Table of Contents

1. INTRODUCTION	4
1.1 Servers	4
1.2 Clients	5
1.2.1 Operators	5
1.3 Channels	5
1.3.1 Channel Operators	6
2. THE IRC SPECIFICATION	7
2.1 Overview	7
2.2 Character codes	7
2.3 Messages	7
2.3.1 Message format in 'pseudo' BNF	8
2.4 Numeric replies	10
3. IRC Concepts	10
3.1 One-to-one communication	10
3.2 One-to-many	11
3.2.1 To a list	11
3.2.2 To a group (channel)	11
3.2.3 To a host/server mask	12
3.3 One to all	12

3.3.1 Client to Client	12
3.3.2 Clients to Server	12
3.3.3 Server to Server	12
4. MESSAGE DETAILS	13
4.1 Connection Registration	13
4.1.1 Password message	14
4.1.2 Nickname message	14
4.1.3 User message	15
4.1.4 Server message	16
4.1.5 Operator message	17
4.1.6 Quit message	17
4.1.7 Server Quit message	18
4.2 Channel operations	19
4.2.1 Join message	19
4.2.2 Part message	20
4.2.3 Mode message	21
4.2.3.1 Channel modes	21
4.2.3.2 User modes	22
4.2.4 Topic message	23
4.2.5 Names message	24
4.2.6 List message	24
4.2.7 Invite message	25
4.2.8 Kick message	25
4.3 Server queries and commands	26
4.3.1 Version message	26
4.3.2 Stats message	27
4.3.3 Links message	28
4.3.4 Time message	29
4.3.5 Connect message	29
4.3.6 Trace message	30
4.3.7 Admin message	31
4.3.8 Info message	31
4.4 Sending messages	32
4.4.1 Private messages	32
4.4.2 Notice messages	33
4.5 User-based queries	33
4.5.1 Who query	33
4.5.2 Whois query	34
4.5.3 Whowas message	35
4.6 Miscellaneous messages	35
4.6.1 Kill message	36
4.6.2 Ping message	37
4.6.3 Pong message	37
4.6.4 Error message	38
5. OPTIONAL MESSAGES	38
5.1 Away message	38
5.2 Rehash command	39
5.3 Restart command	39

5.4	Summon message	40
5.5	Users message	40
5.6	Operwall command	41
5.7	Userhost message	42
5.8	Ison message	42
6.	REPLIES	43
6.1	Error Replies	43
6.2	Command responses	48
6.3	Reserved numerics	56
7.	Client and server authentication	56
8.	Current Implementations Details	56
8.1	Network protocol: TCP	57
8.1.1	Support of Unix sockets	57
8.2	Command Parsing	57
8.3	Message delivery	57
8.4	Connection 'Liveness'	58
8.5	Establishing a server-client connection	58
8.6	Establishing a server-server connection	58
8.6.1	State information exchange when connecting	59
8.7	Terminating server-client connections	59
8.8	Terminating server-server connections	59
8.9	Tracking nickname changes	60
8.10	Flood control of clients	60
8.11	Non-blocking lookups	61
8.11.1	Hostname (DNS) lookups	61
8.11.2	Username (Ident) lookups	61
8.12	Configuration file	61
8.12.1	Allowing clients to connect	62
8.12.2	Operators	62
8.12.3	Allowing servers to connect	62
8.12.4	Administrivia	63
8.13	Channel membership	63
9.	Current problems	63
9.1	Scalability	63
9.2	Labels	63
9.2.1	Nicknames	63
9.2.2	Channels	64
9.2.3	Servers	64
9.3	Algorithms	64
10.	Support and availability	64
11.	Security Considerations	65
12.	Authors' Addresses	65

1. INTRODUCTION

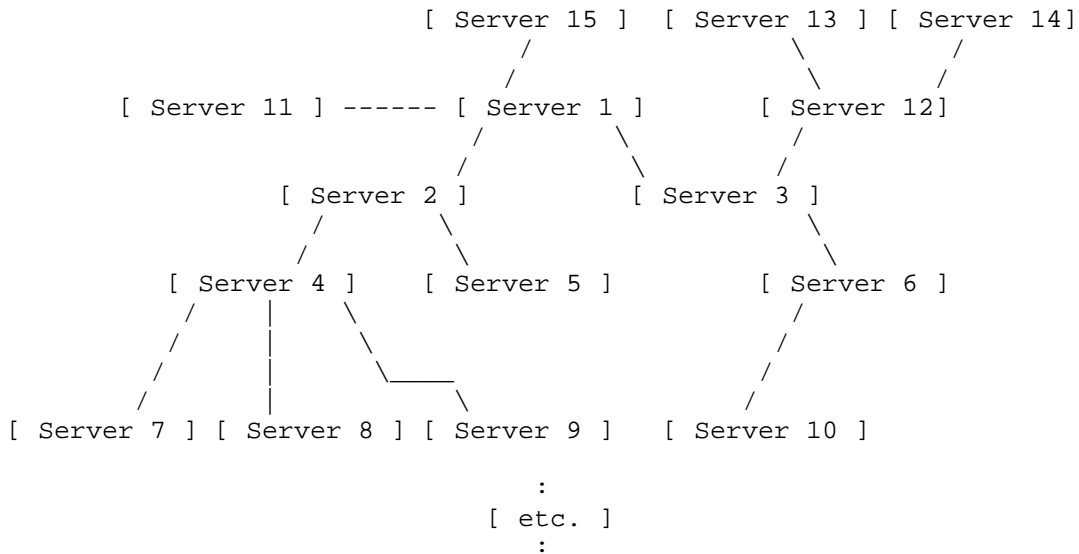
The IRC (Internet Relay Chat) protocol has been designed over a number of years for use with text based conferencing. This document describes the current IRC protocol.

The IRC protocol has been developed on systems using the TCP/IP network protocol, although there is no requirement that this remain the only sphere in which it operates.

IRC itself is a teleconferencing system, which (through the use of the client-server model) is well-suited to running on many machines in a distributed fashion. A typical setup involves a single process (the server) forming a central point for clients (or other servers) to connect to, performing the required message delivery/multiplexing and other functions.

1.1 Servers

The server forms the backbone of IRC, providing a point to which clients may connect to to talk to each other, and a point for other servers to connect to, forming an IRC network. The only network configuration allowed for IRC servers is that of a spanning tree [see Fig. 1] where each server acts as a central node for the rest of the net it sees.



[Fig. 1. Format of IRC server network]

1.2 Clients

A client is anything connecting to a server that is not another server. Each client is distinguished from other clients by a unique nickname having a maximum length of nine (9) characters. See the protocol grammar rules for what may and may not be used in a nickname. In addition to the nickname, all servers must have the following information about all clients: the real name of the host that the client is running on, the username of the client on that host, and the server to which the client is connected.

1.2.1 Operators

To allow a reasonable amount of order to be kept within the IRC network, a special class of clients (operators) is allowed to perform general maintenance functions on the network. Although the powers granted to an operator can be considered as 'dangerous', they are nonetheless required. Operators should be able to perform basic network tasks such as disconnecting and reconnecting servers as needed to prevent long-term use of bad network routing. In recognition of this need, the protocol discussed herein provides for operators only to be able to perform such functions. See sections 4.1.7 (SQUIT) and 4.3.5 (CONNECT).

A more controversial power of operators is the ability to remove a user from the connected network by 'force', i.e. operators are able to close the connection between any client and server. The justification for this is delicate since its abuse is both destructive and annoying. For further details on this type of action, see section 4.6.1 (KILL).

1.3 Channels

A channel is a named group of one or more clients which will all receive messages addressed to that channel. The channel is created implicitly when the first client joins it, and the channel ceases to exist when the last client leaves it. While channel exists, any client can reference the channel using the name of the channel.

Channels names are strings (beginning with a '&' or '#' character) of length up to 200 characters. Apart from the the requirement that the first character being either '&' or '#'; the only restriction on a channel name is that it may not contain any spaces (' '), a control G (^G or ASCII 7), or a comma (',' which is used as a list item separator by the protocol).

There are two types of channels allowed by this protocol. One is a distributed channel which is known to all the servers that are

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts



Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research



With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips



Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.