Kajetan Hinner (2000): Statistics of major IRC networks: methods and summary of user count. M/C: A Journal of Media and Culture 3(4). <originally: http://www.api-network.com/mc/0008/count.html> now: http://www.media-culture.org.au/0008/count.html - Actual figures and updates: www.hinner.com/ircstat/

Abstract

The article explains a successful approach to monitor the major worldwide Internet Relay Chat (IRC) networks. It introduces a new research tool capable of producing detailed and accurate statistics of IRC network's user count. Several obsolete methods are discussed before the still ongoing Socip.perl program is explained. Finally some IRC statistics are described, like development of user figures, their maximum count, IRC channel figures, etc.

Introduction

Internet Relay Chat (IRC) is a text-based service, where people can meet online and chat. All chat is organized in *channels* which a specific topic, like #usa or #linux. A user can be taking part in several channels when connected to an IRC network. For a long time the only IRC network has been EFnet (Eris-Free Network, named after its server eris.berkeley.edu), founded in 1990. The other three major IRC networks are Undernet (1993), DALnet (1994) and IRCnet, which split off EFnet in June 1996. All persons connecting to an IRC network at one time create that IRC network's user space. People are constantly signing on and off, the total number of users ever been to a specific IRC network could be called social space of that IRC network. It is obvious, that the IRC network's social space by far outnumbers its user space.

Several causes led to a separate development of IRC networks: fast growth of user numbers, bad scalability of the IRC protocol and content disagreements, like allowing or prohibiting Robot programs. Today we experience the development of regional based IRC networks, like BrasNet for Brazilian users. IRCnet concentrates mostly on European users, EFnet users mainly are from the Americas and Australia.

Although there has been research on IRC almost from the beginning on (Reid 1991), resources on quantitative development are rare. That was the reason for the development of the scripting language *Perl* IRC robot program Socip. It is running since almost two years now on several IRC systems and gives Internet researchers empirical data of the quantitative development of IRC.

Methodology

Any approach should fulfil the following tasks and needs:

- Store the number of users that are on an IRC network at a given time, e.g. every five minutes.
- Store the number of channels.
- Store the number of servers[1].

It is possible to get this information using the `/lusers` command on an IRC-II client, entered by hand. This was the first approach, it will yield results as in Table 1.

Date	Time	Users	Invisible	Servers	Channels
31.01.95	10:57	2737	2026	93	1637

Table 1 Number of IRC users on January 31st, 1995

During the first months of 1995 it was even possible to get all user information using the `/who **` command. On current major IRC networks with >50000 users this is denied by the IRC Server program which terminates the connection, because it is too slow to accept that amount of data fast enough.

To collect this data manually is a very exhaustive task. I tried several possibilities to run this automatically. The following sections give an overview of my research methodology path.

Eggdrop approach

One of the best-known IRC robot programs is called Eggdrop.[2] It consists of a C core with various modules which can be installed at any time. There were two reasons which made me abandon this approach. The first is a technical reason: All Eggdrop programs compiled and installed by the author created extensive CPU usage on the used Linux computer after some time. The reason for this behavior could not be found out. In any case, it would be impossible to run several Eggdrops simultaneously to research a number of IRC networks. The second

[1] Every IRC networks consists of a number of interconnected servers. A user signs on to a specific server, most usually a geographically near one. All these servers are constantly exchanging information about their connected users and take care of the communication between users.

[2] More information at http://www.xcalibre.com/eggdrop.htm

reason had to do with the statistics which should be obtained: Objective was to get a snapshot of current IRC users and IRC channel use every five minutes. This data should be written in an ASCII file. It was impossible for the author to extend Eggdrop's possibilities in a way that it would periodically submit the `/lusers` command and write the received data into a file. Because of these two reasons (and, moreover, security concerns) the Eggdrop approach was no success.

IrcII script approach

IrcII was the first Unix IRC client. It also had support for a special ircII scripting language. This made it possible to write command files which periodically submitted the `/lusers` command to any chosen IRC Server and logged the command's output. Four different scripts where used to monitor IRCnet, EFnet, DALnet and Undernet from January to October, 1998. These scripts were named Socius_D, Socius_E, Socius_I and Socius_U (depending on the network the script is running at). Every hour each script stores the number of users and channels in a logfile. These logfiles are examined using a Perl script.
There were some drawbacks of the ircII approach. First of all, it usually needs a terminal to run on. This can be avoided using the "screen" package and its virtual screens. It was possible to start ircII, run the scripts, detach, and log off again. The bigger problem, however, was the impossibility to restart ircII and scripts using the task-scheduler crontab or the like, in an automatic fashion (at least the author found no way to achieve this). That's why it was necessary to log into the machine, find out if the scripts are still running, and restart them if needed. Additional disadvantages were the unnecessary lengthy log files and the necessity of providing a second program which extracts the log file data and writes it into a second file, from which the graphs finally could be created. Reconnecting after the Server connection was lost also has been an unsolved problem. So manual checks to see whether the scripts were still running was necessary every day. Usually at least one script was not running after 10 hours. That's why the development of a completely different approach has been started, the one that is still used today.

Perl script only approach

Perl is a powerful script language to handle file-oriented data when speed is not extremely important. Its version 5 flavour allows a lot of modules to use for expansion, including the Net::IRC package. This object-oriented Perl interface to an IRC server enables Perl scripts to connect to an IRC server, use the basic IRC commands, etc.

Smart monitoring

The Socip.pl program includes all server definitions needed to create connections. Right now this Perl program is monitoring ten major IRC networks, including DALnet, EFnet, IRCnet, the Microsoft Network, Talkcity, Undernet and Galaxynet. The program names used are Socip_D.pl, Socip_E.pl, Socip_I.pl, Socip_M.pl, etc. This "**Soc**ial science **I**RC **p**rogram" selects at startup a Nickname from a list. For EFnet, the first nickname used is Socip_E1, the second one would be Socip_E2, etc. Using that nickname, Socip.pl tries to create an IRC connection to a server of the given network. If there is no failure, handlers are set up which take care of proper reactions to IRC messages from the server, like: Ping-pong, message output and reply, etc. Socip.pl joins the channel `#hose` then. This name has no special meaning, but is just a maintenance channel with the additional effect that every now and then real persons meet the robot program in that channel and try to interact with it. Those interactions are logged too. Sitting in that channel, the script sleeps for some time and periodically checks if a certain time span has passed (default: five minutes). After that, the `/lusers` command's output (User, Channel and Server information) is stored in a data file for each IRC network and the IRC network's RRD (Round Robin database) file is updated. This database, which is organised chronologically, offers great detail for recent events and more condensed information for older events. In our case, user and channel information younger than 10 days is stored in five-minute detail. If older than two years, the same information is automatically averaged and stored in a per-day resolution.
In case of network problems Socip.pl acts as necessary. For example, it recognises a connection termination and tries to reconnect after pausing using the next nickname on the list. This prevents nickname collision problems. If there is no response of the IRC Server to `/luser` commands for three times in a row, the next server on the list is accessed.
Special crontab-invoked scripts take care of restarting these Socip.pl scripts when necessary, like: termination of script because of network problems, IRC operator *kill*[3] or power failure. After a reboot all scripts are automatically restarted.

---

[3] Many IRC networks prohibit robot programs in their rules of conduct. IRC operators (marked with an asterisk *, not @ like IRC channel operators by the `/who` command) have the power to destroy a user's server connection using the IRC `kill` command. The user is then removed from the IRC network at once without any chance to avoid this. In addition, a specific user (based on its IP address and/or ident protocol information) can be banned from an IRC network by operator command also. All that can happen to robot programs like Socip.pl

All monitoring is done on a Linux machine (Pentium 120, 32 MB, Debian Linux 2.1) which is up all the time. Processor load is not extensive, and this machine also acts as the Sociology Department's WWW-Server.
The Socip.pl program can be obtained at no cost from http://www.hinner.com. Most IRC networks can be accessed with the original Net::Irc Perl extension, but for some special cases (e.g. Talkcity) an extended version is needed, which can be found also there.

Graphs creation

Now all the data is available in those RRD files and nice graphics can be created. This task is done using the MRTG program by Tobias Oetiker (*multi router traffic grapher*). A crontab-run script updates all IRC graphs four times a day. Each IRC network is visualised through five graphs: Daily, Weekly and Monthly users and channels, accompanied by two graphs showing all known data users/channels and servers. All this information is continuously published on the World Wide Web.[4]

Figures

To get an idea which graphs can be built using mrtg and Socip.pl, some samples are reproduced here. As already mentioned, graphs of all monitored networks are updated four times a day, with five graphs for each IRC network.
Fig. 1 shows the rise of EFnet' users from about 40000 in November 1998 to 65000 in July 2000. Sampled data is oscillating around an average amount, which is resulting from the different time zones of users.

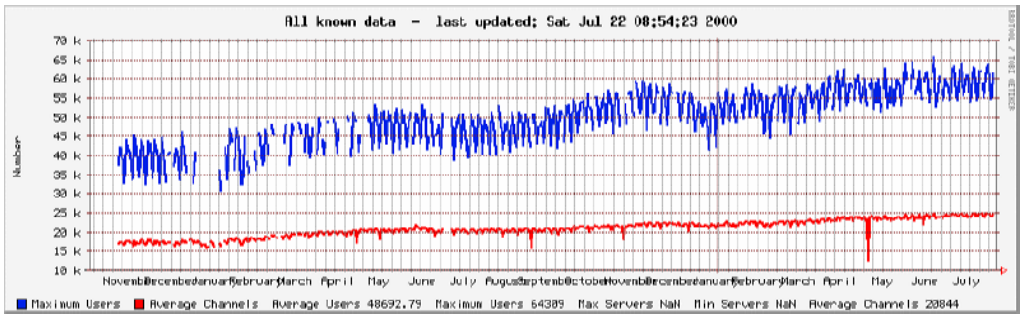

Fig. 1 EFnet - Users and Channels since November 1998

Fig. 2 illustrates the decrease of interconnected EFnet servers over the years. This means, that every server handles more and more users. Reasons for taking IRC servers off the net are security concerns (attacks on the server by malicious persons), new payment schemes, maintenance and cost effort.

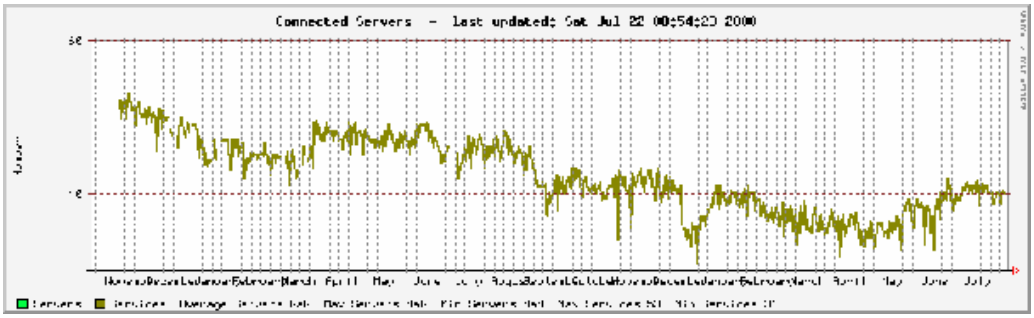

Fig. 2 EFnet - Servers since November 1998

A nice example of a heavily changing weekly graph is Fig. 3, which shows peaks shortly before 6pm CEST[5] and nearly no users shortly after midnight.

---

too.
[4] http://www.hinner.com/ircstat
[5] Central Europe Summer Time, as UTC+2, UTC is Universal Time Coordinated, which is GMT (Greenwich Mean Time). Throughout this article time notions refer to CEST.

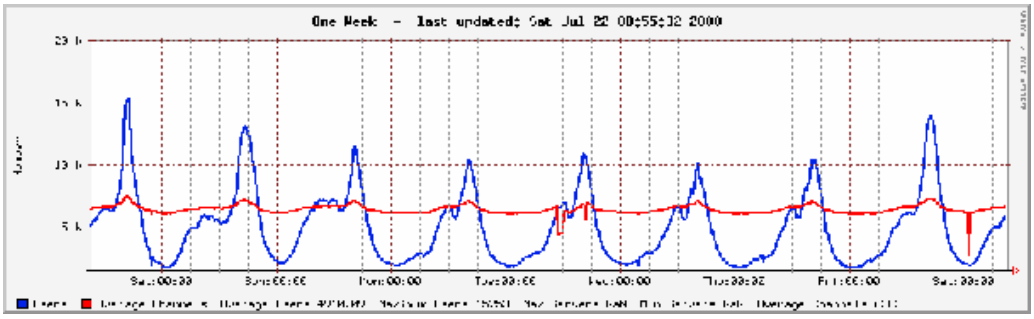

Fig. 3 Galaxynet: Weekly Graph (July, 15th-22nd, 2000)

With even more detail the daily graph portrays user variations: Fig. 4 is taken from Undernet user and channel data. The vertical gap in the graph indicates missing data, caused either by a net split or other network problems.

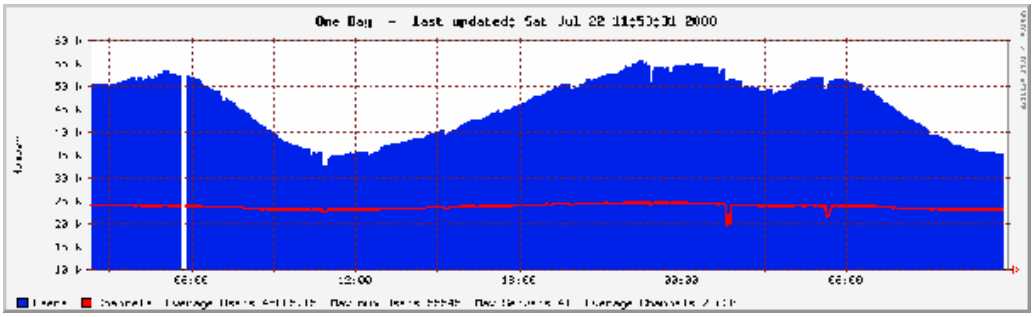

Fig. 4 Undernet: Daily Graph: July, 22nd, 2000

Finally a weekly graph (Fig. 5) is given, this time of the Webchat (http://www.webchat.org) network. It can be seen easily that every day the user count varies from 5000 to nearly 20000, and that channel numbers act accordingly from 2500 to 5000.

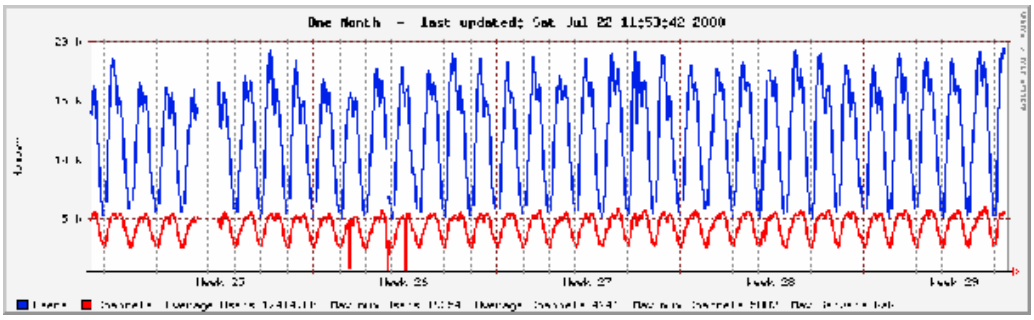

Fig. 5 Webchat: Monthly graph, Week 24-29, 2000

Not every IRC user is connected all the time to an IRC network. This figure may have increased lately with more and more flatrates and cheap internet access offers, but in general most users will sign off the network after some time. This is why IRC is a very dynamic society, with its membership constantly in flux. Maximum user counts only give the highest number of members who were simultaneously online at some point, and one could only guess at the number of total users of the network – that is, including those who are using that IRC service but are not signed on at that time. To answer these questions, more thorough investigation is necessary. Then in- and outflows can be guessed.

Table 2 shows the all time maximum user counts of seven IRC networks, compared to the average numbers of IRC users of the four major IRC networks during the third quarter 1998 (based on available data).

	DALnet	EFnet	Galaxy Net	IRCnet	MS Chat	Undernet	Webchat
Max. 2000	64276	64309	15253	65340	17392	60210	19793
3rd Q. 1998	21000	37000	n/a	24500	n/a	24000	n/a

Table 2 Maximum user counts of selected IRC networks

Compared to the 200-300 users in 1991 and the 7000 IRC-chatters in 1994 the growth is certainly extraordinary: it adds up to a total of 306573 users across all monitored networks. It can be expected that the 500000 IRC users threshold will be passed some time during the year 2001.
As a final remark it should be said that obviously Webchat-Systems will be more and more common in the future. These Chat-Services rely on Java or Javascript programs and not on standard IRC protocols. That means that it will be very hard to monitor such services. The actual number of chat users in the world could have already passed the half million landmark by now.

References

Eggdrop Information, http://www.xcalibre.com/eggdrop.htm
IRC statistics, http://www.hinner.com/ircstat
Mrtg, http://ee-staff.ethz.ch/~oetiker/webtools/mrtg/mrtg.html
Net::IRC, http://www.execpc.com/~corbeau/irc/dev/ or http://www.cpan.org/modules/by-module/Net/
Reid, Elizabeth (1991): Electropolis: Communications and community on Internet Relay Chat, Univ. of Melbourne.
Socip.perl, http://www.hinner.com/irc