

# DSHUB. THE MANUAL.

*Version 1.01.  
Pietricica, 14.10.07*

## **1. Introduction**

Welcome to DSHub world. What are you reading now is my idea of a manual. Hope that it will answer most of your questions and solve the most known problems that people have experienced while using it.

DSHub is a new hubsoft for Direct Connect Network. It uses the ADC protocol currently in development by Jacek Sieka and DC DEV. ADC is the future of DC network and information about it can be found on

<http://dcplusplus.sf.net/ADC.html> .

DSHub website is <http://www.death-squad.ro/dshub>

During the 8 months I worked on this project, I hit myself with a lot of human-based problems, and tried to figure out what people think about it. I've met people that installed DSHub with no word from me, and in a matter of minutes. I've also met people that had no idea what a shell is, or how to install the Java interpreter. ( luckily, DSHub has a GUI now so that people are not forced in using the shell )

DSHub is free and with no charge for everybody. I developed it in the idea of Open Source and for helping other people. Because I care.

That doesn't mean though that I have much responsibility about it. I can assure you that the soft is harmless, but still, I must state that it's your own responsibility, and if you choose to use it, you agree to the terms of the GNU General Public License ( the GNU GPL ) under which the soft is released. Please refer to this license in the file license.txt included in package or at the end of this manual.

## 2. What is ADC ?

ADC is the successor for the old NMDC protocol ( originally created by Jon Hess in the Direct Connect client).

ADC has a number of improvements and new ideas. Among them we can remember:

- Extensibility. Unlike the old NMDC, the new ADC can be virtually improved at any time by just adding extensions to it ( of course, they will be used only if both parties accept it), even the main ADC compatibility is advertised as BASE on connections. Anyone can write a draft for a new extension, make it public and try convincing other developers to add it to their own software.

- Scalability. This feature of ADC protocol makes possible to scale the messages send by clients in small pieces so that the information about them is more accurate and up-to-date. Only information that changed is being sent, so that bandwidth and load can be saved.

- Security. ADC uses unique IDs for every client that allow to globally identify any client. And also this ID cannot be used by other parties since it's created using another PrivateID which is secret, and they work only in pairs. Advantages are : the "clone" problem is being eliminated since a hub cannot allow two clients with same ID, registrations are much more safe based on ID , since it cannot be faked. Also passwords are sent encrypted; this feature eliminates the possibility of package sniffing.

- Safer downloads using TTH. TTH is mandatory for ADC downloading so the problem of corrupted downloads is eliminated.

ADC extensions bring up the Direct Connect out of the "dark ages" of file-sharing. That is possible because:

- REGX is ADC extension that uses Regular Expressions for searches. This improves a lot the searching capabilities and the accuracy of the results.

- ZLIB is an ADC extension that allows downloading and connecting via a GZip stream, which is compressed stream of data, enabling a much faster download for a not so big bandwidth.

- ADCS is the ADC Secure that uses the SSL ( Secure Socket Layer). Logins will be even more safe because the use of certificates, and downloads are encrypted so that nobody can intercept them.

For the future, we can say that a new type of downloading will be implemented , chunk-base , torrent-like.

The SET ( Similarity Enhanced Transfer) it's a new technology that finds alternates for downloading based not on equality but on similarity, for each piece of chunk the file is split into. More info can be found [on http://www.physorg.com/news95436100.html](http://www.physorg.com/news95436100.html).

Some ADC reminders:

- ADC uses the adc:// protocol specifier so you MUST use it in your connecting client ( unless your client has protocol auto-detect capabilities).

- ADC does not have a default port so the port MUST be specified even if its 411 ( NMDC default port )

- NMDC hublists do NOT support ADC so I found out 2 hublists for you :

[www.adchublist.com](http://www.adchublist.com)

[www.hubtracker.com](http://www.hubtracker.com)

Supporting the ADC protocol.

- Direct connect clients do not support ADC unless they are ADC compatible ( mentioned in their features )

- A list of ADC clients ( among with most known mods of them ):

DC++ >0.691, icedc 1.01a, zion >2.04 apexdc >0.3.0, strongdc > 2.01 , zk++ > 0.7, BCDC >0.69, FMDC, Elise or any later version.

### **3.3. Getting DSHub running**

DSHub uses the Java programming language developed by Sun Microsystems Inc. More info about it can be found on <http://java.sun.com/> .

This gives DSHub the advantage of being multi platform. What multi platform means? Exactly what is being said... it works properly on multiple platforms without any change. The most common platforms that I see usage are Windows and Linux. Wondering how this is possible? It's very easy, the Java Virtual Machine makes it possible. You must install a Java Virtual Machine on your system ( this one is system dependant ) and you can run any Java program, so you can run DSHub too.

#### **3.1. Windows Installation**

Installation on Windows shouldn't be so difficult, even for a beginner user. Go to DSHub sourceforge page, that is [www.sf.net/projects/dshub](http://www.sf.net/projects/dshub) and download the latest binary files ( binary files are usually marked with "bin" suffix and source files are marked with "src" suffix). The package may be called something like : dshub-zeta-rc5-bin.zip , this means the package is DSHub, version Zeta, release candidate 5 for this version , and that it's a binary package. Another example could be dshub-omicron-rc21-src.zip , again, this is the DSHub version Omicron , release candidate 21, the source code.

Normally an end-user needs the binary files ( the one you can execute ) and not the source code ( could be used by the ones who might like to improve DSHub or have a look on who I created things).

So, when you are trying to install DSHub, you need the binary package.

Another recommendation, when more versions are available, try getting the latest, because it always has something better then an old one, ok ?

Ok, now that we got that straight, we need to focus on the installation process.

You got the JRE installed? Skip this part then.

Go to <http://java.sun.com> or any other site that allows you to download the JRE ( that is, the Java Runtime Environment ).

DSHub can run with a JRE version 1.6.0.0 the least. So , if you get 1.7 or any latest, it should be ok.

Once you get the JRE installed on your system ( I presume you picked the right one and the right OS ), your system may have changed to reflect this.

Use any archive manager to unzip the DSHub package to any folder at your choice. Go in there using your Windows Explorer. If you see the file DSHub.jar in there with a coffee cup icon, it's super ( if you got extensions not to be shown, you will see just a DSHub file with coffee cup icon , but in fact it still is a .jar file ). Go ahead and double click it and you got the DSHub graphical interface show up, that's all there is in the installation.

There is the case when windows does not register your jar files with double clicking. This case is a bit tricky, and requires some shell skills from you ( remember, that old DOS box ?) .

Get a command prompt running ( you can do it by going to Start/Run/cmd ). You will see a nice prompter with your home folder shown. Now, change the directory to the one you unzipped DSHub. Change directory is the DOS command `cd`, example:

```
D:\Documents and Settings\Administrator>cd DSHub
D:\Documents and Settings\Administrator\DSHub>
```

If you got on another folder, you can always do  
`cd \dshub`

This gets you to the root of the drive and in in DSHub folder. Remember you can use tab for autocomplete folder and file names and X: to change drive, where X is the drive letter, and that names containing spaces should be quoted.

After you got into the DSHub folder, try `java -jar DSHub.jar`

```
> java -jar DSHub.jar
```

```
  Initializing DSHub Zeta ...
```

```
  This program is distributed in the hope that it will be useful,  
  but WITHOUT ANY WARRANTY; without even the implied warranty of  
  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the  
  GNU General Public License for more details.
```

```
  GUI launched...
```

```
  Done.
```

```
  Parsing to Command Mode.Help for info,quit to quit.
```

```
  Server created. Listening on port 411.
```

```
  Start Time:Sun Jul 22 19:58:07 EEST 2007
```

```
>
```

This is what console shows up. The GUI should appear as well. If you want to use GUI only you can run with `javaw` instead of `java`.

Possible problems :

*Could not open jar file* : This is because you did not `cd` to the right folder. Check again.

*Exception in thread...* : You are not using the right JRE version, 1.6 is the least you can use

*'java' is not recognized...* : Either you don't have java installed or its not in your PATH system variable. To correct it, you need to get the path to java ( usually its "c:\Program Files\java\jre1.6\bin" but might vary ) by looking into your computer, and add it to the PATH variable ( right click my computer, go to properties, advanced, system variables, and edit the PATH by adding the path to java).

Here you go, your hub is running ☺

You can go to configure chapter, where you can configure via GUI or via console at DSHub prompter.

## 3.2. Linux Installation

Installation on Linux shouldn't be so difficult, even for a beginner user. Go to DSHub sourceforge page, that is [www.sf.net/projects/dshub](http://www.sf.net/projects/dshub) and download the latest binary files ( binary files are usually marked with "bin" suffix and source files are marked with "src" suffix). The package may

be called something like : dshub-zeta-rc5-bin.zip , this means the package is DSHub, version Zeta, release candidate 5 for this version , and that it's a binary package. Another example could be dshub-omicron-rc21-src.zip , again, this is the DSHub version Omicron , release candidate 21, the source code.

Normally an end-user needs the binary files ( the one you can execute ) and not the source code ( could be used by the ones who might like to improve DSHub or have a look on who I created things).

So, when you are trying to install DSHub, you need the binary package.

Another recommendation, when more versions are available, try getting the latest, because it always has something better then an old one, ok ?

Ok, now that we got that straight, we need to focus on the installation process.

You got the JRE installed? Skip this part then.

If you got a distro that has Java Runtime Environment , this could be good or not. Most distros come with the GIJ , the so called gnu-java. But that comes with version 1.4.2 which is not good for DSHub. 1.6 is the least needed for it to run. When the GIJ will be DSHub compatible, I will post on manual. ( there are also other JVMs but I haven't tested them )

In most cases, you need a Sun JRE.

If you got a rpm-based distro, you can go to <http://java.sun.com> or any other site that allows you to download the JRE.

Or , you can use a package manager ( with all the repos you need ).

Debian / Ubuntu : synaptic

Suse : yast

Gentoo : emerge

Fedora/ Red Hat : yum

Centos : yum



DSHub can run with a JRE version 1.6.0.0 the least. So , if you get 1.7 or any latest, it should be ok.

Once you get the JRE installed on your system ( I presume you picked the right one and the right OS ), your system may have changed to reflect this.

Use any archive manager to unzip the DSHub package to any folder at your choice.

Get a terminal running, change the directory to the one you unzipped DSHub. Change directory is the cd command, example:

```
$ cd /home/DSHub
DSHub/$
```

If you got on another folder, you can always do  
`cd \dshub`

After you got into the DSHub folder, try `java -jar DSHub.jar`

```
$ java -jar DSHub.jar
```

```
  Initializing DSHub Zeta ...
  This program is distributed in the hope that it will be useful,
  but WITHOUT ANY WARRANTY; without even the implied warranty of
  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
  GNU General Public License for more details.
  GUI launched...
  Done.
  Parsing to Command Mode.Help for info,quit to quit.
  Server created. Listening on port 411.
  Start Time:Sun Jul 22 19:58:07 EEST 2007
```

>

This is what console shows up. The GUI should appear as well. If you want to use GUI only you can run with `javaw` instead of `java`.

Possible problems :

*Could not open jar file* : This is because you did not cd to the right folder. Check again.

*Exception in thread...* : You are not using the right JRE version, 1.6 is the least you can use

*Unknown command java* : You don't have Java installed at all, or not the right PATH set.

No window shows up ? : If you are running via tty with no X server or some kind of ssh terminal like putty, you will see

message "GUI not viewable" and you will have to use console to manage the hub. The DSHub has a nice prompter that can take commands like the registered users via DC client.

Also, when running via console, there are possible problems. If you run with `bash &` ( background running ) the software stops responding ( don't know why, but it freezes, can be JVM issue ). So I found out that either you can run via `javaw` ( if you don't need the console at all ), or you can try `screen`. `Screen` is a nice program that most distros include, and you can use it to free your console, but keep the DSHub prompter ready to return to.

To start the screen named DSHub:

```
screen -A -m -d -S DSHub java -jar DSHub.jar
```

To restore the screen :

```
screen -r DSHub
```

To detach from screen and go back to shell prompt, type `Ctrl+a, d`

Here you go, your hub is running 😊

You can go to `configure` chapter, where you can configure via GUI or via console at DSHub prompter.

## 4. Configuring DSHub

DSHub is very easy to configure via GUI. I will explain all the options, and find their equivalent in console and visual interface.

When you first run DSHub, it automatically tries to listen to the TCP 411 port.

DSHub "always" runs.

If you want to stop it, just close it.

This can be done very easy, via the Exit button in GUI or by typing "quit" ( unquoted of course ) and hitting enter.

When hub is (re) started, it tries to apply port setting and listen to connections . Though, this may not have always success. Causes are: insufficient rights ( not Administrator or root ) , or the address to be in use. Anyway, all possible messages are being shown so you can correct them and press "restart" button or give "restart" command in console

"help" – this command shows all other commands with minimal description.

The standard help file ( Eta version ) :

*Available Commands :*

*about -- The program credits.*

*adc -- ADC advanced configuration panel, setting contexts for each command.*

*bancid -- Bans a given cid or the cid of the given online user.*

*banip -- Bans a given ip or the ip of the given online user.*

*bannick -- Bans a given nick, drops if nick online.*

*cfg -- The hub variables.*

*cmdhistory -- Lists the last history\_lines from given commands by logged users.*

*drop -- A kick with no reason or ban time, just drop/Extended drop; use with no arguments to show info.*

*grant -- Offers the possibility of editing an account's profile; use with no arguments for info.*

*gui -- Brings up the gui to server if available.*

*help -- This screen.*

*hideme -- Toggles if you are hidden or not in userlist.*

*history -- Lists the last history\_lines from chat.*

*info -- Lists some useful information about a user,ip or cid.*

*kick -- Kicks out the user given by nick, add extra words for reason/Extended kick; use with no arguments for info.*

*listban -- Lists the current banned CIDs/IPs/nicks.*

*listreg -- Lists the current registered CIDs.*

*mass -- Broadcast message, takes extended parameters; use with no arguments for info.*

*mynick -- Changes your nick to new specified one.*

*password newpass -- Changes your current password, where newpass is the new password.*  
*port x -- Change default port to which hub is listening to x.*  
*quit -- Shuts down hub.*  
*reg CID/online user nick -- Reg the new CID with no password (by default) or the CID of the online user specified by nick.*  
*If already registered, display registration info.*  
*rename -- Renames the user given by nick to new nick given.*  
*restart -- Restarts hub.*  
*stats -- Hub statistics.*  
*topic newtopic -- Where newtopic is the new desired topic; use with no arguments to delete current topic.*  
*unban -- Unbans the specified, looking in CID/IP/nick order.*  
*ureg CID/online user nick -- Unregs the CID/user's CID from database.*  
*usercount -- Info about the current user count.*

### Command explanation:

- a) **About.** Shows the name of the soft developer, aka me.
- b) **Adc.** This command is the ADC advanced configuration panel , which allows enabling and disabling all the client ADC commands that hub may receive, function of their contexts. If you use with no parameters, the following table + info appears:

ADC Advanced Configuration Settings.

-----  
 To modify a value use " adc \_context\_\_name\_ on/off "

Example : "adc bmsg off", where "b" is the context, "msg" is the name and off is the specifier of what to do.

Current Settings :

MSG :	STA:	CTM:	RCM:	INF:	SCH:	RES:	PAS:	SUP:
B on	B off	B off	B off	B on	B on	B off	B off	B off
D on	D on	D on	D on	D off	D on	D on	D off	D off
E on	E on	E on	E on	E off	E on	E on	E off	E off
F on	F off	F off	F off	F off	F on	F off	F off	F off
H on	H on	H off	H off	H off	H off	H off	H on	H on

If you try to use "adc command", you will get a screen similar to this:

ADC Advanced Configuration Settings.

-----  
 EMSG is currently on.

If you use "adc command on/off", the command will be enabled/disabled:

ADC Advanced Configuration Settings.

-----  
 Setting EMSG off.

For info about each command and its context, refer to GUI's [?] boxes or the link to ADC protocol draft.

- c) **Bancid**. This command has a parameter that can be either an CID ( it bans the CID ), or a nick. In last case, the CID of the user with that nick is being banned.
- d) **Banip**. Same as bancid, but on IP.
- e) **Bannick**. Same as bancid, but on nick.
- f) **Cfg**. The command that allows to set the hub by configuration variables. What can be changed about hub ( except port and topic) is here. All variables are described in sections below.
- g) **Cmdhistory**. Same as history, but shows the commands that ops have taken. ( Except password ;)
- h) **Drop**. A kick with no ban. Can take extended arguments like kick and mass.
- i) **Grant**. A command that can change the profile of a registered user, by adding or removing dynamically attributes. There is an attribute for each command listed here, plus some attributes regarding the user's access and view.

The grant command:

Usage grant user/CID [[+-]attribute1]\*

+attribute adds the attribute to the registered user, - removes it.

[+-]all adds all possible attributes.

List of attributes: about adc bancid banip bannick cfg cmdhistory drop grant gui help hideme history info kick listban listreg mass mynick password port quit reg rename restart stats topic unban ureg usercount flyable key kickable nickprotected overridefull overridesbare overridespam renameable

**Example:** grant Pietry +kick+info+stats-adc-cfg

**Will create a screen :**

Editing Account: ELQ4PY6NGDN4DR6L27HYUACXI6YYQ6DO3PBOCPI

kick modified to true

info modified to true

stats modified to true

adc modified to false

cfg modified to false

Done.

You cannot grant an attribute you don't possess, and you can't grant if you aren't a granter.

The attribute all tries to grant all attributes to specified account.

- j) **Gui.** This command brings up the GUI on the server if hidden via the HideMe button. Else, the message : "GUI not viewable" is printed.
- k) **Help.** Shows the help screen.
- l) **Hideme.** This command hides the user taking it from the other users, not appearing in user list anymore. With no arguments, has a toggle usage. Hidden status is being kept even when reconnect.
- m) **History.** This command shows the last history\_lines from chat, who are being kept in hub memory. History\_lines is a cfg variable. This is particularly useful when you want to see a chat that you missed ( connection fail, a night's chat, etc. )
- n) **Info.** This is a versatile command. The argument can be an online user. In which case, the user info is printed ( among with reg info if user is registered). If the argument is an ip/cid, then info about it is shown ( who is using it ).
- o) **Kick.** Same as mass, can have extended features, this command disconnects users and tempbans them for the amount in cfg variable kick\_time.  
**The ban is CID based.**
- p) **Listban.** Lists the current bans and with a last nick for each one.
- q) **Listreg.** Minimal command that brings up a list of regs and the last nick they were seen with. In the future , will be deprecated or improved.
- r) **Mass.** Command that sends a mass message to users on the hub. To what users send? It's selectable via regex or fields.

Mass Command:

Broadcasting in DSHub is very simple now.

Classic mass:

Mass message to all users, example: "mass all text".

Extended mass has way more advantages and can be used very efficiently with a large hub.

Extended mass features:

Sending to users that match a certain regular expression:

Example: `!mass \[RO\].* text --` this command sends mass to all users that have their nick starting with [RO]

Example: `!mass .. text --` this command sends mass to all users with 2 letter nicks

This type of mass command accepts just any regular expression.

Sending to users that have their fields checked:

Example: `!mass share<1024 text --` this command just sends text to all users with share less then 1 gigabyte.

Example: `!mass sl=1 text--` this command just sends text to all users with exactly one open slot.

Example: `!mass su!tcp4 text --` this command just sends text to all passive users.

Extended mass has the operators `>`, `<`, `=`, `!`

And a list of possible fields : all ( to everybody ) share, sl (slots), ni (nick length),su(supports, accepts only = or !, example: `!mass su=tcp4 text`),hn(normal hubs count),hr(registered hub count),ho(op hub count),aw(away, 1 means normal away, 2 means extended away),rg (1-registered, 0 otherwise, registered means not op),op ( 1 - op, 0 - otherwise , op means it has key).

- s) **Mynick.** Changes the current nick of oneself, to the one given as argument in command.
- t) **Password.** At any time, the user taking the command may change his registration password. The command does not appear in cmdhistory.
- u) **Port.** This command changes the default port hub is supposed to run on. It's the only command that needs a restart to apply. Until restart, nothing happens. Requires integer argument.
- v) **Quit.** This command shuts down completely, and to start it again you need to open it again from server.

- w) **Reg.** This command creates a standard reg. There are possible situation of this command's behavior, according to the parameter. If a CID is supplied, the CID is being regged ( if a user is currently using the CID, is being regged ). If a nick is supplied, actually his CID is being regged. Nick has nothing to do with the reg, is just a easy way to reg a CID ( to take from a online user ). If a CID or a user is already regged, information about it is being shown, the reginfo. So , this command servers as info for a reg too.  
Also, the reg is made with no password, just CID check. See the password command on how a user can set his password.
- x) **Rename.** First argument is the nick of an online user, and second the new nick the user is supposed to have. Simply changes the nick. Ops can be renamed only if the rename\_ops variable is set to 1.
- y) **Restart.** This command restarts hub completely, dropping all users, after a 5 second timeout.
- z) **Stats.** Shows all the hub statistics, including start time, usercount and OS version where the hub runs on.
- aa) **Topic.** Changes the current hub topic, to the string given, if nothing given, the topic is deleted.
- bb) **Unban.** This command is a generic ban deleter. In DSHub there are 3 types of banning: IP, CID and nick. Unban can take any of these 3 parameters, and unban it ( delete it from ban database ).
- cc) **Ureg.** This command unregs a CID or a online user's CID. Same like reg, the argument can be a CID or a online user, in which case he is deleted. The name ureg and not unreg, is Unix style.
- dd) **Usercount.** This command shows the number of users that are currently online hub and the number of sockets open ( users trying to login ).



## 5. Cfg Variables.

Cfg Variables list:

timeout_login	20	-- Number of seconds for hub to wait for connecting users until kick them out.
hub_name	hub of Administrator	-- Hub name to display in main window.
max_ni	64	-- Maximum nick size, integer.
min_ni	1	-- Minimum nick size, integer.
max_de	128	-- Maximum description size, integer.
max_share	10485760	-- Maximum share size, long
min_share	0	-- Minimum share size, long
max_sl	1000	-- Maximum slot number, integer.
min_sl	0	-- Minimum slot number, integer.
max_em	128	-- Maximum e-mail string size, integer.
max_hubs_op	70	-- Maximum hubs where user is op, integer.
max_hubs_reg	30	-- Maximum hubs where user is reg, integer.
max_hubs_user	200	-- Maximum hubs where user is user, integer.
max_sch_chars	256	-- Maximum search chars, integer.
min_sch_chars	3	-- Minimum search chars, integer.
max_chat_msg	512	-- Maximum chat message size, integer.
max_users	1000	-- Maximum number of online users, integer.
kick_ops	0	-- 0 = ops can't be kicked/banned, other value = they can be kicked/banned.
rename_ops	0	-- 0 = ops can't be renamed, other value = they can be.
history_lines	50	-- Number of lines to keep in chat history.
opchat_name	OpChat	-- The Operator Chat Bot Nick.
opchat_desc	BoT	-- The Operator Chat Bot Description.
kick_time	300	-- The time to ban a user with a kick, in seconds.
msg_banned	Have a nice day and don't forget to smile !	
-- The additional message to show to banned users when connecting.		
msg_full	Have a nice day and don't forget to smile !	
-- Message to be shown to connecting users when hub full.		
reg_only	0	-- 1 = registered only hub. 0 = otherwise.
nick_chars	ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890[ ]()-.,;`~*%^\$#@!+=_{ }<>: -- Chars that could be used for a nick, String.	
chat_interval	500	-- Interval between chat lines, millis, Integer.

```

keep_alive_interval      120          -- Interval between keep_alive
messages, seconds, Integer.
save_logs                1           -- 1 = logs are saved to file, 0
otherwise.
automagic_search         36          -- Interval between automagic
searches for each user, seconds, Integer.
search_log_base          2000        -- Logarithmic base for user
searches interval, millis, Integer.
search_steps             6           -- Maximum nr of search steps
allowed until reset needed, Integer.
search_spam_reset        300         -- Interval until search_steps
is being reset, seconds, Integer.
msg_search_spam          Search ticket Reserved.
Please be patient while search
is being processed.
Do NOT close this window or start other search
or you will lose this search !      -- Message that appears as a
result when search is delayed, String.
bot_name                 DSHub       -- Hub security bot name,
String.
bot_desc                 www.death-squad.ro/dshub      -- Hub
security bot description, String.

```

This is the cfg variables list. The current list of all hub variables that may be changed via the cfg command. To see them all, use *cfg list*. To change one of them, use *cfg < variable name > < new value >*

I think that the variables are quite clear, any question, you can ask on forums or on my test hubs ( address mentioned on site ).

But there is necessary to explain the logarithmic search scale I implemented in DSHub.

DSHub has now powerful searching features.

First, we need to make a distinction between the automagic and the user searches.

First type is made by client at a regular interval and DSHub keeps a linear spam setting.

Second type are user searches ( manual searches ) that the user takes.

For this type (because of the human factor) DSHub keeps a logarithmic spam setting.

This way, the 2nd search is at  $\text{search\_log\_base}$  interval, but third, is at  $\text{search\_log\_base}^2$

and so on, until the power gets to max\_steps.

After this point, the user needs to wait search\_spam\_reset seconds to get his burst back.

The searches are being kept in queue ( not ignored ! ) and are processed once the timeout is completed so user doesnt need to search again but just wait for his search to be completed.

The messages appears as a fictive result in his search box, which will be filled once the search string is being sent to others.

## **6. Conclusion.**

DSHub has innovative ideas; it's the first ADC hubsoft with a nice graphical user interface ( thanks MAGY for GUI ideas and support ), first with an logarithmic search scale ( not linear ), the first that can take commands using regex and field based.

It's still in development and will implement more and more interesting ideas I have. If you got any of yourself, I am open to them, so just paste of forum and I will get to you.

I really hope that DSHub will make people happier and they will have fun using it.

I would like to thank all others who helped me and supported me all this time.

I love you all and hope we will hear each other soon.