DATASPY NETWORK X BETA 0.5

Users Guide

```
      Status : Released-Sunday, 31 March 2002
 Short Title : DSNX_0.5B_MANUAL.pdf
    Revision : 0
   Copyright : All rights reserved.
```

Dataspy Network X

Introduction

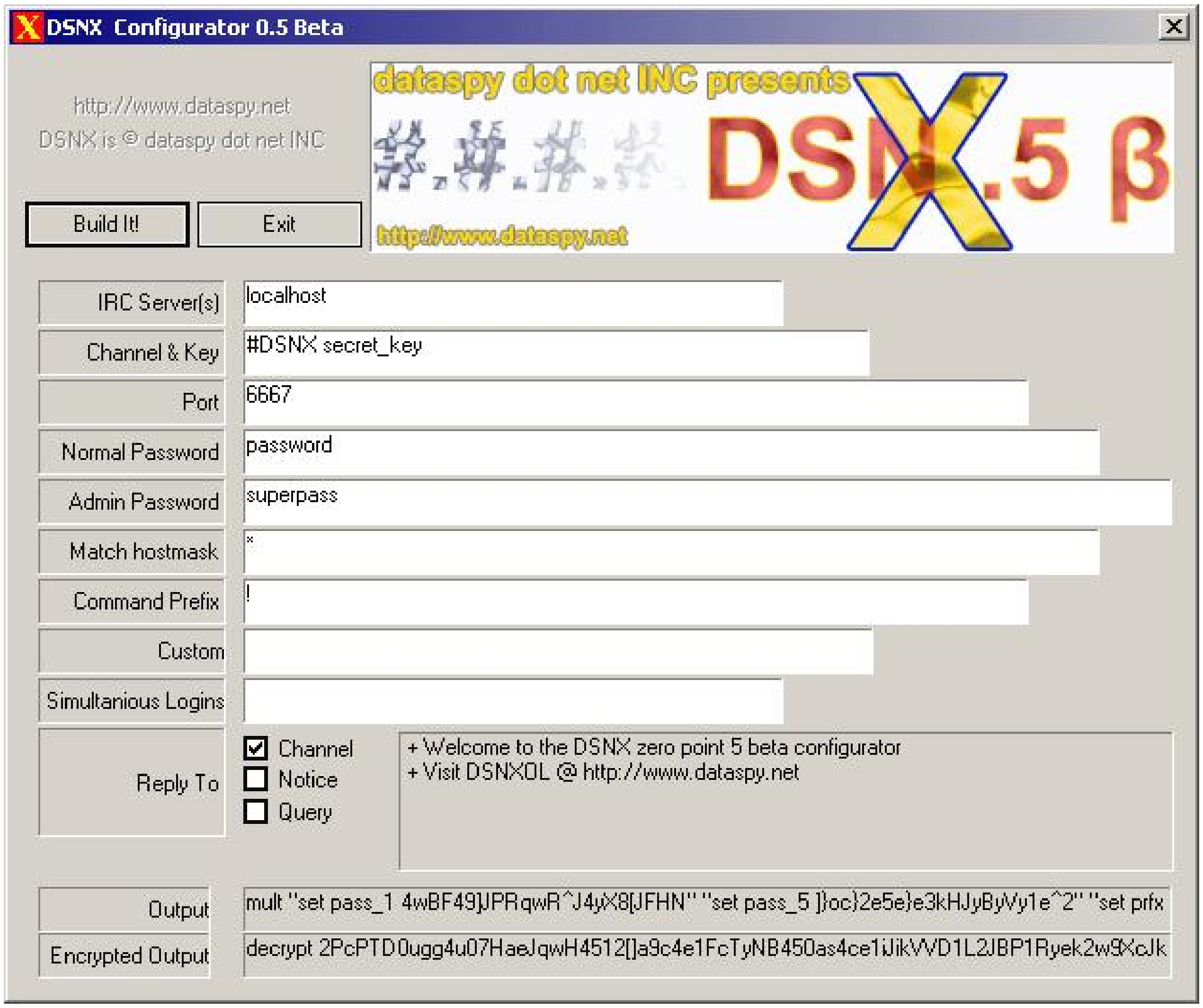
DSNX is an advanced, open source, modular, non–interactive IRC client (An irc robot). It provides the subsystem for a versatile internet technology deployment across multiple systems. Through the patented DSNX technology dataspy dot net INC can provide your fortune 500 company with the ultimate solution.

There are a number of other IRC robots currently available, and even more in development, but none that offer the performance, stability and wide range of features like DSNX does.

So sit back, *relax* and learn how you can use DSNX technology in *your* business today!

Getting Started

Once you have obtained your DSNX license you can continue to deploy the DSNX client.


DSNX Configurator 0.5 Beta
http://www.dataspy.net
DSNX is © dataspy dot net INC
Build It!
Exit
dataspy dot net INC presents
DSNX.5 β
http://www.dataspy.net

IRC Server(s)	localhost
Channel & Key	#DSNX secret_key
Port	6667
Normal Password	password
Admin Password	superpass
Match hostmask	*
Command Prefix	!
Custom	
Simultanious Logins	
Reply To	[✓ Channel] Notice Query

+ Welcome to the DSNX zero point 5 beta configurator
+ Visit DSNXOL @ http://www.dataspy.net

Output: mult "set pass_1 4wBF49]JPRqwR^J4yX8[JFHN" "set pass_5]}oc}2e5e}e3kHJyByVy1e^2" "set prfx

Encrypted Output: decrypt 2PcPTD0ugg4u07HaeJqwH4512[]a9c4e1FcTyNB450as4ce1iJikVVD1L2JBP1Ryek2w9XcJk

When you first run the DSNX Configurator, it looks like the above screen shot.

Configurator Options:

IRC Server(s) –

The server(s) where the DSNX client connects to upon startup. If more than one server is specified (separated by spaces) a random one from the list will be chosen to connect to. If unable to connect to that one, the next one in the list will be picked, and so forth.

Channel & Key –

The channel (including key) the DSNX client joins upon connecting to the irc server.

Port –

The port of the irc server to connect to.

Normal Password –

<PREFIX>set pass_1 MD5_Hash

Password required to access to all commands except ‘set’ ‘upgrade’ and ‘update’.

Admin Password –

<PREFIX>set pass_5 MD5_Hash

Password required to access all commands.

Match Hostmask –

<PREFIX>set hsts hostmark(s)

IRC hostmask (nick!ident@host) that must be matched before a ‘login’ is allowed. Characters ‘*’ and ‘?’ indicate anything and a single number respectively.

Eg: *nick* or *!nick@host or * for everyone.

Command Prefix –

```
<PREFIX>set prfx prefix
```

The prefix required before all commands. If no prefix is set, none is required before commands. This isn’t recommended however as the Client will keep replying with errors when you aren’t typing to it, as it cannot distinguish between normal chat and a command.

Custom –

Enter any custom commands you would like executed upon startup – excluding prefix. (not required)
Eg: “plugins add *.dsnx”

Simultaneous Logins –

```
<PREFIX>set mlgn number
```

The maximum number of simultaneous logins per mainbot. No number indicates no limit.

Reply To -

```
<PREFIX>set rptc ‘c’ or ‘n’ or ‘q’
```

Default reply to channel or notice or query.

Startup

Upon initialisation, the DSNX client reads the startup information from the registry, which is then executed.

Such startup information could include:

```
SET HSTS *
SET RPTC c
SET PASS_1 MD5-HASH
M #channel irc.server.com port
M “#channel key” irc.server.com
```

Etc

Other Notes

Parsing input: The DSNX client has specific input parsing.

It splits commands by spaces.

Eg: <prefix>raw privmsg #c :d e f gh I k

Would be split into:

```
“raw” “privmsg” “#C” “:d” “e” “f” “gh” “I” “k”
  0        1       2    3    4   5    6    7   8
```

You can tell the dsnxclient not to split something by spaces by using quotation marks: ‘”’

Eg: <prefix>”raw” “privmsg #c :hi whaaaza”

Would be split into:

```
“raw” “privmsg #c :hi whaaaza”
  0              1
```

Also you can put a a backslash char to specify its not a splitting mark

Eg: <prefix>raw “privmsg #chan :hi \”nt”

Would split into:

“raw” “privmsg #chan :hi “nt”

rather than:

“raw” privmsg #chan :hi “ “nt”

Authentication levels: The DSNX client provides auth levels, which are between 0 (zero) and 5 (five). The passwords for each level can be unique, and are set using the 'PASS_$' setting where $ is a number between 1 (one) and 5 (five) to indicate which authentication level the password applies to.

An authentication level of 0 means the user has not logged in.

If no password has been set for a particular auth level, users are unable to login for that level.

See the reference for the *login* and *set* commands as mentioned.

Decrypt – Decrypts previously encrypted text

```
<PREFIX>D ENCRYPTED_TEXT
```

Once decrypted, if the checksum is correct, the text will be executed with the same auth_lvl as the user calling *Decrypt*.

```
Eg: .D ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890
<Dclient> OUTPUT_IS_VARIABLE
```

Make sure you have read the reference for the *Encrypt* function as well, as these two functions complement each other.

Login – Logs a user into the DSNXClient

`<PREFIX>login password`

The DSNXClient will check the password against the passwords supplied for admin and normal login, if it matches it will log the user into that auth level.

Note: If a user who is logged in either:

Leaves a channel that the DSNXClient is in,
Quits from the irc server
Joins a channel the DSNXClient is in or
Changes nicknames they are automatically logged out.

If their hostmask changes they are also logged out.

Encrypt – Encrypts text for use with *Decrypt*

<PREFIX>E TEXT_TO_ENCRYPT

Encrypts the text using a 128 bit encryption algorithm, adds a checksum and converts it to a HEX format suitable for transmition over IRC.

```
Eg: .E "r PRIVMSG #channel :Unencrypted"
<Dclient> USER-
C[1O]3[brQf42dAtA2x[I^bjW]]tKv3l98E43MW{Sl8
4tO}0G0Ov]p]d}pv3K^nv{0CdWj5[
```

Notes about *encrypt*:

A- If the DSNX client that is doing the encryption & decryption has joined any channel where a topic is set, the encryption will only be valid until the DSNX client joins another channel with any topic. However, if the bot joins another channel, then rejoins the original channel (and the topic hasn't been changed in the original channel), the encryption is valid again. In other words, the last channel the DSNX client has joined determines the encryptions validity.

B- If there is no topic set in the last channel the DSNX client joined, this also generates valid encryption (i.e. the type used at start-up) which can be used by any client, at any time, so long as the last channel the DSNX client joined had no topic set.

C- Note that you don't need to include the command prefix in any commands except ones directly sent to the DSNX client on IRC.

Info – Prints current DSNXClient stats.

```
<PREFIX>Info
```

Reply: (version up 000:00 on winxx with xx threads)

Includes the version, how long the process has been running, the parent operating system and the number of threads running in the dsnx client (Eg ident server and mainbot = 2 threads).

```
Logout – Logs the user out of the dsnxclient
    <PREFIX>logout
```

Mainbot – Creates a mainbot.

```
<PREFIX>mainbot channel+key <server> <port>
```

Will create a new mainbot in the specified channel on the specified server and port.

If server or port are not specified the current server and ports will be used.

```
Eg <PREFIX>mainbot #dsnx irc.lcirc.net
   <PREFIX>mainbot “#channel secret_key”
server.com
```

```
Nick  – Gets the DSNXClient to change nickname
   <PREFIX>Nick <max_length>

   Chooses a random nick. Default max length
   of nicknames is 9 chars.
```

Plugins – Manipulate plugins

```
<PREFIX>Plugins add search_mask
```

Will search the specified path for plugins and try to load them. Will not load plugins more than once.

```
Eg: <prefix>plugins add c:\*.dsnx
   <prefix>plugins add *.dll
```

```
<PREFIX>Plugins del name
```

Unloads the specified plugin.

```
Eg: <prefix>plugins del “portscan v2”
```

```
<PREFIX>Plugins list
```

Prints the current plugins loaded and the functions they provide

Quit – Quits the DSNXClient from the irc server.

```
    <PREFIX>quit <number>
```

If number is specified will quit all the dsnx clients on that process.

```
Raw – Send a raw IRC message
    <PREFIX>raw <params>…

    Some examples:

    Raw join #channel key
    Raw privmsg #channel :lol im a dsnxclient
    Raw part #channel
    Etc…
```

slist – Manages server lists

<PREFIX>slist new name server1 server2 <…>

Creates a server list called 'name' with the servers specified. Unlimited servers allowed (within reason :D)

The server list name can then be used in place of any other server name within the DSNXClient, and upon connecting a random server will be chosen from the list.

If an error occurs while connecting to any server the next one in the list is tried and so on..

Eg:
Slist new cool_servers irc.cool.com cool.redirect.com better.redirect.com other.redirect.com

Mainbot #channel cool_servers

(Would connect to a random one of those servers)

slist new port_redirects irc.dla.net irc.eu.dla.net irc.us.dla.net

portr new 6667 port_redirects

(Now if you connected to port 6667 on the dsnxclient machine it would redirect you to a random server from that list)

<PREFIX>slist del name

Deletes the named server list

<PREFIX>slist list

Lists the server lists available

```
webdl – Download a file from a http server
    <PREFIX>webdl url <file>

    Downloads url and if <file> is specified,
    saves it to <file> otherwise chooses a
    random file name with the same extension as
    the one being downloaded.
```

Run – runs a file on the dsnx client machine

    <PREFIX>run file <don’t_hide>

    Runs file, and if anything is specified
    after file it will run it with SW_SHOW
    meaning it isn’t hidden.

alias – Manages aliases

```
<PREFIX>alias new alias_name alias_specs
```

Alias_name must be unique (ie can’t be a command already in the DSNXClient) and must also not contain spaces.

Alias specs is the command executed when <prefix>’alias_name params’ is called.

Variables available in alias_specs are:

```
$user          -User calling the command
$chan          -channel the command is
called
$me            -dsnxclients nickname
$<1-9>         -params 1 to 9
$+             -removes spaces
```

Eg:

```
Alias new mmmk “raw privmsg $chan :$user
type $prfx $+ mmk $1”
```

Note that commands within the alias do not require a prefix, so its raw not <prefix>raw or mainbot not <prefix>mainbot

```
<PREFIX>alias del alias_name
```

Removes alias_name

portr – Manages port redirects/proxys

```
<PREFIX>portr new in_port <server>
<out_port>
```

If <server> and <out_port> aren’t specified, creates a HTTP CONNECT proxy on in_port, otherwise creates a simple redirect on in_port to server out_port.

A http CONNECT proxy is used in mIRC and ICQ etc as http under firewall settings.

```
<PREFIX>portr del listening_port
```

Removes the proxy or port redirect listening on listening_port

Md5 – Calculates the md5 hash of param1

```
<PREFIX>md5 string
```

Returns the md5 hash of string as used in the pass_number settings.

Set startup – updates the startup cmds in the registry

```
<PREFIX>setstartup new_command(s)
```

Replaces the startup commands in the registry with param1 (You can get the new startup commands you wish to use from the editserver)

Make sure you specify non-splitting by using quotation marks. Eg setstartup “decrypt asd0as9d0as89d0”

Uninstall – Removes the startup reference

```
<PREFIX>uninstall
```

Note: You will have to call this twice to confirm.

Set – updates settings

```
<PREFIX>set setting_name <new_value>
```

If new_value isn’t specified, it will delete the setting called setting_name.

If no setting by setting_name exists one will be created.

If one already exists it will be updated.