TeamSpeak 3
ServerQuery Manual (2011-11-17)

Content

Introduction

ServerQuery is a command-line interface built into the *TeamSpeak 3 Server* which allows powerful scripting and automation tools to be built based on the exact same instruction set and functionality provided by the *TeamSpeak 3 Client*. For example, you can use scripts to automate the management of virtual servers or nightly backups. In short, you can perform operations more efficiently by using *ServerQuery* scripts than you can by using a user interface.

This manual describes the general *ServerQuery* usage and syntax and provides examples for all commands available.

How to Establish a ServerQuery Connection

Connecting to a *ServerQuery* interface can be done by using a character-mode terminal client such as *Telnet* or *PuTTY*. Basically, a *ServerQuery* client is acting like a real client, except it's unable to send or receive voice data.

Per default, the *TeamSpeak 3 Server* is waiting for incoming *ServerQuery* connections on port *10011* (TCP). On success, the server should welcome you with a `TS3` prompt.

Command Syntax

This section describes the syntax of all *ServerQuery* commands.

Summary of Command Syntax

ServerQuery commands follow the general syntax of:

```
command [parameter...] [option...]
```

The command is a single word which may contain lowercase letters, digits and underscore symbols (a-z 0-9 _) followed by a whitespace. A parameter block is made up of one or more key-value pairs separated by a whitespace. In addition, various commands support options which are specified with a leading minus. For example:

```
command key1=value1 key2=value2 -option1 -option2
```

Some commands accept grouped or nested parameters. Therefore allowing you to apply an action on more than one object. Those parameters are separated by a pipe symbol (|). For example:

```
command key1=value1|key1=value2|key1=value3
```

The pipe symbol (|) is also used to separate list items (e.g. multiple clients in a virtual servers `clientlist`).

Examples of Command Syntax

The following provides some common *ServerQuery* syntax examples:

```
serverlist
clientlist -uid -away -groups
clientdbfind pattern=ScP
clientdbfind pattern=FPMPSC6MXqXq751dX7BKVOJniSo= -uid
clientkick reasonid=5 reasonmsg=Go\saway! clid=1|clid=2|clid=3
channelmove cid=16 cpid=1 order=0
sendtextmessage targetmode=2 target=12 msg=Hello\sWorld!
```

ESCAPING

You cannot use whitespaces or any special characters in parameters. Instead, the *TeamSpeak 3 Server* supports the use of escape patterns which can be used to insert newlines, tabs or other special characters into a parameter string. The same escape patterns are used to clean up the servers output and prevent parsing issues.

Here's an example on how to escape a parameter string correctly.

RIGHT:

```
serveredit virtualserver_name=TeamSpeak\s]\p[\sServer
```

WRONG:

```
serveredit virtualserver_name=TeamSpeak ]|[ Server
```

The following characters need to be escaped if they are to be used:

NAME	*CHAR*	*ASCII*	*REPLACE CHAR*	*REPLACE ASCII*
Backslash	\	92	\\	92 92
Slash	/	47	\/	92 47
Whitespace	" "	32	\s	92 115
Pipe	|	124	\p	92 112
Bell	\a	7	\a	92 97
Backspace	\b	8	\b	92 98
Formfeed	\f	12	\f	92 102
Newline	\n	10	\n	92 110
Carriage Return	\r	13	\r	92 114
Horizontal Tab	\t	9	\t	92 116
Vertical Tab	\v	11	\v	92 118

Whitelisting and Blacklisting

The *TeamSpeak 3 Server* includes flood protection technology for the *ServerQuery* interface which means that a *ServerQuery* client can only execute a finite number of commands per time unit. These limits are controlled by the following server instance properties:

- SERVERINSTANCE_SERVERQUERY_FLOOD_COMMANDS
- SERVERINSTANCE_SERVERQUERY_FLOOD_TIME

Per default, the *TeamSpeak 3 Server* will not allow more than 10 commands within 3 seconds from the same source. You can use the following command to modify these settings:

```
instanceedit serverinstance_serverquery_flood_commands=10 serverinstance_serverquery_flood_time=3
```

If you're using automated scripts or web applications to manage your servers, it's most likely possible that you'll exceed those limits. Therefore, the *TeamSpeak 3 Server* provides whitelisting and blacklisting features for the *ServerQuery* interface.

How to Use the Whitelist

The whitelist is a list of approved hosts that are allowed to ignore the flood protection settings on a *TeamSpeak 3 Server*. For example, if you're using a web administration interface, we strongly recommend that you add the IP address of your web server to the whitelist file. In a new installation of the *TeamSpeak 3 Server*, this file is called *query_ip_whitelist.txt* and it contains the loopback IP address of your server (127.0.0.1). You can enter an infinite number of IP addresses to the whitelist, one IP address per line.

```
85.25.120.233
80.190.225.233
75.125.142.2
194.97.114.2
79.218.0.0/16
127.0.0.1
```

The TeamSpeak 3 Server also supports Classless Inter-Domain Routing (CIDR) notation so you can easily add an entire network to your whitelists. CIDR notation uses a syntax of specifying IP addresses using the base address of the network followed by a slash and the size of the routing prefix, e.g., 192.168.0.0/16 (IPv4) and 2001:db8::/32 (IPv6).

How to Use the Blacklist

The blacklist is a list of hosts that, for one reason or another, are being denied access to the *ServerQuery* interface. In a new installation of the *TeamSpeak 3 Server*, this file is called *query_ip_blacklist.txt* and is empty. The syntax of this file is equal to the whitelist, e.g. 0.0.0.0/0 will refuse all incoming connections.

Command Reference

This is a list of the commands available when using the *TeamSpeak 3 ServerQuery* interface.

help

Provides information about *ServerQuery* commands. Used without parameters, `help` lists and briefly describes every command.

Permissions:

b_serverinstance_help_view

Syntax:

```
help [{command}]
```

Example:

```
help serverinfo
  Usage: serverinfo

  Displays detailed configuration information about a virtual server including ID, number of clients online,
  configuration, etc.

  error id=0 msg=ok
```

quit

Closes the *ServerQuery* connection to the *TeamSpeak 3 Server* instance.

Syntax:

```
quit
```

Example:

```
quit
```

login

Authenticates with the *TeamSpeak 3 Server* instance using given *ServerQuery* login credentials.

Related Permissions:

b_serverquery_login

Syntax:

```
login client_login_name={username} client_login_password={password}
login {username} {password}
```

Example:

```
login client_login_name=xyz client_login_password=xyz
  error id=0 msg=ok
```

logout

Deselects the active virtual server and logs out from the server instance.

Permissions:

b_serverquery_login

Syntax:

```
logout
```

Example:

```
logout
  error id=0 msg=ok
```

VERSION

Displays the servers version information including platform and build number.

Permissions:
b_serverinstance_version_view

Syntax:
```
version
```

Example:
```
version
  version=3.0.0-alpha4 build=9155 platform=Linux
  error id=0 msg=ok
```

HOSTINFO

Displays detailed connection information about the server instance including uptime, number of virtual servers online, traffic information, etc.

For detailed information, see Server Instance Properties.

Permissions:
b_serverinstance_info_view

Syntax:
```
hostinfo
```

Example:
```
hostinfo
  instance_uptime=1903203 host_timestamp_utc=1259337246 virtualservers_running_total=1
  connection_filetransfer_bandwidth_sent=0 …
  error id=0 msg=ok
```

INSTANCEINFO

Displays the server instance configuration including database revision number, the file transfer port, default group IDs, etc.

For detailed information, see Server Instance Properties.

Permissions:
b_serverinstance_info_view

Syntax:
```
instanceinfo
```

Example:
```
instanceinfo
  serverinstance_database_version=11 serverinstance_filetransfer_port=30033
  serverinstance_template_guest_serverquery_group=1 serverinstance_template_serveradmin_group=3 …
  error id=0 msg=ok
```

INSTANCEEDIT

Changes the server instance configuration using given properties.

For detailed information, see Server Instance Properties.

Permissions:
b_serverinstance_modify_settings

Syntax:
```
instanceedit [instance_properties…]
```

Example:
```
instanceedit serverinstance_filetransfer_port=1337
  error id=0 msg=ok
```

BINDINGLIST

Displays a list of IP addresses used by the server instance on multi-homed machines.

Permissions:
b_serverinstance_binding_list

Syntax:
```
bindinglist
```

Example:
```
bindinglist
  ip=0.0.0.0
  error id=0 msg=ok
```

USE

Selects the virtual server specified with `sid` or `port` to allow further interaction. The *ServerQuery* client will appear on the virtual server and acts like a real *TeamSpeak 3 Client*, except it's unable to send or receive voice data.

If your database contains multiple virtual servers using the same UDP port, `use` will select a random virtual server using the specified `port`.

Permissions:
b_virtualserver_select

Syntax:
```
use [sid={serverID}] [port={serverPort}] [-virtual]
use {serverID} [-virtual]
```

Example:
```
use sid=1
  error id=0 msg=ok
```

SERVERLIST

Displays a list of virtual servers including their ID, status, number of clients online, etc. If you're using the `-all` option, the server will list all virtual servers stored in the database. This can be useful when multiple server instances with different machine IDs are using the same database. The machine ID is used to identify the server instance a virtual server is associated with.

The status of a virtual server can be either `online`, `offline`, `deploy running`, `booting up`, `shutting down` and `virtual online`. While most of them are self-explanatory, `virtual online` is a bit more complicated.

Please note that whenever you select a virtual server which is currently stopped, it will be started in virtual mode which means you are able to change its configuration, create channels or change permissions, but no regular *TeamSpeak 3 Client* can connect. As soon as the last *ServerQuery* client deselects the virtual server, its status will be changed back to `offline`.

Permissions:
b_serverinstance_virtualserver_list

Syntax:
```
serverlist [-uid] [-short] [-all] [-onlyoffline]
```

Example:
```
serverlist
  virtualserver_id=1 virtualserver_port=9987 virtualserver_status=online virtualserver_clientsonline=6 …
  error id=0 msg=ok
```

SERVERIDGETBYPORT

Displays the database ID of the virtual server running on the UDP port specified by `virtualserver_port`.

Permissions:
b_serverinstance_virtualserver_list

Syntax:
```
serveridgetbyport virtualserver_port={serverPort}
```

Example:
```
serveridgetbyport virtualserver_port=9987
  server_id=1
  error id=0 msg=ok
```

SERVERDELETE

Deletes the virtual server specified with `sid`. Please note that only virtual servers in stopped state can be deleted.

Permissions:
b_virtualserver_delete

Syntax:
```
serverdelete sid={serverID}
```

Example:
```
serverdelete sid=1
  error id=0 msg=ok
```

SERVERCREATE

Creates a new virtual server using the given properties and displays its ID, port and initial administrator privilege key. If `virtualserver_port` is not specified, the server will test for the first unused UDP port.

The first virtual server will be running on UDP port 9987 by default. Subsequently started virtual servers will be running on increasing UDP port numbers.

For detailed information, see Virtual Server Properties.

Permissions:
b_virtualserver_create

Syntax:
```
servercreate virtualserver_name={serverName} [virtualserver_properties…]
```

Example:
```
servercreate virtualserver_name=TeamSpeak\s]\p[\sServer virtualserver_port=9988
virtualserver_maxclients=32
  sid=2 virtualserver_port=9988 token=eKnFZQ9EK7G7MhtuQB6+N2B1PNZZ6OZL3ycDp2OW
  error id=0 msg=ok
```

SERVERSTART

Starts the virtual server specified with `sid`. Depending on your permissions, you're able to start either your own virtual server only or all virtual servers in the server instance.

Permissions:
b_virtualserver_start_any
b_virtualserver_start

Syntax:
```
serverstart sid={serverID}
```

Example:
```
serverstart sid=1
  error id=0 msg=ok
```

SERVERSTOP

Stops the virtual server specified with `sid`. Depending on your permissions, you're able to stop either your own virtual server only or all virtual servers in the server instance.

Permissions:
b_virtualserver_stop_any
b_virtualserver_stop

Syntax:
```
serverstop sid={serverID}
```

Example:
```
serverstop sid=1
  error id=0 msg=ok
```

SERVERPROCESSSTOP

Stops the entire *TeamSpeak 3 Server* instance by shutting down the process.

Permissions:
b_serverinstance_stop

Syntax:
```
serverprocessstop
```

Example:
```
serverprocessstop
  error id=0 msg=ok
```

SERVERINFO

Displays detailed configuration information about the selected virtual server including unique ID, number of clients online, configuration, etc.

For detailed information, see Virtual Server Properties.

Permissions:
b_virtualserver_info_view

Syntax:
```
serverinfo
```

Example:
```
serverinfo
  virtualserver_port=9987 virtualserver_unique_identifier=zrPkjznB1tMnRwj01xx7RxXjqeY=
  virtualserver_name=TeamSpeak\s]I[\sServer …
  error id=0 msg=ok
```

SERVERREQUESTCONNECTIONINFO

Displays detailed connection information about the selected virtual server including uptime, traffic information, etc.

Permissions:
b_virtualserver_connectioninfo_view

Syntax:
```
serverrequestconnectioninfo
```

Example:
```
serverrequestconnectioninfo
  connection_filetransfer_bandwidth_sent=0 connection_filetransfer_bandwidth_received=0
  connection_packets_sent_total=241454 …
  error id=0 msg=ok
```

SERVEREDIT

Changes the selected virtual servers configuration using given properties. Note that this command accepts multiple properties which means that you're able to change all settings of the selected virtual server at once.

For detailed information, see Virtual Server Properties.

Permissions:

b_virtualserver_modify_name
b_virtualserver_modify_welcomemessage
b_virtualserver_modify_maxclients
b_virtualserver_modify_reserved_slots
b_virtualserver_modify_password
b_virtualserver_modify_default_servergroup
b_virtualserver_modify_default_channelgroup
b_virtualserver_modify_default_channeladmingroup
b_virtualserver_modify_ft_settings
b_virtualserver_modify_ft_quotas
b_virtualserver_modify_channel_forced_silence
b_virtualserver_modify_complain
b_virtualserver_modify_antiflood
b_virtualserver_modify_hostmessage
b_virtualserver_modify_hostbanner
b_virtualserver_modify_hostbutton
b_virtualserver_modify_port
b_virtualserver_modify_autostart
b_virtualserver_modify_needed_identity_security_level
b_virtualserver_modify_priority_speaker_dimm_modificator
b_virtualserver_modify_log_settings
b_virtualserver_modify_icon_id
b_virtualserver_modify_weblist
b_virtualserver_modify_min_client_version
b_virtualserver_modify_codec_encryption_mode

Syntax:

```
serveredit [virtualserver_properties…]
```

Example:

```
serveredit virtualserver_name=TeamSpeak\s]\p[\sServer virtualserver_maxclients=32
  error id=0 msg=ok
```

SERVERGROUPLIST

Displays a list of server groups available. Depending on your permissions, the output may also contain global ServerQuery groups and template groups.

Permissions:

b_serverinstance_modify_querygroup
b_serverinstance_modify_templates
b_virtualserver_servergroup_list

Syntax:

```
servergrouplist
```

Example:

```
servergrouplist
  sgid=9 name=Server\sAdmin type=1 iconid=300 savedb=1|sgid=10 name=Normal type=1 iconid=0 savedb=1|sgid=11 …
  error id=0 msg=ok
```

SERVERGROUPADD

Creates a new server group using the name specified with `name` and displays its ID. The optional `type` parameter can be used to create ServerQuery groups and template groups. For detailed information, see Definitions.

Permissions:
b_virtualserver_servergroup_create

Syntax:
```
servergroupadd name={groupName} [type={groupDbType}]
```

Example:
```
servergroupadd name=Server\sAdmin
  sgid=13
  error id=0 msg=ok
```

SERVERGROUPDEL

Deletes the server group specified with `sgid`. If `force` is set to *1*, the server group will be deleted even if there are clients within.

Permissions:
b_virtualserver_servergroup_delete

Syntax:
```
servergroupdel sgid={groupID} force={1|0}
```

Example:
```
servergroupdel sgid=13
  error id=0 msg=ok
```

SERVERGROUPCOPY

Creates a copy of the server group specified with `ssgid`. If `tsgid` is set to *0*, the server will create a new group. To overwrite an existing group, simply set `tsgid` to the ID of a designated target group. If a target group is set, the `name` parameter will be ignored.

The `type` parameter can be used to create ServerQuery groups and template groups. For detailed information, see Definitions.

Permissions:
b_virtualserver_servergroup_create
i_group_modify_power
i_group_needed_modify_power

Syntax:
```
servergroupcopy ssgid={sourceGroupID} tsgid={targetGroupID} name={groupName} type={groupDbType}
```

Example:
```
servergroupcopy ssgid=6 tsgid=0 name=My\sGroup\s(Copy) type=1
  sgid=21
  error id=0 msg=ok
```

SERVERGROUPRENAME

Changes the name of the server group specified with `sgid`.

Permissions:
i_group_modify_power
i_group_needed_modify_power

Syntax:
```
servergrouprename sgid={groupID} name={groupName}
```

Example:
```
servergrouprename sgid=13 name=New\sName
  error id=0 msg=ok
```

SERVERGROUPPERMLIST

Displays a list of permissions assigned to the server group specified with `sgid`. If the `-permsid` option is specified, the output will contain the permission names instead of the internal IDs.

Permissions:

b_virtualserver_servergroup_permission_list

Syntax:

```
servergrouppermlist sgid={groupID} [-permsid]
```

Example:

```
servergrouppermlist sgid=13
  permid=8470 permvalue=1 permnegated=0 permskip=0|permid=8475 permvalue=1 …
  error id=0 msg=ok
```

SERVERGROUPADDPERM

Adds a set of specified permissions to the server group specified with `sgid`. Multiple permissions can be added by providing the four parameters of each permission. A permission can be specified by `permid` or `permsid`.

Permissions:

i_group_modify_power
i_group_needed_modify_power
i_permission_modify_power

Syntax:

```
servergroupaddperm sgid={groupID} [permid={permID}…] [permsid={permName}…]
permvalue={permValue}… permnegated={1|0}… permskip={1|0}…
```

Examples:

```
servergroupaddperm sgid=13 permid=17276 permvalue=50 permnegated=0 permskip=0|permid=21415
permvalue=20 permnegated=0
  error id=0 msg=ok

servergroupaddperm sgid=3 permsid=b_virtualserver_modify_maxclients permvalue=1 permnegated=0
permskip=0
  error id=0 msg=ok
```

SERVERGROUPDELPERM

Removes a set of specified permissions from the server group specified with `sgid`. Multiple permissions can be removed at once. A permission can be specified by `permid` or `permsid`.

Permissions:

i_group_modify_power
i_group_needed_modify_power
i_permission_modify_power

Syntax:

```
servergroupdelperm sgid={groupID} [permid={permID}…] [permsid={permName}…]
```

Examples:

```
servergroupdelperm sgid=16 permid=17276|permid=21415
  error id=0 msg=ok

servergroupdelperm sgid=3 permsid=b_virtualserver_modify_maxclients
  error id=0 msg=ok
```

SERVERGROUPADDCLIENT

Adds a client to the server group specified with `sgid`. Please note that a client cannot be added to default groups or template groups.

Permissions:
i_group_member_add_power
i_group_needed_member_add_power

Syntax:
```
servergroupaddclient sgid={groupID} cldbid={clientDBID}
```

Example:
```
servergroupaddclient sgid=16 cldbid=3
  error id=0 msg=ok
```

SERVERGROUPDELCLIENT

Removes a client specified with `cldbid` from the server group specified with `sgid`.

Permissions:
i_group_member_remove_power
i_group_needed_member_remove_power

Syntax:
```
servergroupdelclient sgid={groupID} cldbid={clientDBID}
```

Example:
```
servergroupdelclient sgid=16 cldbid=3
  error id=0 msg=ok
```

SERVERGROUPCLIENTLIST

Displays the IDs of all clients currently residing in the server group specified with `sgid`. If you're using the optional `-names` option, the output will also contain the last known nickname and the unique identifier of the clients.

Permissions:
b_virtualserver_servergroup_client_list

Syntax:
```
servergroupclientlist sgid={groupID} [-names]
```

Example:
```
servergroupclientlist sgid=16
  cldbid=7|cldbid=8|cldbid=9|cldbid=11|cldbid=13|cldbid=16|cldbid=18|cldbid=29|cldbid=32|cldbid=34|cldbid=37|cldbid=4
  0|cldbid=47|cldbid=53
  error id=0 msg=ok
```

SERVERGROUPSBYCLIENTID

Displays all server groups the client specified with `cldbid` is currently residing in.

Syntax:
```
servergroupsbyclientid cldbid={clientDBID}
```

Example:
```
servergroupsbyclientid cldbid=18
  name=Server\sAdmin sgid=6 cldbid=18
  error id=0 msg=ok
```

SERVERSNAPSHOTCREATE

Displays a snapshot of the selected virtual server containing all settings, groups and known client identities. The data from a server snapshot can be used to restore a virtual servers configuration, channels and permissions using the `serversnapshotdeploy` command.

Permissions:
b_virtualserver_snapshot_create

Syntax:
```
serversnapshotcreate
```

Example:
```
serversnapshotcreate
  hash=bnTd2E1kNITHjJYRCFjgbKKO5P8=|virtualserver_unique_identifier=zrPkjznB1tMnRwjO1xx7RxXjqeY=
  virtualserver_name=TeamSpeak\s]I[\sServer …
  error id=0 msg=ok
```

SERVERSNAPSHOTDEPLOY

Restores the selected virtual servers configuration using the data from a previously created server snapshot. Please note that the *TeamSpeak 3 Server* does **NOT** check for necessary permissions while deploying a snapshot so the command could be abused to gain additional privileges.

Permissions:
b_virtualserver_snapshot_deploy

Syntax:
```
serversnapshotdeploy virtualserver_snapshot
```

Example:
```
serversnapshotdeploy
  hash=bnTd2E1kNITHjJYRCFjgbKKO5P8=|virtualserver_unique_identifier=zrPkjznB1tMnRwjO1xx7RxXjq= … error id=0 msg=ok
```

SERVERNOTIFYREGISTER

Registers for a specified category of events on a virtual server to receive notification messages. Depending on the notifications you've registered for, the server will send you a message on every event in the view of your *ServerQuery* client (e.g. clients joining your channel, incoming text messages, server configuration changes, etc). The event source is declared by the `event` parameter while `id` can be used to limit the notifications to a specific channel.

Permissions:
b_virtualserver_notify_register

Syntax:
```
servernotifyregister event={server|channel|textserver|textchannel|textprivate} [id={channelID}]
```

Example:
```
servernotifyregister event=server
  error id=0 msg=ok
```

SERVERNOTIFYUNREGISTER

Unregisters all events previously registered with `servernotifyregister` so you will no longer receive notification messages.

Permissions:
b_virtualserver_notify_unregister

Syntax:
```
servernotifyunregister
```

Example:
```
servernotifyunregister
  error id=0 msg=ok
```

SENDTEXTMESSAGE

Sends a text message a specified target. The type of the target is determined by `targetmode` while `target` specifies the ID of the recipient, whether it be a virtual server, a channel or a client.

For detailed information, see Definitions.

Permissions:

i_client_private_textmessage_power
i_client_needed_private_textmessage_power
b_client_server_textmessage_send
b_client_channel_textmessage_send

Syntax:

```
sendtextmessage targetmode={1-3} target={serverID|channelID|clientID} msg={text}
```

Example:

```
sendtextmessage targetmode=2 target=1 msg=Hello\sWorld!
  error id=0 msg=ok
```

LOGVIEW

Displays a specified number of entries from the servers log. If `instance` is set to 1, the server will return lines from the master logfile (`ts3server_0.log`) instead of the selected virtual server logfile.

Permissions:

b_serverinstance_log_view
b_virtualserver_log_view

Syntax:

```
logview [lines={1-100}] [reverse={1|0}] [instance={1|0}] [begin_pos={n}]
```

Example:

```
logview lines=30
  last_pos=403788 file_size=411980 l=\p\slistening\son\s0.0.0.0:9987 …
  error id=0 msg=ok
```

LOGADD

Writes a custom entry into the servers log. Depending on your permissions, you'll be able to add entries into the server instance log and/or your virtual servers log. The `loglevel` parameter specifies the type of the entry.

For detailed information, see Definitions.

Permissions:

b_serverinstance_log_add
b_virtualserver_log_add

Syntax:

```
logadd loglevel={1-4} logmsg={text}
```

Example:

```
logadd loglevel=4 logmsg=Informational\smessage!
  error id=0 msg=ok
```

GM

Sends a text message to all clients on all virtual servers in the *TeamSpeak 3 Server* instance.

Permissions:

b_serverinstance_textmessage_send

Syntax:

```
gm msg={text}
```

Example:

```
gm msg=Hello\sWorld!
  error id=0 msg=ok
```

CHANNELLIST

Displays a list of channels created on a virtual server including their ID, order, name, etc. The output can be modified using several command options.

Permissions:

b_virtualserver_channel_list

Syntax:

```
channellist [-topic] [-flags] [-voice] [-limits] [-icon]
```

Example:

```
channellist -topic
  cid=15 pid=0 channel_order=0 channel_name=Default\sChannel channel_topic=Default\sChannel\shas\sno\s[b]topic[\/b]
  total_clients=2|cid=16 …
  error id=0 msg=ok
```

CHANNELINFO

Displays detailed configuration information about a channel including ID, topic, description, etc.

For detailed information, see Channel Properties.

Permissions:

b_channel_info_view

Syntax:

```
channelinfo cid={channelID}
```

Example:

```
channelinfo cid=1
  channel_name=Default\sChannel channel_topic=Default\sChannel\shas\sno\s[b]topic[\/b]
  channel_description=This\sis\sthe\sdefault\schannel …
  error id=0 msg=ok
```

CHANNELFIND

Displays a list of channels matching a given name pattern.

Permissions:

b_virtualserver_channel_search

Syntax:

```
channelfind [pattern={channelName}]
```

Example:

```
channelfind pattern=default
  cid=15 channel_name=Default\sChannel
  error id=0 msg=ok
```

CHANNELMOVE

Moves a channel to a new parent channel with the ID `cpid`. If `order` is specified, the channel will be sorted right under the channel with the specified ID. If `order` is set to *0*, the channel will be sorted right below the new parent.

Permissions:

i_channel_min_depth
i_channel_max_depth
b_channel_modify_parent
b_channel_modify_sortorder

Syntax:

channelmove cid={channelID} cpid={channelParentID} [order={channelSortOrder}]

Example:

```
channelmove cid=16 cpid=1 order=0
  error id=0 msg=ok
```

CHANNELCREATE

Creates a new channel using the given properties and displays its ID. Note that this command accepts multiple properties which means that you're able to specifiy all settings of the new channel at once.

For detailed information, see Channel Properties.

Permissions:

i_channel_min_depth
i_channel_max_depth
b_channel_create_child
b_channel_create_permanent
b_channel_create_semi_permanent
b_channel_create_temporary
b_channel_create_with_topic
b_channel_create_with_description
b_channel_create_with_password
b_channel_create_modify_with_codec_speex8
b_channel_create_modify_with_codec_speex16
b_channel_create_modify_with_codec_speex32
b_channel_create_modify_with_codec_celtmono48
i_channel_create_modify_with_codec_maxquality
i_channel_create_modify_with_codec_latency_factor_min
b_channel_create_with_maxclients
b_channel_create_with_maxfamilyclients
b_channel_create_with_sortorder
b_channel_create_with_default
b_channel_create_with_needed_talk_power

Syntax:

```
channelcreate channel_name={channelName} [channel_properties…]
```

Example:

```
channelcreate channel_name=My\sChannel channel_topic=My\sTopic
  cid=16
  error id=0 msg=ok
```

CHANNELDELETE

Deletes an existing channel by ID. If `force` is set to *1*, the channel will be deleted even if there are clients within. The clients will be kicked to the default channel with an appropriate reason message.

Permissions:

b_channel_delete_permanent
b_channel_delete_semi_permanent
b_channel_delete_temporary
b_channel_delete_flag_force

Syntax:

```
channeldelete cid={channelID} force={1|0}
```

Example:

```
channeldelete cid=16 force=1
  error id=0 msg=ok
```

CHANNELEDIT

Changes a channels configuration using given properties. Note that this command accepts multiple properties which means that you're able to change all settings of the channel specified with `cid` at once.

For detailed information, see Channel Properties.

Permissions:

i_channel_min_depth
i_channel_max_depth
b_channel_modify_parent
b_channel_modify_make_default
b_channel_modify_make_permanent
b_channel_modify_make_semi_permanent
b_channel_modify_make_temporary
b_channel_modify_name
b_channel_modify_topic
b_channel_modify_description
b_channel_modify_password
b_channel_modify_codec
b_channel_create_modify_with_codec_speex8
b_channel_create_modify_with_codec_speex16
b_channel_create_modify_with_codec_speex32
b_channel_create_modify_with_codec_celtmono48
b_channel_modify_codec_quality
b_channel_create_modify_with_codec_maxquality
b_channel_modify_codec_latency_factor
b_channel_modify_make_codec_encrypted
b_channel_modify_maxclients
b_channel_modify_maxfamilyclients
b_channel_modify_sortorder
b_channel_modify_needed_talk_power
i_channel_modify_power
i_channel_needed_modify_power

Syntax:

```
channeledit cid={channelID} [channel_properties…]
```

Example:

```
channeledit cid=15 channel_codec_quality=3 channel_description=My\sDescription
  error id=0 msg=ok
```

CHANNELGROUPLIST

Displays a list of channel groups available on the selected virtual server.

Permissions:

b_virtualserver_channelgroup_list
b_serverinstance_modify_templates

Syntax:

```
channelgrouplist
```

Example:

```
channelgrouplist
  cgid=1 name=Channel\sAdmin type=2 iconid=100 savedb=1|cgid=2 …
  error id=0 msg=ok
```

CHANNELGROUPADD

Creates a new channel group using a given name and displays its ID. The optional `type` parameter can be used to create ServerQuery groups and template groups.

For detailed information, see Definitions.

Permissions:
b_virtualserver_channelgroup_create

Syntax:
```
channelgroupadd name={groupName} [type={groupDbType}]
```
Example:
```
channelgroupadd name=Channel\sAdmin
  cgid=13
  error id=0 msg=ok
```

CHANNELGROUPDEL

Deletes a channel group by ID. If `force` is set to *1*, the channel group will be deleted even if there are clients within.

Permissions:
b_virtualserver_channelgroup_delete

Syntax:
```
channelgroupdel cgid={groupID} force={1|0}
```

Example:
```
channelgroupdel cgid=13
  error id=0 msg=ok
```

CHANNELGROUPCOPY

Creates a copy of the channel group specified with `scgid`. If `tcgid` is set to *0*, the server will create a new group. To overwrite an existing group, simply set `tcgid` to the ID of a designated target group. If a target group is set, the `name` parameter will be ignored. The `type` parameter can be used to create ServerQuery groups and template groups.

For detailed information, see Definitions.

Permissions:
b_virtualserver_channelgroup_create
i_group_modify_power
i_group_needed_modify_power

Syntax:
```
channelgroupcopy scgid={sourceGroupID} tsgid={targetGroupID} name={groupName} type={groupDbType}
```

Example:
```
channelgroupcopy scgid=4 tcgid=0 name=My\sGroup\s(Copy) type=1
  cgid=33
  error id=0 msg=ok
```

CHANNELGROUPRENAME

Changes the name of a specified channel group.

Permissions:
i_group_modify_power
i_group_needed_modify_power

Syntax:
```
channelgrouprename cgid={groupID} name={groupName}
```

Example:
```
channelgrouprename cgid=13 name=New\sName
  error id=0 msg=ok
```

CHANNELGROUPADDPERM

Adds a set of specified permissions to a channel group. Multiple permissions can be added by providing the two parameters of each permission. A permission can be specified by `permid` or `permsid`.

Permissions:

i_group_modify_power
i_group_needed_modify_power
i_permission_modify_power

Syntax:

```
channelgroupaddperm cgid={groupID} [permid={permID}…] [permsid={permName}…] permvalue={permValue}…
```

Example:

```
channelgroupaddperm cgid=13 permid=17276 permvalue=50|permid=21415 permvalue=20
  error id=0 msg=ok
```

CHANNELGROUPPERMLIST

Displays a list of permissions assigned to the channel group specified with `cgid`. If the `–permsid` option is specified, the output will contain the permission names instead of the internal IDs.

Permissions:

b_virtualserver_channelgroup_permission_list

Syntax:

```
channelgrouppermlist cgid={groupID} [-permsid]
```

Example:

```
channelgrouppermlist cgid=13
  permid=8470 permvalue=1 permnegated=0 permskip=0|permid=8475 permvalue=1 …
  error id=0 msg=ok
```

CHANNELGROUPDELPERM

Removes a set of specified permissions from the channel group. Multiple permissions can be removed at once. A permission can be specified by `permid` or `permsid`.

Permissions:

i_group_modify_power
i_group_needed_modify_power
i_permission_modify_power

Syntax:

```
channelgroupdelperm cgid={groupID} [permid={permID}…] [permsid={permName}…]
```

Example:

```
channelgroupdelperm cgid=16 permid=17276|permid=21415
  error id=0 msg=ok
```

CHANNELGROUPCLIENTLIST

Displays all the client and/or channel IDs currently assigned to channel groups. All three parameters are optional so you're free to choose the most suitable combination for your requirements.

Permissions:

b_virtualserver_channelgroup_client_list

Syntax:

```
channelgroupclientlist [cid={channelID}] [cldbid={clientDBID}] [cgid={groupID}]
```

Example:

```
channelgroupclientlist cid=2 cgid=9
  cid=2 cldbid=9 cgid=9|cid=2 cldbid=24 cgid=9|cid=2 cldbid=47 cgid=9
  error id=0 msg=ok
```

SETCLIENTCHANNELGROUP

Sets the channel group of a client to the ID specified with `cgid`.

Permissions:

i_group_member_add_power
i_group_needed_member_add_power
i_group_member_remove_power
i_group_needed_member_remove_power

Syntax:

```
setclientchannelgroup cgid={groupID} cid={channelID} cldbid={clientDBID}
```

Example:

```
setclientchannelgroup cgid=13 cid=15 cldbid=20
  error id=0 msg=ok
```

CHANNELPERMLIST

Displays a list of permissions defined for a channel.

Permissions:

b_virtualserver_channel_permission_list

Syntax:

```
channelpermlist cid={channelID} [-permsid]
```

Example:

```
channelpermlist cid=2
  cid=2 permid=4353 permvalue=1 permnegated=0 permskip=0|permid=17276 permvalue=50 …
  error id=0 msg=ok
```

CHANNELADDPERM

Adds a set of specified permissions to a channel. Multiple permissions can be added by providing the two parameters of each permission. A permission can be specified by `permid` or `permsid`.

Permissions:

i_group_modify_power
i_group_needed_modify_power
i_permission_modify_power

Syntax:

```
channeladdperm cid={channelID} [permid={permID}…] [permsid={permName}…] permvalue={permValue}…
```

Example:

```
channeladdperm cid=16 permid=17276 permvalue=50|permid=21415 permvalue=20
  error id=0 msg=ok
```

CHANNELDELPERM

Removes a set of specified permissions from a channel. Multiple permissions can be removed at once. A permission can be specified by `permid` or `permsid`.

Permissions:

i_group_modify_power
i_group_needed_modify_power
i_permission_modify_power

Syntax:

```
channeldelperm cid=123 [permid={permID}…] [permsid={permName}…]
```

Example:

```
channeldelperm cid=16 permid=17276|permid=21415
  error id=0 msg=ok
```

CLIENTLIST

Displays a list of clients online on a virtual server including their ID, nickname, status flags, etc. The output can be modified using several command options.

Please note that the output will only contain clients which are currently in channels you're able to subscribe to.

Permissions:
b_virtualserver_client_list
i_channel_subscribe_power
i_channel_needed_subscribe_power

Syntax:
```
clientlist [-uid] [-away] [-voice] [-times] [-groups] [-info] [-icon] [-country]
```

Example:
```
clientlist -away
  clid=5 cid=7 client_database_id=40 client_nickname=ScP client_type=0 client_away=1
  client_away_message=not\shere|clid=6 …
  error id=0 msg=ok
```

CLIENTINFO

Displays detailed configuration information about a client including unique ID, nickname, client version, etc.

Permissions:
b_client_info_view

Syntax:
```
clientinfo clid={clientID}
```

Example:
```
clientinfo clid=6
  client_unique_identifier=P5H2hrN6+gpQI4n\/dXp3p17vtYO= client_nickname=Rabe85 client_version=3.0.0-
  alpha24\s[Build:\s8785]\s(UI:\s8785) …
  error id=0 msg=ok
```

CLIENTFIND

Displays a list of clients matching a given name pattern.

Permissions:
b_virtualserver_client_search

Syntax:
```
clientfind pattern={clientName}
```
Example:
```
clientfind pattern=sven
  clid=7 client_nickname=Sven
  error id=0 msg=ok
```

CLIENTEDIT

Changes a clients settings using given properties.

For detailed information, see Client Properties.

Permissions:
b_client_modify_description
b_client_set_talk_power

Syntax:
```
clientedit clid={clientID} [client_properties…]
```

Example:
```
clientedit clid=10 client_description=Best\sguy\sever!
  error id=0 msg=ok
```

CLIENTDBLIST

Displays a list of client identities known by the server including their database ID, last nickname, etc.

Permissions:

b_virtualserver_client_dblist

Syntax:

```
clientdblist [start={offset}] [duration={limit}] [-count]
```

Example:

```
clientdblist
  cldbid=7 client_unique_identifier=DZhdQU58qyooEK4Fr8Ly738hEmc= client_nickname=MuhChy client_created=1259147468
  client_lastconnected=1259421233 error id=0 msg=ok
```

CLIENTDBINFO

Displays detailed database information about a client including unique ID, creation date, etc.

Permissions:

b_virtualserver_client_dbinfo

Syntax:

```
clientdbinfo cldbid={clientDBID}
```

Example:

```
clientdbfind cldbid=4
  client_unique_identifier=FPMPSC6MXqXq751dX7BKVOJniSo= client_nickname=ScP client_created=1265411019
  error id=0 msg=ok
```

CLIENTDBFIND

Displays a list of client database IDs matching a given pattern. You can either search for a clients last known nickname or his unique identity by using the `-uid` option.

Permissions:

b_virtualserver_client_dbsearch

Syntax:

```
clientdbfind pattern={clientName|clientUID} [-uid]
```

Example:

```
clientdbfind pattern=sven
  cldbid=56
  error id=0 msg=ok
```

CLIENTDBEDIT

Changes a clients settings using given properties.

For detailed information, see Client Properties.

Permissions:

b_client_modify_dbproperties
b_client_modify_description
b_client_set_talk_power

Syntax:

```
clientdbedit cldbid={clientDBID} [client_properties…]
```

Example:

```
clientdbedit cldbid=56 client_description=Best\sguy\sever!
  error id=0 msg=ok
```

CLIENTDBDELETE

Deletes a clients properties from the database.

Permissions:
b_client_delete_dbproperties

Syntax:
```
clientdbdelete cldbid={clientDBID}
```

Example:
```
clientdbdelete cldbid=56
  error id=0 msg=ok
```

CLIENTGETIDS

Displays all client IDs matching the unique identifier specified by `cluid`.

Syntax:
```
clientgetids cluid={clientUID}
```

Example:
```
clientgetids cluid=dyjxkshZP6bzOn3bnwFQ1CkwZOM=
  cluid=dyjxkshZP6bzOn3bnwFQ1CkwZOM= clid=1 name=Janko
  error id=0 msg=ok
```

CLIENTGETDBIDFROMUID

Displays the database ID matching the unique identifier specified by `cluid`.

Syntax:
```
clientgetdbidfromuid cluid={clientUID}
```

Example:
```
clientgetdbidfromuid cluid=dyjxkshZP6bzOn3bnwFQ1CkwZOM=
  cluid=dyjxkshZP6bzOn3bnwFQ1CkwZOM= cldbid=32
  error id=0 msg=ok
```

CLIENTGETNAMEFROMUID

Displays the database ID and nickname matching the unique identifier specified by `cluid`.

Syntax:
```
clientgetnamefromuid cluid={clientUID}
```

Example:
```
clientgetnamefromuid cluid=dyjxkshZP6bzOn3bnwFQ1CkwZOM=
  cluid=dyjxkshZP6bzOn3bnwFQ1CkwZOM= cldbid=32 name=Janko
  error id=0 msg=ok
```

CLIENTGETNAMEFROMDBID

Displays the unique identifier and nickname matching the database ID specified by `cldbid`.

Syntax:
```
clientgetnamefromdbid cldbid={clientDBID}
```

Example:
```
clientgetnamefromdbid cldbid=32
  cluid=dyjxkshZP6bzOn3bnwFQ1CkwZOM= cldbid=32 name=Janko
  error id=0 msg=ok
```

CLIENTSETSERVERQUERYLOGIN

Updates your own *ServerQuery* login credentials using a specified username. The password will be auto-generated.

Permissions:
b_client_create_modify_serverquery_login

Syntax:
```
clientsetserverquerylogin client_login_name={username}
```

Example:
```
clientsetserverquerylogin client_login_name=admin
  client_login_password=+r\/TQqvR
  error id=0 msg=ok
```

CLIENTUPDATE

Change your ServerQuery clients settings using given properties.

For detailed information, see Client Properties.

Syntax:
```
clientupdate [client_properties…]
```

Example:
```
clientupdate client_nickname=ScP\s(query)
  error id=0 msg=ok
```

CLIENTMOVE

Moves one or more clients specified with `clid` to the channel with ID `cid`. If the target channel has a password, it needs to be specified with `cpw`. If the channel has no password, the parameter can be omitted.

Permissions:
i_client_move_power
i_client_needed_move_power

Syntax:
```
clientmove clid={clientID}… cid={channelID} [cpw={channelPassword}]
```

Example:
```
clientmove clid=5|clid=6 cid=3
  error id=0 msg=ok
```

CLIENTKICK

Kicks one or more clients specified with `clid` from their currently joined channel or from the server, depending on `reasonid`. The `reasonmsg` parameter specifies a text message sent to the kicked clients. This parameter is optional and may only have a maximum of 40 characters.

For detailed information, see Definitions.

Permissions:
i_client_kick_from_server_power
i_client_kick_from_channel_power
i_client_needed_kick_from_server_power
i_client_needed_kick_from_channel_power

Syntax:
```
clientkick clid={clientID}… reasonid={4|5} [reasonmsg={text}]
```
Example:
```
clientkick clid=5|clid=6 reasonid=4 reasonmsg=Go\saway!
  error id=0 msg=ok
```

CLIENTPOKE

Sends a poke message to the client specified with `clid`.

Permissions:

i_client_poke_power
i_client_needed_poke_power

Syntax:

```
clientpoke clid={clientID}… msg={text}
```

Example:

```
clientpoke clid=5 msg=Wake\sup!
  error id=0 msg=ok
```

CLIENTPERMLIST

Displays a list of permissions defined for a client.

Permissions:

b_virtualserver_client_permission_list

Syntax:

```
clientpermlist cldbid={clientDBID} [-permsid]
```

Example:

```
clientpermlist cldbid=2
  cldbid=2 permid=4353 permvalue=1 permnegated=0 permskip=0|permid=17276 permvalue=50 permnegated=0
  permskip=0|permid=21415 …
  error id=0 msg=ok
```

CLIENTADDPERM

Adds a set of specified permissions to a client. Multiple permissions can be added by providing the three parameters of each permission. A permission can be specified by `permid` or `permsid`.

Permissions:

i_group_modify_power
i_group_needed_modify_power
i_permission_modify_power

Syntax:

```
clientaddperm cldbid={clientDBID} [permid={permID}…] [permsid={permName}…]
permvalue={permValue}… permskip={1|0}…
```

Example:

```
clientaddperm cldbid=16 permid=17276 permvalue=50 permskip=1|permid=21415 permvalue=20 permskip=0
  error id=0 msg=ok
```

CLIENTDELPERM

Removes a set of specified permissions from a client. Multiple permissions can be removed at once. A permission can be specified by `permid` or `permsid`.

Permissions:

i_group_modify_power
i_group_needed_modify_power
i_permission_modify_power

Syntax:

```
channeldelperm cldbid={clientDBID} [permid={permID}…] [permsid={permName}…]
```

Example:

```
clientdelperm cldbid=16 permid=17276|permid=21415
  error id=0 msg=ok
```

CHANNELCLIENTPERMLIST

Displays a list of permissions defined for a client in a specific channel.

Permissions:

b_virtualserver_channelclient_permission_list

Syntax:

```
channelclientpermlist cid={channelID} cldbid={clientDBID} [-permsid]
```

Example:

```
channelclientpermlist cid=12 cldbid=3
  cid=12 cldbid=3 permid=4353 permvalue=1 permnegated=0 permskip=0|permid=17276 permvalue=50 permnegated=0
  permskip=0|permid=21415 …
  error id=0 msg=ok
```

CHANNELCLIENTADDPERM

Adds a set of specified permissions to a client in a specific channel. Multiple permissions can be added by providing the three parameters of each permission. A permission can be specified by `permid` or `permsid`.

Permissions:

i_group_modify_power
i_group_needed_modify_power
i_permission_modify_power

Syntax:

```
channelclientaddperm cid={channelID} cldbid={clientDBID} [permid={permID}…] [permsid={permName}…]
permvalue={permValue}…
```

Example:

```
channelclientaddperm cid=12 cldbid=3 permid=17276 permvalue=50|permid=21415 permvalue=20
  error id=0 msg=ok
```

CHANNELCLIENTDELPERM

Removes a set of specified permissions from a client in a specific channel. Multiple permissions can be removed at once. A permission can be specified by `permid` or `permsid`.

Permissions:

i_group_modify_power
i_group_needed_modify_power
i_permission_modify_power

Syntax:

```
channelclientdelperm cid={channelID} cldbid={clientDBID} [permid={permID}…] [permsid={permName}…]
```

Example:

```
channelclientdelperm cid=12 cldbid=3 permid=17276|permid=21415
  error id=0 msg=ok
```

PERMISSIONLIST

Displays a list of permissions available on the server instance including ID, name and description.

Permissions:

b_serverinstance_permission_list

Syntax:

```
permissionlist
```

Example:

```
permissionlist
  permid=21413 permname=b_client_channel_textmessage_send permdesc=Send\stext\smessages\sto\schannel|permid=21414
  permname=i_client_talk_power …
  error id=0 msg=ok
```

PERMIDGETBYNAME

Displays the database ID of one or more permissions specified by `permsid`.

Permissions:

b_serverinstance_permission_list

Syntax:

```
permidgetbyname permsid={permName}…
```

Example:

```
permidgetbyname permsid=b_serverinstance_help_view|permsid=b_serverinstance_info_view
  permsid=b_serverinstance_help_view permid=4353|permsid=b_serverinstance_info_view permid=4355
  error id=0 msg=ok
```

PERMOVERVIEW

Displays all permissions assigned to a client for the channel specified with `cid`. If `permid` is set to *0*, all permissions will be displayed. A permission can be specified by `permid` or `permsid`.

Permissions:

b_client_permissionoverview_view

Syntax:

```
permoverview cid={channelID} cldbid={clientDBID} [permid={permID}…] [permsid={permName}…]
```

Example:

```
permoverview cldbid=57 cid=74 permid=0
  t=0 id1=5 id2=0 p=37 v=1 n=0 s=0|t=0 id1=5 id2=0 p=38 v=1 n=0 s=0 …
  error id=0 msg=ok
```

PERMGET

Displays the current value of the permission specified with `permid` or `permsid` for your own connection. This can be useful when you need to check your own privileges.

Permissions:

b_client_permissionoverview_own

Syntax:

```
permget permid={permID}
permget permsid={permName}
```

Example:

```
permget permid=21174
  permsid=i_client_move_power permid=21174 permvalue=100
  error id=0 msg=ok

permget permsid=i_client_move_power
  permsid=i_client_move_power permid=21174 permvalue=100
  error id=0 msg=ok
```

PERMFIND

Displays detailed information about all assignments of the permission specified with `permid`. The output is similar to `permoverview` which includes the type and the ID of the client, channel or group associated with the permission. A permission can be specified by `permid` or `permsid`.

Permissions:

b_virtualserver_permission_find
b_serverinstance_permission_find

Syntax:

```
permfind [permid={permID}…] [permsid={permName}…]
```

Example:

```
permfind permid=4353
  t=0 id1=1 id2=0 p=4353|t=0 id1=2 id2=0 p=4353
  error id=0 msg=ok
```

PERMRESET

Restores the default permission settings on the selected virtual server and creates a new initial administrator token. Please note that in case of an error during the `permreset` call - e.g. when the database has been modified or corrupted - the virtual server will be deleted from the database.

Permissions:
b_virtualserver_permission_reset

Syntax:
```
permreset
```

Example:
```
permreset
  token=eKnFZQ9EK7G7MhtuQB6+N2B1PNZZ6OZL3ycDp2OW
  error id=0 msg=ok
```

PRIVILEGEKEYLIST

Displays a list of privilege keys available including their type and group IDs. Tokens can be used to gain access to specified server or channel groups.

A privilege key is similar to a client with administrator privileges that adds you to a certain permission group, but without the necessity of a such a client with administrator privileges to actually exist. It is a long (random looking) string that can be used as a ticket into a specific server group.

Permissions:
b_virtualserver_token_list

Syntax:
```
privilegekeylist
```

Example:
```
privilegekeylist
  token=88CVUg\/zkujt+y+WfHdko79UcM4R6uyCL6nEfy3B token_type=0 token_id1=9 token_id2=0 …
  error id=0 msg=ok
```

PRIVILEGEKEYADD

Create a new token. If `tokentype` is set to *0*, the ID specified with `tokenid1` will be a server group ID. Otherwise, `tokenid1` is used as a channel group ID and you need to provide a valid channel ID using `tokenid2`.

The `tokencustomset` parameter allows you to specify a set of custom client properties. This feature can be used when generating tokens to combine a website account database with a TeamSpeak user. The syntax of the value needs to be escaped using the ServerQuery escape patterns and has to follow the general syntax of:

```
ident=ident1 value=value1|ident=ident2 value=value2|ident=ident3 value=value3
```

Permissions:
b_virtualserver_token_add

Syntax:
```
privilegekeyadd tokentype={1|0} tokenid1={groupID} tokenid2={channelID}
[tokendescription={description}] [tokencustomset={customFieldSet}]
```

Example:
```
tokenadd tokentype=0 tokenid1=6 tokenid2=0 tokendescription=Test\stoken\swith\scustom\sset
tokencustomset=ident=forum_user\svalue=Sven\sPaulsen\pident=forum_id\svalue=123
  token=eKnFZQ9EK7G7MhtuQB6+N2B1PNZZ6OZL3ycDp2OW
  error id=0 msg=ok
```

PRIVILEGEKEYDELETE

Deletes an existing token matching the token key specified with `token`.

Permissions:
b_virtualserver_token_delete

Syntax:
```
privilegekeydelete token={tokenKey}
```

Example:
```
privilegekeydelete token=eKnFZQ9EK7G7MhtuQB6+N2B1PNZZ6OZL3ycDp2OW
  error id=0 msg=ok
```

PRIVILEGEKEYUSE

Use a token key gain access to a server or channel group. Please note that the server will automatically delete the token after it has been used.

Permissions:
b_virtualserver_token_use

Syntax:
```
privilegekeyuse token={tokenKey}
```

Example:
```
privilegekeyuse token=eKnFZQ9EK7G7MhtuQB6+N2B1PNZZ6OZL3ycDp2OW
  error id=0 msg=ok
```

MESSAGELIST

Displays a list of offline messages you've received. The output contains the senders unique identifier, the messages subject, etc.

Syntax:
messagelist

Example:
```
messagelist
  msgid=4 cluid=xwEzb5ENOaglVHu9oelK++reUyE= subject=Test timestamp=1259439465 flag_read=0 …
  error id=0 msg=ok
```

MESSAGEADD

Sends an offline message to the client specified by `cluid`.

Syntax:
```
messageadd cluid={clientUID} subject={subject} message={text}
```

Example:
```
messageadd cluid=oHhi9WzXLNEFQOwAu4JYKGU+C+c= subject=Hi! message=Where\aare\syou?!?
  error id=0 msg=ok
```

MESSAGEDEL

Deletes an existing offline message with ID `msgid` from your inbox.

Syntax:
```
messagedel msgid={messageID}
```

Example:
```
messagedel msgid=4
  error id=0 msg=ok
```

MESSAGEGET

Displays an existing offline message with ID `msgid` from your inbox. Please note that this does not automatically set the `flag_read` property of the message.

Syntax:

```
messageget msgid={messageID}
```

Example:

```
messageget msgid=4
  msgid=4 cluid=xwEzb5ENOaglVHu9oelK++reUyE= subject=Hi! message=Where\aare\syou?!?
  error id=0 msg=ok
```

MESSAGEUPDATEFLAG

Updates the `flag_read` property of the offline message specified with `msgid`. If flag is set to *1*, the message will be marked as read.

Syntax:

```
messageupdateflag msgid={messageID} flag={1|0}
```

Example:

```
messageupdateflag msgid=4 flag=1
  error id=0 msg=ok
```

COMPLAINLIST

Displays a list of complaints on the selected virtual server. If `tcldbid` is specified, only complaints about the targeted client will be shown.

Permissions:

b_client_complain_list

Syntax:

```
complainlist [tcldbid={targetClientDBID}]
```

Example:

```
complainlist tcldbid=3
  tcldbid=3 tname=Julian fcldbid=56 fname=Sven message=Bad\sguy! timestamp=1259440948 …
  error id=0 msg=ok
```

COMPLAINADD

Submits a complaint about the client with database ID `tcldbid` to the server.

Permissions:

i_client_complain_power
i_client_needed_complain_power

Syntax:

```
complainadd tcldbid={targetClientDBID} message={text}
```

Example:

```
complainadd tcldbid=3 message=Bad\sguy!
  error id=0 msg=ok
```

COMPLAINDELALL

Deletes all complaints about the client with database ID `tcldbid` from the server.

Permissions:

b_client_complain_delete

Syntax:

```
complaindelall tcldbid={targetClientDBID}
```

Example:

```
complaindelall tcldbid=3
  error id=0 msg=ok
```

COMPLAINDEL

Deletes the complaint about the client with ID `tcldbid` submitted by the client with ID `fcldbid` from the server.

Permissions:
b_client_complain_delete
b_client_complain_delete_own

Syntax:
```
complaindel tcldbid={targetClientDBID} fcldbid={fromClientDBID}
```

Example:
```
complaindel tcldbid=3 fcldbid=4
  error id=0 msg=ok
```

BANCLIENT

Bans the client specified with ID `clid` from the server. Please note that this will create two separate ban rules for the targeted clients IP address and his unique identifier.

Permissions:
i_client_ban_power
i_client_needed_ban_power

Syntax:
```
banclient clid={clientID} [time={timeInSeconds}] [banreason={text}]
```

Example:
```
banclient clid=4 time=3600
  banid=2
  banid=3
  error id=0 msg=ok
```

BANLIST

Displays a list of active bans on the selected virtual server.

Permissions:
b_client_ban_list

Syntax:
```
banlist
```

Example:
```
banlist
  banid=7 ip=1.2.3.4 created=1259444002242 invokername=Sven invokercldbid=56 invokeruid=oHhi9WzXLNEFQOwAu4JYKGU+C+c=
  reason enforcements=0
  error id=0 msg=ok
```

BANADD

Adds a new ban rule on the selected virtual server. All parameters are optional but at least one of the following must be set: `ip`, `name`, or `uid`.

Permissions:
b_client_ban_create

Syntax:
```
banadd [ip={regexp}] [name={regexp}] [uid={clientUID}]
[time={timeInSeconds}] [banreason={text}]
```

Example:
```
banadd ip=1.2.3.4 banreason=just\s4\sfun
  banid=1
  error id=0 msg=ok
```

BANDEL

Deletes the ban rule with ID `banid` from the server.

Permissions:
b_client_ban_delete
b_client_ban_delete_own

Syntax:
bandel banid={banID}

Example:
```
bandel banid=3
  error id=0 msg=ok
```

BANDELALL

Deletes all active ban rules from the server.

Permissions:
b_client_ban_delete

Syntax:
```
bandelall
```

Example:
```
bandelall
  error id=0 msg=ok
```

FTINITUPLOAD

Initializes a file transfer upload. `clientftfid` is an arbitrary ID to identify the file transfer on client-side. On success, the server generates a new `ftkey` which is required to start uploading the file through *TeamSpeak 3*'s file transfer interface.

Permissions:
i_ft_file_upload_power
i_ft_needed_file_ upload _power
i_ft_quota_mb_upload_per_client

Syntax:
```
ftinitupload clientftfid={clientFileTransferID} name={filePath} cid={channelID}
cpw={channelPassword} size={fileSize} overwrite={1|0} resume={1|0}
```

Example:
```
ftinitupload clientftfid=1 name=\/image.iso cid=5 cpw= size=673460224 overwrite=1 resume=0
  clientftfid=1 serverftfid=6 ftkey=itRNdsIOvcBiBg\/Xj4Ge51ZSrsShHuid port=30033 seekpos=0
  error id=0 msg=ok
```

FTINITDOWNLOAD

Initializes a file transfer download. `clientftfid` is an arbitrary ID to identify the file transfer on client-side. On success, the server generates a new `ftkey` which is required to start downloading the file through *TeamSpeak 3*'s file transfer interface.

Permissions:
i_ft_file_download_power
i_ft_needed_file_ download _power
i_ft_quota_mb_ download _per_client

Syntax:
```
ftinitdownload clientftfid={clientFileTransferID} name={filePath} cid={channelID}
cpw={channelPassword} seekpos={seekPosition}
```

Example:
```
ftinitdownload clientftfid=1 name=\/image.iso cid=5 cpw= seekpos=0
  clientftfid=1 serverftfid=7 ftkey=NrOga\/4d2GpYC5oKgxuclTO37X83ca\/1 port=30033 size=673460224
  error id=0 msg=ok
```

FTLIST

Displays a list of running file transfers on the selected virtual server. The output contains the path to which a file is uploaded to, the current transfer rate in bytes per second, etc.

Permissions:
b_ft_transfer_list

Syntax:
```
ftlist
```

Example:
```
ftlist
  clid=2 path=files\/virtualserver_1\/channel_5 name=image.iso size=673460224 sizedone=450756 clientftfid=2
  serverftfid=6 sender=0 status=1 current_speed=60872.8 …
  error id=0 msg=ok
```

FTGETFILELIST

Displays a list of files and directories stored in the specified channels file repository.

Permissions:
i_ft_file_browse_power
i_ft_needed_file_browse_power

Syntax:
```
ftgetfilelist cid={channelID} cpw={channelPassword} path={filePath}
```

Example:
```
ftgetfilelist cid=2 cpw= path=\/
  cid=2 path=\/ name=Stuff size=0 datetime=1259415210 type=0|name=Pic1.PNG size=563783 datetime=1259425462
  type=1|name=Pic2.PNG …
  error id=0 msg=ok
```

FTGETFILEINFO

Displays detailed information about one or more specified files stored in a channels file repository.

Permissions:
i_ft_file_browse_power
i_ft_needed_file_browse_power

Syntax:
```
ftgetfileinfo cid={channelID} cpw={channelPassword} name={filePath}…
```

Example:
```
ftgetfileinfo cid=2 cpw= path=\/Pic1.PNG|cid=2 cpw= path=\/Pic2.PNG
  cid=2 path=\/ name=Stuff size=0 datetime=1259415210 type=0|name=Pic1.PNG size=563783 datetime=1259425462
  type=1|name=Pic2.PNG …
  error id=0 msg=ok
```

FTSTOP

Stops the running file transfer with server-side ID `serverftfid`.

Syntax:
```
ftstop serverftfid={serverFileTransferID} delete={1|0}
```

Example:
```
ftstop serverftfid=2 delete=1
  error id=0 msg=ok
```

FTDELETEFILE

Deletes one or more files stored in a channels file repository.

Permissions:
i_ft_file_delete_power
i_ft_needed_file_delete_power

Syntax:
```
ftdeletefile cid={channelID} cpw={channelPassword} name={filePath}…
```

Example:
```
ftdeletefile cid=2 cpw= name=\/Pic1.PNG|name=\/Pic2.PNG
  error id=0 msg=ok
```

FTCREATEDIR

Creates new directory in a channels file repository.

Permissions:
i_ft_directory_create_power
i_ft_needed_file_directory_create_power

Syntax:
```
ftcreatedir cid={channelID} cpw={channelPassword} dirname={dirPath}
```

Example:
```
ftcreatedir cid=2 cpw= dirname=\/My\sDirectory
  error id=0 msg=ok
```

FTRENAMEFILE

Renames a file in a channels file repository. If the two parameters `tcid` and `tcpw` are specified, the file will be moved into another channels file repository.

Permissions:
i_ft_file_rename_power
i_ft_needed_file_rename_power

Syntax:
```
ftrenamefile cid={channelID} cpw={channelPassword} [tcid={targetChannelID}]
[tcpw={targetChannelPassword}] oldname={oldFilePath} newname={newFilePath}
```

Example:
```
ftrenamefile cid=2 cpw= tcid=3 tcpw=secret oldname=\/Pic3.PNG newname=\/Pic3.PNG
  error id=0 msg=ok
```

CUSTOMSEARCH

Searches for custom client properties specified by `ident` and `value`. The value parameter can include regular characters and SQL wildcard characters (e.g. `%`).

Syntax:
```
customsearch ident={ident} pattern={pattern}
```

Example:
```
customsearch ident=forum_account pattern=%ScP%
  cldbid=2 ident=forum_account value=ScP
  error id=0 msg=ok
```

CUSTOMINFO

Displays a list of custom properties for the client specified with `cldbid`.

Syntax:

```
custominfo cldbid={clientDBID}
```

Example:

```
custominfo cldbid=3
  cldbid=3 ident=forum_account value=ScP|ident=forum_id value=123
  error id=0 msg=ok
```

WHOAMI

Displays information about your current *ServerQuery* connection including your loginname, etc.

Syntax:

```
whoami
```

Example:

```
whoami
  virtualserver_status=online virtualserver_id=1 virtualserver_unique_identifier=zrPkjznB1tMnRwjO1xx7RxXjqeY=
  client_channel_id=2 …
  error id=0 msg=ok
```

SERVER INSTANCE PROPERTIES

This is a list of properties available for the server instance:

NAME	*CHANGABLE*
INSTANCE_UPTIME *Uptime in seconds*	No
HOST_TIMESTAMP_UTC *Current server date and time as UTC timestamp*	No
VIRTUALSERVERS_RUNNING_TOTAL *Number of virtual servers running*	No
CONNECTION_FILETRANSFER_BANDWIDTH_SENT *Current bandwidth used for outgoing file transfers (Bytes/s)*	No
CONNECTION_FILETRANSFER_BANDWIDTH_RECEIVED *Current bandwidth used for incoming file transfers (Bytes/s)*	No
CONNECTION_PACKETS_SENT_TOTAL *Total amount of packets sent*	No
CONNECTION_PACKETS_RECEIVED_TOTAL *Total amount of packets received*	No
CONNECTION_BYTES_SENT_TOTAL *Total amount of bytes sent*	No
CONNECTION_BYTES_RECEIVED_TOTAL *Total amount of bytes received*	No
CONNECTION_BANDWIDTH_SENT_LAST_SECOND_TOTAL *Average bandwidth used for outgoing data in the last second (Bytes/s)*	No
CONNECTION_BANDWIDTH_RECEIVED_LAST_SECOND_TOTAL *Average bandwidth used for incoming data in the last second (Bytes/s)*	No
CONNECTION_BANDWIDTH_SENT_LAST_MINUTE_TOTAL *Average bandwidth used for outgoing data in the last minute (Bytes/s)*	No
CONNECTION_BANDWIDTH_RECEIVED_LAST_MINUTE_TOTAL *Average bandwidth used for incoming data in the last minute (Bytes/s)*	No
SERVERINSTANCE_DATABASE_VERSION *Database revision number*	No
SERVERINSTANCE_ GUEST_SERVERQUERY_GROUP *Default ServerQuery group ID*	Yes
SERVERINSTANCE_TEMPLATE_SERVERADMIN_GROUP *Default template group ID for administrators on new virtual servers (used to create initial token)*	Yes
SERVERINSTANCE_FILETRANSFER_PORT *TCP port used for file transfers*	Yes
SERVERINSTANCE_MAX_DOWNLOAD_TOTAL_BANDWITDH *Max bandwidth available for outgoing file transfers (Bytes/s)*	Yes
SERVERINSTANCE_MAX_UPLOAD_TOTAL_BANDWITDH *Max bandwidth available for incoming file transfers (Bytes/s)*	Yes
SERVERINSTANCE_TEMPLATE_SERVERDEFAULT_GROUP *Default server group ID used in templates*	Yes
SERVERINSTANCE_TEMPLATE_CHANNELDEFAULT_GROUP *Default channel group ID used in templates*	Yes
SERVERINSTANCE_TEMPLATE_CHANNELADMIN_GROUP *Default channel administrator group ID used in templates*	Yes
VIRTUALSERVERS_TOTAL_MAXCLIENTS *Max number of clients for all virtual servers*	No
VIRTUALSERVERS_TOTAL_CLIENTS_ONLINE *Number of clients online on all virtual servers*	No
VIRTUALSERVERS_TOTAL_CHANNELS_ONLINE *Number of channels on all virtual servers*	No
SERVERINSTANCE_SERVERQUERY_FLOOD_COMMANDS *Max number of commands allowed in <SERVERINSTANCE_SERVERQUERY_FLOOD_TIME> seconds*	Yes
SERVERINSTANCE_SERVERQUERY_FLOOD_TIME *Timeframe in seconds for <SERVERINSTANCE_SERVERQUERY_FLOOD_COMMANDS> commands*	Yes
SERVERINSTANCE_SERVERQUERY_FLOOD_BAN_TIME *Time in seconds used for automatic bans triggered by the ServerQuery flood protection*	Yes

VIRTUAL SERVER PROPERTIES

This is a list of properties available for virtual servers:

NAME	*CHANGABLE*
VIRTUALSERVER_NAME *Name of the virtual server*	Yes
VIRTUALSERVER_WELCOMEMESSAGE *Welcome message of the virtual server*	Yes
VIRTUALSERVER_MAXCLIENTS *Number of slots available on the virtual server*	Yes
VIRTUALSERVER_PASSWORD *Password of the virtual server*	Yes
VIRTUALSERVER_FLAG_PASSWORD *Indicates whether the server has a password set or not*	No
VIRTUALSERVER_CLIENTSONLINE *Number of clients connected to the virtual server*	No
VIRTUALSERVER_QUERYCLIENTSONLINE *Number of ServerQuery clients connected to the virtual server*	No
VIRTUALSERVER_CHANNELSONLINE *Number of channels created on the virtual server*	No
VIRTUALSERVER_CREATED *Creation date and time of the virtual server as UTC timestamp*	No
VIRTUALSERVER_UPTIME *Uptime in seconds*	No
VIRTUALSERVER_HOSTMESSAGE *Host message of the virtual server*	Yes
VIRTUALSERVER_HOSTMESSAGE_MODE *Host message mode of the virtual server (see Definitions)*	Yes
VIRTUALSERVER_DEFAULT_SERVER_GROUP *Default server group ID*	Yes
VIRTUALSERVER_DEFAULT_CHANNEL_GROUP *Default channel group ID*	Yes
VIRTUALSERVER_DEFAULT_CHANNEL_ADMIN_GROUP *Default channel administrator group ID*	Yes
VIRTUALSERVER_PLATFORM *Operating system the server is running on*	No
VIRTUALSERVER_VERSION *Server version information including build number*	No
VIRTUALSERVER_MAX_DOWNLOAD_TOTAL_BANDWIDTH *Max bandwidth for outgoing file transfers on the virtual server (Bytes/s)*	Yes
VIRTUALSERVER_MAX_UPLOAD_TOTAL_BANDWIDTH *Max bandwidth for incoming file transfers on the virtual server (Bytes/s)*	Yes
VIRTUALSERVER_HOSTBANNER_URL *Host banner URL opened on click*	Yes
VIRTUALSERVER_HOSTBANNER_GFX_URL *Host banner URL used as image source*	Yes
VIRTUALSERVER_HOSTBANNER_GFX_INTERVAL *Interval for reloading the banner on client-side*	Yes
VIRTUALSERVER_COMPLAIN_AUTOBAN_COUNT *Number of complaints needed to ban a client automatically*	Yes
VIRTUALSERVER_COMPLAIN_AUTOBAN_TIME *Time in seconds used for automatic bans triggered by complaints*	Yes
VIRTUALSERVER_COMPLAIN_REMOVE_TIME *Time in seconds before a complaint is deleted automatically*	Yes
VIRTUALSERVER_MIN_CLIENTS_IN_CHANNEL_BEFORE_FORCED_SILENCE *Number of clients in the same channel needed to force silence*	Yes
VIRTUALSERVER_PRIORITY_SPEAKER_DIMM_MODIFICATOR *Client volume lowered automatically while a priority speaker is talking*	Yes
VIRTUALSERVER_ANTIFLOOD_POINTS_TICK_REDUCE *Anti-flood points removed from a client for being good*	Yes

NAME	*CHANGABLE*
VIRTUALSERVER_ANTIFLOOD_POINTS_NEEDED_WARNING *Anti-flood points needed to receive a warning message from the server*	Yes
VIRTUALSERVER_ANTIFLOOD_POINTS_NEEDED_KICK *Anti-flood points needed to be kicked from the server*	Yes
VIRTUALSERVER_ANTIFLOOD_POINTS_NEEDED_BAN *Anti-flood points needed to be banned from the server*	Yes
VIRTUALSERVER_ANTIFLOOD_POINTS _BAN_TIME *Time in seconds used for automatic bans triggered by the flood protection*	Yes
VIRTUALSERVER_CLIENT_CONNECTIONS *Total number of clients connected to the virtual server since it was last started*	No
VIRTUALSERVER_QUERY_CLIENT_CONNECTIONS *Total number of ServerQuery clients connected to the virtual server since it was last started*	No
VIRTUALSERVER_HOSTBUTTON_TOOLTIP *Text used for the tooltip of the host button on client-side*	Yes
VIRTUALSERVER_HOSTBUTTON_GFX_URL *Text used for the tooltip of the host button on client-side*	Yes
VIRTUALSERVER_HOSTBUTTON_URL *URL opened on click on the host button*	Yes
VIRTUALSERVER_DOWNLOAD_QUOTA *Download quota for the virtual server (MByte)*	Yes
VIRTUALSERVER_UPLOAD_QUOTA *Download quota for the virtual server (MByte)*	Yes
VIRTUALSERVER_MONTH_BYTES_DOWNLOADED *Number of bytes downloaded from the virtual server on the current month*	No
VIRTUALSERVER_MONTH_BYTES_UPLOADED *Number of bytes uploaded to the virtual server on the current month*	No
VIRTUALSERVER_TOTAL_BYTES_DOWNLOADED *Number of bytes downloaded from the virtual server since it was last started*	No
VIRTUALSERVER_ TOTAL_BYTES_UPLOADED *Number of bytes uploaded to the virtual server since it was last started*	No
VIRTUALSERVER_UNIQUE_IDENTIFER *Unique ID of the virtual server*	No
VIRTUALSERVER_ID *Database ID of the virtual server*	No
VIRTUALSERVER_MACHINE_ID *Machine ID identifying the server instance associated with the virtual server in the database*	Yes
VIRTUALSERVER_PORT *UDP port the virtual server is listening on*	Yes
VIRTUALSERVER_AUTOSTART *Indicates whether the server starts automatically with the server instance or not*	Yes
CONNECTION_FILETRANSFER_BANDWIDTH_SENT *Current bandwidth used for outgoing file transfers (Bytes/s)*	No
CONNECTION_FILETRANSFER_BANDWIDTH_RECEIVED *Current bandwidth used for incoming file transfers (Bytes/s)*	No
CONNECTION_PACKETS_SENT_TOTAL *Total amount of packets sent*	No
CONNECTION_PACKETS_RECEIVED_TOTAL *Total amount of packets received*	No
CONNECTION_BYTES_SENT_TOTAL *Total amount of bytes sent*	No
CONNECTION_BYTES_RECEIVED_TOTAL *Total amount of bytes received*	No
CONNECTION_BANDWIDTH_SENT_LAST_SECOND_TOTAL *Average bandwidth used for outgoing data in the last second (Bytes/s)*	No
CONNECTION_BANDWIDTH_RECEIVED_LAST_SECOND_TOTAL *Average bandwidth used for incoming data in the last second (Bytes/s)*	No
CONNECTION_BANDWIDTH_SENT_LAST_MINUTE_TOTAL *Average bandwidth used for outgoing data in the last minute (Bytes/s)*	No
CONNECTION_BANDWIDTH_RECEIVED_LAST_MINUTE_TOTAL *Average bandwidth used for incoming data in the last minute (Bytes/s)*	No

NAME	*CHANGABLE*
VIRTUALSERVER_STATUS *Status of the virtual server (online | virtual online | offline | booting up | shutting down | ...)*	Yes
VIRTUALSERVER_ LOG_CLIENT *Indicates whether the server logs events related to clients or not*	Yes
VIRTUALSERVER_ LOG_QUERY *Indicates whether the server logs events related to ServerQuery clients or not*	Yes
VIRTUALSERVER_ LOG_CHANNEL *Indicates whether the server logs events related to channels or not*	Yes
VIRTUALSERVER_ LOG_PERMISSIONS *Indicates whether the server logs events related to permissions or not*	Yes
VIRTUALSERVER_ LOG_SERVER *Indicates whether the server logs events related to server changes or not*	Yes
VIRTUALSERVER_ LOG_FILETRANSFER *Indicates whether the server logs events related to file transfers or not*	Yes
VIRTUALSERVER_MIN_CLIENT_VERSION *Min client version required to connect*	Yes
VIRTUALSERVER_NEEDED_IDENTITY_SECURITY_LEVEL *Minimum client identity security level required to connect to the virtual server*	Yes
VIRTUALSERVER_NAME_PHONETIC *Phonetic name of the virtual server*	Yes
VIRTUALSERVER_ICON_ID *CRC32 checksum of the virtual server icon*	Yes
VIRTUALSERVER_RESERVED_SLOTS *Number of reserved slots available on the virtual server*	Yes
VIRTUALSERVER_TOTAL_PACKETLOSS_SPEECH *The average packet loss for speech data on the virtual server*	No
VIRTUALSERVER_TOTAL_PACKETLOSS_KEEPALIVE *The average packet loss for keepalive data on the virtual server*	No
VIRTUALSERVER_TOTAL_PACKETLOSS_CONTROL *The average packet loss for control data on the virtual server*	No
VIRTUALSERVER_TOTAL_PACKETLOSS_TOTAL *The average packet loss for all data on the virtual server*	No
VIRTUALSERVER_TOTAL_PING *The average ping of all clients connected to the virtual server*	No
VIRTUALSERVER_IP *The IPv4 address the virtual server is listening on*	No
VIRTUALSERVER_WEBLIST_ENABLED *Indicates whether the server appears in the global web server list or not*	Yes
VIRTUALSERVER_CODEC_ENCRYPTION_MODE *The global codec encryption mode of the virtual server*	Yes
VIRTUALSERVER_FILEBASE *The directory where the virtual servers filebase is located*	No
VIRTUALSERVER_HOSTBANNER_MODE *The display mode for the virtual servers hostbanner (see Definitions)*	Yes
VIRTUALSERVER_ASK_FOR_PRIVILEGEKEY *Indicates whether the initial privilege key for the virtual server has been used or not*	No

CHANNEL PROPERTIES

This is a list of properties available for channels:

NAME	*CHANGABLE*
CHANNEL_NAME *Name of the channel*	Yes
CHANNEL_TOPIC *Topic of the channel*	Yes
CHANNEL_DESCRIPTION *Description of the channel*	Yes
CHANNEL_PASSWORD *Password of the channel*	Yes
CHANNEL_FLAG_PASSWORD *Indicates whether the channel has a password set or not*	No
CHANNEL CODEC *Codec used by the channel (see Definitions)*	Yes
CHANNEL_CODEC_QUALITY *Codec quality used by the channel*	Yes
CHANNEL_MAXCLIENTS *Individual max number of clients for the channel*	Yes
CHANNEL_MAXFAMILYCLIENTS *Individual max number of clients for the channel family*	Yes
CHANNEL_ORDER *ID of the channel below which the channel is positioned*	Yes
CHANNEL_FLAG_PERMANENT *Indicates whether the channel is permanent or not*	Yes
CHANNEL_FLAG_SEMI_PERMANENT *Indicates whether the channel is semi-permanent or not*	Yes
CHANNEL_FLAG_TEMPORARY *Indicates whether the channel is temporary or not*	Yes
CHANNEL_FLAG_DEFAULT *Indicates whether the channel is the virtual servers default channel or not*	Yes
CHANNEL_FLAG_MAXCLIENTS_UNLIMITED *Indicates whether the channel has a max clients limit or not*	Yes
CHANNEL_FLAG_MAXFAMILYCLIENTS_UNLIMITED *Indicates whether the channel has a max family clients limit or not*	Yes
CHANNEL_FLAG_MAXFAMILYCLIENTS_INHERITED *Indicates whether the channel inherits the max family clients from his parent channel or not*	Yes
CHANNEL_NEEDED_TALK_POWER *Needed talk power for this channel*	Yes
CHANNEL_NAME_PHONETIC *Phonetic name of the channel*	Yes
CHANNEL_FILEPATH *Path of the channels file repository*	No
CHANNEL_FORCED_SILENCE *Indicates whether the channel is silenced or not*	No
CHANNEL_ICON_ID *CRC32 checksum of the channel icon*	Yes
CHANNEL_CODEC_IS_UNENCRYPTED *Indicates whether speech data transmitted in this channel is encrypted or not*	Yes
CPID *The channels parent ID*	Yes
CID *The channels ID*	No

CLIENT PROPERTIES

This is a list of properties available for clients:

NAME	*CHANGABLE*
CLIENT_UNIQUE_IDENTIFIER *Unique ID of the client*	No
CLIENT_NICKNAME *Nickname of the client*	Yes
CLIENT_VERSION *Client version information including build number*	No
CLIENT_PLATFORM *Operating system the client is running on*	No
CLIENT_INPUT_MUTED *Indicates whether the client has their microphone muted or not*	No
CLIENT_OUTPUT_MUTED *Indicates whether the client has their speakers muted or not*	No
CLIENT_INPUT_HARDWARE *Indicates whether the client has enabled their capture device or not*	No
CLIENT_OUTPUT_HARDWARE *Indicates whether the client has enabled their playback device or not*	No
CLIENT_DEFAULT_CHANNEL *Default channel of the client*	No
CLIENT_LOGIN_NAME *Username of a ServerQuery client*	No
CLIENT_DATABASE_ID *Database ID of the client*	No
CLIENT_CHANNEL_GROUP_ID *Current channel group ID of the client*	No
CLIENT_SERVER_GROUPS *Current server group IDs of the client separated by a comma*	No
CLIENT_CREATED *Creation date and time of the clients first connection to the server as UTC timestamp*	No
CLIENT_LASTCONNECTED *Creation date and time of the clients last connection to the server as UTC timestamp*	No
CLIENT_TOTALCONNECTIONS *Total number of connections from this client since the server was started*	No
CLIENT_AWAY *Indicates whether the client is away or not*	No
CLIENT_AWAY_MESSAGE *Away message of the client*	No
CLIENT_TYPE *Indicates whether the client is a ServerQuery client or not*	No
CLIENT_FLAG_AVATAR *Indicates whether the client has set an avatar or not*	No
CLIENT_TALK_POWER *The clients current talk power*	No
CLIENT_TALK_REQUEST *Indicates whether the client is requesting talk power or not*	No
CLIENT_TALK_REQUEST_MSG *The clients current talk power request message*	No
CLIENT_IS_TALKER *Indicates whether the client is able to talk or not*	Yes
CLIENT_MONTH_BYTES_DOWNLOADED *Number of bytes downloaded by the client on the current month*	No
CLIENT_MONTH_BYTES_UPLOADED *Number of bytes uploaded by the client on the current month*	No
CLIENT_TOTAL_BYTES_DOWNLOADED *Number of bytes downloaded by the client since the server was started*	No
CLIENT_TOTAL_BYTES_UPLOADED *Number of bytes uploaded by the client since the server was started*	No

NAME	*CHANGABLE*
CLIENT_IS_PRIORITY_SPEAKER *Indicates whether the client is a priority speaker or not*	No
CLIENT_UNREAD_MESSAGES *Number of unread offline messages in this clients inbox*	No
CLIENT_NICKNAME_PHONETIC *Phonetic name of the client*	No
CLIENT_DESCRIPTION *Brief description of the client*	Yes
CLIENT_NEEDED_SERVERQUERY_VIEW_POWER *The clients current ServerQuery view power*	No
CONNECTION_FILETRANSFER_BANDWIDTH_SENT *Current bandwidth used for outgoing file transfers (Bytes/s)*	No
CONNECTION_FILETRANSFER_BANDWIDTH_RECEIVED *Current bandwidth used for incoming file transfers (Bytes/s)*	No
CONNECTION_PACKETS_SENT_TOTAL *Total amount of packets sent*	No
CONNECTION_PACKETS_RECEIVED_TOTAL *Total amount of packets received*	No
CONNECTION_BYTES_SENT_TOTAL *Total amount of bytes sent*	No
CONNECTION_BYTES_RECEIVED_TOTAL *Total amount of bytes received*	No
CONNECTION_BANDWIDTH_SENT_LAST_SECOND_TOTAL *Average bandwidth used for outgoing data in the last second (Bytes/s)*	No
CONNECTION_BANDWIDTH_RECEIVED_LAST_SECOND_TOTAL *Average bandwidth used for incoming data in the last second (Bytes/s)*	No
CONNECTION_BANDWIDTH_SENT_LAST_MINUTE_TOTAL *Average bandwidth used for outgoing data in the last minute (Bytes/s)*	No
CONNECTION_BANDWIDTH_RECEIVED_LAST_MINUTE_TOTAL *Average bandwidth used for incoming data in the last minute (Bytes/s)*	No
CONNECTION_CLIENT_IP *The IPv4 address of the client*	No
CLIENT_IS_CHANNEL_COMMANDER *Indicates whether the client is a channel commander or not*	Yes
CLIENT_ICON_ID *CRC32 checksum of the client icon*	Yes
CLIENT_COUNTRY *The country identifier of the client (i.e. DE)*	No

DEFINITIONS

The following enumerations can be used to change the behavior of various *ServerQuery* commands:

```
enum HostMessageMode {
  HostMessageMode_LOG = 1,                // 1: display message in chatlog
  HostMessageMode_MODAL,                  // 2: display message in modal dialog
  HostMessageMode_MODALQUIT               // 3: display message in modal dialog and close connection
};

enum HostBannerMode {
  HostMessageMode_NOADJUST = 0,           // 0: do not adjust
  HostMessageMode_IGNOREASPECT,           // 1: adjust but ignore aspect ratio (like TeamSpeak 2)
  HostMessageMode_KEEPASPECT              // 2: adjust and keep aspect ratio
};

enum Codec {
  CODEC_SPEEX_NARROWBAND = 0,             // 0: speex narrowband     (mono, 16bit, 8kHz)
  CODEC_SPEEX_WIDEBAND,                   // 1: speex wideband       (mono, 16bit, 16kHz)
  CODEC_SPEEX_ULTRAWIDEBAND,              // 2: speex ultra-wideband (mono, 16bit, 32kHz)
  CODEC_CELT_MONO                         // 3: celt mono            (mono, 16bit, 48kHz)
};

enum CodecEncryptionMode {
  CODEC_CRYPT_INDIVIDUAL = 0,             // 0: configure per channel
  CODEC_CRYPT_DISABLED,                   // 1: globally disabled
  CODEC_CRYPT_ENABLED                     // 2: globally enabled
};

enum TextMessageTargetMode {
  TextMessageTarget_CLIENT = 1,           // 1: target is a client
  TextMessageTarget_CHANNEL,              // 2: target is a channel
  TextMessageTarget_SERVER                // 3: target is a virtual server
};

enum LogLevel {
  LogLevel_ERROR = 1,                     // 1: everything that is really bad
  LogLevel_WARNING,                       // 2: everything that might be bad
  LogLevel_DEBUG,                         // 3: output that might help find a problem
  LogLevel_INFO                           // 4: informational output
};

enum ReasonIdentifier {
  REASON_KICK_CHANNEL = 4,                // 4: kick client from channel
  REASON_KICK_SERVER                      // 5: kick client from server
};

enum PermissionGroupDatabaseTypes {
  PermGroupDBTypeTemplate = 0,            // 0: template group     (used for new virtual servers)
  PermGroupDBTypeRegular,                 // 1: regular group      (used for regular clients)
  PermGroupDBTypeQuery                    // 2: global query group (used for ServerQuery clients)
};

enum PermissionGroupTypes {
  PermGroupTypeServerGroup = 0,           // 0: server group permission
  PermGroupTypeGlobalClient,              // 1: client specific permission
  PermGroupTypeChannel,                   // 2: channel specific permission
  PermGroupTypeChannelGroup,              // 3: channel group permission
  PermGroupTypeChannelClient              // 4: channel-client specific permission
};

enum TokenType {
  TokenServerGroup = 0,                   // 0: server group token (id1={groupID} id2=0)
  TokenChannelGroup                       // 1: channel group token (id1={groupID} id2={channelID})
};

enum PermissionAutoUpdateTypes {
  PermissionAutoUpdateQG = 0,             // 0: target will be handled as Query Guest
  PermissionAutoUpdateQA,                 // 1: target will be handled as Query Admin
  PermissionAutoUpdateSA,                 // 2: target will be handled as Server Admin
  PermissionAutoUpdateSN,                 // 3: target will be handled as Server Normal
  PermissionAutoUpdateSG,                 // 4: target will be handled as Server Guest
  PermissionAutoUpdateCA,                 // 5: target will be handled as Channel Admin
  PermissionAutoUpdateCO,                 // 6: target will be handled as Channel Operator
  PermissionAutoUpdateCV,                 // 7: target will be handled as Channel Voice
  PermissionAutoUpdateCG,                 // 8: target will be handled as Channel Guest
};
```