BFH PC Server Remote Administration Protocol

This is the remote-administration protocol used by BFH PC Server for Xpack 4 (Betrayal)~~R2.~~

~~It is work-in-progress; features are first added to the game, and then controlling commands are added to the Remote Administration interface.~~

Contents

About

This document describes how to communicate with the Remote Administration interface that is present in BFH PC servers. The protocol is bidirectional, and allows clients to send commands to the server as well as the server to send events to clients.

The protocol is designed for machine-readability, not human-readability. It is the basis for all graphical remote administration tools.

Low-level protocol

Packet format

int32

32-bit unsigned integer

1 byte	bits 7..0 of value
1 byte	bits 15..8 of value
1 byte	bits 23..16 of value
1 byte	bits 31..24 of value

Word

int32	Size	Number of bytes in word, excluding trailing null byte
char<>	Content	Word contents -- must not contain any null bytes
char	Terminator	Trailing null byte

Packet

int32	Sequence	Bit 31: 0 = The command in this command/response pair originated on the server 1 = The command in this command/response pair originated on the client Bit 30: 0 = Request, 1 = Response Bits 29..0: Sequence number (this is used to match requests/responses in a full duplex transmission)
int32	Size	Total size of packet, in bytes
int32	NumWords	Number of words following the packet header
Word<N>	Words	N words

A packet cannot be more than 16384 bytes in size.

Protocol behavior

The client communicates with the server using a request/response protocol. Each request contains a sequence number which grows monotonically, a flag which indicates whether the command originated on the client or the server, and one word containing the command name. In addition to this, a command can have zero or more arguments.

Every request must be acknowledged by a response. The response includes the same sequence number, and the same origin flag. However, it has the response flag set.

Sequence numbers are unique within one server-client connection. Thus, the same sequence number can be used when the server is communicating with different clients.

Responses must contain at least one word. The first word can be one of the following:

OK	- request completed successfully
UnknownCommand	- unknown command
InvalidArguments	- Arguments not appropriate for command
<other>	- command-specific error

OK is the only response which signifies success.
Subsequent arguments (if any) are command-specific.

The server is guaranteed to adhere to this protocol specification. If the client violates the protocol, the server may close the connection without any prior notice.

Comments

The format of the Words portion of a packet is designed such that it shall be easy to split it into individual words in both C++ and Python. Any numerical arguments are always transferred in string form (not in raw binary form).

The protocol is designed to be fully bidirectional.

Parameter formats

String
An 8bit ASCII string. Must not contain any characters with ASCII code 0.

Boolean
Two possible values:

true
false

HexString
A stream of hexadecimal digits. The stream must always contain an even number of digits. Allowed characters are: 0123456789ABCDEF

Password
A password is from 0 up to 16 characters in length, inclusive. The allowed characters are: abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789

Filename
A filename is from 1 up to 240 characters in length, inclusive. The allowed characters are: abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789._-

Clantag
A clan tag is from 0 to an unknown number of characters in length. At the time of writing, it is unclear which the allowed characters are.

Player name
The “player name” (referred to as “Soldier name” in-game) is the persona name which the player chose when logging in to EA Online. The exact specification of a player name (length, valid characters, etc.) is currently unclear. This is sometimes abbreviated as “name” in the document.

GUID
The GUID is a unique identifier for a player. It is 35 characters long, consists of the prefix “EA_” immediately followed by a 32-character HexString.

TeamId

An integer.

Team 0 is neutral. Depending on gamemode, there are up to 16 non-neutral teams, numbered 1…16.

SquadId
An integer.

Squad 0 is “no squad”. Depending on gamemode, there are up to 32 squads numbered 1…32.
Note that squad IDs are local within each team; that is, to uniquely identify a squad you need to specify both a Team ID and a Squad ID.

Player subset
Several commands – such as admin.listPlayers – take a player subset as argument.

A player subset is one of the following:

all	- all players on the server
team <team number: Team ID>	- all players in the specified team
squad <team number: Team ID> <squad number: Squad ID>	- all players in the specified team+squad
player <player name: string>	- one specific player

Timeout

Some commands, such as bans, take a timeout as argument.

A timeout is one of the following:

perm	- permanent
rounds <number of rounds: integer>	- number of rounds
seconds <number of seconds: integer>	- number of seconds

Id-type

Some commands, such as bans, take an id-type as argument

An id-type is one of the following:

name	- Soldier name
ip	- IP address
guid	- Player's GUID

Player info block

The standard set of info for a group of players contains a lot of different fields. To reduce the risk of having to do backwards-incompatible changes to the protocol, the player info block includes some formatting information.

<number of parameters>	- number of parameters for each player
N x <parameter type: string>	- the parameter types that will be sent below
<number of players>	- number of players following
M x N x <parameter value>	- all parameter values for player 0, then all parameter values for player 1, etc.

Current parameters:

name	string	- player name
guid	GUID	- player's GUID
teamId	Team ID	- player's current team
squadId	Squad ID	- player's current squad
kills	integer	- number of kills, as shown in the in-game scoreboard
deaths	integer	- number of deaths, as shown in the in-game scoreboard
score	integer	- score, as shown in the in-game scoreboard
rank	integer	- the rank of the player
ping	integer	- ping between the server and player

Team scores

This describes the number of tickets, or kills, for each team in the current round.

<number of entries: integer>	- number of team scores that follow
N x <score: integer>	- score for all teams
<target score: integer>	- when any team reaches this score, the match ends

IpPortPair

A string on the following format:

<IPv4 address>:<port number>

MapList

This describes the set of maps which the server rotates through. Format is as follows:

<number of maps: integer>	- number of maps that follow
<number of words per map: integer>	- number of words per map
<map name: string>	- name of map
<gamemode name: string>	- name of gamemode
<number of rounds: integer>	- number of rounds to play on map before switching

The reason for the <number of words per map> specification is future proofing; in the future, DICE might add extra words per map after the first three. However, the first three words are very likely to remain the same.

Server events

Most commands require the client to be logged in. Before the client has logged in, only 'login.plainText', 'login.hashed', 'logout', 'version', 'listPlayers', 'serverInfo' and 'quit' commands are available.

Summary

Command	Description
player.onAuthenticated	Player with name <soldier name> has joined the server
player.onJoin	Player with name <soldier name> has joined the server
player.onLeave	with name <soldier name> has left the server
player.onSpawn	Player with name <soldier name> has spawned in
player.onKill	Player with name <killing soldier name> has killed <killed soldier name>
player.onChat	Chat message has been sent to a group of people
player.onSquadChange	Player might have changed squad
player.onTeamChange	Player might have changed team
punkBuster.onMessage	PunkBuster server has output a message
server.onMaxPlayerCountChange	Effective max player count has changed
server.onLevelLoaded	Level has loaded
server.onRoundOver	Round has ended
server.onRoundOverPlayers	Player stats at end-of-round
server.onRoundOverTeamScores	Team stats at end-of-round

Player events

Request: player.onAuthenticated <soldier name: string>
Response: OK
Effect: Player with name <soldier name> has joined the server

Request: player.onJoin <soldier name: string> <id: GUID>
Response: OK
Effect: Player with name <soldier name> has joined the server

Request: player.onLeave <soldier name: string> <soldier info: player info block>
Response: OK
Effect: Player with name <soldier name> has left the server; his last set of stats were <soldier info>

Request: player.onSpawn <soldier name: string> <team: Team ID>
Response: OK
Effect: Player with name <soldier name> has spawned in, with team <team>
NOTE The <team> specifier is probably superfluous information and might get removed in the future

Request: player.onKill <killing soldier name: string> <killed soldier name: string> <weapon: string> <headshot: boolean>
Response: OK
Effect: Player with name <killing soldier name> has killed <killed soldier name>
Suicide indication is unknown at this moment.
If the server kills the player (through admin.killPlayer), the result is unknown.

Request: player.onChat <source soldier name: string> <text: string> <target players: player subset>
Response: OK
Effect: Player with name <source soldier name> (or the server, or the server admin) has sent chat message <text> to <target players>
Comment: If <source soldier name> is "Server", then the message was sent from the server rather than from an actual player

Request: player.onSquadChange <soldier name: player name> <team: Team ID> <squad: Squad ID>
Response: OK
Effect: Player might have changed squad

Request: player.onTeamChange <soldier name: player name> <team: Team ID> <squad: Squad ID>
Response: OK
Effect: Player might have changed team

Misc

Request: punkBuster.onMessage <message: string>
Response: OK
Effect: PunkBuster server has output a message
Comment: The entire message is sent as a raw string. It may contain newlines and whatnot.
Request: server.onMaxPlayerCountChange <count: int>
Response: OK
Effect: The game server has changed the effective max player count

Level/Round

Request: server.onLevelLoaded <level name: string> <gamemode: string> <roundsPlayed: int> <roundsTotal: int>
Response: OK
Effect: Level has completed loading, and will start in a bit

Request: server.onRoundOver <winning team: Team ID>
Response: OK
Effect: The round has just ended, and <winning team> won

Request: server.onRoundOverPlayers <end-of-round soldier info: player info block>
Response: OK
Effect: The round has just ended, and <end-of-round soldier info> is the final detailed player stats

Request: server.onRoundOverTeamScores <end-of-round scores: team scores>
Response: OK
Effect: The round has just ended, and <end-of-round scores> is the final ticket/kill/life count for each team

Client commands

Most commands require the client to be logged in. Before the client has logged in, only 'login.plainText', 'login.hashed', 'logout', 'version', 'serverInfo', 'listPlayers' and 'quit' commands are available.

Server type

For BFH, a new server type has been added on top of the BF3 legacy ones. “Official” is a sort of even more restricted “Ranked” server type. In Official kick and ban commands will not work and most other commands are restricted to certain values.

Summary

Command	*Description*
Misc	
login.plainText <password>	Attempt to login to game server with password
login.hashed	Retrieves the salt, used in the hashed password login process
login.hashed <passwordHard>	Sends a hashed password to the server, in an attempt to log in
serverinfo	Query for brief server info
logout	Logout from game server
quit	Disconnect from server
version	Reports game server type, and build ID
currentLevel	Return current map running on game server
listPlayers <players>	Return list of a group of players on the server, without GUIDs
Admin	
admin.eventsEnabled <enabled>	Set whether or not the server will send events to the current connection
admin.help	Report which commands the server knows about
admin.kickPlayer <player name> <reason>	Kick player <soldier name> from server
admin.killPlayer <player name>	Kill a player without scoring effects
admin.listPlayers <players>	Return list of a group of players on the server
admin.movePlayer <name> <teamId> <squadId> <forceKill>	Move a player to another team and squad
admin.password <password>>	Set the admin password for the server
admin.say <message> <players>	Send a chat message to a group of players
admin.shutDown <gracefulShutdown>	Shuts down the game server
admin.yell <message> <duration> <players>	Show a large on-screen message for a group of players
Banning	
banList.add <id-type> <id, timeout> reason>	Add player/IP/GUID to ban list for a certain amount of time
banList.clear	Clears ban list
banList.list <startIndex>	Return part of the list of banned players/IPs/GUIDs
banList.load	Load list of banned players/IPs/GUIDs from file
banList.remove <id-type> <id>	Remove player/IP/GUID from ban list
banList.save	Save list of banned players/IPs/GUIDs to file
FairFight	
fairFight.activate	Attempts to activate FairFight if it is not currently running
fairfight.deactivate	Attempts to deactivate FairFight if it is currently running
fairFight.isActive	Returns whether or not FairFight currently is active
Map list	
mapList.add <map> <gamemode> <rounds>	Insert map at specified offset in map list

<offset>	
mapList.availableMaps <filter>	Return list of available maps or gamemodes
mapList.clear	Clear map list
mapList.endRound <teamId>	End current round, declaring the specified team as winners
mapList.getMapIndices	Get indices for current & next map
mapList.getRounds	Get current round and number of rounds
mapList.list <startIndex>	Returns part of the map list
mapList.load	Load list of maps from disk
mapList.remove <index>	Remove specified map from map list
mapList.restartRound	Restart current round
mapList.runNextRound	Abort current round and move on to next
mapList.save	Save list of maps to disk
mapList.setNextMapIndex <index>	Set which map to switch to at end of current round

Player

player.idleDuration <player name>	Get idle duration for a soldier
player.isAlive <player name>	Check if the soldier is alive
player.ping <player name>	Get a soldiers ping to the server

PunkBuster

punkBuster.activate	Attempt to activate PunkBuster if it is not currently running
punkBuster.isActive	Returns whether or not PunkBuster currently is active
punkBuster.pb_sv_command <command>	Send a raw PunkBuster command to the PunkBuster server

Reserved Slots

reservedSlotsList.add <player name>	Add <name> to list of players who can use the reserved slots
reservedSlotsList.aggressiveJoin <enabled>	Set if the server should kick to make room for VIP
reservedSlotsList.clear	Clear reserved slots list
reservedSlotsList.list	Retrieve list of players who can utilize the reserved slots
reservedSlotsList.load	Load list of reserved soldier names from file
reservedSlotsList.remove <player name>	Remove <name> from list of players who can use the reserved slots
reservedSlotsList.save	Save list of reserved soldier names to file

Spectator list

//The Spectator list will only be in effect if vars.alwaysAllowSpectators is set to false

spectatorList.add <player name>	Add player to the spectator list
spectatorList.clear	Clears spectator list
spectatorList.list <startIndex>	Return part of the list of spectators
spectatorList.remove <player name>	Remove player from spectator list?
spectatorList.save	Save list of spectators to file

Squad

squad.leader <teamId> <squadId> <player name>	Get/Set the leader of a squad
squad.listActive <teamId>	Get all squads that have players in them on a specific team
squad.listPlayers <teamId> <squadId>	Get player count and names of soldiers in a specific squad
squad.private <teamId> <squadId> <private>	Get/Set whether a squad is private or not

Variables

Command	Description	When
vars.3dSpotting <enabled>	Set if spotted targets are visible in the 3d-world	
vars.3pCam <enabled>	Set if allowing to toggle to third person vehicle cameras	
vars.alwaysAllowSpectators <enabled>	Set whether spectators need to be in the spectator list before joining	ROS
vars.autoBalance <enabled>	Set if the server should autobalance	
vars.bulletDamage <modifier: percent>	Set bullet damage scale factor	ROR
vars.curatedUnlockList <index>	Set Curated Unlock List to use (0 – Disabled, 1 – Pistols Only, 2 – Melee Only)	
vars.hacker <enabled>	Set if Hacker is allowed or not on the game server	
vars.forceReloadWholeMags <enabled>	Set hardcore reload on or off	
vars.frameHistoryTime <modifier: float>	Set maximum latency to force process of frame data	
vars.friendlyFire <enabled>	Set if the server should allow team damage	
vars.gameModeCounter <modifier: integer>	Set scale factor for number of tickets to end round	
vars.gamePassword <password>	Set the game password for the server	
vars.hitIndicatorsEnabled <enabled>	Set if hit indicators are enabled or not	
vars.hud <enabled>	Set if HUD is enabled	
vars.idleBanRounds <enabled>	Set how many rounds idle timeout should ban (if at all)	
vars.idleTimeout <time>	Set idle timeout	
vars.killCam <enabled>	Set if killcam is enabled	
vars.killFeed <enabled>	Set if Kill Feed is enabled	
vars.maxPlayers <numPlayers>	Set desired maximum number of players	
vars.maxSpectators <numSpectators>	Set desired maximum number of spectators	
vars.minimap <enabled>	Set if minimap is enabled	
vars.miniMapSpotting <enabled>	Set if spotted targets are visible on the minimap	ROR
vars.mpExperience <experience>	Set the MP Experience of the game server	ROS
vars.nameTag <enabled>	Set if nametags should be displayed	
vars.onlySquadLeaderSpawn <enabled>	Set if players can only spawn on their squad leader	
vars.optimizeBandwidth <enabled>	Set if the server will optimize the bandwidth when High Tick Rate is enabled (>30 Hz)	
vars.outHighFrequency <tickrate>	Sets the tickrate of the server	NR
vars.playerManDownTime <modifier: percent>	Man Down time scale factor	ROR
vars.playerRespawnTime <modifier: percent>	Set player respawn time scale factor	
vars.preset <serverPreset> <lockPresetSetting>	Set the server preset. If lockPresetSetting is set to true, the preset will override any settings that conflicts with it and make sure that these settings cannot be changed without setting the server to Custom	
vars.randomStartingMap <enabled>	Controls if the starting map of the rotation is chosen randomly	
vars.regenerateHealth <enabled>	Set if health regeneration should be active	
vars.requireReadyPlayersToStart <enabled>	Set if needs ready players to start the round	
vars.roundLockdownCountdown <time>	Set the duration of pre-round	
vars.roundRestartPlayerCount <numPlayers>	Set minimum numbers of players to go from in-round to warm-up	
vars.roundStartPlayerCount <numPlayers>	Set minimum numbers of players to go from warm-up to pre-round/in- round	
vars.roundStartReadyPlayersPercent <modifier:	Players ready needed to start round scale factor	

percent>		
vars.roundStartReadyPlayersPercentRoundBased <modifier: percent>	Players ready needed to start round in round based modes (Rescue, Crosshair) scale factor	
vars.roundTimeLimit <modifier: percent>	Set percentage of the default time limit value	
vars.roundWarmupTimeout <time>	Set time to transition in to game round after player requirement has been met	
vars.roundsToWin <numRounds>	Set number of rounds needed to win	NR
vars.serverDescription <description>	Set server description	
vars.serverMessage <message>	Set the server welcome message	
vars.serverName <name>	Set the server name	
vars.serverTickTime	Shows the time per frame in the server (in ms)	
vars.serverType <type>	Set the server type: Official, Ranked, Unranked or Private	ROS
vars.soldierHealth <modifier: percent>	Set soldier max health scale factor	
vars.team1FactionOverride <factionId>	Set the faction for team 1	
vars.team2FactionOverride <factionId>	Set the faction for team 2	
vars.team3FactionOverride <factionId>	Set the faction for team 3	
vars.team4FactionOverride <factionId>	Set the faction for team 4	
vars.teamKillCountForKick<count>	Set number of teamkills allowed during a round	
vars.teamKillKickForBan <count>	Set number of team-kill kicks that will lead to permaban	
vars.teamKillValueDecreasePerSecond <count>	Set kill-value decrease per second	
vars.teamKillValueForKick <count>	Set max kill-value allowed for a player before he/she is kicked	
vars.teamKillValueIncrease <count>	Set kill-value increase for a teamkill	
vars.teamSwitchingAllowed <enabled>	Allow switching teams	
vars.TeamSwitchCooldown <time>	Set seconds for cooldown when switching teams	
vars.ticketBleedRate <modifier: percent>	Set the percentage of the ticket bleed rate	
vars.unlockMode <type>	Set what weapons are unlocked for players on the server	
vars.vehicleSpawnAllowed <enabled>	Set whether vehicles should spawn in-game	
vars.vehicleSpawnDelay <modifier: percent>	Set vehicle spawn delay scale factor	

ROS: Read Only after startup
ROR: Read Only in Ranked
NR: Next Round

Misc

Request: login.plainText <password: string>
Response: OK - Login successful, you are now logged in regardless of prior status
Response: InvalidPassword - Login unsuccessful, logged-in status unchanged
Response: PasswordNotSet - Login unsuccessful, logged-in status unchanged
Response: InvalidArguments
Effect: Attempt to login to game server with password <password>
Comments: If you are connecting to the admin interface over the internet, then use login.hashed instead to avoid having evildoers sniff the admin password

Request: login.hashed
Response: OK <salt: HexString> - Retrieved salt for the current connection
Response: PasswordNotSet - No password set for server, login impossible
Response: InvalidArguments
Effect: Retrieves the salt, used in the hashed password login process
Comments: This is step 1 in the 2-step hashed password process. When using this people cannot sniff your admin password.

Request: login.hashed <passwordHash: HexString>
Response: OK - Login successful, you are now logged in regardless of prior status
Response: PasswordNotSet - No password set for server, login impossible
Response: InvalidPasswordHash - Login unsuccessful, logged-in status unchanged
Response: InvalidArguments
Effect: Sends a hashed password to the server, in an attempt to log in
Comments: This is step 2 in the 2-step hashed password process. When using this people cannot sniff your admin password.

Request: serverInfo
Response: OK <serverName: string> <current playercount: integer> <effective max playercount: integer>
<current gamemode: string> <current map: string>
<roundsPlayed: integer> <roundsTotal: string> <scores: team scores> <onlineState: online state>
<ranked: boolean> <punkBuster: boolean> <hasGamePassword: boolean>
<serverUpTime: seconds> <roundTime: seconds> <gameIpAndPort: IpPortPair>
<punkBusterVersion: string> <joinQueueEnabled: boolean>
<region: string> <closestPingSite: string> <country: string> <matchMakingEnabled: boolean>
<blazePlayerCount: integer> <blazeGameState: string>
Response: InvalidArguments
Effect: Query for brief server info.
Comments: This command can be performed without being logged in.
Some of the return values will be empty or zero when the server isn't fully up and running or between levels.
Some return values are not yet implemented, and will therefore be zero.

Request: logout
Response: OK - You are now logged out regardless of prior status
Response: InvalidArguments
Effect: Logout from game server

Request: quit

Response: OK
Response: InvalidArguments
Effect: Disconnect from server

Request: version
Response: OK BFH <version> Response:
InvalidArguments
Effect: Reports game server type, and build ID
Comments: Game server type and build ID uniquely identify the server, and the protocol it is running.

Request: currentLevel
Response: OK <map: string>
Response: InvalidArguments
Effect: Reports the current map running on the game server

Request: listPlayers <players: player subset>
Response: OK <player info>
Response: InvalidArguments
Effect: Return list of all players on the server, but with zeroed out GUIDs

Admin

Request: admin.eventsEnabled <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set whether or not the server will send events to the current connection

Request: admin.help
Response: OK <all commands available on server, as separate words>
Response: InvalidArguments
Effect: Report which commands the server knows about

Request: admin.kickPlayer <soldier name: player name> <reason: string>
Response: OK - Player did exist, and got kicked
Response: InvalidArguments
Response: PlayerNotFound - Player name doesn't exist on server
Effect: Kick player <soldier name> from server
Comments: Reason text is optional. Default reason is “Kicked by administrator”.

Request: admin.killPlayer <name: player name>
Response: OK
Response: InvalidArguments
Response: InvalidPlayerName
Response: SoldierNotAlive
Effect: Kill a player without any stats effect

Request: admin.listPlayers <players: player subset>
Response: OK <player info>

Response: InvalidArguments
Effect: Return list of all players on the server; including guids

Request: admin.movePlayer <name: player name> <teamId: TeamId> <squadId: SquadId> <forceKill: boolean>
Response: OK
Response: InvalidArguments
Response: InvalidTeamId
Response: InvalidSquadId
Response: InvalidPlayerName
Response: InvalidForceKill
Response: PlayerNotDead - Player is alive and forceKill is false
Response: SetTeamFailed
Response: SetSquadFailed
Effect: Move a player to another team and/or squad
Comment: Only works if player is dead. This command will kill player if forceKill is true

Request: admin.password <password: player name>
Response: OK - for set operation
Response: OK <password> - for get operation
Response: InvalidArguments
Response: InvalidPassword - password does not conform to password format rules
Effect: Set the admin password for the server, use it with an empty string("") to reset

Request: admin.say <message: string> <players: player subset>
Response: OK
Response: InvalidArguments
Response: InvalidTeam
Response: InvalidSquad
Response: PlayerNotFound
Response: TooLongMessage
Effect: Send a chat message to players. The message must be less than 128 characters long.

Request: admin.shutDown <gracefulShutdown: boolean> <timeInSeconds: integer>
Response: OK
Response: InvalidArguments
Effect: Game server shuts down
Comment: If no value given, command will perform an immediate shutdown. If set to true, the server will shut down at the end of round or when the set time has been reached

Request: admin.yell <message: string> <duration: seconds> <players: player subset>
Response: OK
Response: InvalidArguments
Response: PlayerNotFound
Response: MessageIsTooLong
Effect: Show an obnoxious message on players' screens for the specified duration.
If duration is left out, a default of 10 seconds is assumed.
If players are left out, the message will go to all players.
The message must be less than 256 characters long.

Banning

Request: banList.add <id-type: id-type> <id: string> <timeout: timeout> <reason: string>
Response: OK
Response: InvalidArguments
Response: BanListFull
Effect: Add player to ban list for a certain amount of time
Comments: Adding a new name/IP/GUID ban will replace any previous ban for that name/IP/GUID
timeout can take three forms:
perm - permanent <default>
rounds <integer> - until the given number of rounds has passed
seconds <integer> - number of seconds until ban expires
Id-type can be any of these
name – A soldier name
ip – An IP address
guid – A player guid
Id could be either a soldier name, ip address or guid depending on id-type.
Reason is optional and defaults to "Banned by admin"; max length 80 chars.

Request: banList.clear
Response: OK
Response: InvalidArguments
Effect: Clears ban list

Request: banList.list <startOffset: integer>
Response: OK <player ban entries>
Response: InvalidArguments
Effect: Return a section of the list of banned players' name/IPs/GUIDs.
Comment: 6 words (Id-type, id, ban-type, seconds left, rounds left, and reason) are received for every ban in the list.
If no startOffset is supplied, it is assumed to be 0.
At most 100 entries will be returned by the command.
To retrieve the full list, perform several banList.list calls with increasing offset until the server returns 0 entries.
(There is an unsolved synchronization problem hidden there: if a ban expires during this process, then one other entry will be skipped during retrieval. There is no known workaround for this.)

Request: banList.load
Response: OK
Response: InvalidArguments
Response: InvalidIdType
Response: InvalidBanType
Response: InvalidTimeStamp - A time stamp could not be read
Response: IncompleteBan - Incomplete ban entry at end of file
Response: AccessError - Could not read from file
Effect: Load list of banned players/IPs/GUIDs from file
Comment: 6 lines (Id-type, id, ban-type, seconds left, rounds left, and reason) are retrieved for every ban in the list.
Entries read before getting InvalidIdType, InvalidBanType, InvalidTimeStamp and IncompleteBan

is still loaded.

Request: banList.remove <id-type: id-type> <id: string>
Response: OK
Response: InvalidArguments
Response: NotFound - Id not found in banlist; banlist unchanged
Effect: Remove name/ip/guid from banlist

Request: banList.save
Response: OK
Response: InvalidArguments
Response: AccessError - Could not save to file
Effect: Save list of banned players/IPs/GUIDs to file
Comment: 6 lines (Id-type, id, ban-type, seconds left, rounds left, and reason) are stored for every ban in the list. Every line break has windows "\r\n" characters.

FairFight

Request: fairFight.activate
Response: OK AlreadyActive
Response: OK Connecting
Effect: Attempt to activate FairFight server module if it currently is inactive

Request: fairFight.deactivate
Response: OK
Response: Denied
Effect: Attempt to deactivate FairFight server module if it is currently active

Request: fairFight.isActive
Response: OK <active: Boolean>
Effect: Query whether the FairFight server module is active

MapList

Request: mapList.add <map: string> <gamemode: string> <rounds: integer> <index: integer>
Response: OK
Response: InvalidArguments
Response: InvalidMap – incorrect map name
Response: InvalidGameModeOnMap – gamemode does not exist for that map
Response: InvalidRoundsPerMap – number of rounds must be 1 or greater
Response: Full – Map list maximum size has been reached
Response: InvalidMapIndex – Index value is out of range
Effect: Adds the map <map>, with gamemode <gamemode>, for <rounds> rounds, to the maplist. If <index> is not specified, it is appended to the end; otherwise, it is inserted before the map which is currently at position <index>.

Request: mapList.availableMaps <filter: string>
Response: OK <map name> <list of available gamemodes>
Response: OK <gamemode> <list of available maps>

Response: InvalidArguments
Effect: Return list of available maps or gamemodes
Comment: The only two strings accepted are “perMap” and “perGameMode”

Request: mapList.clear
Response: OK
Response: InvalidArguments
Effect: Clears the map list.

Request: mapList.endRound <winner: Team ID>
Response: OK
Response: InvalidArguments
Effect: End the current round, declaring <winner> as the winning team

Request: mapList.getMapIndices
Response: OK <current map index: integer> <next map index: integer>
Response: InvalidArguments
Effect: Returns the index of the map that is currently being played, and the index of the next map to run.

Request: mapList.getRounds
Response: OK <current round: integer> <total rounds to play on this map: integer>
Response: InvalidArguments
Effect: Returns the (1-based) current round number, and total number of rounds before switching map.

Request: mapList.list <startIndex>
Response: OK < map list: MapList>
Response: InvalidArguments
Effect: Returns a section of the map list.
If no startOffset is supplied, it is assumed to be 0.
At most 100 entries will be returned by the command.
To retrieve the full list, perform several mapList.list calls with increasing offset until the server returns 0 entries.
(There is an unsolved synchronization problem hidden there: if the map list is edited by another RCON client during this process, then entries may be missed during retrieval. There is no known workaround for this.)

Request: mapList.load
Response: OK
Response: InvalidArguments
Response: AccessError - File I/O error
Response: InvalidMap - Incorrect map name
Response: InvalidGameModeOnMap - Gamemode does not exist for that map
Response: InvalidRoundsPerMap - Number of rounds must be 1 or greater
Response: Full - Map list maximum size has been reached
Effect: Clears the map list and loads it from disk again.
Comments: If loading fails, the map list will be in an undefined state.

Request: mapList.remove <index: integer>
Response: OK
Response: InvalidArguments

Response: InvalidMapIndex – Index value is out of range
Effect: Removes the map at offset <index> from the maplist.

Request: mapList.restartRound
Response: OK
Response: InvalidArguments
Effect: Restarts the current round, without going through the end-of-round sequence.

Request: mapList.runNextRound
Response: OK
Response: InvalidArguments
Effect: Switches immediately to the next round, without going through the end-of-round sequence.

Request: mapList.save
Response: OK
Response: InvalidArguments
Response: AccessError – File I/O error
Effect: Saves the map list to disk.

Request: mapList.setNextMapIndex <index: integer>
Response: OK
Response: InvalidArguments
Response: InvalidMapIndex – Index value is out of range
Effect: Specifies which map to switch to once the current round completes. If there are rounds remaining on the current map, those rounds will be skipped.

Player

Request: player.idleDuration <soldier name: player name>
Response: OK <idleDuration: float>
Response: InvalidArguments
Response: InvalidPlayerName
Effect: Returns the amount of seconds that a certain player has been idle for

Request: player.isAlive <soldier name: player name>
Response: OK <alive: boolean>
Response: InvalidArguments
Response: InvalidPlayerName
Effect: Returns whether the player is alive or not

Request: player.ping <soldier name: player name>
Response: OK <soldier name: player name> <ping: integer>
Response: InvalidArguments
Response: InvalidPlayerName
Effect: Returns the player's ping

PunkBuster

Request: punkBuster.activate
Response: OK

Effect: Attempt to activate PunkBuster server module if it currently is inactive

Request: punkBuster.isActive
Response: OK <active: Boolean>
Effect: Query whether the PunkBuster server module is active

Request: punkBuster.pb_sv_command <command: string>
Response: OK - Command sent to PunkBuster server module
Response: InvalidArguments
Response: InvalidPbServerCommand - Command does not begin with "pb_sv_"
Effect: Send a raw PunkBuster command to the PunkBuster server
Comment: The entire command is to be sent as a single string. Don't split it into multiple words.

Reserved Slots

Request: reservedSlotsList.add <id: string>
Response: OK
Response: InvalidArguments
Response: InvalidName
Response: PlayerAlreadyInList
Response: Full
Effect: Add player to VIP list

Request: reservedSlotsList.aggressiveJoin
Response: OK
Response: InvalidArguments
Effect: If set to true, a non-VIP player will be kicked to give room when a VIP enters the queue.

Request: reservedSlotsList.clear
Response: OK
Response: InvalidArguments
Effect: Clears VIP list

Request: reservedSlotsList.list <startOffset: integer>
Response: OK <player entries>
Response: InvalidArguments
Effect: Return a section of the list of VIP players' names.
Comment: 1 line for each player
If no startOffset is supplied, it is assumed to be 0.
At most 100 entries will be returned by the command.
To retrieve the full list, perform several reservedSlots.list calls with increasing offset until the server returns 0 entries.

Request: reservedSlotsList.load
Response: OK
Response: PlayerAlreadyInList
Response: InvalidArguments
Response: Full
Response: InvalidName
Response: AccessError - Could not read from file
Response: IncompleteRead - Could not read the full file

Effect: Load list of VIP players from file
Comment: 1 line for each entry with player name

Request: reservedSlotsList.remove <id-type: id-type> <id: string>
Response: OK
Response: InvalidArguments
Response: PlayerNotInList
Effect: Remove player from the VIP list

Request: reservedSlostsList.save
Response: OK
Response: InvalidArguments
Response: AccessError - Could not save to file
Effect: Save list of VIP player names to file
Comment: 1 line for each player name.
Every line break has windows "\r\n" characters.

Spectator list

Request: spectatorList.add <id: string>
Response: OK
Response: InvalidArguments
Effect: Add player to spectator list

Request: spectatorList.clear
Response: OK
Response: InvalidArguments
Effect: Clear the spectator list

Request: spectatorList.list
Response: OK
Response: InvalidArguments
Effect: Return list of players in the spectator list

Request: spectatorList.load
Response: OK
Response: InvalidArguments
Effect: Loads the spectator list from file

Request: spectatorList.remove <id: string>
Response: OK
Response: InvalidArguments
Effect: Remove player from spectator list

Request: spectatorList.save
Response: OK
Response: InvalidArguments
Effect: Save spectator list to file

Squad

Request: squad.leader <teamId: integer> <squadId: integer> <soldier name: string>

Response: OK <soldier name: string> - for get operation
Response: OK - for set operation
Response: InvalidArguments
Response: InvalidTeam
Response: InvalidSquad
Response: EmptySquad
Effect: Gets or sets who's the leader of a squad.

Request: squad.listActive <teamId: integer>
Response: OK <squadCount: integer> <squadIds: integer>
Response: InvalidArguments
Response: InvalidTeam
Effect: Returns the number of active squads on a team together with the squad ids

Request: squad.listPlayers <teamId: integer> <squadId: integer>
Response: OK <playerCount: integer> <soldier names: string>
Response: InvalidArguments
Response: InvalidTeam
Response: InvalidSquad
Effect: Returns the number of players in the squad together with the player names

Request: squad.private <teamId: integer> <squadId: integer> <private: boolean>
Response: OK <private: boolean> - for get operation
Response: OK - for set operation
Response: InvalidArguments
Response: InvalidTeam
Response: InvalidSquad
Response: EmptySquad
Effect: Gets or sets whether a certain squad is private or not.

Variables

Request: vars.3dSpotting <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set if spotted targets are visible in the 3d-world
Delay: Works after map switch

Request: vars.3pCam <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set if players should be allowed to switch to third-person vehicle cameras
Delay: Unknown

Request: vars.alwaysAllowSpectators <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation

Response: CommandIsReadOnly
Effect: Set whether spectators are allowed to join without being on the spectator list
Comment: This command can only be used during startup

Request: vars.autoBalance <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set if the server should autobalance

Request: vars.bulletDamage <modifier: integer>
Response: OK - for set operation
Response: OK <modifier: integer> - for get operation
Response: InvalidArguments
Effect: Set bullet damage scale factor, in percent
Delay: Instantaneous

Request: vars.curatedUnlockList <index: integer>
Response: OK - for set operation
Response: OK <index: integter> - for get operation
Response: InvalidArguments
Effect: Set a Curated Unlock List enabled in the server
Delay: Works after map switch

Request: vars.hacker <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set if Hacker is allowed or not on the game server
Delay: Works after map switch

Request: vars.forceReloadWholeMags <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set hardcore reload on or off
Delay: Works after map switch

Request: vars.frameHistoryTime <modifier: float>
Response: OK - for set operation
Response: OK <modifier: float> - for get operation
Response: InvalidArguments
Effect: Set maximum latency to force process of frame data
Delay: Instantaneous

Request: vars.friendlyFire <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Response: LevelNotLoaded - for set operation

Effect: Set if the server should allow team damage
Delay: Works after round restart
Comment: Not available during level load.

Request: vars.gameModeCounter <modifier: integer> Response:
OK - for set operation Response:
OK < modifier: integer > - for get operation Response:
InvalidArguments
Effect: Set scale factor for number of tickets to end round, in percent
Delay: Instantaneous

Request: vars.gamePassword <password: password>
Response: OK - for set operation
Response: OK <password> - for get operation
Response: InvalidArguments
Response: InvalidPassword - password does not conform to password format rules
Response: InvalidConfig - password can't be set if ranked is enabled
Effect: Set the game password for the server, use it with an empty string("") to reset

Request: vars.hitIndicatorsEnabled <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set if hit indicators are enabled or not
Delay: Works after map switch

Request: vars.hud <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set if players hud is available
Delay: Works after round restart

Request: vars.idleBanRounds <numRounds: integer>
Response: OK - for set operation
Response: OK <rounds: integer> - for get operation
Response: InvalidArguments
Effect: Set how many rounds an idle-kick person should be banned
Set to 0 to disable ban mechanism
Delay: Instantaneous

Request: vars.idleTimeout <time: seconds>
Response: OK - for set operation
Response: OK <time: seconds> - for get operation
Response: InvalidArguments
Effect: Set how many seconds a player can be idle before he/she is kicked from server
Set to 0 to disable idle kick
Delay: Instantaneous

Request: vars.killCam <enabled: boolean>

Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set if killcam is enabled
Delay: Works after map switch

Request: vars.killFeed <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set if Kill Feed is enabled
Delay: Works after map switch

Request: vars.maxPlayers <nr of players: integer>
Response: OK - for set operation
Response: OK <nr of players: integer> - for get operation
Response: InvalidArguments
Response: InvalidNrOfPlayers - vars.maxPlayers capped to 64
Effect: Set desired maximum number of players
Comment: The effective maximum number of players is also effected by the server provider, and the game engine. If the desired maximum number of players is set to something that is accepted by the server, the effective maximum number of players will usually change within a second.
If the value is currently not accepted, then the server will continue to check every 10 seconds and change the effective count whenever the game engine allows it. It can only be changed at runtime on UNRANKED and PRIVATE servers.

Request: vars.maxSpectators <numSpectators: integer>
Response: OK - for set operation
Response: OK <numSpectators: integer>- for get operation
Response: InvalidArguments
Effect: Set desired maximum number of spectators

Request: vars.minimap <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set if minimap is enabled
Delay: Works after map switch

Request: vars.miniMapSpotting <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set if spotted targets are visible on the minimap
Delay: Works after map switch

Request: vars.mpExperience <experience: string>
Response: OK - for set operation
Response: OK <experience: string> - for get operation
Response: CommanIsReadOnly
Response: InvalidArguments
Effect: Set the MP experience of the server: HEIST, HOTWIRE, BLOODMONEY, HOSTAGE, HIT, TWS, TWL or

TDM.
Comment: This command can only be used during startup

Request: vars.nameTag <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set if nametags should be displayed
Delay: Works after map switch

Request: vars.onlySquadLeaderSpawn <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set if players can only spawn on their squad leader
Delay: Instantaneous

Request: vars.optimizeBandwidth <enabled:boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set if the server will optimize the bandwidth when High Tick rate is enabled (>30 Hz)
Delay: Instantaneous

Request: vars.outHighFrequency <tickrate: integer>
Response: OK - for set operation
Response: OK <tickrate: integer> - for get operation
Response: InvalidArguments
Effect: Set if the server will optimize the bandwidth when High Tick Rate is enabled (>30 Hz)
Delay: Next Round

Request: vars.playerManDownTime <modifier: integer>
Response: OK - for set operation
Response: OK < modifier: integer > - for get operation
Response: InvalidArguments
Effect: Set player man down time scale factor, in percent
Delay: Instantaneous

Request: vars.playerRespawnTime <modifier: integer>
Response: OK - for set operation
Response: OK < modifier: integer > - for get operation
Response: InvalidArguments
Effect: Set player respawn time scale factor, in percent
Delay: Instantaneous (Read Only in Ranked)

Request: vars.preset <serverPreset: string> <lockPresetSetting: boolean>
Response: OK - for set operation
Response: OK <serverPreset: string> <lockPresetSetting: boolean>- for get operation
Response: CommandIsReadOnly
Response: InvalidArguments

Effect: Set the server preset: NORMAL, HARDCORE, CLASSIC, or CUSTOM. If lockPresetSetting is set to true, the preset will override any settings that conflicts with it and make sure that these settings cannot be changed without setting the server to Custom
Comment: This command can only be used during startup.

Request: vars.randomStartingMap <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Controls if the starting map of the rotation is chosen randomly
Delay: Instantaneous

Request: vars.regenerateHealth <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set if players health regeneration is active
Delay: Instantaneous

Request: vars.requireReadyPlayersToStart <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set if needs ready players to start the round
Delay: Instantaneous

Request: vars.roundLockdownCountdown <time: seconds>
Response: OK <time: seconds> - for set operation
Response: OK <time: seconds> - for get operation
Response: InvalidArguments
Effect: Set the duration of pre-round
Delay: Takes effect next round
Comment: Allowed durations are between 15 and 30 seconds for ranked servers, and between 10 and 900 seconds for unranked servers. If the value gets clamped during a set operation, then the clamped value is returned as part of the response

Request: vars.roundRestartPlayerCount <numPlayers: integer>
Response: OK <numPlayers: integer> - for set operation
Response: OK < numPlayers: integer> - for get operation
Response: InvalidArguments
Effect: Set the minimum number of players for the round to restart in pre-round
Delay: Takes effect next round
Comment: If the server is ranked, and the value gets clamped during a set operation, then the clamped value is returned as part of the response

Request: vars.roundStartPlayerCount <nuPlayers: integer>
Response: OK <numPlayers: integer> - for set operation
Response: OK < numPlayers: integer> - for get operation
Response: InvalidArguments
Effect: Set the minimum number of players required to begin a round

Delay: Takes effect next round
Comment: If the server is ranked, and the value gets clamped during a set operation, then the clamped value is returned as part of the response

Request: vars.roundStartReadyPlayersPercent <modifier: percentage>
Response: OK <numPlayers: integer> - for set operation
Response: OK < numPlayers: integer> - for get operation
Response: InvalidArguments
Effect: Players ready needed to start round scale factor
Delay: Takes effect next round

Request: vars.roundStartReadyPlayersPercentRoundBased <modifier: percentage>
Response: OK <numPlayers: integer> - for set operation
Response: OK < numPlayers: integer> - for get operation
Response: InvalidArguments
Effect: Players ready needed to start round in round based modes (Rescue, Crosshair) scale factor
Delay: Takes effect next round

Request: vars.roundTimeLimit <modifier: percentage>
Response: OK - for set operation
Response OK <modifier: percentage> - for get operation
Response: InvalidArguments
Effect: Set percentage of the default time limit value
Delay: Takes effect next round
Comment: If set to 0, there will be no time limit

Request: vars.roundWarmupTimeout <time: integer>
Response: OK - for set operation
Response: OK <time: integer> - for get operation
Response: InvalidArguments
Effect: Set time to transition in to game round after player requirement has been met
Delay: Takes effect next round

Request: vars.serverDescription <description: string>
Response: OK - for set operation
Response: OK <description: string> - for get operation
Response: InvalidArguments
Effect: Sets the server description. This string is displayed on the server's detail page on Battlelog. This string must be less than 256 characters in length.

Request: vars.serverMessage <description: string>
Response: OK - for set operation
Response: OK <description: string> - for get operation
Response: InvalidArguments
Effect: Sets the server welcome message. This message will be displayed via an admin.yell to each player the first time that player deploys in on the server. The message is displayed for 5 seconds. This string must be less than 256 characters in length.

Request: vars.serverName <name: string>
Response: OK - for set operation
Response: OK <name> - for get operation

Response: InvalidArguments
Response: TooLongName - for set operation
Effect: Set server name

Request: vars.serverTickTime
Response: OK <type: string> - for get operation
Response: InvalidArguments
Effect: Shows the time per frame in the server (in ms)
Comments: Instantaneous

Request: vars.serverType <type: string>
Response: OK - for set operation
Response: OK <type: string> - for get operation
Response: CommandIsReadOnly
Response: InvalidArguments
Effect: Set the server type: Official, Ranked, Unranked or Private
Comments: This command can only be used during startup

Request: vars.soldierHealth <modifier: integer>
Response: OK - for set operation
Response: OK <modifier: integer> - for get operation
Response: InvalidArguments
Effect: Set soldier max health scale factor, in percent
Comments: Instantaneous

Request: vars.team1FactionOverride <factionId: integer>
Response: OK - for set operation
Response: OK <factionId: integer> - for get operation
Response: InvalidArguments
Effect: Set the faction of team 1
Delay: Takes effect next round

Request: vars.team2FactionOverride <factionId: integer>
Response: OK - for set operation
Response: OK <factionId: integer> - for get operation
Response: InvalidArguments
Effect: Set the faction of team 2
Delay: Takes effect next round

Request: vars.team3FactionOverride <factionId: integer>
Response: OK - for set operation
Response: OK <factionId: integer> - for get operation
Response: InvalidArguments
Effect: Set the faction of team 3
Delay: Takes effect next round

Request: vars.team4FactionOverride <factionId: integer>
Response: OK - for set operation
Response: OK <factionId: integer> - for get operation
Response: InvalidArguments

Effect: Set the faction of team 4
Delay: Takes effect next round

Request: vars.teamKillKickForBan <count: integer>
Response: OK - for set operation
Response: OK <count: integer> - for get operation
Response: InvalidArguments
Effect: Set how many teamkill-kicks will lead to a permanent ban
Set 0 to disable feature
Delay: Instantaneous

Request: vars.teamKillValueDecreasePerSecond <count: integer>
Response: OK - for set operation
Response: OK <count: integer> - for get operation
Response: InvalidArguments
Effect: Set how much every player's kill-value should decrease per second
Delay: Instantaneous

Request: vars.teamKillCountForKick <count: integer>
Response: OK - for set operation
Response: OK <count: integer> - for get operation
Response: InvalidArguments
Effect: Set number of teamkills allowed during one round, before the game kicks the player in question
Set to 0 to disable kill counting
Delay: Instantaneous

Request: vars.teamKillValueForKick <count: integer>
Response: OK - for set operation
Response: OK <count: integer> - for get operation
Response: InvalidArguments
Effect: Set the highest kill-value allowed before a player is kicked for teamkilling
Set to 0 to disable kill value mechanism
Delay: Instantaneous

Request: vars.teamKillValueIncrease <count: integer>
Response: OK - for set operation
Response: OK <count: integer> - for get operation
Response: InvalidArguments
Effect: Set the value of a teamkill (adds to the player's current kill-value)
Delay: Instantaneous

Request: vars.teamSwitchingAllowed <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set whether switching teams is allowed
Delay: Instantaneous

Request: vars.teamSwitchCooldown <count: float>
Response: OK - for set operation

Response: OK <count: float> - for get operation
Response: InvalidArguments
Effect: Set seconds for cooldown when switching teams
Delay: Instantaneous

Request: vars.ticketBleedRate <modifier: percent>
Response: OK - for set operation
Response: OK <modifier: percent> - for get operation
Response: InvalidArguments
Effect: Set the percentage of the ticket bleed rate
Delay: Takes effect next round

Request: vars.unlockMode <type: string>
Response: OK - for set operation
Response: OK <type: string> - for get operation
Response: InvalidArguments
Effect: Set what weapons are unlocked for players on the server
Delay: Takes effect next round
Comment: Allowed strings are – all, common, none, stats, list, blacklist – the last two arguments are currently not working

Request: vars.vehicleSpawnAllowed <enabled: boolean>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set whether vehicles should spawn in-game
Delay: Instantaneous

Request: vars.vehicleSpawnDelay <modifier: integer>
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set vehicle spawn delay scale factor, in percent
Delay: Instantaneous