BF3 PC Server Remote Administration Protocol

This is the remote-administration protocol used by BF3 PC Server R11.

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 BFBC2 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 behaviour

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.

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.

Team ID

An integer.

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

Squad ID
An integer.

Squad 0 is “no squad”. Depending on gamemode, there are up to 8 squads numbered 1..8.
Note that squad ID 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
round	- until end of round
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

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.

Unlock mode

Specifies which set of unlocks should be available to players. Current values are available:

all	- all unlocks for player, including purchased DLCs
common	- all unlocks for player, excluding DLCs
stats	- unlocks for player according to his stats
none	- only the base set of weapons

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.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>
Response: OK
Effect: Player with name <source soldier name> (or the server, or the server admin) has sent chat message <text> to some people
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.

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.

Summary

Command	Description
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
admin.help	Report which commands the server knows about
logout	Logout from game server
quit	Disconnect from server
version	Reports game server type, and build ID
listPlayers <players>	Return list of a group of players on the server, without GUIDs
admin.eventsEnabled <enabled>	Set whether or not the server will send events to the current connection
admin.password [password]	Set the admin password for the server
admin.say <message, players>	Send a chat message to a group of players
admin.kickPlayer <soldier name, reason>	Kick player <soldier name> from server
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.killPlayer <name>	Kill a player without scoring effects
punkBuster.isActive	Returns whether or not PunkBuster currently is active
punkBuster.activate	Attempt to activate PunkBuster if it is not currently running
punkBuster.pb_sv_command <command>	Send a raw PunkBuster command to the PunkBuster server
gameAdmin.load	Load list of game admins from file
gameAdmin.save	Save list of game admins to file
gameAdmin.add <player> <restrLevel>	Add player to list of game admins
gameAdmin.remove <player>	Remove player from list of game admins
gameAdmin.list	Return list of game admins
banList.load	Load list of banned players/IPs/GUIDs from file
banList.save	Save list of banned players/IPs/GUIDs to file
banList.add <id-type, id, timeout, reason>	Add player/IP/GUID to ban list for a certain amount of time
banList.remove <id-type, id>	Remove player/IP/GUID from ban list
banList.clear	Clears ban list
banList.list [startIndex]	Return part of the list of banned players/IPs/GUIDs
reservedSlotsList.load	Load list of reserved soldier names from file
reservedSlotsList.save	Save list of reserved soldier names to file
reservedSlotsList.add <name>	Add <name> to list of players who can use the reserved slots
reservedSlotsList.remove <name>	Remove <name> from list of players who can use the reserved slots
reservedSlotsList.clear	Clear reserved slots list
reservedSlotsList.list	Retrieve list of players who can utilize the reserved slots
mapList.load	Load list of maps from disk
mapList.save	Save list of maps to disk
mapList.add <map> <gamemode> <rounds> <offset>	Insert map at specified offset in map list
mapList.remove <index>	Remove specified map from map list
mapList.clear	Clear map list
mapList.list	Returns entire map list

mapList.setNextMapIndex <index> Set which map to switch to at end of current round
mapList.getMapIndices Get indices for current & next map
mapList.getRounds Get current round and number of rounds
mapList.endRound <teamID> End current round, declaring the specified team as winners
mapList.runNextRound Abort current round and move on to next
mapList.restartRound Restart current round
mapList.availableMaps Currently broken!
vars.ranked Change the server between ranked/unranked modevars.serverName
[name] Set the server name
vars.gamePassword [password] Set the game password for the server
vars.autoBalance [enabled] Set if the server should autobalance
vars.roundStartPlayerCount [numPlayers] Set minimum numbers of players to go from preround to in-round
vars.roundRestartPlayerCount [numPlayers] Set minimum numbers of players to go from in-round to preround
vars.serverMessage TODO: document
vars.killRotation TODO: document
vars.friendlyFire [enabled] Set if the server should allow team damage
vars.maxPlayers [nr of players] Set desired maximum number of players
vars.bannerUrl [url] TODO: document
vars.serverDescription [description] TODO: document
vars.killCam [enabled] Set if killcam is enabled
vars.miniMap [enabled] Set if minimap is enabled
vars.hud [enabled] Set if HUD is enabled
vars.crossHair [enabled] Set if crosshair for all weapons is enabled
vars.3dSpotting [enabled] Set if spotted targets are visible in the 3d-world
vars.miniMapSpotting [enabled] Set if spotted targets are visible on the minimap
vars.nameTag [enabled] Set if nametags should be displayed
vars.3pCam [enabled] Set if allowing to toggle to third person vehicle cameras
vars.regenerateHealth [enabled] Set if health regeneration should be active
vars.teamKillCountForKick [count] Set number of teamkills allowed during a round
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.teamKillValueDecreasePerSecond [count] Set kill-value decrease per second
vars.teamKillKickForBan [count] Set number of team-kill kicks that will lead to permaban
vars.idleTimeout [time] Set idle timeout
vars.idleBanRounds [enabled] Set how many rounds idle timeout should ban (if at all)
vars.vehicleSpawnAllowed [enabled] Set whether vehicles should spawn in-game
vars.vehicleSpawnDelay [modifier: percent] Set vehicle spawn delay scale factor
vars.soldierHealth [modifier: percent] Set soldier max health scale factor
vars.playerRespawnTime [modifier: percent] Set player respawn time scale factor
vars.playerManDownTime [modifier: percent] Set player man-down time scale factor
vars.bulletDamage [modifier: percent] Set bullet damage scale factor
vars.gameModeCounter [modifier: integer] Set scale factor for number of tickets to end round
vars.onlySquadLeaderSpawn [enabled] Set if players can only spawn on their squad leader
vars.unlockMode [mode] Set weapons & gadgets to be available on an unranked server

vars.roundsPerMap TODO: investigate this

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: 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 BF3 <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: listPlayers <players: player subset>
Response: OK <player info>
Response: InvalidArguments
Effect: Return list of all players on the server, but with zeroed out GUIDs

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.password [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
Effect: Set the admin password for the server, use it with an empty string("") to reset

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

PunkBuster

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

Request: punkBuster.activate
Response: OK
Effect: Attempt to activate PunkBuster server module if it currently is inactive

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.

Query

Request: serverInfo
Response: OK <serverName: string> <current playercount: integer> <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>

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.

Communication

Request: admin.say <message: string> <players: player subset>

Response: OK

Response: InvalidArguments

Response: InvalidTeam

Response: InvalidSquad

Response: InvalidPlayer

Response: TooLongMessage

Effect: Send a chat message to players. The message must be less than 128 characters long.

NOTE current it is only possible to send messages to all, teams or squads – not to individual players!

Manage players

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.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: Team ID> <squadId: Squad ID> <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.killPlayer <name: player name>
Response: OK
Response: InvalidArguments
Response: InvalidPlayerName
Response: SoldierNotAlive
Effect: Kill a player without any stats effect

Banning

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: 5 lines (Id-type, id, ban-type, time and reason) are retrieved for every ban in the list.
Entries read before getting InvalidIdType, InvalidBanType, InvalidTimeStamp and IncompleteBan is still loaded.

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: 5 lines (Id-type, id, ban-type, time and reason) are stored for every ban in the list.
Every line break has windows "\r\n" characters.

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]
- round - until end of round
- 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.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.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: The list starts with a number telling how many bans the call returns.
After that, 5 words (Id-type, id, ban-type, time 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.)

MapList

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.save
Response: OK
Response: InvalidArguments

Response: AccessError – file I/O error
Effect: Saves the map list to disk.

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

Request: mapList.list
Response: OK <entire map list: MapList>
Response: InvalidArguments
Effect: Returns the entire map list.

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.

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

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

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

Request: mapList.availableMaps <…>
Response: OK
Effect: Currently broken!

Variables

Request: vars.ranked [ranked: boolean]
Response: OK - for set operation
Response: OK <ranked: boolean> - for get operation
Response: InvalidArguments
Effect: Change the server between ranked/unranked mode
Comments: This command can only be used during startup. It can only be used to switch the server from ranked to unranked mode; the server can never switch back to ranked mode again.

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.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.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.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.maxPlayers [nr of players: integer]
Response: OK - for set operation
Response: OK <nr of players: integer> - for get operation
Response: InvalidArguments
Response: InvalidNrOfPlayers - Player limit must be in the range 8..32
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

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.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.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.crossHair [enabled: boolean]
Response: OK - for set operation
Response: OK <enabled: boolean> - for get operation
Response: InvalidArguments
Effect: Set if crosshair for all weapons is enabled
Delay: Works after map switch

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.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.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.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.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.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.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.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 permaban
Set to 0 to disable feature
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.idleBanRounds [number of rounds: 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.roundStartPlayerCount [nr of players: integer]
Response: OK [nr of players: integer] - for set operation
Response: OK < nr of players: integer > - for get operation
Response: InvalidArguments
Effect: Set the minimum number of players required to begin a round
Delay: Takes effect next round
Note: 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.roundRestartPlayerCount [nr of players: integer]
Response: OK [nr of players: integer] - for set operation
Response: OK < nr of players: 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

Note: 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.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 < modifier: integer > - for get operation
Response: InvalidArguments
Effect: Set vehicle spawn delay scale factor, in percent
Delay: Instantaneous

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
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

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.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.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.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.unlockMode [mode: Unlock mode]
Response: OK - for set operation
Response: OK <mode: Unlock mode> - for get operation
Response: InvalidArguments
Effect: Set which group of weapons/unlock should be available to players on an unranked server
Delay: Instantaneous, but if changed while players are on the server, strange things will happen