

MoH PC Server Administration

This is the server admin manual for MoH PC Server R9.

Contents

Game server operation.....	3
Files which may be accessible to the server admin	3
Remote administration interface.....	4
Commands.....	4
Events.....	4
GUI tools.....	4
Startup script.....	4
Log files	4
Admin log	4
Status log.....	4
Accounts, soldier names, and GUIDs	4
Player slots.....	5
How many players does a server support?	5
Spectators	5
Ban system.....	5
banlist.txt format	5
Communicating with other players.....	6
Sending messages from the admin interface	6
Profanity filter.....	6
Text chat moderation	6
Text chat moderation list.....	6
textchatmoderationlist.txt format.....	7
Text chat spam filter.....	7
Map handling.....	7
Overview	7
Controlling map switching.....	7
maplist.txt format	8
Autobalance	8

There is an autobalancer built into the game. Currently it doesn't work all that great. As a work-around, you can try out the TrueBalance plug-in for PProCon.	8
Team-kill kicker.....	8
Idle timeout.....	8
Ranked and unranked servers.....	8
Preround & round warmup.....	9
Reconfiguring the game modes.....	9
Normal, modified, and hardcore game modes.....	9
Normal/Modified/Hardcore, and the server browser	9
Normal/Modified/Hardcore, and Matchmake.....	10
Level-independent settings.....	10
Server region.....	11

Game server operation

When the game server first starts up, it reads a set of configuration files from disk. Some of these are managed by the RSP, and some by the server administrator.

The game server will then cycle through a series of maps. Game clients can connect to the server and play on the maps.

Control of the game server is done through a “Remote Administration” interface. This is a TCP port (kind of like a terminal interface). There are both Python scripts and GUI tools which control the game server through this mechanism.

Players can indirectly communicate with the GUI tools by sending special chat commands, which the GUI tools react upon.

The game server writes a set of log files to disk while it is running; these can be inspected by the server admin.

Files which may be accessible to the server admin

EA decides which files the RSP may make available to the admin. It is up to each RSP how to facilitate this, and the extent to which access is given. The list below contains all the files which the RSP is allowed to give the server admin full access to:

- AdminScripts/*.txt
- AdminLogs/*.log
- pb/svss/*
- pb/svlogs/*
- pb/*.cfg
- pb/pbbans.dat
- pb/pbucon.use
- pb/sv_viol.log
- pb/sv_cheat.log
- banlist.txt
- maplist.txt
- reservedSlots.txt
- reservedSpectateSlots.txt
- textchatmoderationlist.txt

In addition, the server admin may have limited access to a few lines in ServerOptions.ini.

Remote administration interface

The remote administration interface is a two-way channel for sending and receiving commands from the game server. Before the remote administration can be used, a remote admin password must be set, either via AdminScripts/startup.txt or ServerOptions.ini.

Do notice that the remote admin interface is normally case sensitive.

Commands

There's a Python script, called "CommandConsole.py", that can be used to connect to the remote administration interface. Once connected, there is an assortment of commands available that can be sent. See "MoH PC Remote Administration Protocol.pdf" for the full list.

Events

The game server can also send events when specific things happen in-game. For instance, when a player joins or leaves the server, when a round ends, or when anyone says anything through the chat. The Python script called "EventConsole.py" can be used to listen to these events.

GUI tools

There are several GUI tools constructed, which make it easier to control the game server. We'd recommend that you use them rather than relying on CommandConsole.py / EventConsole.py for everyday use.

Startup script

The game server will process the file named AdminScripts/startup.txt during bootup. Each line in that file will be executed as a remote administration command. If there are errors, they will be detailed in the status log.

Log files

Admin log

Files named AdminLogs/AdminLog*.log log every command & event that the remote administration interface performs.

Status log

Any files named AdminLogs/StatusLog*.log describe what the server is doing. If your server is acting strangely, or perhaps ignoring some of your configuration options, look in the latest log file –you might find some hints there.

Accounts, soldier names, and GUIDs

Every MoH PC player has exactly one EA account. The player can then have one or more soldier names.

The PunkBuster GUID is tied to the EA account. So is the "EA GUID".

The PB GUID is used with all PB-services, while the EA GUID is used with any non-PB-related functions in the game server.

PunkBuster GUIDs are 32-digit hexstrings.

EA GUIDs are the prefix “EA_” followed by a 32-digit hexstring.

Player slots

How many players does a server support?

By default it is set to 24 players but it is also determined by 2 factors:

- The RSP has a max-cap which they can set per server
- Finally, the admin also has a max cap that can be set (**vars.maxPlayerLimit**)

The current max number of players is the minimum of the two, however never less than 8 or more than 24.

The current effective value can be queried through **vars.currentPlayerLimit**.

Spectators

There is a bug in the game client which sometimes makes the game client hang during loading when joining in as spectator. We have already fixed this internally. The fix will be included in a client update soon. Until then, we advise you not to use spectators.

Ban system

The game server has an internal ban system. This system is independent from PunkBuster’s banlist. At startup, ban entries are read from the file named banlist.txt. During runtime, the **banList.*** commands can be used to manipulate the banlist.

Players can be banned either on their soldier name, or on their EA GUID. Banning someone on their soldier name is not particularly effective – if it’s a determined griefer then he/she will just create a new soldier and return. Banning someone on their EA GUID is much more effective.

To find out someone’s EA GUID, perform admin.serverInfo while that person is playing on your server. Or – inspect the AdminLog.

People can be banned either for a few seconds, until the end of the current round, or permanently.

The banlist can contain up to 10.000 entries.

banlist.txt format

Each entry in the banlist occupies 5 lines.

The first line specifies what the ban is on:

guid – ban on EA GUID
name – ban on soldier name
ip – ban on game client IP address

The second line specifies the GUID/name/IP that the ban applies to

The third line specifies the duration of the ban:

perm – permanent
round – until the end of the current round
seconds – until the given time is reached

The fourth line contains the timestamp for a “seconds”-type ban; otherwise it is unused.

The fifth line contains the reason for being banned. Max length 80 characters.

Communicating with other players

Sending messages from the admin interface

The **admin.say** command sends regular chat messages to players.

Profanity filter

The game server supports a profanity filter. It filters chat messages that are sent from game clients. It is on by default. You can enable/disable it through the **vars.profanityFilter** command.

Messages sent via **admin.say** does not get profanity filtered.

Text chat moderation

The game server can be set to three different chat moderation modes – free, moderated and muted. In addition, each player belongs to one of four moderation levels – muted, normal, voice, or admin. This matrix describes when a given user can chat on a given server:

	Server: Free	Server: Moderated	Server: Muted
Player: Admin	Yes	Yes	Yes
Player: Voice	Yes	Yes	No
Player: Normal	Yes	No	No
Player: Muted	No	No	No

The chat moderation mode of the server is controlled via **vars.textChatModerationMode**.

The moderation level of a player depends on the contents of the text chat moderation list (see next section). Users which are not listed are assumed to have a moderation level of normal.

Text chat moderation list

The moderation level of a player is Normal, per default.

If you want to change a player's moderation level to something else (for instance, making the server admin able to chat regardless of what's going on on the server), you do so by adding entries to the text chat moderation list.

The list is loaded from disk when the server starts up.

During runtime, the **textChatModerationList.*** commands can be used to edit the list. This is useful for muting a specific player, for instance.

textchatmoderationlist.txt format

This file contains one line per player. The line is formatted as follows:

<moderation level>:<player name> - with no spaces

The moderation level should be "admin", "voice" or "muted". No players should be listed with "normal" level in the file.

Text chat spam filter

There is a text-chat spam filter built into the game server. It is active for all players whose moderation level (see chapter "Text chat moderation") is not set to Admin.

It works like this:

If a player sends X text messages within Y seconds, that player will be temporarily muted for the next Z seconds. The three parameters are controllable through:

X – **vars.textChatSpamTriggerCount** [default: 5 messages]

Y – **vars.textChatSpamDetectionTime** [default: 10seconds]

Z – **vars.textChatSpamCoolDownTime** [default: 60seconds]

Setting vars.textChatSpamTriggerCount to 0 will disable the spam filter altogether.

Map handling

Overview

The maplist determines in which sequence the maps will be played, and how many rounds will be played on each map.

Upon startup, the maplist.txt file is read. During runtime, the **mapList.*** commands can be used to edit the set of maps.

When the same map is played for several rounds, the teams will switch sides after a run.

Controlling map switching

mapList.* can be used to edit the maplist while the server is running.

mapList.nextLevelIndex controls which position in the maplist that the server currently is at.

admin.runNextRound switchies to the next round, without finishing the current.

admin.restartRound makes all players reload the current map, and restarts the current round.

admin.endRound declares a specific team as the winning team, and then moves directly to the end-of-round screen.

maplist.txt format

Subsequent lines specify map names, and optionally the number of rounds that the map should be played. Default number of rounds is 2.

Autobalance

There is an autobalancer built into the game. Currently it doesn't work all that great. As a work-around, you can try out the TrueBalance plugin for PProCon. Team-kill kicker

When friendly fire is enabled, there is a system which can automatically kick players for excessive teamkilling. It tracks how many kills (in total) each player has performed during the current round. Also, it has a scoring system, where each kill increases the score by a set amount, and the score decreases at a constant rate.

If either the kill count or the kill score goes above set thresholds, the player is kicked for teamkilling.

You can control the settings of this functionality using **vars.teamKill***.

Idle timeout

If a player doesn't give any input within a specific period of time, he/she will be kicked due to idling. You can change the time interval / disable the idle timeout through **vars.idleTimeout**.

Ranked and unranked servers

Servers can run in two modes: ranked, and unranked.

When servers run in ranked mode, player progression (scoring etc) is reported to EA's master servers. Players will gain unlocks etc through continued play on ranked servers.

On the other hand, there are more rules on ranked servers than on unranked. Some server functions (such as setting a game password) are unavailable when running a ranked server.

Ranked servers must also run PunkBuster.

When running a ranked server, it is not possible to disable the use of pre-round.

Unranked servers do not allow people any progression or stats tracking, but give more leeway as to how the game is configured.

Preround & round warmup

MOH uses a preround system. This means that until enough players have joined the server, game mode rules aren't active and neither is scoring. When a set number of players have joined, the level will reload and start over, this time with scoring and game mode rules active. While playing, if the number of players connected to the server drop below a lower limit, preround rules will apply again (scoring and game mode rules will be deactivated). The preround limits can be modified using the admin interface.

vars.preRoundLimit lets you change the lower and upper pre round limits. To disable the use of preround, set both limits to 1, ie. "vars.preRoundLimit 1 1".

When running a ranked server, the lower limit can't be set lower than 2 and upper to 4.

When a level starts with game mode rules active, there will be a short period of warmup. During this time any player that spawns will not be able to move or fire. When a set number of players have spawned in on each team, a counter will appear and when that reaches zero, gameplay will commence. These rules can be modified using a set of commands.

admin.roundStartTimerEnabled can be used to enable/disable the warmup time.

vars.roundStartTimerPlayersLimit can be used to change the number of players that need to spawn, on each team, for gameplay to commence.

vars.roundStartTimerDelay sets the number of seconds on the warmup counter. If set to -1, the value defined on the level will be used.

Reconfiguring the game modes

This is usually done through a set of console commands.

Normal, modified, and hardcore game modes

If all settings on the game server are in their default state, the server runs in "Normal" mode. Some settings can be changed while the server remains in "Normal" mode.

Changing some settings will convert the server to "Modified" mode. See info below.

Changing the Hardcore setting will convert the server to "Hardcore" mode.

Normal/Modified/Hardcore, and the server browser

Players can see whether a server is normal / modified / hardcore in the server browser. A grey skull icon indicates a "Modified" mode. A yellow skull indicates a "Hardcore" mode.

If the player chooses to filter for hardcore OFF, only Normal servers are retrieved.

If the player chooses not to filter for hardcore, all servers are retrieved.

If the player chooses to filter for hardcore ON, only Hardcore servers are retrieved.

Normal/Modified/Hardcore, and Matchmake

Matchmake is mainly used by casual players. As such, it will only send players to Normal servers. In addition, it will attempt to send players to servers which are not full, not empty, and in the same part of the world. Plus, it will attempt to pair up players with other players of equal skill.

Level-independent settings

The following level-independent settings are available:

vars.serverName controls the name of the server, as seen in server browser

vars.adminPassword controls password used for login to remote admin interface

vars.gamePassword - if set, players must enter a password when connecting to server

vars.hardCore – when set, player health is reduced, most in-game UI is disabled, friendly fire is enabled and other things.

Setting this makes the server “Hardcore”.

vars.ranked - when set, stats are reported to EA’s master servers.

Currently, this is set via ServerOptions.ini and read-only while the server is running.

When set, vars.gamePassword is forced off and vars.punkBuster is forced on.

vars.skillLimit – When set, players with a lower or higher skill than the specified values will not be allowed into the server.

This is set via AdminScripts/startup.txt and read-only while the server is running.

vars.friendlyFire – when set, people can inflict damage on others in the same team. Setting this makes the server “Modified”.

vars.noCrossHairs – when set, guns have no crosshairs in the center of the screen. Disabling crosshairs makes the server “Modified”.

vars.noSpotting – when set, spotted targets are not marked with icons in the 3D world. Disabling spotting makes the server “Modified”.

vars.noUnlocks – when set, unlocked weapons could not be used. Disabling unlocks makes the server “Modified”.

vars.noAmmoPickups – when set, ammo pickups will not be available. Disabling ammo pickups makes the server “Modified”.

vars.realisticHealth – when set, players will not regenerate health after damage. Enabling realistic health makes the server “Modified”.

vars.supportAction – when set, support action will be available.

vars.preRoundLimit – sets the “upper” and “lower” limits to control pre rounds. If there is more players or equal than “upper”, the pre round will start. If a player left and the number of players is less or equal to the “lower” limit, the level will restart.

vars.clanTeams – when set, players with equal clan tag will be placed in the same team.

vars.serverDescription - The description needs to be less than 400 characters long; the character ‘|’ acts as line-break char.

vars.bannerUrl - The banner url needs to be max 63 characters long. The banner needs to be a 512x64 picture smaller than 127kb, in .PNG format

Server region

A server is listed under one of the seven continents in the server browser. This setting is controlled by your RSP. It should be set to the region where the server is physically located. If you are placed in the wrong continent, contact your RSP.