

# Mobile Forces Dedicated Server Setup Guide

## Abstract

The purpose of this document is to provide the reader with guidance and suggestions on how to setup a mobile forces dedicated server. It will cover basic setup of a server for both Windows and Linux, including server partitioning, recommended ini file settings, recommended network settings and creation of automatic map rotations.

This document assumes that the reader is comfortable with working from the command prompt, understands the concept of partitions, is familiar with the directory layout of Mobile Forces, is comfortable using an editor to edit the mobileforces.ini file.

## Authors

WalkerB - *Friday Night Gaming Server Administrator*

# Contents

What is a Dedicated Server ?

Basic Server Recommendations

Windows Setup

Linux Setup

UCC Server Command Line

Basic Server Settings

Tuning Network Settings

Tuning Game Settings

Dealing with Bots

Creating Automated Map Rotations

## What is a Dedicated Server ?

When you run Mobile Forces normally, it will bring up the graphical menu where you will choose your character settings, game settings and keys. You will then join a game, playing in nice 3d graphics showing you the game world around you.

This is your 'client' which shows you the first or third person perspective of what is going on around you in the game world. It's job is to draw the scene around you as quickly and accurately as possible.

Where does your client get it's information on what is happening to you relative to everyone else who is in the game ? This is where the 'server' comes in. Your client receives information from the server as to the world around you (where everyone is, whose shooting at you etc) and passes information back to the server (where you are, who you are shooting at etc).

The server's job is to process this information as quickly and accurately as possible, so that its master view of the world is up to date.

There are two types of servers - listen servers and dedicated servers.

Listen servers can be thought of as 'super clients'. Here one player will set up his machine as a listen server and all of the other players will connect to it. Note here that there is a player playing on the server itself. A listen server has to handle normal client duties for its player as well as server duties for everyone connected. This is the simplest type of server to setup and is usually ran for a quick LAN game between players.

Dedicated servers dispose of all of the client functions and work solely as a server. That is they do not use the graphical rendering engine to provide a view of the world. Instead they run a text based log which informs the server operator what is going on. Here there is no player on the server itself, instead everyone must connect to the server from another machine. This is the type of server most commonly used on the Internet as it does not require an operator to be on it all the time. It also has much less overhead (no heavy graphical rendering engine) than a listen server, which means its resources can be used much more effectively.

This document deals with the basic installation and setup of a dedicated server for Mobile Forces.

## Basic Server Recommendations

Before we get started into anything Mobile Forces or operating system specific, there are a number of basic server recommendations that should be kept in mind for any server that will be placed on the Internet.

### Partitioning

This is one of the main areas that someone who does not know much about servers will make a mistake on. When setting up a typical workstation people will go for the easiest solution and format the hard drive as one big partition containing all the free space. From a support point of view for workstations this makes sense - 1 drive, 1 partition. For a server this is the kiss of death.

For a server you should split the operating system up from the applications (in this case games) it will host. You should also create a separate partition for the pagefile. This will provide two main benefits - resilience and resistance to fragmentation. Resilience from the fact that the OS and data have been separated - if you lose one due to filesystem corruption, the other stands a much better chance of still being intact. Resistance to fragmentation from the fact that the OS and the pagefile are now in separate partitions away from the data (in this case games) meaning that the only fragmentation on the games partition will come from files written by the games themselves (generally a lot less than when combined with the OS and the pagefile). This means that your server will retain its performance for longer with less defragmentation necessary.

### Security

The Internet is a dangerous place these days (less so than some people would have you believe, but still worth keeping a healthy paranoia about). Before you connect any system to the Internet - regardless of the OS - you should make sure that it is patched to the latest level (latest service pack and or patches) and make sure that any unused services (such as the notoriously insecure web services) are disabled. Firewalling the server is worth considering, but it should be a lower priority than making sure that the server is properly patched and secured in the first place. Note well that most attacks on servers are coming from automated scripts that don't care if your server is a web server or a MF gaming server, so thinking that no one would be interested in a gaming server is the wrong way to look at it.

Also remember that security is a process and not a goal - what is secure right now might not necessarily be secure in a months time. You should on a regular basis re-evaluate the security of your server and apply any patches that have been released in the intervening time.

## Windows Setup

Installing Windows with a Mobile Forces server setup in mind is relatively simple. For the reasons outlined previously it is a good idea to partition your hard drive to separate your OS, pagefile and game data.

For dedicated server usage the I would rank the Windows operating systems in the following order of preference:

- Windows 2000 Server
- Windows NT 4 Server
- Windows XP
- Windows 98
- Windows ME

The size of the OS partition is dependant upon the version of Windows, but I would suggest using between 2 and 4 G of space. This is enough to comfortably install any version of windows in.

The size of the Pagefile partition is dependant upon the amount of memory installed in your system. Once Windows is installed on the OS partition, you will then have to create the pagefile partition and move the pagefile to it. The pagefile partition should generally be about twice the size of the physical memory that you have installed in the machine. Once you have that partition created, moving the pagefile to it varies by version. For Windows 2000/XP/NT you have to right click on My Computer, go to the Advanced (2000/XP) or Performance (NT) tab and then click on Performance Options (2000) / Settings (XP) and then finally click on Change in the Virtual Memory box. Move the paging file to your new partition setting the Initial size to be the size of your memory and the maximum size to be 1.5 times the amount of memory. Finally set the old pagefile size to 0 and 0 and then close it all down and reboot. For Windows 98/ME you have to right click on My Computer, choose properties and click on the Performance tab. Click on the Virtual Memory button and then select 'Let Me Specify My Own' and fill in the drive letter of the partition on size as for 2000/XP/NT above.

Now create your games partition big enough to hold the installs of the games that you want to host. Install Mobile Forces to a folder on this partition and then install the patch to the same folder.

Note that what is being described here is a setup of a dedicated server that will be used mainly for the purpose of hosting Mobile Forces games. To just host an occasional Mobile Forces game you can skip going through this setup and simply tune your existing Mobile Forces INI file as described in the following pages.

## Linux Setup

Installing Linux with a Mobile Forces server setup in mind is a little bit more complicated than it is for Windows. The variant of Linux that you choose to install must support the use of the WINE windows emulator as there is currently no native Linux port of Mobile Forces. This is not as big of a problem as it seems as almost all of the recent Linux distributions come with WINE installed.

The partitioning scheme for Linux is more complicated than it is for Windows, but not massively so. It still breaks down into the same categories of separating the operating system from the data. First you need a /boot partition to hold the files necessary to boot the Linux kernel, these are small and a 50M partition will more than cover what is required. The OS itself then needs a / (or root) partition, giving this between 2 and 4G of disk space will be sufficient for our needs. Creating a separate /var partition of about 1G in size is a very wise idea as this is where Linux writes it's log files etc and this is the partition which will see the most fragmentation.

Linux uses a separate partition for its swapfile by default, so create a swap partition that is roughly twice the size of your installed memory.

Finally create a partition to hold the game files (big enough to hold Mobile Forces). You cannot install Mobile Forces directly in Linux so you will have to install and patch Mobile Forces on a Windows machine and then copy the entire Mobile Forces directory to the games partition on your Linux machine.

Once complete your partition table should look something like this :-

/boot	50M	
SWAP	2x Installed Memory	
/	2 - 4G	
/var	1G	
/games		Big enough to hold all the installed games you want to host

## UCC Server Command Line

In order to run a dedicated server, it should be done from the command line. The command to do this for Mobile Forces exists in the Mobile Forces\System directory - UCC.exe. In order to make this function as a dedicated server you have to call it as shown below (from within the Mobile Forces\System directory) :-

```
UCC server mapname?game=gametype INI=serverinifile USERINI=profileini
MULTIHOME=ip
LOG=logfile PORT=portnumber
```

Note well that the above is typed all on one line and not broken over two.

Where *mapname* is the name of the map to start on (ie mf-airport), *gametype* is the type of game to run (Rage.RageDeathMatch, Rage.RageTeamGame, RageGame.RageCTF, RageGame.RageDomination, RageGame.RageDetonation, RageGame.RageSafeCracker, RageGame.TrailerGame, RageGame.Captains and Racing.mentalrace), *serverinifile* is the location of the server configuration file that you want to use (usually MobileForces.ini), *profileini* is the location of the profile configuration file that you want to use (usually profile00.ini), *ip* is the IP address that you want the server to listen on (only usually required if the machine has more than one network card), *portnumber* is the port that you want the server to listen on (default is 7777 and only necessary to be changed if you want to run multiple servers on the same machine) and *logfile* is the name of the file to log information to (defaults to UCC.log unless set).

For Linux you have to use WINE to run the UCC command, as shown below :-

```
wine --winver nt40 --debugmsg fixme-all ucc.exe server
mapname?game=gametype INI=serverinifile USERINI=profileini MULTIHOME=ip
LOG=logfile PORT=portnumber
```

Again this is all on one line and the x=y entries all have the same meanings as above. Note well that for wine to work, you must first create an empty ~/.wine directory. Note that you only have to do this the first time you run wine.

To add a mutator to the server all you have to do is modify the *mapname*?game=*gametype* section of the command line to read *mapname*?game=*gametype*?mutator=*a,b,c* where *a,b* and *c* are the mutators that you want the server to start up with.



## Basic Server Settings

Ok so the UCC SERVER command line starts the server, but how is it actually configured - where do you set its name and things like that ? There are two .ini files that exist in the MobileForces\System directory that control the behaviour of the server. The main server configuration file is MobileForces.ini - this controls things like network settings, game specific settings and server specific settings. Bots are controlled separately in the profile00.ini file. Here the skill level of bots on the server is configured.

For the basic server settings such as name and memory allocated we will look for sections within the MobileForces.ini file.

### [Engine.GameEngine] Section

This section lists what packages will be loaded on the server (must match the clients that connect) and more importantly to us the cache size that the game will allocate. The entry CacheSizeMegs=x is the entry that controls how much server memory is dedicated by the game to caching. You want to set this just high enough to be able to cache everything the game needs without wasting free memory. I would suggest CacheSizeMegs=40 as a reasonable value.

### [Engine.GameReplicationInfo] Section

This is the section where the server name, admin details and message of the day (motd) are set.

ServerName should be set to the name as you want it to appear to others, AdminName and AdminEmail should be filled in with the details of the server administrator (that's you...) and then MOTDLine1 through 4 can be filled in with a brief message to display to players as they connect and enter the server.

### [Engine.GameInfo] Section

The final of the three basic server setup sections, this is where the maximum number of players and spectators is set.

The MaxPlayers= entry is where you set the maximum number of players that will be allowed on your server. Set this conservatively at first until you know what your server can handle as Mobile Forces is a very CPU intensive server. The MaxSpectators= entry controls the number of spectators allowed on your server. Spectators take just as much CPU as a regular player does, so it is recommended to keep this low (0 or 1).

Passwording of the server is accomplished here as well. AdminPassword= should always be set and only you and trusted administrators should know it. If you want the game to be private then GamePassword= can also be set and then only players who know the password can connect to the server.

## Tuning Network Settings

One of the main things that you have to do to the MobileForces.ini file is to tune the network settings to suit the connection and hardware that you are running on. The defaults are set up to favour broadband players for a server with good hardware and lots of bandwidth. For most setups these will need to be modified to match what you want the server to do.

### [IpDrv.TcpNetDriver] Section

The three most important entries here are MaxClientRate, NetServerMaxTickRate and AllowDownloads.

MaxClientRate is how you cap the amount of bandwidth (in bytes per second) that each incoming client will be allocated. This defines how smooth the game will feel to them, and for really large games this has to be big enough to accomodate all of the replication data. A rough and ready rule for setting the MaxClientRate is to use the following formula :

$$(((ServerUpstreamBandwidth * 1000) * 0.75) / MaxPlayers) / 8)$$

Where *ServerUpstreamBandwidth* is the amount of upstream bandwidth that is available to you in kilobits per second and *MaxPlayers* is the maximum number of players that you want to support. This gives you a rate that will keep you bandwidth usage to 75% of your upstream bandwidth which should keep the game smooth. Note well that if you use high rate numbers and have a 56k player join then the whole server will become lagged, lower rate numbers seem to be less effected by this.

NetServerMaxTickRate is the rate at which the server will update the game world to the clients per second. Dropping this will reduce the CPU usage of the game, but will make use of vehicles erratic as their movement is much faster than walking and thus much more prone to prediction error. Do not drop it below 30 to avoid this problem.

AllowDownloads flags the server as to whether or not it will allow clients to download any missing files they have when they try to connect. The best example of this would be if you were hosting a custom map on your server. If this was set to true and a client connected who did not have the map then it would be downloaded automatically (at bandwidth expense!) to their machine. If AllowDownloads was set to false then the client would be notified of the missing file and denied access to the server.

## **[IpDrv.HTTPDownload] Section**

If you have AllowDownloads set to true from the previous section and have a lot of clients join who don't have the custom files then a lot of your server bandwidth will be eaten up providing those files. To provide a way around this problem you can use this section to redirect clients to a webserver to retrieve the files (hopefully on another segment of bandwidth).

RedirectToURL can be set to the URL of where to retrieve the files in question. This URL must hold the files to be downloaded in its base directory.

UseCompression can be set to true to provide much faster downloads, but the files that are placed in the redirection URL must first be compressed using the UCC Compress command. Once compressed, the package/map will have the same name as the original file with a .uz extension added to the end of the file.

## **[IpDrv.UdpServerUplink] Section**

This section allows you to control whether or not the server will announce itself to the master servers and be browsable from the Internet. If you set DoUplink to true then the server will try to connect to the MasterServerAddress listed below and announce itself. The UpdateMinutes entry controls how frequently it will send updates on its status to the master server. Note well that the server must be able to connect to port 27900 on the Master server to announce itself.

## **[IpDrv.UdpBeacon] Section**

This section allows you to control whether or not your server will announce itself on your Lan. If you set DoBeacon to true then the server will send out beacon UDP broadcasts to announce itself to your local lan. As these are broadcasts they will not be routed outside of your local area network and as such are unnecessary for an Internet Server.

## Tuning Game Settings

Now we have the basic server set up, named and the network settings correctly sized for our connection and what we want to do. Now we have to configure the settings for the games such as how long we want the rounds to be, or how many frags to win. The first thing that you have to understand is that all of the game types are sub types of Deathmatch and that settings made in the deathmatch section will filter down to the other gametypes unless explicitly reset in the section for that gametype.

### [Rage.RageDeathmatch] Section

This section is the section that controls the game settings for the Deathmatch gametype.

AirControl sets the level of control that a player will have on their direction in the air (falling or jumping) 1.00 is full control 0.35 is 35%. The bChangeLevels variable must be set to true to allow for map rotation - if it is set to false then the server will keep using the current map only.

InitialBots sets the number of bots that will be spawned into the map when a player joins. MinPlayers sets the minimum number of players required on the server. If the number of players is less than this then bots will be spawned to make the numbers up.

FragLimit sets the number of kills necessary to win the map and TimeLimit sets the time limit in minutes for the map. Note well that setting either of these two to 0 means no limit.

### [Rage.RageTeamGame] Section

This section is the section that defines the settings for the Squad Deathmatch gametype.

bBalanceTeams controls whether or not the automatic balancing of teams is turned on. When set to true, players will be moved around to balance the teams. bPlayersBalanceTeams controls whether players will be moved around to balance the teams (true) or bots will be spawned to balance the teams (false). bNoTeamChanges when set to true will stop players from changing teams during the game.

FriendlyFireScale sets as a percentage (ie 1.0 is 100 and 0.3 is 30) the amount of damage that friendly fire will do. MaxTeamSize sets the maximum size that a team can be and RespawnWait sets the time in seconds that a player will have to wait before respawning.

FragLimit sets the number of combined kills that are necessary for one team to win and TimeLimit sets the time limit for the map. Note well that setting either of these to 0 means no limit.

### **[RageGame.RageCTF] Section**

This section is the section that defines the settings for the Capture The Flag gametype. Note well that this section will inherit any settings not explicitly defined from the Squad DM and Deathmatch settings.

bNoTeamChanges, FriendlyFireScale and MaxTeamSize all have the same meanings as they do for the [Rage.RageTeamGame] section.

FragLimit sets the required number of flag captures to win and TimeLimit sets the time limit in minutes for the level. Setting either of them to 0 means no limit.

Note well that it is a good idea to add a RespawnWait=5 to this section to add a 5 second delay in respawning - this makes for a less basecamping game and encourages attacking.

### **[RageGame.RageDomination] Section**

This section is the section that defines the settings for the Holdout gametype. Note well that this section will inherit any settings not explicitly defined from the Squad DM and Deathmatch settings.

bNoTeamChanges, FriendlyFireScale and MaxTeamSize all have the same meanings as they do for the [Rage.RageTeamGame] section.

FragLimit sets the number of minutes that the control point must be held for the team to win and TimeLimit sets the time limit in minutes for the level. Setting either of them to 0 means no limit.

### **[RageGame.RageDetonation] Section**

This section is the section that defines the settings for the Detonation gametype. Note well that this section will inherit any settings not explicitly defined from the Squad DM and Deathmatch settings.

bNoTeamChanges, FriendlyFireScale and MaxTeamSize all have the same meanings as they do for the [Rage.RageTeamGame] section.

FragLimit sets the number of detonations required for the team to win and TimeLimit sets the time limit in minutes for the level. Setting either of them to 0 means no limit.

## **[RageGame.RageSafeCracker] Section**

This section is the section that defines the settings for the SafeCraker gametype. Note well that this section will inherit any settings not explicitly defined from the Squad DM and Deathmatch settings.

bNoTeamChanges, FriendlyFireScale and MaxTeamSize all have the same meanings as they do for the [Rage.RageTeamGame] section.

TimeLimit sets the timelimit for the first round of the game. The second round timelimit is defined by how long it takes the attacking team to steal the gold. If you set TimeLimit to 0 then this means that there will be no timelimit on the first round and the second round timelimit will be set to how long it took the attacking team to steal the gold.

Note well that it is a good idea to add a RespawnWait=5 to this section to add a 5 second delay in respawning - this makes for a less basecamping game and encourages attacking.

## **[RageGame.TrailerGame] Section**

This section is the section that defines the settings for the TrailerGame gametype. Note well that this section will inherit any settings not explicitly defined from the Squad DM and Deathmatch settings.

bNoTeamChanges, FriendlyFireScale and MaxTeamSize all have the same meanings as they do for the [Rage.RageTeamGame] section.

FragLimit sets the number of times that the trailer has to be delivered to the enemy base to win the game and TimeLimit sets the timelimit on the level. Setting either FragLimit or TimeLimit to 0 means no limit.

## **[Racing.mentalrace] Section**

This section is the section that defines the settings for the MentalRace gametype. You will not have or need this section if you do not have the MentalRace mod installed. Note well that this section will inherit any settings not explicitly defined from the Squad DM and Deathmatch settings.

bNoTeamChanges, FriendlyFireScale and MaxTeamSize all have the same meanings as they do for the [Rage.RageTeamGame] section.

FragLimit sets the number of laps to be completed to win the level and TimeLimit sets the timelimit on the level. Setting either FragLimit or TimeLimit to 0 means no limit.

## Dealing with Bots

The settings for dealing with bots are split across both of the configuration INI files.

The settings that control the number of bots that will be used in the game is controlled by the MobileForces.ini file and can be found in the [Rage.RageDeathMatch] section.

**InitialBots** - this sets the number of bots that will be initially spawned to make the number of players in the server up to the minimum number required by the server (see below).

**MinPlayers** - this sets the minimum number of players that the server requires; if the number of human players is below this number then the server will spawn bots to make the number up to this.

Bots can also be used to balance teams by setting the bPlayersBalanceTeams entry in the [Rage.RageTeamGame] section to false. If the bBalanceTeams entry is true then the server is spawn bots as required to balance the teams and not move players around.

The settings that actually control the bots themselves is to be found in the [Rage.RageBotInfo] section of the Profile00.ini file.

**Difficulty** - this sets the skill level of the bots (0 is retardobot, 7 is never miss)

**bAdjustSkill** - this sets whether the bots will adjust their skill level to match the opposition (true) or not (false)

**bRandomOrder** - this sets whether or not the bots should be used in a random order (if false then Coffey is always first red bot, Heywood is always first blue bot etc).

Between the two files you have complete control over the bots skills and how they are deployed. Note however that sometimes the game will get confused if a lot of people enter and leave the server and it is possible to end up with two bots on one side and no one on the other.

## Creating Automated Map Rotations

The UCC command line will start a server up on a specified map, but what do you do when that level is complete - how do you get the server to perform an automated map rotation ?

The answer is that there are a number of settings in the MobileForces.ini file that control the behaviour of automated map rotation. First of the bChangeLevels entry in the [Rage.RageDeathMatch] section must be set to true - without this the server will simply loop the level it was started on.

The actual map rotation itself is set in the [Engine.Maplist] section, which allows you to define map rotations of up to 32 maps. An example of this would be :-

```
[Engine.Maplist]
Maps[0]=mf-airport.umf
Maps[1]=mf-Carpark.umf
Maps[2]=mf-western.umf
Maps[3]=
Maps[4]=
Note well that ALL blank entries must exist
Maps[31]=
MapNum=0
```

Which would give us a three map rotation of airport, carpark and western. Note well that all gametypes except for Mental Racing support automated map rotation. The MapNum entry at the bottom is used by the server to keep track of the current map in the rotation.

Another point worth noting is that each entry in the maplist is actually processed by the ServerTravel function and as such is actually a URL. This allows you to do some advanced stuff like change the gametype or add mutators for each map. For example :-

```
[Engine.Maplist]
Maps[0]=mf-airport.umf?game=Rage.RageTeamGame
Maps[1]=mf-airport.umf?game=RageGame.RageCTF?mutator=FNG.LowGrav
Maps[2]=mf-Carpark.umf?game=RageGame.RageDetonation
Maps[3]=mf-Carpark.umf?game=Rage.RageTeamGame
Maps[4]=
Note well that ALL blank entries must exist
Maps[31]=
MapNum=0
```

Which would give us a four map rotation that consisted of Airport Squad DM style, then Airport CTF style with the FNG.Lowgrav mutator active, the Carpark detonation style and finally Carpark Squad DM style.