

# asss User's Guide – 1.6.1

Grelminar <grelminar@yahoo.com>

May 27, 2025

## 1 Introduction

asss is a new server for Subspace and Continuum. It was written from scratch by Grelminar (grelminar@yahoo.com), with help from a bunch of other people (see the Acknowledgements section). The name asss is an acronym for “a small subspace server.”

Although care has been taken to remain compatible with the original Subspace server, known as subgame, players, and especially staff and administrators, should be aware that asss is a different piece of software. It has many features that subgame is missing, but it is also missing some from subgame. The features that are common to both may work different. They will have different bugs. In short, don't expect everything to work the same as in subgame, because it won't.

### 1.1 Platform and Requirements

asss was developed primarily on a Linux system on the Intel x86 platform. Although some effort has been spent making it run on Windows also, people running it on non-Linux systems should not expect everything to work perfectly: there may be missing features and it may run slower.

The requirements for building and running asss are pretty minimal: The system should have the pthreads library (any recent Linux system should), Berkeley DB 4.0 or greater (older versions won't work) (optional), mysql (optional), Python 2.2 or newer (optional), and zlib. To compile asss from source (on either Linux or Windows), the include files for those libraries must be installed, as well as a C compiler. If you've obtained the source from CVS, you'll also need the Python interpreter in order to generate certain files. If you're using a tarball instead, it will come with those files present already.

The Makefile contains some information about which parts of it might need to be modified for your environment. You'll probably have to modify the paths that point to installed libraries.

It also contains some options that you can change to customize the build process to your environment. You can turn debugging, optimization, and profiling on and off by changing the values of the `debug`, `opt`, and `prof` Makefile variables to `yes` or `no`.

If you're missing mysql, you should comment out or change the `have_mysql` variable. That will disable building all modules that require mysql, which is currently only the alias database. If you're missing Berkeley DB, you should comment/change the line that sets `have_bdb`. This will disable the scoring modules and the `dbtool`. And if you're missing Python, you should comment/change the line that sets `have_python`, which will disable the Python module loader.

The Makefile should work without excessive modification on Linux, FreeBSD, Cygwin, and Mingw32.

Currently, only 32-bit Intel platforms are supported because of byte-order issues. Eventually, asss will be able to run on other architectures, but for now, Intel will have to do.

## 2 File Layout

The server always access files relative to the directory it was started from, and it expects to have certain files and directories in certain places. That means that to run multiple copies of

the server on one machine, you should make sure that each one is started from its own home directory.

Here's what a typical zone's file layout should look like:

```
/home/myzone
+ news.txt
+ scrty
+ scrty1
+ bin
| + asss
| + dbtool
| + backtrace
| + scoring.so
| + security.so
| + fg_wz.py
| + fg_turf.py
| + ...
|
+ arenas
| + (default)
| | + arena.conf
| |
| + (public)
| | + arena.conf
| |
| + duel
| | + arena.conf
| |
| + pb
|   + arena.conf
|   + balls.conf
|   + pb.lvl
|
+ conf
| + global.conf
| + modules.conf
| + groupdef.conf
| + groupdef.dir
| | + default
| | + mod
| | + smod
| | + sysop
| |
| + svb
|   + svb.conf
|   + prizeweights
|   + misc
|   + ship-warbird
|   + ...
|
+ log
| + asss.log
| + asss.log.1
|
+ maps
| + zone1-pub.lvl
```

```
| + another.lv1
|
+ data
  + data.db
```

The most important directory is **bin**. This directory should contain the main ass binary, as well as all files containing modules to be loaded by the main binary.

**conf** contains config files that affect the server as a whole. Among the important files are **modules.conf**, which specifies the list of modules to load at startup, **global.conf**, which contains config settings for the whole server, **groupdef.conf**, which describes which capabilities belong to each group, and **staff.conf**, which assigns groups to various players. **groupdef.conf** uses files in the **groupdef.dir** subdirectory to ensure more powerful groups have all the capabilities of lesser ones.

**conf** can also contain partial config files for arenas to include. The default directory structure contains an **svs** directory, with the Standard VIE Settings, split into multiple files, by ship and function.

**log** will be used by the server to deposit any log files that it creates.

**data** is used to keep the database holding all persistent information, including scores. Information for all arenas is kept in the same database file.

**maps** is an optional directory that the server will search for **.lv1** files in. These files can also be located in arena directories, so this isn't a required directory. It might simplify administration, though, to keep all map files in this directory.

Each arena gets its own subdirectory in the **arenas** directory that holds config files, maps, and other data. Two subdirectories are special: **(public)** is used for all public directories, and **(default)** is used for all arenas for which a directory doesn't exist. Note that it's ok for **(public)** to not exist, in which case public arenas will use the configuration from **(default)**.

Each arena directory must contain a file named **arena.conf**, which contains the settings for that arena. For ease of administration, this file may **#include** other config files in either the same directory, or the global **conf** directory.

The file **news.txt** should be located in the base of the zone directory as well, unless another location is specified in **global.conf**.

## 2.1 Running ass

### 2.1.1 Command line arguments

There are currently three things you can give ass on the command line:

- A directory name on the command line will be interpreted as the name of a directory containing the zone files (as described in the previous section). If no directory is specified, the current directory will be used.
- The optional switch **--daemonize** (abbreviated **-d**) tells it to fork into the background before starting up. You might want to use this when running ass from a startup script.
- Another switch, **--chroot** (abbreviated **-c**), tells it to attempt to chroot to the zone directory before starting up. See the next section for more information on this.

### 2.1.2 Running chrooted

If you want to increase the security of your host, you can run ass chrooted. This means that it will run with its root directory set to the zone directory, and it won't be able to access any files outside of that directory.

You need to do a bit of preparatory work before chroot can work. You'll need to make a **lib** directory in the zone directory containing all the shared libraries needed by any modules you'll be loading. On my machine, I needed to put the following files from **/lib** and **/usr/lib** in there: **ld-linux.so.2**, **libc.so.6**, **libpthread.so.0**, **libz.so.1**, **libm.so.6**, and **libdb-4.0.so**.

You'll also have to make sure that nothing within the zone directory is a symlink pointing outside of the zone directory. So you'll need a separate copy of the `bin` directory and shared settings files for each separate zone. It's also a good idea (although not strictly necessary) to create an `etc` directory with limited `passwd` and `group` files, and also things like `ld.so.conf`, `hosts`, and `nsswitch.conf`.

If you're using Python modules, you'll also need a copy of the Python standard library in your chroot environment. It's usually found in `/usr/lib/python2.3`. Note that you can hard-link the actual files to avoid wasting space.

In order to do a chroot, `asss` needs to be run as root. It won't continue running as root, of course: as soon as it successfully chroots, it drops its privileges and runs as a normal user. The user it runs as depends how it was run: if the `asss` binary is installed `setuid-root`, it will always drop to the user who invoked the binary. If it's actually run by the root user, it will use the contents of the `USER` environment variable to control which user to drop to. So to run it as user "nobody" from a script running as root (like `rc.local`), you can run something like `env USER=nobody /path/to/asss /zone/dir --chroot --daemonize`.

### 3 Modules

Almost all of the functionality of `asss` is split into many small modules. Currently modules can be written in either C or Python. Most core modules are written in C, since the Python module support is still somewhat experimental.

C modules are in separate libraries with the extension `.so` (on Unix) or `.dll` (on Windows). One shared library can contain any number of modules.

There are currently 79 modules that are part of `asss`, but each zone might have some custom-developed modules for their zone as well.

When the server starts up, it loads all of the modules listed in the file `modules.conf`. Once it's running, more modules can be loaded with the `?insmod` command, and modules can be unloaded with `?rmmod`. The current list of loaded modules can be examined with `?lsmmod`.

The `modules.conf` file has a special format that's slightly different from the rest of the config files. It has no sections. Each line should contain a "module specifier." A module specifier is something of the form `filename:module` for C modules, or `<py> filename` for Python modules. The filename part should be the name of the file containing the module, without the extension (`.so` or `.dll` or `.py`). The module part should be a module name that's contained in the file. The colon separating them is just a colon. Comments in the `modules.conf` file are indicated by an initial semicolon or pound sign.

If a particular zone has no need for a particular module (e.g., Chaos Zone doesn't have any flags or balls, so it doesn't need those modules), it shouldn't load those modules. Only loading the modules that are actually used for a zone will decrease the memory usage of the server and may make it run faster.

Once a module is loaded into the server, it has full access to the server's data, including player IP addresses, machine id's, scores, and passwords. It can also access files on the machine it is running on, and make network connections, and it can easily crash or deadlock the server. Python is a safe language, so a module written in Python can't crash the server. It can still deadlock it, though, and still has arbitrary access to the system. Thus, **admins and sysops should be careful to only load modules from sources that they trust.**

### 4 Capabilities

The old Subspace server supported a very limited notion of authority: There were moderators, super moderators, and sysops. Each level allowed access to more and more commands. Additionally, moderators and above could see private freqs and private arenas, and bypass freq and arena size limits.

`asss` is much more flexible. It lets sysops and admins assign any set of powers to any group of people. In the `asss` model, each of the above powers, plus a few more, like energy viewing, is

assigned a capability name. Each command also gets a capability name (actually, each command gets two, one for using the command with public messages, and one for using it with private messages). Whenever the server needs to determine if a player can take a certain action, it asks the capability manager, which replies either yes or no.

The capability manager loads the file `conf/groupdef.conf`, which uses the files in `conf/groupdef.dif`, to determine which groups have which capabilities.

The server comes with one capability manager, contained in the `capman` module, but there's no reason why another one couldn't be used if your zone has peculiar needs for assigning people powers.

## 4.1 Capability names

The most common capability names are for commands. If a player tries to run a command, say, `?lastlog`, the server would query the capability manager with the name `cmd_lastlog`. If a player uses a command as a private message, as in `:annoying_player:?freqkick`, the capability name used would be `privcmd_freqkick`.

There are several non-command capabilities that are currently used in the server:

- `seeprivarena` controls whether private arena names are sent to a player for the `?arena` command.
- `seeprivfreq` determines if a player sees private freqs in the freq listing.
- `findinprivs` is needed by a player running `?find` for the server to report the names of private arenas. (Not implemented yet.)
- `seepd` allows players to see other ship's energy and specials from spectator mode. ("epd" stands for extra position data.)
- `seesysoplogall` allows a player to see all important log messages in the zone.
- `seesysoplogarena` only allows a player to see only important log messages having to do with the arena he is currently in.
- `seemodchat` allows players to see the moderator chat.
- `sendmodchat` controls who can send moderator chat messages. Usually, these two capabilities would be given to the same people.
- `uploadfile` allows a player to upload files. Note that the player must also have the `cmd_putfile` to upload a file using that command.
- `bypasslock` allows players to switch ships even though the arena or themselves have been locked into a ship or into spectator mode by a staff member.
- `bypasssecurity` lets players use unauthorized clients, or prevents kicking off for security checksum failures.
- `invisiblespectator` makes players not show up on the list given when the person they are spectating uses the `?spec` command.
- `unlimitedchat` allows a player (e.g., a bot) to bypass chat flooding checks.
- `changesettings` lets clients use the settings change packet (required for `?quickfix/?getsettings`).
- `isstaff` makes players show up in `?listmod` output.
- `seeallstaff` allows a player to see all non-default-group players, even if they lack `isstaff`.

## 4.2 The default capability manager

The default capability manager works with groups. Each group has a set of capabilities, and players are assigned to groups. To check if a player has a certain capability, the capability manager simply checks if the group he's in has that capability.

To determine which groups have which capabilities, the `groupdef.conf` file is used. It should have a section for each group, and a line within that section for each capability.

To determine which players belong to which groups, the `staff.conf` file is used. Each section in the file corresponds to an arena<sup>1</sup>, except for the special section (`global`), which applies to and overrides all other arena settings. Keys are player names, and values are groups. So a setting like “`Grelminar=sysop`” in the (`global`) section would give Grelminar sysop powers in all arenas, while a setting “`ZippyDan=smod`” in the `pb` section would give ZippyDan smod powers in arenas `pb`, `pb1`, `pb2`, etc.

The command `?setgroup` can be used to control group assignment without editing the `staff.conf` file manually.

The default capability manager also supports passwords for groups, although using this feature is strongly discouraged. It is intended for sysops or other staff members to gain privileged access when the zone isn't connected to a billing server to provide authentication.<sup>2</sup> To use it, add keys to the `GroupPasswords` section, of the form “`group = password`”.

### 4.2.1 Emulating the old system

Using the default manager, it's relatively easy to set up ass to emulate the old server's moderator, super moderator, and sysop model: The `groupdef.conf` file looks like this:

```
; conf/groupdef.conf

[default]
#include groupdef.dir/default

[mod]
#include groupdef.dir/default
#include groupdef.dir/mod

[smod]
#include groupdef.dir/default
#include groupdef.dir/mod
#include groupdef.dir/smod

[sysop]
#include groupdef.dir/default
#include groupdef.dir/mod
#include groupdef.dir/smod
#include groupdef.dir/sysop
```

The files in `groupdef.dir` contain simply lists of capabilities. Each group includes the file for itself, as well as the files for the lesser powerful groups. The way `groupdef.conf` includes files means that smods will have all the capabilities of mods, plus more, sysops will have more than smods, etc.

---

<sup>1</sup>Actually an arena group name; see the section on arena groups.

<sup>2</sup>But there's a better way to do this: if you load the `auth_file` module before `billing`, the server will fall back to using `auth_file` when the billing server is not connected. Staff members can set passwords using the `?passwd` command (specific to `auth_file`), and they will have access to their usual group.

## 5 Logging

asss has extensive logging capabilities. Any remotely interesting event in the game will generate a log message, which will be passed to any number of loaded logging handlers.

### 5.1 Levels

There are five importance levels defined for log messages: `DRIVEL` is unimportant information that you probably don't want to see, but is logged anyway, just in case. `INFO` is basic information about common, unexceptional events. `MALICIOUS` is for exceptional conditions that are caused by players sending bad data to the server. These might be indications of cheating or other illicit activity. They also might be caused by abnormal network conditions. `WARN` is for error conditions that can be worked around, or aren't too catastrophic. `ERROR` is for really really horrible error conditions. These usually indicate misconfigured servers or bugs in the server itself.

### 5.2 What is logged?

There are currently 381 distinct log messages in the server. By type, there are 37 `ERROR` messages, 111 `WARN` messages, 81 `MALICIOUS` messages, 65 `INFO` messages, and 87 `DRIVEL` messages.

### 5.3 Filtering

Log handlers support a common method of filtering that give you lots of control over which handlers see which messages.

By default, all messages are seen by all handlers. To limit messages to a handler `log_foo`, create a section with the same name as the handler in `global.conf`. The keys in that section will be module names, and the values will be a set of priority levels to allow, specified by listing the first letters of the allowed levels. The special key `all` will be used for modules not listed. For example:

```
; this keeps flag positions and ball fires from appearing in the log
; file, but allows other DRIVEL messages.
[log_file]
all = DIMWE
flags = IMWE
balls = IMWE

; this allows all messages to go to the console except those from
; cmdman.
[log_console]
all = DIMWE
cmdman = none

; this lets only important messages (malicious and error) go to sysops
[log_sysop]
all = ME
```

### 5.4 Commands

In general, all commands run by anyone are logged, at level `INFO`, along with their parameters and targets. Some commands, however, contain personal or sensitive information that might be abused by zone staff who can view logs. To prevent this abuse, there is a hardcoded list of commands whose parameters don't get logged (they get replaced by `...` in the log messages).

## 5.5 Handlers

The current log handlers are:

- **log\_console** simply writes all log messages to standard out, which is usually the terminal that ass's is started from. Usually, ass's will run detached from any terminal, so this is primarily intended for debugging.
- **log\_file** write all log messages to a file. The name of the file is controlled by the `Log:LogFile` configuration option. The command `?admlogfile` may be used to flush or reopen the log file while the server is running. ass's always appends to a single file. If log rotation is desired, it should be accomplished with an external program such as `logrotate`.
- **log\_sysop** informs players of log events within the game. "Important" messages, as defined by the logging filter, are sent to players with the capabilities `seesysoplogall` and `seesysoplogarena`. Players with the latter capability only see log messages that originated in the arena. This logging module also implements the `?lastlog` command.
- **log\_staff** broadcasts log lines generated by a specific set of commands to all online staff. The intention is to let other staff members know when one of them uses certain important commands. The default set of commands is `?warn`, `?kick`, and `?setcm`, although the list is configurable with the `log_staff:commands` setting in `global.conf`.

## 6 New Features

### 6.1 Arena groups

To make the process of creating multiple arenas with identical settings easier, ass's supports arena groups. If an arena name ending with a number is requested, the configuration and other data for that arena will be taken from the directory named by that arena without the number at the end. So arenas `smallpb1`, `smallpb2`, `smallpb3`, etc. will all be identical in configuration to `smallpb`, which uses data in the directory `arenas/smallpb`.

Persistent data (e.g., scores) are also partially shared between arenas in the same group. Data in the "forever" and "per-reset" intervals will be shared, but data in the "per-game" interval will be kept separate between different arenas in the group.

The group name of an arena (the name without the number at the end) is also used for determining staff groups.

### 6.2 Freq Management

#### Requires module: `freqowners`

If the arena controller allows it, private freqs can now be owned. The first player to move to a particular private freq becomes an owner for that freq. An owner can kick non-owners off of his freq by sending them the command `?freqkick`. An owner can share owner privileges to other players by sending them the command `?giveowner`. The spec freq can't be owned.

The config variable `Team:AllowFreqOwners` controls whether to enable freq ownership. It defaults to on.

#### Requires module: `fm_password`

The `fm_password` module implements password-protected freqs. It's a freq manager module meant to sit on top of another freq manager (like `fm_normal`). The `?freqpwd` can be used by anyone on a private freq to set a password. To join a freq with a password, players must use the `?joinpwd` command before attempting to switch to the protected freq.



## 6.3 Arena limiting

**Requires module:** `arenaperm`

Any arena can specify a `General:NeedCap` value in its config file. If present, players will not be allowed to enter the arena unless they have the specified capability.

## 6.4 Moderator chat

asss includes an actual moderator chat system, which should be an improvement over the `?cheater`-based systems in use currently.

Mod chat messages begin with a backslash (`\`), and are displayed in dark red (the same color as sysop warning messages). Who is allowed to send and receive mod chat is controlled by two capabilities: `seemodchat` and `sendmodchat`, which allow players to see and send mod chat.

## 6.5 Multiple commands

You can specify multiple commands on one line by putting a vertical bar (`|`) directly after the command character (`? or *`), and the separating commands with more bars. Do not put any spaces or other characters between the bars and the start of the command. Multiple private commands are supported, but you can't mix public and private commands on the same line. There's a hard limit of five commands on one line. Example: `?|flagreset|shipreset|prize warp|aa go!`.

## 6.6 Built-in alias database

**Requires module:** `mysql`, `aliasdb`

asss includes a hastily-written alias database. The alias database depends on `mysql` support, although it's written so that it should be easy to port to another relational database if necessary.

All logins are automatically entered if the `aliasdb` module is loaded. There are several ways to query the database: `?alias` lets you do general-purpose queries, `?qip` allows you to query by IP address range. `?rawquery` allows you to make custom queries with most SQL commands. You can find the documentation for these commands in the Commands section.

The `?last` command uses the alias database to find the last 10 people to log in.

## 6.7 Authentication

Ok, so this isn't new, but it's greatly expanded in functionality: authentication can now be done with things other than billing servers, and some authentication modules can be "stacked."

For example, one useful auth module is `auth_file`, which uses a file of hashed passwords to authenticate users. This module is intended for use by private servers who want to allow a small group of people (say, a squad) to play together, and not allow anyone else in. It can also be used as a fallback module by the `billing` module (which acts as an auth module, among other things). This means if the billing server is connected, login requests will be authenticated against the billing server, but if it isn't, they get passed to `auth_file`.

If the user is listed in the file and supplies a correct password, he will be allowed access and be granted groups. If not, he will be either accepted or rejected depending on the value of `General:AllowUnknown` setting in `passwd.conf`. If an unknown player is allowed, he will *not* be assigned groups based on name. (That will also not happen if no auth modules are loaded.)

The `auth_file` module also allows you to lock a specific player name out of a zone.

**Note:** In the default configuration, `auth_file` is insecure, since anyone can log in as a player that has no password set, and then set one. See the documentation for `General:RequireAuthenticationToSetPassword` and also read the comments in the default `passwd.conf`.

To use a fallback module for the `billing` module, simply make sure that that module is loaded before `billing` is loaded.

Two more authentication modules are intended to be layered on top of the basic ones: `auth_prefix` lets only staff members log in with certain punctuation characters as prefixes to their real account names (controlled by the `prefix_x` capabilities), and `auth_ban` implements

simple banning by machine-id, as an authentication layer. It provides the `?kick`, `?listmidbans`, and `?delmidban` commands to control the ban list.

## 6.8 Multiple “public” arenas

asss supports a general player placement interface to decide which arena a player should be placed in upon entering the zone. The most useful arena placing interface is `ap_multipub`, which has the effect of creating multiple “public” arenas.

To use `ap_multipub`, simply make sure it’s loaded from `modules.conf` (somewhere near the end is good). It is controlled by two settings in the global config file: `General:PublicArenas` is a whitespace-separated list of public arena *types* (not names). For example, if `General:PublicArenas` is set to `“pb turf wz,”` the server will start placing people in the arena named `pb1`, then when that gets full, it will move to `turf1`, then `wz1`, `pb2`, etc. To control how many people it will put in each arena, use `General:DesiredPlaying`, which is a count of *playing* players (i.e., not spectators).

# 7 Lag Control

## 7.1 Lag Measurement

Lag, which includes both latency and packetloss, is difficult to measure accurately and control. asss does as well as it can with limited information.

There are several ways that the server collects latency information: Position packets sent from the client contain timestamps that the server can compare to its own current time to determine approximately how long the packet took to get there. This is complicated by the fact that the times on the server and client aren’t always perfectly synchronized. Reliable packets need to be acknowledged, and the round-trip time between the sending of a reliable packet and the receipt of its acknowledgement can be measured. That will be equal to approximately twice the one-way latency, but that isn’t exact either because the two trips might take different amounts of time. Finally, the client can measure latency using the same techniques, and periodically send its results to the server for processing.

Packetloss is slightly easier: the client and server can keep track of how many packets each has sent and recieved, and compare numbers periodically. Reliable packets also provide opportunities to measure packetloss: if a reliable packet isn’t acknowledged within the timeout, the server knows either the original packet or the acknowledgement got lost. If a reliable packet is recieved twice, the server knows the acknowledgement got lost. Again, the client can also measure these numbers and send the results to the server.

## 7.2 Settings and Actions

There is one global setting for lag, `Lag:CheckInterval` which controls how often each player’s lag numbers are checked to perform actions. It’s specified in ticks. Each arena can specify its own lag limits. All of the parameters described below go in the `Lag` section in the arena’s configuration file (or a file included from it).

There are four main values that lag actions are based on: average ping (determined by an exponential averaging scheme, based on S2C, C2S, and reliable pings), S2C packet loss, S2C weapons packet loss, and C2S packet loss. Each value has four thresholds associated with it: one controls when a player gets forced into spectator mode, one controls when a player is allowed to pick up flags and balls, and two control weapons ignoring. The units of the settings concerning latency are milliseconds, and the units of the settings concerning packetloss are tenths of a percent (i.e., fractions out of 1000).

Forcing into spec is easy enough: if the value is over the threshold when a player is examined, he’s forced into spec. Disabling flags and balls also works on a simple threshold: if the value is above it, the player won’t be allowed to pick up any flags or balls. If he’s currently carrying a flag or ball, and one of the values moves over the limit, he’ll get to keep it.

Weapon ignoring is slightly more complicated: There are two thresholds, one to start ignoring weapons, and one where all weapons will be ignored. If all of the values are below their respective starting thresholds, none of the player's weapons will be ignored. If one of them is higher, a percent of incoming weapons from that player to be ignored is calculated by interpolation between the starting threshold (0%) and the higher threshold (100%). If multiple values are above their starting threshold, the percent of weapons that gets ignored is the maximum of the percent ignored from each value. C2S packetloss doesn't cause weapon ignoring, since C2S packetloss generally gives the player a disadvantage, not an advantage.

The names of these settings are: `PingToSpec`, `PingToStartIgnoringWeapons`, `PingToIgnoreAllWeapons`, `PingToDisallowFlags`, `S2CLossToSpec`, `S2CLossToStartIgnoringWeapons`, `S2CLossToIgnoreAllWeapons`, `S2CLossToDisallowFlags`, `WeaponLossToSpec`, `WeaponLossToStartIgnoringWeapons`, `WeaponLossToIgnoreAllWeapons`, `WeaponLossToDisallowFlags`, `C2SLossToSpec`, and `C2SLossToDisallowFlags`. Their functions should be clear from their names and the above description.

One final setting `SpikeToSpec`, determines the length of time that the server can receive no packets from a player before forcing him into spectator mode.

### 7.3 Bandwidth Throttling

asss supports bandwidth throttling for players on slower connections. To make the game fairer, packets are prioritized depending on their function. For example, weapons packets will be preferred over chat messages when deciding how to use up the last few bytes of allotted bandwidth. The server will also reserve a certain percentage of the total bandwidth for packets of certain priorities. Techniques similar to those used in modern TCP implementations are used to dynamically adjust the bandwidth limit to players based on their connection quality.

## 8 Regions and Extended LVL Files

asss supports an extension to the classic map file format. The extension format is mostly backwards-compatible, so extended lvl files should work with any program that supports classic lvl files.

Extended lvl files can contain various types of additional data. The simplest is a list of text attributes about the map, such as a name, a version, the map and tileset creators, and the programs used to create it. This data can be viewed with the `?mapinfo` command.

The other important data that extended lvl files can contain, that ass uses for new functionality, is regions.

### 8.1 What are regions?

A "region" is a named set of tiles, with optional associated attributes. The tiles in a region can be an arbitrary set, and don't have to be in a certain shape or be connected (although things are more efficient if they are). A region has a name, which can be used to refer to it from a module (the details of how to do this are in the developer's guide, or at least will be eventually). Regions can also have several attributes which are interpreted by ass itself.

### 8.2 Region attributes

Currently supported region attribute are: `no-antiwarp`, `no-weapons`, `no-flags`, and `autowarp`. If a region has the `no-antiwarp` attribute, the server will clear the antiwarp bit of any players in that region, effectively disabling their antiwarp (although it will appear to them that their antiwarp is still on; you should consider this when designing a zone using this feature). The `no-weapons` attribute means that the server will ignore weapons from players in the region, although again, the players themselves will still see their own weapons on their screen. The `no-flags` attribute prevents flags from being dropped in that region. The `autowarp` attribute (which requires the `autowarp` module to function) will automatically move a player to a different location, optionally in a different arena, when the region is entered.

### 8.3 Making extended lvl files

lvltool, which is available on the assb web site, is a simple command-line tool to manipulate extended lvl files. It's more of a proof-of-concept than a useful tool, although if you put enough effort in, you can use it to create arbitrary extended lvl files.

The "Continuum Level / Ini Tool," version 1-1-05 or later, is a graphical map editor, written in Java, that also supports creating extended lvl files.

## 9 Virtual Servers

assb allows one server process to appear to clients as several different servers. The primary advantage of this feature is that players connecting to all virtual servers are treated the same internally and can move between arenas and communicate as if they connected to the same server.

To create virtual servers, you have to tell the `net` module to listen on more than one port. You do this by creating additional sections in `global.conf` named "Listen1," "Listen2," etc. Each setting must specify a port, and can also optionally specify a virtual server identifier, and a specific IP address to bind to.

Virtual server identifiers are used in several ways: if you are using an arena placing module that supports them (e.g., `ap_multipub`), the server id will be used as the arena basename to place players who connect through that port in.

The `directory` module also supports virtual servers: it will create one directory entry for each virtual server. The server name and description can be different for each virtual server. To specify them, create "Name" and "Description" settings in the section "Directory-*servername*" for each virtual server identifier. If either of those settings is missing from that section, it will fall back to their values in the "Directory" section.

Finally, an example to make this all clear:

```
;; global.conf

;; listen on 3 different ports:

; players connecting to port 2000 will be send to a random arena.
[Listen1]
Port = 2000

; players who connect to 5000 will be sent to pb1, pb2, etc.
[Listen2]
Port = 5000
ConnectAs = pb

; port 7500 will send them to aswz by default, and so on.
[Listen3]
Port = 7500
ConnectAs = aswz

; this will force the server to listen on an internal interface only
; and send those players to a secret arena:
[Listen4]
BindAddress = 192.168.0.23
Port = 3300
ConnectAs = #secret

[Directory]
;; point to the directory servers you want to be listed on. using
```

```
;; default port and password.
Server1 = sscentral.one.com
Server2 = sscentral.two.com

;; now describe what this server is called by default:
Name = A Testing Zone
Description = Testing happens here.

[Directory-pb]
;; specify the name and description for pb:
Name = PowerBall
Description = Play with balls!

[Directory-aswz]
;; specify only name for aswz:
Name = A Small Warzone
```

## 10 Using dbtool

FIXME!

## 11 Command Reference

These are all of the commands that the server currently recognizes. Not all of them will always be available. If a command requires a module that's not one of the core modules, that will be indicated above its description. Most other commands require the `playercmd` module.

Possible targets are listed for each command. The targets can be “none,” which refers to commands typed as public (arena) messages, “player,” for commands that can target specific players, “freq,” for commands that can target a whole freq at a time (with either ' or " ), or some restriction of one of those.

Each command also describes any required or optional arguments.

Note that the section doesn't list who is allowed to run a particular command, because that is determined by the capability manager, which can be fully customized for each particular server.

### a

**Possible targets:** player, freq, or arena

**Arguments:** <text>

Displays the text as an arena (green) message to the targets.

### aa

**Possible targets:** player, freq, or arena

**Arguments:** <text>

Displays the text as an anonymous arena (green) message to the targets.

### acceptfile

**Requires module:** sendfile

**Possible targets:** none

**Arguments:** none

Accept a file that has been offered to you.

## actives

**Possible targets:** arena

Syntax: ?specactives Lists everyone who is Available and not Idle. See Also: ?idles ?specactives

## addallowed

**Requires module:** auth\_file

**Possible targets:** none

**Arguments:** <player name>

Adds a player to passwd.conf with no set password. This will allow them to log in when AllowUnknown is set to false, and has no use otherwise.

## addcredits

**Possible targets:** Player

**Arguments:** none

Private message a player ?addcredits <amount> in order To give extra credits to that player.

## admlogfile

**Possible targets:** none

**Arguments:** flush or reopen

Administers the log file that the server keeps. There are two possible subcommands: **flush** flushes the log file to disk (in preparation for copying it, for example), and **reopen** tells the server to close and re-open the log file (to rotate the log while the server is running).

## alias

**Requires module:** aliasdb

**Possible targets:** player or none

**Arguments:** [<name>]

Queries the alias database for players matching from the name, ip, or macid of the target. Only works on MySQL 4 or later.

## apfo

**Requires module:** <py> hockey

**Arguments:** 0 or 1, for the team. Example '?addpt 1'

This command will force add a teams point.

## apfz

**Requires module:** <py> hockey

This command will force add a teams point.

## arena

**Possible targets:** none

**Arguments:** [-a] [-t]

Lists the available arenas. Specifying -a will also include empty arenas that the server knows about. The -t switch forces the output to be in text even for regular clients (useful when using the Continuum chat window).

## attmod

**Possible targets:** none

**Arguments:** [-d] <module name>

Attaches the specified module to this arena. Or with -d, detaches the module from the arena.

## available

**Possible targets:** self

Syntax: ?available Command Aliases: ?av Marks you as Available. Certain games require you to be Available to be picked.

## az

**Possible targets:** none

**Arguments:** <text>

Displays the text as an anonymous arena (green) message to the whole zone.

## ballcount

**Possible targets:** none

**Arguments:** [<new # of balls> | +<balls to add> | -<balls to remove>]

Displays or changes the number of balls in the arena. A number without a plus or minus sign is taken as a new count. A plus signifies adding that many, and a minus removes that many. Continuum currently supports only eight balls.

## ballinfo

**Possible targets:** none

**Arguments:** none

Displays the last known position of balls, as well as the player who is carrying it or who fired it, if applicable.

## best

**Possible targets:** none

**Arguments:** none

Displays your best track record for this arena.

## botfeature

**Possible targets:** none

**Arguments:** [+/-seeallposn] [+/-seeownposn]

Enables or disables bot-specific features. **seeallposn** controls whether the bot gets to see all position packets. **seeownposn** controls whether you get your own mirror position packets.

## cancelfile

**Requires module:** sendfile

**Possible targets:** none

**Arguments:** none

Withdraw your previously offered files.

## cd

**Possible targets:** none

**Arguments:** [<server directory>]

Changes working directory for file transfer. Note that the specified path must be an absolute path; it is not considered relative to the previous working directory. If no arguments are specified, return to the server's root directory.

## credits

**Possible targets:** Player, none

**Arguments:** none

If no target is selected, the ?credits command will list your own credits. Otherwise, private message a player ?credits in order to view how many credits they currently have.

## delfile

**Possible targets:** none

**Arguments:** <server pathname>

Delete a file from the server. Paths are relative to the current working directory.

## delmidban

**Requires module:** auth\_ban

**Possible targets:** none

**Arguments:** <machine id>

Removes a machine id ban.

## destroy

**Possible targets:** none

**Arguments:** -c, -s

Using ?destroy -c will clear the entire credits database and return the top player. Using ?destroy -s will clear the player scores table and return the top players of certain sections.

## detmod

**Possible targets:** none

**Arguments:** <module name>

Detaches the module from the arena.

## disablecmdgroup

**Possible targets:** none

**Arguments:** <command group>

Disables all the commands in the specified command group and released the modules that they require. This can be used to release interfaces so that modules can be unloaded or upgraded without unloading playercmd (which would be irreversible).

## donate

**Possible targets:** Player

**Arguments:** none

Private message a player ?donate <amount> in order To donate some of your own credits to that player.



## dropturret

**Requires module:** autoturret

**Possible targets:** none

**Arguments:** [<freq>]

Drops a turret right where your ship is. The turret will fire 10 level 1 bombs, 1.5 seconds apart, and then disappear.

## enablecmdgroup

**Possible targets:** none

**Arguments:** <command group>

Enables all the commands in the specified command group. This is only useful after using ?disablecmdgroup.

## endinterval

**Possible targets:** none

**Arguments:** [-g] [-a <arena group name>] <interval name>

Causes the specified interval to be reset. If -g is specified, reset the interval at the global scope. If -a is specified, use the named arena group. Otherwise, use the current arena's scope. Interval names can be game; reset; or maprotation.

## energy

**Possible targets:** arena or player

**Arguments:** [-t] [-n] [-s]

If sent as a priv message, turns energy viewing on for that player. If sent as a pub message, turns energy viewing on for the whole arena (note that this will only affect new players entering the arena). If -t is given, turns energy viewing on for teammates only. If -n is given, turns energy viewing off. If -s is given, turns energy viewing on/off for spectator mode.

## faceoff

**Requires module:** <py> hockey

Min. players: 2 This is an attach module created by resoL This command will start the faceoff for the hockey game

## find

**Possible targets:** none

**Arguments:** <all or part of a player name>

Tells you where the specified player is right now. If you specify only part of a player name, it will try to find a matching name using a case insensitive substring search.

## flaginfo

**Possible targets:** none

**Arguments:** none

Displays information (status, location, carrier) about all the flags in the arena.

## flagreset

**Possible targets:** none

**Arguments:** none

Causes the flag game to immediately reset.

## forceding

**Requires module:** turf\_reward

**Possible targets:** none

**Arguments:** none

Forces a reward to take place immediately in your current arena.

## forcestats

**Requires module:** turf\_stats

**Possible targets:** none

**Arguments:** none

Displays stats to arena for previous dings.

## freqkick

**Requires module:** freqowners

**Possible targets:** player

**Arguments:** none

Kicks the player off of your freq. The player must be on your freq and must not be an owner himself. The player giving the command, of course, must be an owner.

## freqpwd

**Requires module:** fm\_password

**Possible targets:** none

**Arguments:** <password>

Sets a password for your freq. Public freqs and the spec freq cannot have passwords.

## gamerecord

**Requires module:** record

**Possible targets:** none

**Arguments:** status | record <file> | play <file> | pause | restart | stop

TODO: write more here.

## geta

**Possible targets:** none

**Arguments:** section:key

Displays the value of an arena setting. Make sure there are no spaces around the colon.

## getcm

**Possible targets:** player or arena

**Arguments:** none

Prints out the chat mask for the target player, or if no target, for the current arena. The chat mask specifies which types of chat messages are allowed.

## getcmdb

**Possible targets:** player

**Arguments:** [player]

Prints out the chat mask for the target player. The chat mask specifies which types of chat messages are allowed. If no target specified, lists all saved player chat masks.

## getfile

**Possible targets:** none

**Arguments:** <filename>

Transfers the specified file from the server to the client. The filename is considered relative to the current working directory.

## getg

**Possible targets:** none

**Arguments:** section:key

Displays the value of a global setting. Make sure there are no spaces around the colon.

## getgroup

**Possible targets:** player or none

**Arguments:** none

Displays the group of the player, or if none specified, you.

## giveall

**Possible targets:** none

**Arguments:** none

Gives all players in the arena credits.

## giveball

**Possible targets:** player or none

**Arguments:** [-f] [<ballid>]

Moves the specified ball to you, or to target player. If no ball is specified, ball id 0 is assumed. If -f is specified, the ball is forced onto the player and there will be no shot timer, and if the player is already carrying a ball it will be dropped where they are standing. If -f is not specified, then the ball is simply moved underneath a player for him to pick up, but any balls already carried are not dropped.

## giveowner

**Requires module:** freqowners

**Possible targets:** player

**Arguments:** none

Allows you to share freq ownership with another player on your current private freq. You can't remove ownership once you give it out, but you are safe from being kicked off yourself, as long as you have ownership.

## grplogin

**Possible targets:** none

**Arguments:** <group name> <password>

Logs you in to the specified group, if the password is correct.

## hashpasswd

**Requires module:** auth\_file

**Possible targets:** none

**Arguments:** <player name>:<new password>

Generates the password hash to be used in passwd.conf for the given player name and password.

## help

**Possible targets:** none

**Arguments:** <command name> | <setting name (section:key)>

Displays help on a command or config file setting. Use **section:** to list known keys in that section. Use **:** to list known section names.

## host

**Possible targets:** none

**Arguments:** request, none, event

By default, ?host <msg> will send a message to any online moderator requesting that an event be hosted. If the player using this command is a staff member, and if no parameters are given, ?host will list the available events in that arena. If an event is specified (e.g. ?host elim), it will attempt to start that event.

## howto

**Requires module:** <py> hockey

Commands: start, stop, faceoff, apfz, apfo, rpz, rpfo, howto. For help on a specific command, use ?c command

## idles

**Possible targets:** arena

Syntax: ?idles Lists everyone who is Idle. See Also: ?actives

## info

**Possible targets:** player

**Arguments:** none

Displays various information on the target player, including which client they are using, their resolution, IP address, how long they have been connected, and bandwidth usage information.

## insmod

**Possible targets:** none

**Arguments:** <module specifier>

Immediately loads the specified module into the server.

## jackpot

**Possible targets:** none

**Arguments:** none or <arena name> or all

Displays the current jackpot for this arena, the named arena, or all arenas.

## joinpwd

**Requires module:** fm\_password

**Possible targets:** none

**Arguments:** <password>

Sets your joining password, which will be used to check if you can join password-protected freqs.

## kick

**Requires module:** auth\_ban

**Possible targets:** player

**Arguments:** [-s seconds | -t seconds | -m minutes | seconds] [message]

Kicks the player off of the server, with an optional timeout. (-s number, -t number, or number for seconds, -m number for minutes.) For kicks with a timeout, you may provide a message to be displayed to the user. Messages appear to users on timeout as 'You have been temporarily kicked for <MESSAGE>.

## lag

**Possible targets:** none or player

**Arguments:** [-v]

Displays lag information about you or a target player. Use -v for more detail. The format of the ping fields is 'last average (min-max):'

## laghist

**Possible targets:** none or player

**Arguments:** [-r]

Displays lag histograms. If a -r is given, do this histogram for 'reliable' latency instead of c2s pings.

## last

**Possible targets:** none

**Arguments:** none

Tells you the last 10 people to log in.

## lastlog

**Requires module:** log\_sysop

**Possible targets:** none or player

**Arguments:** [<number of lines>] [<limiting text>]

Displays the last <number> lines in the server log (default: 10). If limiting text is specified, only lines that contain that text will be displayed. If a player is targeted, only lines mentioning that player will be displayed.

## listarena

**Possible targets:** none

**Arguments:** <arena name>

Lists the players in the given arena.

## listcommand

**Possible targets:** none, Player

Displays all the commands that you (or the specified player) can use. Commands in the arena section are specific to the current arena. The symbol before the command specifies how you can use the command: A dot '.' means you use the command without sending it to a player, it might apply to the entire zone, the current arena or to yourself. A slash '/' means you can send the command in a private message to a player, the effects will then apply to that player only. A colon ':' means you can send the command in a private message to a player in a different arena

## listmidbans

**Requires module:** auth\_ban

**Possible targets:** none

**Arguments:** none

Lists the current kicks (machine-id bans) in effect.

## listmod

**Possible targets:** none

**Arguments:** none

Lists all staff members logged on, which arena they are in, and which group they belong to.

## local\_password

**Requires module:** auth\_file

**Possible targets:** none

**Arguments:** <new password>

Changes your local server password. Note that this command only changes the password used by the auth\_file authentication mechanism (used when the billing server is disconnected). This command does not involve the billing server.

## lock

**Possible targets:** player, freq, or arena

**Arguments:** [-n] [-s] [-t <timeout>]

Locks the specified targets so that they can't change ships. Use ?unlock to unlock them. By default, ?lock won't change anyone's ship. If -s is present, it will spec the targets before locking them. If -n is present, it will notify players of their change in status. If -t is present, you can specify a timeout in seconds for the lock to be effective.

## lockarena

**Possible targets:** arena

**Arguments:** [-n] [-a] [-i] [-s]

Changes the default locked state for the arena so entering players will be locked to spectator mode. Also locks everyone currently in the arena to their ships. The -n option means to notify players of their change in status. The -a options means to only change the arena's state, and not lock current players. The -i option means to only lock entering players to their initial ships, instead of spectator mode. The -s means to spec all players before locking the arena.

## lsmod

**Possible targets:** none

**Arguments:** [-a] [-s] [<text>]

Lists all the modules currently loaded into the server. With -a, lists only modules attached to this arena. With -s, sorts by name. With optional 'text', limits modules displayed to those whose names contain the given text. By using <c> or <py> as the 'text', modules of a specific module loader are listed

## makearena

**Possible targets:** none

**Arguments:** <arena name>

Creates a directory for the new directory under 'arenas/'

## mapinfo

**Possible targets:** none

**Arguments:** none

Displays some information about the map in this arena.

## mark

**Possible targets:** Player, Arena

Syntax: /?mark, ?mark <player>, ?mark (list only) Marks player in purple on your radar.

## modinfo

**Possible targets:** none

**Arguments:** <module name>

Displays information about the specified module. This might include a version number, contact information for the author, and a general description of the module.

## moveball

**Possible targets:** none

**Arguments:** <ballid> <xtile> <ytile>

Moves the specified ball to the specified coordinates.

## moveflag

**Possible targets:** none

**Arguments:** <flag id> <owning freq> [<x coord> <y coord>]

Moves the specified flag. You must always specify the freq that will own the flag. The coordinates are optional: if they are specified, the flag will be moved there, otherwise it will remain where it is.

## netstats

**Possible targets:** none

**Arguments:** none

Prints out some statistics from the network layer.

## neutflag

**Possible targets:** none

**Arguments:** <flag id>

Neuts the specified flag in the middle of the arena.

## newgame

**Requires module:** `sgm_cnc2`

Basically restarts the game.

## notavailable

**Possible targets:** self

Syntax: ?notavailable Command Aliases: ?nav Marks you as Not Available. Certain games will prevent you from being picked when Not Available.

## notify

**Possible targets:** none

**Arguments:** <message>

Sends the message to all online staff members.

## objimage

**Possible targets:** any

**Arguments:** <id> <image>

Change the image associated with an object id. Object commands: ?objon ?objoff ?objset ?objmove ?objimage ?objlayer ?objtimer ?objmode ?objinfo ?objlist

## objinfo

**Possible targets:** none

**Arguments:** <id>

Reports all known information about the object. Object commands: ?objon ?objoff ?objset ?objmove ?objimage ?objlayer ?objtimer ?objmode ?objinfo ?objlist

## objlayer

**Possible targets:** any

**Arguments:** <id> <layer code>

Change the image associated with an object id. Layer codes: BelowAll AfterBackground AfterTiles AfterWeapons AfterShips AfterGauges AfterChat TopMost Object commands: ?objon ?objoff ?objset ?objmove ?objimage ?objlayer ?objtimer ?objmode ?objinfo ?objlist

## objlist

**Possible targets:** none

**Arguments:** none

List all ServerControlled object id's. Use ?objinfo <id> for attributes Object commands: ?objon ?objoff ?objset ?objmove ?objimage ?objlayer ?objtimer ?objmode ?objinfo ?objlist

## objmode

**Possible targets:** any

**Arguments:** <id> <mode code>

Change the mode associated with an object id. Mode codes: ShowAlways EnterZone EnterArena Kill Death ServerControlled Object commands: ?objon ?objoff ?objset ?objmove ?objimage ?objlayer ?objtimer ?objmode ?objinfo ?objlist

## objmove

**Possible targets:** any

**Arguments:** <id> <x> <y> (for map obj) or <id> [CBSGFETROWV]<0/1> [CBSGFETROWV]<0/1> (screen obj)

Moves an LVZ map or screen object. Coordinates are in pixels. Object commands: ?objon ?objoff ?objset ?objmove ?objimage ?objlayer ?objtimer ?objmode ?objinfo ?objlist

## objoff

**Possible targets:** any

**Arguments:** object id

Toggles the specified object off. Object commands: ?objon ?objoff ?objset ?objmove ?objimage ?objlayer ?objtimer ?objmode ?objinfo ?objlist



## objon

**Possible targets:** any

**Arguments:** object id

Toggles the specified object on. Object commands: ?objon ?objoff ?objset ?objmove ?objimage ?objlayer ?objtimer ?objmode ?objinfo ?objlist

## objset

**Possible targets:** any

**Arguments:** [+/-]object id [+/-]id ...

Toggles the specified objects on/off. Object commands: ?objon ?objoff ?objset ?objmove ?objimage ?objlayer ?objtimer ?objmode ?objinfo ?objlist

## objtimer

**Possible targets:** any

**Arguments:** <id> <timer>

Change the timer associated with an object id. Object commands: ?objon ?objoff ?objset ?objmove ?objimage ?objlayer ?objtimer ?objmode ?objinfo ?objlist

## obscene

**Possible targets:** none

**Arguments:** none

Toggles the obscene word filter.

## owner

**Possible targets:** none

**Arguments:** none

Displays the arena owner.

## pausetimer

**Possible targets:** none

**Arguments:** none

Toggles the timer between paused and unpaused. The timer must have been created with ?timer.

## points

**Possible targets:** any

**Arguments:** <points to add>

Adds the specified number of points to the targets' flag points.

## prize

**Possible targets:** player, freq, or arena

**Arguments:** see description

Gives the specified prizes to the target player(s).

Prizes are specified with an optional count, and then a prize name (e.g. 3 reps, anti). Negative prizes can be specified with a '-' before the prize name or the count (e.g. -prox, -3 bricks, 5 -guns). More than one prize can be specified in one command. A count without a prize name means random. For compatability, numerical prize ids with # are supported.

## putfile

**Possible targets:** none

**Arguments:** <client filename>[:<server filename>]

Transfers the specified file from the client to the server. The server filename, if specified, will be considered relative to the current working directory. If omitted, the uploaded file will be placed in the current working directory and named the same as on the client.

## putmap

**Possible targets:** none

**Arguments:** <map file>

Transfers the specified map file from the client to the server. The map will be placed in maps/uploads/<arenabasename>.lvl, and the setting General:Map will be changed to the name of the uploaded file.

## putzip

**Possible targets:** none

**Arguments:** <client filename>[:<server directory>]

Uploads the specified zip file to the server and unzips it in the specified directory (considered relative to the current working directory), or if none is provided, the working directory itself. This can be used to efficiently send a large number of files to the server at once, while preserving directory structure.

## pwd

**Possible targets:** none

**Arguments:** none

Prints the current working directory. A working directory of :indicates the server's root directory.

## py

**Requires module:** <py> exec

**Possible targets:** any

**Arguments:** <python code>

Executes arbitrary python code. The code runs in a namespace containing all of the ass module, plus three helpful preset variables: **me** is yourself, **t** is the target of the command, and **a** is the current arena. You can write multi-line statements by separating lines with semicolons (be sure to get the indentation correct). Output written to stdout (e.g., with print) is captured and displayed to you, as are any exceptions raised in your code.

## qip

**Possible targets:** none

**Arguments:** <ip address or pattern>

Queries the alias database for players connecting from that ip. Queries can be an exact address, ?qip 216.34.65.%, or ?qip 216.34.65.0/24.

## quickfix

**Requires module:** quickfix

**Possible targets:** none

**Arguments:** <limiting text>

Lets you quickly change arena settings. This will display some list of settings with their current values and allow you to change them. The argument to this command can be used to limit the list of settings displayed. (With no arguments, equivalent to ?getsettings in subgame.)

## rawquery

**Possible targets:** none

**Arguments:** <sql code>

Performs a custom sql query on the alias data. The text you type after ?rawquery will be used as the WHERE clause in the query. Examples: ?rawquery name like '%blah%' ?rawquery macid = 34127563 order by lastseen desc

## recyclearena

**Possible targets:** none

**Arguments:** none

Recycles the current arena without kicking players off.

## recyclezone

**Possible targets:** none

**Arguments:** none

Immediately shuts down the server, exiting with EXIT\_RECYCLE. The run-asss script, if it is being used, will notice EXIT\_RECYCLE and restart the server.

## redirect

**Possible targets:** any

**Arguments:** <redirect alias> | <ip>:<port>[:<arena>]

**Requires module:** redirect

Redirects the target to a different zone.

## reloadconf

**Possible targets:** none

**Arguments:** [-f] [path]

With no args, causes the server to reload any config files that have been modified since they were loaded. With -f, forces a reload of all open files. With a string, forces a reload of all files whose pathnames contain that string.

## removecreds

**Possible targets:** Player

**Arguments:** none

Private message a player ?removecredits <amount> in order To remove a specified amount of credits.

## renfile

**Possible targets:** none

**Arguments:** <old filename>:<new filename>

Rename a file on the server. Paths are relative to the current working directory.

## reply

**Possible targets:** player

**Arguments:** <message>

Sends a private message to a player. Useful for logging replies to moderator help requests.

## **resetcredits**

**Possible targets:** Player

**Arguments:** none

Private message a player ?resetcredits in order To reset their credits to the default amount.

## **resetgame**

**Possible targets:** none

**Arguments:** none

Resets soccer game scores and balls.

## **rmgroup**

**Possible targets:** player

**Arguments:** none

Removes the group from a player, returning him to group 'default'. If the group was assigned for this session only, then it will be removed for this session; if it is a global group, it will be removed globally; and if it is an arena group, it will be removed for this arena.

## **rmmod**

**Possible targets:** none

**Arguments:** <module name>

Attempts to unload the specified module from the server.

## **rpfo**

**Requires module:** <py> hockey

This command will force remove a teams point.

## **rpfz**

**Requires module:** <py> hockey

This command will force remove a teams point.

## **rules**

**Possible targets:** none

**Arguments:** none

If a game is currently in play, this command will display the rules.

## **score**

**Possible targets:** none

**Arguments:** none

Returns current score of the soccer game in progress.

## **scorerreset**

**Possible targets:** none or player

**Arguments:** none

Resets your own score, or the target player's score.

## send

**Possible targets:** player

**Arguments:** <arena name>

Sends target player to the named arena. (Works on Continuum users only.)

## sendfile

**Requires module:** sendfile

**Possible targets:** player

**Arguments:** none

Offer someone a file from your client's directory. Only one file can be offered at once.

## set\_local\_password

**Requires module:** auth\_file

**Possible targets:** player

**Arguments:** none

If used on a player that has no local password set, it will set their local password to the password they used to log in to this session.

## seta

**Possible targets:** none

**Arguments:** [-t] section:key=value

Sets the value of an arena setting. Make sure there are no spaces around either the colon or the equals sign. A -t makes the setting temporary.

## setcm

**Possible targets:** player or arena

**Arguments:** see description

Modifies the chat mask for the target player, or if no target, for the current arena. The arguments must all be of the form (-|+)(pub|pubmacro|freq|nmefreq|priv|chat|modchat|all) or -t <seconds>. A minus sign and then a word disables that type of chat, and a plus sign enables it. The special type all means to apply the plus or minus to all of the above types. -t lets you specify a timeout in seconds. The mask will be effective for that time, even across logouts.

Examples:

- If someone is spamming public macros: :player:?setcm -pubmacro -t 600
- To disable all blue messages for this arena: ?setcm -pub -pubmacro
- An equivalent to \*shutup: :player:?setcm -all
- To restore chat to normal: ?setcm +all

Current limitations: You can't currently restrict a particular frequency. Leaving and entering an arena will remove a player's chat mask, unless it has a timeout.

## setcmdb

**Possible targets:** player

**Arguments:** see description

Unlike ?setcm this command doesn't modify the existing chatmask of the target. It only sets it based on the arguments. The arguments must all be of the form (playername:) (-|+)(pub|pubmacro|freq|nmefreq|priv|modchat|all) or (-t <seconds>). A minus sign and then a word disables that type of chat, and a plus sign enables

it. The special type **all** means to apply the plus or minus to all of the above types. **-t** lets you specify a timeout in seconds, **-t 0** (the default) will not expire. An argument of **+all** will delete the chatmask database record.

Examples:

- To prevent a player from talking in pub `:player:?setcmdb -pub -pubmacro`
- To prevent a player from talking in pub `?setcmdb player: -pub -pubmacro -t 86400`
- To restore chat to normal: `:player:?setcmdb +all`
- To restore chat to normal: `?setcmdb player: +all`

## setcredits

**Possible targets:** Player

**Arguments:** none

Private message a player `?setcredits <amount>` in order To specify a new value.

## setfreq

**Possible targets:** player, freq, or arena

**Arguments:** `[-f] <freq number>`

Moves the targets to the specified freq. If **-f** is specified, this command ignores the arena freqman.

## setg

**Possible targets:** none

**Arguments:** `[-t] section:key=value`

Sets the value of a global setting. Make sure there are no spaces around either the colon or the equals sign. A **-t** makes the setting temporary.

## setgroup

**Possible targets:** player

**Arguments:** `[-a] [-p] <group name>`

Assigns the group given as an argument to the target player. The player must be in group **default**, or the server will refuse to change his group. Additionally, the player giving the command must have an appropriate capability: **setgroup\_foo**, where **foo** is the group that he's trying to set the target to.

The optional **-p** means to assign the group permanently. Otherwise, when the target player logs out or changes arenas, the group will be lost.

The optional **-a** means to make the assignment local to the current arena, rather than being valid in the entire zone.

## setjackpot

**Possible targets:** none

**Arguments:** `<new jackpot value>`

Sets the jackpot for this arena to a new value.

## setscore

**Possible targets:** none

**Arguments:** `<freq 0 score> [<freq 1 score> [... [<freq 7 score>]]]`

Changes score of current soccer game, based on arguments. Only supports first eight freqs, and arena must be in absolute scoring mode (Soccer:CapturePoints < 0).

## setship

**Possible targets:** player, freq, or arena

**Arguments:** [-f] <ship number>

Sets the targets to the specified ship. The argument must be a number from 1 (Warbird) to 8 (Shark), or 9 (Spec). If -f is specified, this command ignores the arena freqman.

## shipreset

**Possible targets:** player, freq, or arena

**Arguments:** none

Resets the target players' ship(s).

## shutdown

**Possible targets:** none

**Arguments:** [-r]

Immediately shuts down the server, exiting with `EXIT_NONE`. If -r is specified, exit with `EXIT_RECYCLE` instead. The `run-asss` script, if it is being used, will notice `EXIT_RECYCLE` and restart the server.

## spawnball

**Possible targets:** none

**Arguments:** [<ballid>]

Resets the specified existing ball back to its spawn location. If no ball is specified, ball id 0 is assumed.

## spec

**Possible targets:** any

**Arguments:** none

Displays players spectating you. When private, displays players spectating the target.

## specactives

**Possible targets:** arena

Syntax: ?actives Lists everyone who is Available, in Spectator Mode, and not Idle. See Also: ?idles ?actives

## specall

**Possible targets:** player, freq, or arena

**Arguments:** none

Sends all of the targets to spectator mode.

## start

**Requires module:** <py> prodem

Min. players: 2 All ships start as shark, kill an equal or lower number ship to get promoted. If you are killed by a higher number ship then you, you will be demoted. First player to kill another Warbird using a Warbird wins the game.

## stats

**Possible targets:** player or none

**Arguments:** [forever|game|reset]

Prints out some basic statistics about the target player, or if no target, yourself. An interval name can be specified as an argument. By default, the per-reset interval is used.

## stop

**Requires module:** <py> hockey

This command will stop the hockey game if one has started.

## takeownership

**Requires module:** freqowners

**Possible targets:** none

**Arguments:** none

Makes you become owner of your freq, if your freq doesn't have an owner already.

## time

**Possible targets:** none

**Arguments:** none

Displays the time that has passed since the start of the race.

## timer

**Possible targets:** none

**Arguments:** <minutes>[:<seconds>]

Set arena timer to minutes:seconds, only in arenas with TimedGame setting off. Note, that the seconds part is optional, but minutes must always be defined (even if zero). If successful, server replies with ?time response.

## timereset

**Possible targets:** none

**Arguments:** none

Reset a timed game, but only in arenas with Misc:TimedGame in use.

## togglecreds

**Possible targets:** none

**Arguments:** none

Using ?togglecreds will either turn the credits counter (display) on or off.

## trackbest

**Possible targets:** none

**Arguments:** none

Displays the overall best track record for this arena.

## turfinfo

**Requires module:** turf\_reward

**Possible targets:** none

**Arguments:** none

Displays the current settings / requirements to receive awards.



## **turfresetflags**

**Requires module:** turf\_reward

**Possible targets:** none

**Arguments:** none

Resets the turf\_reward module's and flags module's flag data in your current arena.

## **turfresettimer**

**Requires module:** turf\_reward

**Possible targets:** none

**Arguments:** none

Resets the ding timer in your current arena.

## **turfstats**

**Requires module:** turf\_stats

**Possible targets:** none

**Arguments:** none

Gets stats to previous dings.

## **turftime**

**Requires module:** turf\_reward

**Possible targets:** none

**Arguments:** none

Displays the amount of time till next ding.

## **unlock**

**Possible targets:** player, freq, or arena

**Arguments:** [-n]

Unlocks the specified targets so that they can now change ships. An optional **-n** notifies players of their change in status.

## **unlockarena**

**Possible targets:** arena

**Arguments:** [-n] [-a]

Changes the default locked state for the arena so entering players will not be locked to spectator mode. Also unlocks everyone currently in the arena to their ships The **-n** options means to notify players of their change in status. The **-a** option means to only change the arena's state, and not unlock current players.

## **unmark**

**Possible targets:** Player, Arena

Syntax: /?unmark, ?unmark <player> Removes mark on a player.

## **uptime**

**Possible targets:** none

**Arguments:** none

Displays how long the server has been running.

## usage

**Possible targets:** player or none

**Arguments:** none

Displays the usage information (current hours and minutes logged in, and total hours and minutes logged in), as well as the first login time, of the target player, or you if no target.

## userdbadm

**Possible targets:** none

**Arguments:** status|drop|connect

The subcommand 'status' reports the status of the user database server connection. 'drop' disconnects the connection if it's up, and 'connect' reconnects after dropping or failed login.

## userdbid

**Possible targets:** player or none

**Arguments:** none

Displays the user database server id of the target player, or yours if no target.

## userid

**Possible targets:** player or none

**Arguments:** none

Displays the user database id of the target player, or yours if no target.

## version

**Possible targets:** none

**Arguments:** none

Prints out the version and compilation date of the server. It might also print out some information about the machine that it's running on.

## warn

**Possible targets:** player

**Arguments:** <message>

Send a red warning message to a player.

## warpto

**Possible targets:** player, freq, or arena

**Arguments:** <x coord> <y coord>

Warps target player to coordinate x,y.

## watchdamage

**Possible targets:** player, none

**Arguments:** [0 or 1]

Turns damage watching on and off. If sent to a player, an argument of 1 turns it on, 0 turns it off, and no argument toggles. If sent as a public command, only ?watchdamage 0 is meaningful, and it turns off damage watching on all players.

## watchgreen

**Requires module:** <py> watchgreen

**Possible targets:** player or arena

**Arguments:** none

If sent to a player, turns on green watching for that player. If sent as a public message, turns off all your green watching.

## where

**Possible targets:** player

**Arguments:** none

Displays the current location (on the map) of the target player.

## z

**Possible targets:** none

**Arguments:** <text>

Displays the text as an arena (green) message to the whole zone.

## zone

**Possible targets:** none

**Arguments:** none

Displays the name of this zone.

# 12 Configuration Reference

All config files used by ass (except `modules.conf`) have the same format and conventions. The format is roughly based on, and is backwards compatible with, the Windows `.ini` file format, so `server.cfg` files can be used as-is, although you'll probably need to add a few settings to get things working well.

Config files are processed line-by-line. All leading and trailing whitespace is ignored. A line is a comment if the first character (ignoring whitespace) is a semicolon or a forward slash. If the first character is a pound sign, it signals a preprocessor directive. These directives work very much like C preprocessor directives: `#include` allows one config file to include another. `#define` allows macros to be defined. Macros cannot currently take arguments. To reference the definition of a macro, you have to use `$(MACRONAME)`, not just the name of the macro. The parens can be omitted if the character after the end of the macro name isn't alphanumeric. `#ifdef`, `#ifndef`, `#else`, and `#endif` allow conditional inclusion of sections based on whether a specific macro is defined or not. If a line ends with a backslash, it denotes a line continuation: the following line of the file (or more if that line ends with a backslash) is appended to the original line before it is processed.

The start of a section is a line starting with an open bracket and ending with a closing bracket. The text between the brackets is the section name. Any line containing an equals sign is a value: the text before the equals is the key name (minus leading and trailing whitespace) and the text after (again minus whitespace) is the value. Section names and values are case-insensitive, but the case of values is preserved. Lines that don't contain an equals sign also specify keys, and their associated value is the empty string. Value-less keys are used primarily in the capability manager, where the presence or absence of a capability is all that's important.

If a key name contains a colon, it is treated specially: the text before the colon is treated as the section name for this key only (it doesn't modify the idea of the "current section") and the text after the colon is the key name.

The following sections describe specific settings. They are sorted alphabetically by section and then by key. The settings are listed with the section and key names separated by a colon.

The section name “All” isn’t a real section name but means the setting is present in a section for each ship.

## 12.1 Global settings

### **Arenas:PermanentArenas**

**Type:** String

A list of the names of arenas to permanently set up when asss is started up.

### **Billing:GroupID**

**Type:** Integer

**Default:** 1

GroupID identifying zone to user database server.

### **Billing:HonorScoreResetRequests**

**Type:** Boolean

**Default:** Yes

Whether to reset scores when the billing server says it is time to

### **Billing:IP**

**Type:** String

The ip address of the user database server (no dns hostnames allowed).

### **Billing:LocalChatPrefix**

**Type:** String

Secret prefix to prepend to local chats

### **Billing:Password**

**Type:** String

The password to log in to the user database server with.

### **Billing:Port**

**Type:** Integer

**Default:** 1850

The port to connect to on the user database server.

### **Billing:Proxy**

**Type:** String

This setting allows you to specify an external program that will handle the billing server connection. The program should be prepared to speak the asss billing protocol over its standard input and output. It will get two command line arguments, which are the ip and port of the billing server, as specified in the Billing:IP and Billing:Port settings. The program name should either be an absolute pathname or be located on your \$PATH.

### **Billing:RetryInterval**

**Type:** Integer

**Default:** 180

How many seconds to wait between tries to connect to the user database server.

### **Billing:ScoreID**

**Type:** Integer

**Default:** 0

Score realm.

### **Billing:ServerID**

**Type:** Integer

**Default:** 0

ServerID identifying zone to user database server.

**Billing:ServerName**

**Type:** String

The server name to send to the user database server.

**Billing:ServerNetwork**

**Type:** String

The network name to send to the billing server. A network name should identify a group of servers (e.g., SSCX).

**Billing:StaffChatPrefix**

**Type:** String

Secret prefix to prepend to staff chats

**Billing:StaffChats**

**Type:** String

Comma separated staff zone local list.

**Chat:CommandLimit**

**Type:** Integer

**Default:** 5

How many commands are allowed on a single line.

**Chat:FilterMode**

**Type:** Boolean

If true, replace obscene words with garbage characters, otherwise suppress whole line.

**Chat:FloodLimit**

**Type:** Integer

**Default:** 10

How many messages needed to be sent in a short period of time (about a second) to qualify for chat flooding.

**Chat:FloodShutup**

**Type:** Integer

**Default:** 60

How many seconds to disable chat for a player that is flooding chat messages.

**Chat:ForceFilter**

**Type:** Boolean

If true, players will always start with the obscenity filter on by default. If false, use their preference.

**Chat:MessageReliable**

**Type:** Boolean

**Default:** Yes

Whether to send chat messages reliably.

**Chat:Obscene**

**Type:** String

A space-separated list of obscene words to filter. Words starting with a question mark are encoded with rot-13.

**Config:CheckModifiedFilesInterval**

**Type:** Integer

**Default:** 1500

How often to check for modified config files on disk (in ticks).

**Config:FlushDirtyValuesInterval**

**Type:** Integer

**Default:** 500

How often to write modified config settings back to disk (in ticks).

**Directory:Description**

**Type:** String

The server description to send to the directory server. See Directory:Name for more information about the section name.

**Directory:Name**

**Type:** String

The server name to send to the directory server. Virtual servers will use section name 'Directory-<vs-name>' for this and other settings in this section, and will fall back to 'Directory' if that section doesn't exist. See Net:Listen help for how to identify virtual servers.

**Directory:Password**

**Type:** String

**Default:** cane

The password used to send information to the directory server. Don't change this.

**Directory:Port**

**Type:** Integer

**Default:** 4991

The port to connect to for the directory server.

**General:NewsFile**

**Type:** String

**Default:** news.txt

The filename of the news file.

**General:NewsRefreshMinutes**

**Type:** Integer

**Default:** 5

How often to check for an updated news.txt.

**General:PublicArenas**

**Type:** String

**Requires module:** ap\_multipub

A list of public arena types that the server will place people in when they don't request a specific arena.

**Lag:CheckInterval**

**Type:** Integer

**Default:** 300

How often to check each player for out-of-bounds lag values (in ticks).

**Listen:AllowCont**

**Type:** Boolean

**Default:** Yes

Whether Continuum clients are allowed to connect to this port.

**Listen:AllowVIE**

**Type:** Boolean

**Default:** Yes

Whether VIE protocol clients (i.e., Subspace 1.34 and bots) are allowed to connect to this port.

**Listen:BindAddress**

**Type:** String

The interface address to bind to. This is optional, and if omitted, the server will listen on all available interfaces.

**Listen:ConnectAs**

**Type:** String

This setting allows you to treat clients differently depending on which port they connect to. It serves as a virtual server identifier for the rest of the server. The standard arena placement module will use this as the name of a default arena to put clients who connect through this port in.

**Listen:Port**

**Type:** Integer

The port that the game protocol listens on. Sections named Listen1 through Listen9 are also supported. All Listen sections must contain a port setting.

**Log:DatedLogsPath**

**Type:** String

**Default:** log

If using dated log files, the path to put the files in.

**Log:FileFlushPeriod**

**Type:** Integer

**Default:** 10

How often to flush the log file to disk (in minutes).

**Log:LogFile**

**Type:** String

**Default:** asss.log

The name of the log file. Has no effect when using dated log files.

**Log:NewFilePeriod**

**Type:** Integer

**Default:** 3

How often to open a new log file (in days). Has no effect when writing to an undated log file. If less than 1, the same file will be used until the server restarts.

**Log:UseDatedLogs**

**Type:** Boolean

**Default:** No

Whether to use filenames in the format YYMMDD.log

**mysql:database**

**Type:** String

**Requires module:** mysql

The database on the mysql server to use.

**mysql:hostname**

**Type:** String

**Requires module:** mysql

The name of the mysql server.

**mysql:password**

**Type:** String

**Requires module:** mysql

The password to log in to the mysql server as.

**mysql:user**

**Type:** String

**Requires module:** mysql

The mysql user to log in to the server as.

**Net:ChatListen**

**Type:** String

**Requires module:** chatnet

Where to listen for chat protocol connections. Either 'port' or 'ip:port'. Net:Listen will be used if this is missing.

**Net:ChatMessageDelay**

**Type:** Integer

**Default:** 20 mod: chatnet

The delay between sending messages to clients using the text-based chat protocol. (To limit bandwidth used by non-playing clients.)

**Net:DropTimeout**

**Type:** Integer

**Default:** 3000

How long to get no data from a client before disconnecting him (in ticks).

**Net:InternalClientPort**

**Type:** String

The bind port for the internal client socket (used to communicate with biller and dirserver). \*/

**Net:Listen**

**Type:** String

A designation for a port and ip to listen on. Format is one of 'port', 'port:connectas', or 'ip:port:connectas'. Listen1 through Listen9 are also supported. A missing or zero-length 'ip' field means all interfaces. The 'connectas' field can be used to treat clients differently depending on which port or ip they use to connect to the server. It serves as a virtual server identifier for the rest of the server.

**Net:MaxOutlistSize**

**Type:** Integer

**Default:** 500

How many S2C packets the server will buffer for a client before dropping him.

**Net:SimplePingPopulationMode**

**Type:** Integer

**Default:** 1

Display what value in the simple ping response (used by continuum)? 1 = display total player count (default); 2 = display playing count (in ships); 3 = alternate between 1 and 2

**Peer0:Address**

**Type:** String

Send and receive peer packets to/from this IP address

**Peer0:Arenas**

**Type:** String

A list of arena's that belong to the peer. This server will redirect players that try to go to this arena. These arena's will also be used for find and will be shown in ?arena. If you are also



using Peer0:RenameArenas, you should put the local arena name here; this is the one you would see in the ?arena list if you are in this zone

**Peer0:IncludeInPopulation**

**Type:** Boolean

If set, include the population count of this peer in the ping protocol.

**Peer0:Password**

**Type:** String

Peers must agree upon a common password

**Peer0:Port**

**Type:** Integer

Send and receive peer packets to/from this UDP port

**Peer0:ProvidesDefaultArenas**

**Type:** Boolean

If set, any arena that would normally end up as (default) will be redirected to this peer zone

**Peer0:ReceiveMessages**

**Type:** Boolean

If set, display the zone (\*zone) and alert messages from this peer

**Peer0:RelayArenas**

**Type:** String

A list of arena's of this peer that will be relayed to other peers.

**Peer0:RenameArenas**

**Type:** String

A list of arena's that belong to the peer which should be renamed to a different name locally. For example 'foo=bar,0=twpublic' will display the remote 'foo' arena as 'bar' instead

**Peer0:SendDummyArenas**

**Type:** String

A list of arena's that we send to the peer with a single dummy player. Instead of the full player list. This will keep the arena in the arena list of the peer with a fixed count of 1

**Peer0:SendMessages**

**Type:** Boolean

If set, forward alert and zone (?z) messages to the peer

**Peer0:SendOnly**

**Type:** Boolean

If set, we send data to our peer but we reject any that we might receive

**Peer0:SendPlayerList**

**Type:** Boolean

If set, send a full arena and player list to the peer. Otherwise only send a summary of our population

**Peer0:SendZeroPlayerCount**

**Type:** Boolean

If set and SendPlayerList is not set, always send a population count of 0

**Persist:SyncSeconds**

**Type:** Integer

**Default:** 180

The interval at which all persistent data is synced to the database.

**Redirects:name;**

**Type:** String

Settings in the Redirects section correspond to arena names. If a player tries to go to an arena name listed in this section, they will be redirected to the zone specified as the value of the setting. The format of values is 'ip:port[:arena]'. \*/

**Security:SecurityKickoff**

**Type:** Boolean

**Default:** No

Whether to kick players off of the server for violating security checks.

**SGCompat:TemporarySet**

**Type:** Boolean

If this setting is 0, the '?set' command will be mapped to '?seta'. If this setting is 1, the '?set' command will be mapped to '?seta -t'.

## 12.2 Arena settings

**All:AfterburnerEnergy**

**Type:** Integer

Amount of energy required to have 'Afterburners' activated

**All:AntiWarpEnergy**

**Type:** Integer

Amount of energy required to have 'Anti-Warp' activated (thousanth per tick)

**All:AntiWarpStatus**

**Type:** Integer

**Range:** 0-2

Whether ships are allowed to receive 'Anti-Warp' 0=no 1=yes 2=yes/start-with

**All:AttachBounty**

**Type:** Integer

Bounty required by ships to attach as a turret

**All:BombBounceCount**

**Type:** Integer

Number of times a ship's bombs bounce before they explode on impact

**All:BombFireDelay**

**Type:** Integer

delay that ship waits after a bomb is fired until another weapon may be fired (in ticks)

**All:BombFireEnergy**

**Type:** Integer

Amount of energy it takes a ship to fire a single bomb

**All:BombFireEnergyUpgrade**

**Type:** Integer

Extra amount of energy it takes a ship to fire an upgraded bomb. i.e.  $L2 = \text{BombFireEnergy} + \text{BombFireEnergyUpgrade}$

**All:BombSpeed**

**Type:** Integer

How fast bombs travel

**All: BombThrust**

**Type:** Integer

Amount of back-thrust you receive when firing a bomb

**All: BrickMax**

**Type:** Integer

Maximum number of Bricks allowed in ships

**All: BulletFireDelay**

**Type:** Integer

Delay that ship waits after a bullet is fired until another weapon may be fired (in ticks)

**All: BulletFireEnergy**

**Type:** Integer

Amount of energy it takes a ship to fire a single L1 bullet

**All: BulletSpeed**

**Type:** Integer

How fast bullets travel

**All: BurstMax**

**Type:** Integer

Maximum number of Bursts allowed in ships

**All: BurstShrapnel**

**Type:** Integer

Number of bullets released when a 'Burst' is activated

**All: BurstSpeed**

**Type:** Integer

How fast the burst shrapnel is for this ship

**All: CloakEnergy**

**Type:** Integer

Amount of energy required to have 'Cloak' activated (thousandths per tick)

**All: CloakStatus**

**Type:** Integer

**Range:** 0-2

Whether ships are allowed to receive 'Cloak' 0=no 1=yes 2=yes/start-with

**All: DamageFactor**

**Type:** Integer

How likely a the ship is to take damage (ie. lose a prize) (0=special-case-never, 1=extremely likely, 5000=almost never)

**All: DecoyMax**

**Type:** Integer

Maximum number of Decoys allowed in ships

**All: DisableFastShooting**

**Type:** Boolean

If firing bullets, bombs, or thors is disabled after using afterburners (1=enabled) (Cont .36+)

**All: DoubleBarrel**

**Type:** Boolean

Whether ships fire with double barrel bullets

**All:EmpBomb**

**Type:** Boolean

Whether ships fire EMP bombs

**All:Gravity**

**Type:** Integer

How strong of an effect the wormhole has on this ship (0 = none)

**All:GravityTopSpeed**

**Type:** Integer

Ship are allowed to move faster than their maximum speed while effected by a wormhole. This determines how much faster they can go (0 = no extra speed)

**All:InitialBombs**

**Type:** Other

**Range:** 0-3

Initial level a ship's bombs fire

**All:InitialBounce**

**Type:** Integer

**Default:** 0

**Range:** 0-1

Initial bouncing bullets upon spawn

**All:InitialBounty**

**Type:** Integer

Number of 'Greens' given to ships when they start

**All:InitialBrick**

**Type:** Integer

Initial number of Bricks given to ships when they start

**All:InitialBurst**

**Type:** Integer

Initial number of Bursts given to ships when they start

**All:InitialDecoy**

**Type:** Integer

Initial number of Decoys given to ships when they start

**All:InitialEnergy**

**Type:** Integer

Initial amount of energy that the ship can have

**All:InitialGuns**

**Type:** Integer

**Range:** 0-3

Initial level a ship's guns fire

**All:InitialMultifire**

**Type:** Integer

**Default:** 0

**Range:** 0-1

Multifire upon spawn

**All:InitialPortal****Type:** Integer

Initial number of Portals given to ships when they start

**All:InitialProx****Type:** Integer**Default:** 0**Range:** 0-1

Prox upon spawn

**All:InitialRecharge****Type:** Integer

Initial recharge rate, or how quickly this ship recharges its energy

**All:InitialRepel****Type:** Integer

Initial number of Repels given to ships when they start

**All:InitialRocket****Type:** Integer

Initial number of Rockets given to ships when they start

**All:InitialRotation****Type:** Integer

Initial rotation rate of the ship (0 = can't rotate, 400 = full rotation in 1 second)

**All:InitialShield****Type:** Integer**Default:** 0**Range:** 0-1

Temporary shields upon spawn

**All:InitialShrapnel****Type:** Integer**Default:** 0**Range:** 0-31

one or more shrapnel prizes upon spawn

**All:InitialSpeed****Type:** Integer

Initial speed of ship (0 = can't move)

**All:InitialSuper****Type:** Integer**Default:** 0**Range:** 0-1

Temporary super upon spawn

**All:InitialThor****Type:** Integer

Initial number of Thor's Hammers given to ships when they start

**All:InitialThrust****Type:** Integer

Initial thrust of ship (0 = none)

**All:LandmineFireDelay**

**Type:** Integer

Delay that ship waits after a mine is fired until another weapon may be fired (in ticks)

**All:LandmineFireEnergy**

**Type:** Integer

Amount of energy it takes a ship to place a single L1 mine

**All:LandmineFireEnergyUpgrade**

**Type:** Integer

Extra amount of energy it takes to place an upgraded landmine. i.e.  $L2 = \text{LandmineFireEnergy} + \text{LandmineFireEnergyUpgrade}$

**All:LimitPerTeam**

**Type:** Integer

**Requires module:** enf\_shipcount

**Default:** -1

The maximum number of this ship on any given frequency. -1 means no limit.

**All:MaxBombs**

**Type:** Integer

**Range:** 0-3

Maximum level a ship's bombs can fire

**All:MaxGuns**

**Type:** Integer

**Range:** 0-3

Maximum level a ship's guns can fire

**All:MaximumEnergy**

**Type:** Integer

Maximum amount of energy that the ship can have

**All:MaximumRecharge**

**Type:** Integer

Maximum recharge rate, or how quickly this ship recharges its energy

**All:MaximumRotation**

**Type:** Integer

Maximum rotation rate of the ship (0 = can't rotate, 400 = full rotation in 1 second)

**All:MaximumSpeed**

**Type:** Integer

Maximum speed of ship (0 = can't move)

**All:MaximumThrust**

**Type:** Integer

Maximum thrust of ship (0 = none)

**All:MaxMines**

**Type:** Integer

Maximum number of mines allowed in ships

**All:MultiFireAngle**

**Type:** Integer

Angle spread between multi-fire bullets and standard forward firing bullets (111 = 1 degree, 1000 = 1 ship-rotation-point)

**All:MultiFireDelay****Type:** Integer

Delay that ship waits after a multifire bullet is fired until another weapon may be fired (in ticks)

**All:MultiFireEnergy****Type:** Integer

Amount of energy it takes a ship to fire multifire L1 bullets

**All:PortalMax****Type:** Integer

Maximum number of Portals allowed in ships

**All:PrizeShareLimit****Type:** Integer

Maximum bounty that ships receive Team Prizes

**All:Radius****Type:** Integer**Default:** 14**Range:** 0-255

The ship's radius from center to outside, in pixels. (Cont .37+)

**All:RepelMax****Type:** Integer

Maximum number of Repels allowed in ships

**All:RocketMax****Type:** Integer

Maximum number of Rockets allowed in ships

**All:RocketTime****Type:** Integer

How long a Rocket lasts (in ticks)

**All:SeeBombLevel****Type:** Integer**Range:** 0-4

If ship can see bombs on radar (0=Disabled, 1=All, 2=L2 and up, 3=L3 and up, 4=L4 bombs only)

**All:SeeMines****Type:** Boolean

Whether ships see mines on radar

**All:ShieldsTime****Type:** Integer

How long Shields lasts on the ship (in ticks)

**All:ShrapBounce****Type:** Integer**Default:** 0

Whether this ship has bouncing shrapnel -1 = never; 1 = always When set to non 0 weapon packets will be rewritten

**All:ShrapLevel****Type:** Integer**Default:** 0

**Range:** 0-4

The shrapnel level of this ship When set to non 0 weapon packets will be rewritten

**All:ShrapnelMax**

**Type:** Integer

Maximum amount of shrapnel released from a ship's bomb

**All:ShrapnelRate**

**Type:** Integer

Amount of additional shrapnel gained by a 'Shrapnel Upgrade' prize.

**All:SoccerBallFriction**

**Type:** Integer

Amount the friction on the soccer ball (how quickly it slows down – higher numbers mean faster slowdown)

**All:SoccerBallProximity**

**Type:** Integer

How close the player must be in order to pick up ball (in pixels)

**All:SoccerBallSpeed**

**Type:** Integer

Initial speed given to the ball when fired by the carrier

**All:SoccerThrowTime**

**Type:** Integer

Time player has to carry soccer ball (in ticks)

**All:StealthEnergy**

**Type:** Integer

Amount of energy required to have 'Stealth' activated (thousandths per tick)

**All:StealthStatus**

**Type:** Integer

**Range:** 0-2

Whether ships are allowed to receive 'Stealth' 0=no 1=yes 2=yes/start-with

**All:SuperTime**

**Type:** Integer

How long Super lasts on the ship (in ticks)

**All:ThorLevel**

**Type:** Other

**Range:** 0-4

The thor level this ship has. When set to non 0 weapon packets will be rewritten

**All:ThorMax**

**Type:** Integer

Maximum number of Thor's Hammers allowed in ships

**All:ThorShrap**

**Type:** Other

**Range:** 0-31

The amount of shrap thors have When set to non 0 weapon packets will be rewritten

**All:ThorShrapBounce**

**Type:** Integer



**Default:** 0

**Range:** 0-1

Whether this ship has bouncing shrapnel for thors -1 = never; 1 = always. When set to non 0 weapon packets will be rewritten

**All:ThorShrapLevel**

**Type:** Other

**Range:** 0-4

The thor shrapnel level this ship has. When set to non 0 weapon packets will be rewritten

**All:TurretLimit**

**Type:** Integer

Number of turrets allowed on a ship

**All:TurretSpeedPenalty**

**Type:** Integer

Amount the ship's speed is decreased with a turret riding

**All:TurretThrustPenalty**

**Type:** Integer

Amount the ship's thrust is decreased with a turret riding

**All:UpgradeEnergy**

**Type:** Integer

Amount added per 'Energy Upgrade' Prize

**All:UpgradeRecharge**

**Type:** Integer

Amount added per 'Recharge Rate' Prize

**All:UpgradeRotation**

**Type:** Integer

Amount added per 'Rotation' Prize

**All:UpgradeSpeed**

**Type:** Integer

Amount added per 'Speed' Prize

**All:UpgradeThrust**

**Type:** Integer

Amount added per 'Thruster' Prize

**All:XRadarEnergy**

**Type:** Integer

Amount of energy required to have 'X-Radar' activated (thousandths per tick)

**All:XRadarStatus**

**Type:** Integer

**Range:** 0-2

Whether ships are allowed to receive 'X-Radar' 0=no 1=yes 2=yes/start-with

**Bomb:BBombDamagePercent**

**Type:** Integer

Percentage of normal damage applied to a bouncing bomb (in 0.1%)

**Bomb: BombAliveTime**

**Type:** Integer

Time bomb is alive (in ticks)

**Bomb:BombDamageLevel**

**Type:** Integer

Amount of damage a bomb causes at its center point (for all bomb levels)

**Bomb:BombExplodeDelay**

**Type:** Integer

How long after the proximity sensor is triggered before bomb explodes

**Bomb:BombExplodePixels**

**Type:** Integer

Blast radius in pixels for an L1 bomb (L2 bombs double this, L3 bombs triple this)

**Bomb:BombSafety**

**Type:** Boolean

Whether proximity bombs have a firing safety. If enemy ship is within proximity radius, will it allow you to fire

**Bomb:EBombDamagePercent**

**Type:** Integer

Percentage of normal damage applied to an EMP bomb (in 0.1%)

**Bomb:EBombShutdownTime**

**Type:** Integer

Maximum time recharge is stopped on players hit with an EMP bomb

**Bomb:JitterTime**

**Type:** Integer

How long the screen jitters from a bomb hit (in ticks)

**Bomb:ProximityDistance**

**Type:** Integer

Radius of proximity trigger in tiles (each bomb level adds 1 to this amount)

**Brick:AntibrickwarpDistance**

**Type:** Integer

**Default:** 0

Squared smallest distance allowed between players and new bricks before new bricks are cancelled to prevent brickwarping. 0 disables antibrickwarp feature.

**Brick:BrickMode**

**Type:** Enumerated

**Default:** BRICK\_VIE

How bricks behave when they are dropped (BRICK\_VIE=improved vie style, BRICK\_AHEAD=drop in a line ahead of player, BRICK\_LATERAL=drop laterally across player, BRICK\_CAGE=drop 4 bricks simultaneously to create a cage)

**Brick:BrickSpan**

**Type:** Integer

**Default:** 10

The maximum length of a dropped brick.

**Brick:BrickTime**

**Type:** Integer

How long bricks last (in ticks)

**Brick:CountBricksAsWalls****Type:** Boolean**Default:** Yes

Whether bricks snap to the edges of other bricks (as opposed to only snapping to walls)

**Bullet:BulletAliveTime****Type:** Integer

How long bullets live before disappearing (in ticks)

**Bullet:BulletDamageLevel****Type:** Integer

Maximum amount of damage that a L1 bullet will cause

**Bullet:BulletDamageUpgrade****Type:** Integer

Amount of extra damage each bullet level will cause

**Bullet:ExactDamage****Type:** Boolean**Default:** No

Whether to use exact bullet damage (Cont .36+)

**Burst:BurstDamageLevel****Type:** Integer

Maximum amount of damage caused by a single burst bullet

**Chat:RestrictChat****Type:** Integer**Default:** 0

This specifies an initial chat mask for the arena. Don't use this unless you know what you're doing.

**Cost:AntiWarp****Type:** Integer**Default:** 0

Points cost for AntiWarp Ability. 0 to disallow purchase.

**Cost:Bomb****Type:** Integer**Default:** 0

Points cost for Bomb Upgrade. 0 to disallow purchase.

**Cost:Bounce****Type:** Integer**Default:** 0

Points cost for Bouncing Bullets. 0 to disallow purchase.

**Cost:Brick****Type:** Integer**Default:** 0

Points cost for Brick. 0 to disallow purchase.

**Cost:Burst****Type:** Integer**Default:** 0

Points cost for Burst. 0 to disallow purchase.

**Cost:Cloak****Type:** Integer**Default:** 0

Points cost for Cloak Ability. 0 to disallow purchase.

**Cost:Decoy****Type:** Integer**Default:** 0

Points cost for Decoy. 0 to disallow purchase.

**Cost:Energy****Type:** Integer**Default:** 0

Points cost for Energy Upgrade. 0 to disallow purchase.

**Cost:Gun****Type:** Integer**Default:** 0

Points cost for Gun Upgrade. 0 to disallow purchase.

**Cost:MultiFire****Type:** Integer**Default:** 0

Points cost for MultiFire. 0 to disallow purchase.

**Cost:Portal****Type:** Integer**Default:** 0

Points cost for Portal. 0 to disallow purchase.

**Cost:Prox****Type:** Integer**Default:** 0

Points cost for Proximity Bombs. 0 to disallow purchase.

**Cost:PurchaseAnytime****Type:** Boolean**Default:** No

Whether players can buy items outside a safe zone.

**Cost:Recharge****Type:** Integer**Default:** 0

Points cost for Recharge Upgrade. 0 to disallow purchase.

**Cost:Repel****Type:** Integer**Default:** 0

Points cost for Repel. 0 to disallow purchase.

**Cost:Rocket****Type:** Integer**Default:** 0

Points cost for Rocket. 0 to disallow purchase.

**Cost:Rotation****Type:** Integer

**Default:** 0

Points cost for Rotation Upgrade. 0 to disallow purchase.

**Cost:Shield**

**Type:** Integer

**Default:** 0

Points cost for Shields. 0 to disallow purchase.

**Cost:Shrap**

**Type:** Integer

**Default:** 0

Points cost for Shrapnel Upgrade. 0 to disallow purchase.

**Cost:Speed**

**Type:** Integer

**Default:** 0

Points cost for Top Speed. 0 to disallow purchase.

**Cost:Stealth**

**Type:** Integer

**Default:** 0

Points cost for Stealth Ability. 0 to disallow purchase.

**Cost:Super**

**Type:** Integer

**Default:** 0

Points cost for Super. 0 to disallow purchase.

**Cost:Thor**

**Type:** Integer

**Default:** 0

Points cost for Thor. 0 to disallow purchase.

**Cost:Thrust**

**Type:** Integer

**Default:** 0

Points cost for Thrust Upgrade. 0 to disallow purchase.

**Cost:XRadar**

**Type:** Integer

**Default:** 0

Points cost for XRadar. 0 to disallow purchase.

**CTF:NeutAfterKill**

**Type:** Boolean

**Default:** No

If enabled, a killed flagger drops his flag on the ground instead of resetting it. An enemy can then steal it, or a friendly can return it home by touching it.

**CTF:Team0-Name**

**Type:** String

**Default:** Team 0

The name for this freq to display in arena messages

**CTF:Team0-Region**

**Type:** Integer

The region where an enemy player can score

**CTF:Team0-X**

**Type:** Integer

**Default:** 502

The X coordinate where the flag for this team will spawn (in tiles)

**CTF:Team0-Y**

**Type:** Integer

**Default:** 512

The Y coordinate where the flag for this team will spawn (in tiles)

**CTF:Team1-Name**

**Type:** String

**Default:** Team 0

The name for this freq to display in arena messages

**CTF:Team1-Region**

**Type:** Integer

The region where an enemy player can score

**CTF:Team1-X**

**Type:** Integer

**Default:** 502

The X coordinate where the flag for this team will spawn (in tiles)

**CTF:Team1-Y**

**Type:** Integer

**Default:** 512

The Y coordinate where the flag for this team will spawn (in tiles)

**CTF:WinCaptures**

**Type:** Integer

**Default:** 3

How many flag captures a team needs to win the game

**Door:DoorDelay**

**Type:** Integer

How often doors attempt to switch their state

**Door:DoorMode**

**Type:** Integer

Door mode (-2=all doors completely random, -1=weighted random (some doors open more often than others), 0-255=fixed doors (1 bit of byte for each door specifying whether it is open or not))

**DPrizeWeight:AllWeapons**

**Type:** Integer

Likelihood of 'Super!' prize appearing

**DPrizeWeight:AntiWarp**

**Type:** Integer

Likelihood of 'AntiWarp' prize appearing

**DPrizeWeight:Bomb**

**Type:** Integer

Likelihood of 'Bomb Upgrade' prize appearing

**DPrizeWeight:BouncingBullets**

**Type:** Integer

Likelihood of 'Bouncing Bullets' prize appearing

**DPrizeWeight:Brick**

**Type:** Integer

Likelihood of 'Brick' prize appearing

**DPrizeWeight:Burst**

**Type:** Integer

Likelihood of 'Burst' prize appearing

**DPrizeWeight:Cloak**

**Type:** Integer

Likelihood of 'Cloak' prize appearing

**DPrizeWeight:Decoy**

**Type:** Integer

Likelihood of 'Decoy' prize appearing

**DPrizeWeight:Energy**

**Type:** Integer

Likelihood of 'Energy Upgrade' prize appearing

**DPrizeWeight:Glue**

**Type:** Integer

Likelihood of 'Engine Shutdown' prize appearing

**DPrizeWeight:Gun**

**Type:** Integer

Likelihood of 'Gun Upgrade' prize appearing

**DPrizeWeight:MultiFire**

**Type:** Integer

Likelihood of 'MultiFire' prize appearing

**DPrizeWeight:MultiPrize**

**Type:** Integer

Likelihood of 'Multi-Prize' prize appearing

**DPrizeWeight:Portal**

**Type:** Integer

Likelihood of 'Portal' prize appearing

**DPrizeWeight:Proximity**

**Type:** Integer

Likelihood of 'Proximity Bomb' prize appearing

**DPrizeWeight:QuickCharge**

**Type:** Integer

Likelihood of 'Recharge' prize appearing

**DPrizeWeight:Recharge**

**Type:** Integer

Likelihood of 'Full Charge' prize appearing (not 'Recharge')

**DPrizeWeight:Repel**

**Type:** Integer

Likelihood of 'Repel' prize appearing

**DPrizeWeight:Rocket**

**Type:** Integer

Likelihood of 'Rocket' prize appearing

**DPrizeWeight:Rotation**

**Type:** Integer

Likelihood of 'Rotation' prize appearing

**DPrizeWeight:Shields**

**Type:** Integer

Likelihood of 'Shields' prize appearing

**DPrizeWeight:Shrapnel**

**Type:** Integer

Likelihood of 'Shrapnel Upgrade' prize appearing

**DPrizeWeight:Stealth**

**Type:** Integer

Likelihood of 'Stealth' prize appearing

**DPrizeWeight:Thor**

**Type:** Integer

Likelihood of 'Thor' prize appearing

**DPrizeWeight:Thruster**

**Type:** Integer

Likelihood of 'Thruster' prize appearing

**DPrizeWeight:TopSpeed**

**Type:** Integer

Likelihood of 'Speed' prize appearing

**DPrizeWeight:Warp**

**Type:** Integer

Likelihood of 'Warp' prize appearing

**DPrizeWeight:XRadars**

**Type:** Integer

Likelihood of 'XRadars' prize appearing

**Flag:CarryFlags**

**Type:** Integer

Whether the flags can be picked up and carried (0=no, 1=yes, 2=yes-one at a time, 3=yes-two at a time, 4=three, etc..)

**Flag:DropCenter**

**Type:** Boolean

**Default:** No

Whether flags dropped normally go in the center of the map, as opposed to near the player.

**Flag:DropOwned**

**Type:** Boolean

**Default:** Yes

Whether flags you drop are owned by your team.

**Flag:DropRadius**

**Type:** Integer



**Default:** 2

How far from a player do dropped flags appear (in tiles).

**Flag:EnterGameFlaggingDelay**

**Type:** Integer

Time a new player must wait before they are allowed to see flags

**Flag:FlagBlankDelay**

**Type:** Integer

Amount of time that a user can get no data from server before flags are hidden from view for 10 seconds

**Flag:FlagCount**

**Type:** Other

**Default:** 0

**Range:** 0-256

How many flags are present in this arena.

**Flag:FlagDropDelay**

**Type:** Integer

Time before flag is dropped by carrier (0=never)

**Flag:FlagDropResetReward**

**Type:** Integer

Minimum kill reward that a player must get in order to have his flag drop timer reset

**Flag:FlaggerBombFireDelay**

**Type:** Integer

Delay given to flaggers for firing bombs (zero is ships normal firing rate) (do not set this number less than 20)

**Flag:FlaggerBombUpgrade**

**Type:** Boolean

Whether the flaggers get a bomb upgrade

**Flag:FlaggerDamagePercent**

**Type:** Integer

Percentage of normal damage received by flaggers (in 0.1%)

**Flag:FlaggerFireCostPercent**

**Type:** Integer

Percentage of normal weapon firing cost for flaggers (in 0.1%)

**Flag:FlaggerGunUpgrade**

**Type:** Boolean

Whether the flaggers get a gun upgrade

**Flag:FlaggerKillMultiplier**

**Type:** Integer

Number of times more points are given to a flagger (1 = double points, 2 = triple points)

**Flag:FlaggerOnRadar**

**Type:** Boolean

Whether the flaggers appear on radar in red

**Flag:FlaggerSpeedAdjustment**

**Type:** Integer

Amount of speed adjustment player carrying flag gets (negative numbers mean slower)

**Flag:FlaggerThrustAdjustment**

**Type:** Integer

Amount of thrust adjustment player carrying flag gets (negative numbers mean less thrust)

**Flag:FlagReward**

**Type:** Integer

**Requires module:** points\_flag

**Default:** 5000

The basic flag reward is calculated as  $(\text{players in arena})^2 * \text{reward} / 1000$ .

**Flag:FriendlyTransfer**

**Type:** Boolean

**Default:** Yes

Whether you get a teammates flags when you kill him.

**Flag:NeutCenter**

**Type:** Boolean

**Default:** No

Whether flags that are neut-dropped go in the center, as opposed to near the player who dropped them.

**Flag:NeutOwned**

**Type:** Boolean

**Default:** No

Whether flags you neut-drop are owned by your team.

**Flag:NoDataFlagDropDelay**

**Type:** Integer

Amount of time that a user can get no data from server before flags he is carrying are dropped

**Flag:ResetDelay**

**Type:** Integer

**Default:** 0

The length of the delay between flag games.

**Flag:SafeCenter**

**Type:** Boolean

**Default:** No

Whether flags dropped from a safe zone spawn in the center, as opposed to near the safe zone player.

**Flag:SafeOwned**

**Type:** Boolean

**Default:** Yes

Whether flags dropped from a safe zone are owned by your team, as opposed to neutral.

**Flag:SpawnRadius**

**Type:** Integer

**Default:** 50

How far from the spawn center that new flags spawn (in tiles).

**Flag:SpawnX**

**Type:** Integer

**Default:** 512

The X coordinate that new flags spawn at (in tiles).

**Flag:SpawnY**

**Type:** Integer

**Default:** 512

The Y coordinate that new flags spawn at (in tiles).

**Flag:SplitPoints**

**Type:** Boolean

**Default:** No

Whether to split a flag reward between the members of a freq or give them each the full amount.

**Flag:TeamChangeGrace**

**Type:** Integer

**Default:** 1000

Period of time during which players are allowed to switch back to the winning team after leaving it.

**Flag:TKCenter**

**Type:** Boolean

**Default:** No

Whether flags dropped by a team-kill spawn in the center, as opposed to near the killed player.

**Flag:TKOwned**

**Type:** Boolean

**Default:** Yes

Whether flags dropped by a team-kill are owned by your team, as opposed to neutral.

**Flag:WinDelay**

**Type:** Integer

**Default:** 200

The delay between dropping the last flag and winning (ticks).

**GameCredits:GoalCreditsEnabled**

**Type:** Integer

**Default:** 0

**Range:** 0-1

Enable credits for goals; Note that this requires a module calling credits->Goal(...)

**GameCredits:GoalCreditsPerPlaying**

**Type:** Integer

**Default:** 0

**GameCredits:GoalExtraCredits**

**Type:** Integer

**Default:** 500

**GameCredits:GoalMaximumCredits**

**Type:** Integer

**Default:** 10000

**GameCredits:goalScorerBonusMultiplier**

**Type:** Integer

**Default:** 0

**GameCredits:GoalSplit**

Type: Integer

Default: 0

Range: 0-1

**GameCredits:KillAnnounce**

Type: Integer

Default: 0

Range: 0-1

Send arena messages for kills

**GameCredits:KillCreditsEnabled**

Type: Integer

Default: 0

Range: 0-1

Do players receive credits for kills

**GameCredits:KillCreditsShareBetweenFreqs**

Type: Integer

Default: 1

Range: 0-1

If you kill someone, do you get a portion of the credits for it on your other freqs

**GameCredits:KillerSpreeCredits**

Type: Integer

Default: 0

When the killed is on a spree, the killer receives this much credits extra for every kill

**GameCredits:KillerTeammateCreditsMultiplier**

Type: Integer

Default: 0

The maximum distance a teammate receives credits for kills

**GameCredits:KillExtraCredits**

Type: Integer

Default: 100

Credits added to every kill

**GameCredits:KillMaximumCredits**

Type: Integer

Default: 20000

Maximum Credits you can receive for a kill

**GameCredits:LVZEnabled**

Type: Integer

Default: 10

The maximum number of digits in the lvz

**GameCredits:LVZImageStart**

Type: Integer

Default: -1

**GameCredits:LVZObjectStart**

Type: Integer

Default: -1

**GameCredits:MaxCredits****Type:** Integer**Default:** 10000

The maximum amount of credits someone may have; The credits go to his team members when full

**GameCredits:OreCalculatePer****Type:** Integer**Default:** 100

When calculating ore reward, divide it by this

**GameCredits:OreCreditsEnabled****Type:** Integer**Default:** 0**Range:** 0-1

Enable credits for ore mining; Note that this requires a module calling credits->Ore(...)

**GameCredits:OreCreditsPerPlaying****Type:** Integer**Default:** 0**GameCredits:OreExtraCredits****Type:** Integer**Default:** 500**GameCredits:OreMaximumCredits****Type:** Integer**Default:** 10000**GameCredits:OreScorerBonusPromille****Type:** Integer**Default:** 0

How much more credits does the scorer get then his team members

**GameCredits:OreSplit****Type:** Integer**Default:** 0**Range:** 0-1**GameCredits:OtherFreqGainPermille****Type:** Integer**Default:** 200

How many credits (in permille) you get for freqs you are not on

**General:DesiredPlaying****Type:** Integer**Requires module:** ap\_multipub**Default:** 15

This controls when the server will create new public arenas.

**General:ExtraLevelFilesKeys**

**Type:** Integer

**Default:** 0

**Range:** 0-15

How many extra keys (LevelFiles1, LevelFiles2, etc.) to use to generate the final arena LVZ list.

**General:LevelFiles**

**Type:** String

The main LVZ list for the arena.

**General:Map**

**Type:** String

The name of the level file for this arena.

**General:MaxPlaying**

**Type:** Integer

**Default:** 100

This is the most players that will be allowed to play in the arena at once. Zero means no limit.

**General:NeedCap**

**Type:** String

**Requires module:** arenaperm

If this setting is present for an arena, any player entering the arena must have the capability specified this setting. This can be used to restrict arenas to certain groups of players.

**General:ScoreGroup**

**Type:** String

If this is set, it will be used as the score identifier for shared scores for this arena (unshared scores, e.g. per-game scores, always use the arena name as the identifier). Setting this to the same value in several different arenas will cause them to share scores.

**Kill:BountyIncreaseForKill**

**Type:** Integer

Number of points added to players bounty each time he kills an opponent

**Kill:EnterDelay**

**Type:** Integer

How long after a player dies before he can re-enter the game (in ticks)

**Kill:FixedKillReward**

**Type:** Integer

**Default:** -1

If -1 use the bounty of the killed player to calculate kill reward. Otherwise use this fixed value  
\*/

**Kill:FlagMinimumBounty**

**Type:** Integer

**Default:** 0

The minimum bounty the killing player must have to get any bonus kill points for flags transferred, carried or owned.

**Kill:JackpotBountyPercent**

**Type:** Integer

**Default:** 0

The percent of a player's bounty added to the jackpot on each kill. Units: 0.1%.

**Kill:MaxBonus**

**Type:** Integer

FIXME: fill this in

**Kill:MaxPenalty**

**Type:** Integer

FIXME: fill this in

**Kill:PointsPerCarriedFlag**

**Type:** Integer

**Default:** 0

The number of extra points to give for each flag the killing player is carrying. Note that flags that were transferred to the killer as part of the kill are counted here, so adjust PointsPerKilledFlag accordingly.

**Kill:PointsPerKilledFlag**

**Type:** Integer

**Default:** 100

The number of extra points to give for each flag a killed player was carrying. Note that the flags don't actually have to be transferred to the killer to be counted here.

**Kill:PointsPerTeamFlag**

**Type:** Integer

**Default:** 0

The number of extra points to give for each flag owned by the killing team. Note that flags that were transferred to the killer as part of the kill are counted here, so adjust PointsPerKilledFlag accordingly.

**Kill:RewardBase**

**Type:** Integer

FIXME: fill this in

**Lag:C2SLossToDisallowFlags**

**Type:** Integer

**Default:** 50

The C2S packetloss when a player isn't allowed to pick up flags or balls. Units 0.1%.

**Lag:C2SLossToSpec**

**Type:** Integer

**Default:** 150

The C2S packetloss at which to force a player to spec. Units 0.1%.

**Lag:PingToDisallowFlags**

**Type:** Integer

**Default:** 500

The average ping when a player isn't allowed to pick up flags or balls.

**Lag:PingToIgnoreAllWeapons**

**Type:** Integer

**Default:** 1000

The average ping when all weapons should be ignored.

**Lag:PingToSpec**

**Type:** Integer

**Default:** 600

The average ping at which to force a player to spec.

**Lag:PingToStartIgnoringWeapons**

**Type:** Integer

**Default:** 300

The average ping to start ignoring weapons at.

**Lag:S2CLossToDisallowFlags**

**Type:** Integer

**Default:** 50

The S2C packetloss when a player isn't allowed to pick up flags or balls. Units 0.1%.

**Lag:S2CLossToIgnoreAllWeapons**

**Type:** Integer

**Default:** 500

The S2C packetloss when all weapons should be ignored. Units 0.1%.

**Lag:S2CLossToSpec**

**Type:** Integer

**Default:** 150

The S2C packetloss at which to force a player to spec. Units 0.1%.

**Lag:S2CLossToStartIgnoringWeapons**

**Type:** Integer

**Default:** 40

The S2C packetloss to start ignoring weapons at. Units 0.1%.

**Lag:SpikeToSpec**

**Type:** Integer

**Default:** 3000

The amount of time the server can get no data from a player before forcing him to spectator mode (in ms).

**Lag:WeaponLossToDisallowFlags**

**Type:** Integer

**Default:** 50

The weapon packetloss when a player isn't allowed to pick up flags or balls. Units 0.1%.

**Lag:WeaponLossToIgnoreAllWeapons**

**Type:** Integer

**Default:** 500

The weapon packetloss when all weapons should be ignored. Units 0.1%.

**Lag:WeaponLossToSpec**

**Type:** Integer

**Default:** 150

The weapon packetloss at which to force a player to spec. Units 0.1%.

**Lag:WeaponLossToStartIgnoringWeapons**

**Type:** Integer

**Default:** 40

The weapon packetloss to start ignoring weapons at. Units 0.1%.

**Latency:ClientSlowPacketSampleSize**

**Type:** Integer

Number of packets to sample S2C before checking for kickout

**Latency:ClientSlowPacketTime**

**Type:** Integer

Amount of latency S2C that constitutes a slow packet



**Latency:S2CNoDataKickoutDelay****Type:** Integer

Amount of time a user can receive no data from server before connection is terminated

**Latency:SendRoutePercent****Type:** Integer

Percentage of the ping time that is spent on the C2S portion of the ping (used in more accurately synchronizing clocks)

**Legalship:ArenaMask****Type:** Integer**Requires module:** enf\_legalship**Default:** 255**Range:** 0-255

The ship mask of allowed ships in the arena. 1=warbird, 2=javelin, etc.

**Legalship:Freq0Mask****Type:** Integer**Requires module:** enf\_legalship**Default:** 255**Range:** 0-255

The ship mask allowed on freq 0. Ships must also be allowed on the arena mask. You can define a mask for any freq (FreqXMask).

**Legalship:Freq1Mask****Type:** Integer**Requires module:** enf\_legalship**Default:** 255**Range:** 0-255

The ship mask allowed on freq 1. Ships must also be allowed on the arena mask. You can define a mask for any freq (FreqXMask).

**log\_staff:commands****Type:** String**Requires module:** log\_staff**Default:** 'warn kick setcm'

A list of commands that trigger messages to all logged-in staff.

**Message:AllowAudioMessages****Type:** Boolean

Whether players can send audio messages

**Mine:MineAliveTime****Type:** Integer

Time that mines are active (in ticks)

**Mine:TeamMaxMines****Type:** Integer

Maximum number of mines allowed to be placed by an entire team

**Misc:ActivateAppShutdownTime****Type:** Integer

Amount of time a ship is shutdown after application is reactivated

**Misc:AllowSavedShips****Type:** Integer

Whether saved ships are allowed (do not allow saved ship in zones where sub-arenas may have

differing parameters)

**Misc:AntiwarpFlagShipChange**

**Type:** Integer

prevents players with flags from changing ships while antiwarped.

**Misc:AntiWarpSettleDelay**

**Type:** Integer

How many ticks to activate a fake antiwarp after attaching, portaling, or warping.

**Misc:AntiwarpShipChange**

**Type:** Integer

prevents players without flags from changing ships while antiwarped.

**Misc:BounceFactor**

**Type:** Integer

How bouncy the walls are (16 = no speed loss)

**Misc:DecoyAliveTime**

**Type:** Integer

Time a decoy is alive (in ticks)

**Misc:DisableScreenshot**

**Type:** Boolean

**Default:** No

Whether to disable Continuum's screenshot feature (Cont .37+)

**Misc:ExtraPositionData**

**Type:** Integer

Whether regular players receive sysop data about a ship

**Misc:FrequencyShift**

**Type:** Integer

Amount of random frequency shift applied to sounds in the game

**Misc:GreetMessage**

**Type:** String

The message to send to each player on entering the arena.

**Misc:MaxResArea**

**Type:** Integer

**Default:** 0

Maximum screen area (x\*y) allowed in the arena, Zero means no limit.

**Misc:MaxXres**

**Type:** Integer

**Default:** 0

Maximum screen width allowed in the arena. Zero means no limit.

**Misc:MaxYres**

**Type:** Integer

**Default:** 0

Maximum screen height allowed in the arena. Zero means no limit.

**Misc:NearDeathLevel**

**Type:** Integer

Amount of energy that constitutes a near-death experience (ships bounty will be decreased by

1 when this occurs – used for dueling zone)

**Misc:NoSafeAntiwarp**

**Type:** Integer

**Default:** 0

Disables antiwarp on players in safe zones.

**Misc:PeriodicMessage0**

**Type:** String

10 20 periodic message. 10 is the interval and 20 is the initial delay (in minutes)

**Misc:PeriodicMessage1**

**Type:** String

10 20 periodic message. 10 is the interval and 20 is the initial delay (in minutes)

**Misc:PeriodicMessage2**

**Type:** String

10 20 periodic message. 10 is the interval and 20 is the initial delay (in minutes)

**Misc:PeriodicMessage3**

**Type:** String

10 20 periodic message. 10 is the interval and 20 is the initial delay (in minutes)

**Misc:PeriodicMessage4**

**Type:** String

10 20 periodic message. 10 is the interval and 20 is the initial delay (in minutes)

**Misc:PeriodicMessage5**

**Type:** String

10 20 periodic message. 10 is the interval and 20 is the initial delay (in minutes)

**Misc:PeriodicMessage6**

**Type:** String

10 20 periodic message. 10 is the interval and 20 is the initial delay (in minutes)

**Misc:PeriodicMessage7**

**Type:** String

10 20 periodic message. 10 is the interval and 20 is the initial delay (in minutes)

**Misc:PeriodicMessage8**

**Type:** String

10 20 periodic message. 10 is the interval and 20 is the initial delay (in minutes)

**Misc:PeriodicMessage9**

**Type:** String

10 20 periodic message. 10 is the interval and 20 is the initial delay (in minutes)

**Misc:RegionCheckInterval**

**Type:** Integer

**Default:** 100

How often to check for region enter/exit events (in ticks).

**Misc:SafetyLimit**

**Type:** Integer

Amount of time that can be spent in the safe zone (in ticks)

**Misc:SeeEnergy**

**Type:** Enumerated

**Default:** SEE\_NONE

Whose energy levels everyone can see: SEE\_NONE means nobody else's, SEE\_ALL is everyone's, SEE\_TEAM is only teammates.

**Misc:SelfScoreReset**

**Type:** Boolean

**Default:** No

Whether players can reset their own scores using ?scorereset. \*/

**Misc:SendPositionDelay**

**Type:** Integer

Amount of time between position packets sent by client

**Misc:SheepMessage**

**Type:** String

The message that appears when someone says ?sheep

**Misc:ShipChangeInterval**

**Type:** Integer

**Default:** 500

The allowable interval between player ship changes, in ticks.

**Misc:SlowFrameCheck**

**Type:** Integer

Whether to check for slow frames on the client (possible cheat technique) (flawed on some machines, do not use)

**Misc:SpecSeeEnergy**

**Type:** Enumerated

**Default:** SEE\_NONE

Whose energy levels spectators can see. The options are the same as for Misc:SeeEnergy, with one addition: SEE\_SPEC means only the player you're spectating.

**Misc:SpecSeeExtra**

**Type:** Boolean

**Default:** Yes

Whether spectators can see extra data for the person they're spectating.

**Misc:TeamKillPoints**

**Type:** Boolean

**Default:** No

Whether points are awarded for a team-kill.

**Misc:TickerDelay**

**Type:** Integer

Amount of time between ticker help messages

**Misc:TimedGame**

**Type:** Integer

**Default:** 0

How long the game timer lasts (in ticks). Zero to disable.

**Misc:VictoryMusic**

**Type:** Integer

Whether the zone plays victory music or not

**Misc:WarpPointDelay**

**Type:** Integer

How long a portal is active

**Misc:WarpRadiusLimit**

**Type:** Integer

When ships are randomly placed in the arena, this parameter will limit how far from the center of the arena they can be placed (1024=anywhere)

**Misc:WarpTresholdDelta**

**Type:** Integer

**Default:** 320

The amount of change in a players position that is considered a warp (only while he is flashing). value is in pixels

**Modules:AttachModules**

**Type:** String

This is a list of modules that you want to take effect in this arena. Not all modules need to be attached to arenas to function, but some do.

**Net:AntiWarpSendPercent**

**Type:** Integer

**Default:** 5

Percent of position packets with antiwarp enabled to send to the whole arena.

**Net:BulletPixels**

**Type:** Integer

**Default:** 1500

How far away to always send bullets (in pixels)

**Net:PositionExtraPixels**

**Type:** Integer

**Default:** 8000

How far away to send positions of players on radar

**Net:WeaponPixels**

**Type:** Integer

**Default:** 2000

How far away to always send weapons (in pixels)

**Periodic:RewardDelay**

**Type:** Integer

**Default:** 0

The interval between periodic rewards (in ticks). Zero to disable.

**Periodic:RewardMinimumPlayers**

**Type:** Integer

**Default:** 0

The minimum players necessary in the arena to give out periodic rewards.

**Periodic:RewardPoints**

**Type:** Integer

**Requires module:** points\_periodic

**Default:** 100

Periodic rewards are calculated as follows: If this setting is positive, you get this many points per flag. If it's negative, you get it's absolute value points per flag, times the number of players in the arena.

**Periodic:SendZeroRewards****Type:** Integer**Default:** 1

Whether frequencies with zero points will still get a reward notification during the ding.

**Prize:DeathPrizeTime****Type:** Integer

How long the prize exists that appears after killing somebody

**Prize:DontShareBrick****Type:** Boolean**Default:** No

Whether Brick greens don't go to the whole team.

**Prize:DontShareBurst****Type:** Boolean**Default:** No

Whether Burst greens don't go to the whole team.

**Prize:DontShareThor****Type:** Boolean**Default:** No

Whether Thor greens don't go to the whole team.

**Prize:EngineShutdownTime****Type:** Integer

Time the player is affected by an 'Engine Shutdown' Prize (in ticks)

**Prize:MinimumVirtual****Type:** Integer

Distance from center of arena that prizes/flags/soccer-balls will spawn

**Prize:MultiPrizeCount****Type:** Integer

Number of random greens given with a MultiPrize

**Prize:PrizeDelay****Type:** Integer

How often prizes are regenerated (in ticks)

**Prize:PrizeFactor****Type:** Integer

Number of prizes hidden is based on number of players in game. This number adjusts the formula, higher numbers mean more prizes. (Note: 10000 is max, 10 greens per person)

**Prize:PrizeHideCount****Type:** Integer

Number of prizes that are regenerated every PrizeDelay

**Prize:PrizeMaxExist****Type:** Integer

Maximum amount of time that a hidden prize will remain on screen. (actual time is random)

**Prize:PrizeMinExist****Type:** Integer

Minimum amount of time that a hidden prize will remain on screen. (actual time is random)

**Prize:PrizeNegativeFactor****Type:** Integer

Odds of getting a negative prize. (1 = every prize, 32000 = extremely rare)

**Prize:TakePrizeReliable****Type:** Integer

Whether prize packets are sent reliably (C2S)

**Prize:TeamkillPrize****Type:** Integer**Default:** 0

The prize # to give for a teamkill, if Prize:UseTeamkillPrize=1.

**Prize:UpgradeVirtual****Type:** Integer

Amount of additional distance added to MinimumVirtual for each player that is in the game

**Prize:UseDeathPrizeWeights****Type:** Boolean**Default:** No

Whether to use the DPrizeWeight section for death prizes instead of the PrizeWeight section.

**Prize:UseTeamkillPrize****Type:** Integer**Default:** 0

Whether to use a special prize for teamkills. Prize:TeamkillPrize specifies the prize #.

**PrizeWeight:AllWeapons****Type:** Integer

Likelihood of 'Super!' prize appearing

**PrizeWeight:AntiWarp****Type:** Integer

Likelihood of 'AntiWarp' prize appearing

**PrizeWeight:Bomb****Type:** Integer

Likelihood of 'Bomb Upgrade' prize appearing

**PrizeWeight:BouncingBullets****Type:** Integer

Likelihood of 'Bouncing Bullets' prize appearing

**PrizeWeight:Brick****Type:** Integer

Likelihood of 'Brick' prize appearing

**PrizeWeight:Burst****Type:** Integer

Likelihood of 'Burst' prize appearing

**PrizeWeight:Cloak****Type:** Integer

Likelihood of 'Cloak' prize appearing

**PrizeWeight:Decoy****Type:** Integer

Likelihood of 'Decoy' prize appearing

**PrizeWeight:Energy**

**Type:** Integer

Likelihood of 'Energy Upgrade' prize appearing

**PrizeWeight:Glue**

**Type:** Integer

Likelihood of 'Engine Shutdown' prize appearing

**PrizeWeight:Gun**

**Type:** Integer

Likelihood of 'Gun Upgrade' prize appearing

**PrizeWeight:MultiFire**

**Type:** Integer

Likelihood of 'MultiFire' prize appearing

**PrizeWeight:MultiPrize**

**Type:** Integer

Likelihood of 'Multi-Prize' prize appearing

**PrizeWeight:Portal**

**Type:** Integer

Likelihood of 'Portal' prize appearing

**PrizeWeight:Proximity**

**Type:** Integer

Likelihood of 'Proximity Bomb' prize appearing

**PrizeWeight:QuickCharge**

**Type:** Integer

Likelihood of 'Recharge' prize appearing

**PrizeWeight:Recharge**

**Type:** Integer

Likelihood of 'Full Charge' prize appearing (not 'Recharge')

**PrizeWeight:Repel**

**Type:** Integer

Likelihood of 'Repel' prize appearing

**PrizeWeight:Rocket**

**Type:** Integer

Likelihood of 'Rocket' prize appearing

**PrizeWeight:Rotation**

**Type:** Integer

Likelihood of 'Rotation' prize appearing

**PrizeWeight:Shields**

**Type:** Integer

Likelihood of 'Shields' prize appearing

**PrizeWeight:Shrapnel**

**Type:** Integer

Likelihood of 'Shrapnel Upgrade' prize appearing



**PrizeWeight:Stealth**

**Type:** Integer

Likelihood of 'Stealth' prize appearing

**PrizeWeight:Thor**

**Type:** Integer

Likelihood of 'Thor' prize appearing

**PrizeWeight:Thruster**

**Type:** Integer

Likelihood of 'Thruster' prize appearing

**PrizeWeight:TopSpeed**

**Type:** Integer

Likelihood of 'Speed' prize appearing

**PrizeWeight:Warp**

**Type:** Integer

Likelihood of 'Warp' prize appearing

**PrizeWeight:XRadar**

**Type:** Integer

Likelihood of 'XRadar' prize appearing

**Radar:MapZoomFactor**

**Type:** Integer

A number representing how much the map is zoomed out for radar. (48 = whole map on radar, 49+ = effectively disable radar)

**Radar:RadarMode**

**Type:** Integer

Radar mode (0=normal, 1=half/half, 2=quarters, 3=half/half-see team mates, 4=quarters-see team mates)

**Radar:RadarNeutralSize**

**Type:** Integer

Size of area between blinded radar zones (in pixels)

**Repel:RepelDistance**

**Type:** Integer

Number of pixels from the player that are affected by a repel

**Repel:RepelSpeed**

**Type:** Integer

Speed at which players are repelled

**Repel:RepelTime**

**Type:** Integer

Time players are affected by the repel (in ticks)

**Rocket:RocketSpeed**

**Type:** Integer

Speed value given while a rocket is active

**Rocket:RocketThrust**

**Type:** Integer

Thrust value given while a rocket is active

**Security:MaxDeathWithoutFiring**

**Type:** Integer

**Default:** 5

The number of times a player can die without firing a weapon before being placed in spectator mode.

**Shrapnel:InactiveShrapDamage**

**Type:** Integer

Amount of damage shrapnel causes in it's first 1/4 second of life

**Shrapnel:Random**

**Type:** Boolean

Whether shrapnel spreads in circular or random patterns

**Shrapnel:ShrapnelDamagePercent**

**Type:** Integer

Percentage of normal damage applied to shrapnel (relative to bullets of same level) (in 0.1%)

**Shrapnel:ShrapnelSpeed**

**Type:** Integer

Speed that shrapnel travels

**Soccer:AllowBombs**

**Type:** Boolean

Whether the ball carrier can fire his bombs

**Soccer:AllowGoalByDeath**

**Type:** Boolean

**Default:** No

Whether a goal is scored if a player dies carrying the ball on a goal tile.

**Soccer:AllowGuns**

**Type:** Boolean

Whether the ball carrier can fire his guns

**Soccer:BallBlankDelay**

**Type:** Integer

Amount of time a player can receive no data from server and still pick up the soccer ball

**Soccer:BallBounce**

**Type:** Boolean

Whether the ball bounces off walls

**Soccer:BallCount**

**Type:** Integer

**Default:** 0

The number of balls in this arena.

**Soccer:BallLocation**

**Type:** Boolean

Whether the balls location is displayed at all times or not

**Soccer:CapturePoints**

**Type:** Integer

**Default:** 1

If positive, these points are distributed to each goal/team. When you make a goal, the points get transferred to your goal/team. If one team gets all the points, then they win as well. If

negative, teams are given 1 point for each goal, first team to reach -CapturePoints points wins the game.

**Soccer:DisableBallKilling**

**Type:** Boolean

**Default:** No

Whether to disable ball killing in safe zones (Cont .38+)

**Soccer:DisableWallPass**

**Type:** Boolean

**Default:** No

Whether to disable ball-passing through walls (Cont .38+)

**Soccer:GoalDelay**

**Type:** Integer

**Default:** 0

How long after a goal before the ball appears (in ticks).

**Soccer:KillerIgnorePassDelay**

**Type:** Integer

**Default:** 0

How much 'pass delay' should be trimmed off for someone killing a ball carrier.

**Soccer:MinPlayers**

**Type:** Integer

**Default:** 0

The minimum number of players who must be playing for soccer points to be awarded.

**Soccer:MinTeams**

**Type:** Integer

**Default:** 0

The minimum number of teams that must exist for soccer points to be awarded.

**Soccer:Mode**

**Type:** Enumerated

Goal configuration (\$GOAL\_ALL, \$GOAL\_LEFTRIGHT, \$GOAL\_TOPBOTTOM, \$GOAL\_CORNERS\_3\_1, \$GOAL\_CORNERS\_1\_3, \$GOAL\_SIDES\_3\_1, \$GOAL\_SIDES\_1\_3)

**Soccer:NewGameDelay**

**Type:** Integer

**Default:** -3000

How long to wait between games. If this is negative, the actual delay is random, between zero and the absolute value. Units: ticks.

**Soccer:PassDelay**

**Type:** Integer

How long after the ball is fired before anybody can pick it up (in ticks)

**Soccer:Reward**

**Type:** Integer

**Default:** 0

Negative numbers equal absolute points given, positive numbers use FlagReward formula.

**Soccer:SendTime**

**Type:** Integer

**Default:** 100

**Range:** 25-500

How often the server sends ball positions (in ticks).

**Soccer:SpawnRadius**

**Type:** Integer

**Default:** 20

How far from the spawn center the ball can spawn (in tiles).

**Soccer:SpawnX**

**Type:** Integer

**Default:** 512

**Range:** 0-1023

The X coordinate that the ball spawns at (in tiles).

**Soccer:SpawnX/Y/RadiusN**

**Type:** Integer

**Default:** -1

The spawn coordinates and radius for balls other than the first one. N goes from 1 to 7 (0 is take care of by the settings without a number). If there are more balls than spawns defined, the latter balls will repeat the first spawns in order. For example, with 3 spawns, the fourth ball uses the first spawn, the fifth ball uses the second. If only part of a spawn is undefined, that part will default to the first spawn's setting.

**Soccer:SpawnY**

**Type:** Integer

**Default:** 512

**Range:** 0-1023

The Y coordinate that the ball spawns at (in tiles).

**Soccer:UseFlagger**

**Type:** Boolean

If player with soccer ball should use the Flag:Flagger\* ship adjustments or not

**Soccer:WinBy**

**Type:** Integer

**Default:** 0

Have to beat other team by this many goals

**Spawn:Team0-Radius**

**Type:** Integer

**Default:** 0

How large of a circle from the center point freq 0 can start. (Cont .38+)

**Spawn:Team0-X**

**Type:** Integer

**Default:** 0

If set to a value, this is the X coordinate for the center point where freq 0 will start. (Cont .38+)

**Spawn:Team0-Y**

**Type:** Integer

**Default:** 0

If set to a value, this is the Y coordinate for the center point where freq 0 will start. (Cont .38+)

**Spawn:Team1-Radius**

**Type:** Integer

**Default:** 0

How large of a circle from the center point freq 1 can start. (Cont .38+)

**Spawn:Team1-X****Type:** Integer**Default:** 0

If set to a value, this is the X coordinate for the center point where freq 1 will start. (Cont .38+)

**Spawn:Team1-Y****Type:** Integer**Default:** 0

If set to a value, this is the Y coordinate for the center point where freq 1 will start. (Cont .38+)

**Spawn:Team2-Radius****Type:** Integer**Default:** 0

How large of a circle from the center point freq 2 can start. NOTE: if the Team2 settings are 0, Team0 will apply to evens and Team1 will apply to odds. (Cont .38+)

**Spawn:Team2-X****Type:** Integer**Default:** 0

If set to a value, this is the X coordinate for the center point where freq 2 will start. NOTE: if the Team2 settings are 0, Team0 will apply to evens and Team1 will apply to odds. (Cont .38+)

**Spawn:Team2-Y****Type:** Integer**Default:** 0

If set to a value, this is the Y coordinate for the center point where freq 2 will start. NOTE: if the Team2 settings are 0, Team0 will apply to evens and Team1 will apply to odds. (Cont .38+)

**Spawn:Team3-Radius****Type:** Integer**Default:** 0

How large of a circle from the center point freq 3 can start. NOTE: Repeats, freq 4 will use Team0's, freq 5 will use Team1's, etc. (Cont .38+)

**Spawn:Team3-X****Type:** Integer**Default:** 0

If set to a value, this is the X coordinate for the center point where freq 3 will start. NOTE: Repeats, freq 4 will use Team0's, freq 5 will use Team1's, etc. (Cont .38+)

**Spawn:Team3-Y****Type:** Integer**Default:** 0

If set to a value, this is the Y coordinate for the center point where freq 3 will start. NOTE: Repeats, freq 4 will use Team0's, freq 5 will use Team1's, etc. (Cont .38+)

**Spectator:HideFlags****Type:** Boolean**Default:** No

Whether to show dropped flags to spectators (Cont .36+)

**Spectator:NoXRadar****Type:** Boolean**Default:** No

Whether spectators are disallowed from having X radar (Cont .36+)

**Team:AllowFreqOwners**

**Type:** Boolean

**Default:** Yes

Whether to enable the freq ownership feature in this arena.

**Team:BalancedAgainstEnd**

**Type:** Integer

**Default:** 0

Freqs  $\geq$  BalancedAgainstStart and  $<$  BalancedAgainstEnd will be checked for balance even when players are not changing to or from these freqs. Set End  $<$  Start to disable this check.

**Team:BalancedAgainstStart**

**Type:** Integer

**Default:** 1

Freqs  $\geq$  BalancedAgainstStart and  $<$  BalancedAgainstEnd will be checked for balance even when players are not changing to or from these freqs. Set End  $<$  Start to disable this check.

**Team:DesiredTeams**

**Type:** Integer

**Default:** 2

The number of teams that the freq balancer will form as players enter.

**Team:DisallowTeamSpectators**

**Type:** Boolean

**Default:** No

If players are allowed to spectate outside of the spectator frequency.

**Team:ForceEvenTeams**

**Type:** Integer

**Default:** 0

Whether the default balancer will enforce even teams. Does not apply if a custom balancer module is used.

**Team:IncludeSpectators**

**Type:** Boolean

**Default:** No

Whether to include spectators when enforcing maximum freq sizes.

**Team:InitialSpec**

**Type:** Boolean

**Default:** No

If players entering the arena are always assigned to spectator mode.

**Team:MaxFrequency**

**Type:** Integer

**Default:** 10000

**Range:** 1-10000

One more than the highest frequency allowed. Set this below PrivFreqStart to disallow private freqs.

**Team:MaxPerPrivateTeam**

**Type:** Integer

**Default:** 1000

The maximum number of players on a private freq. Zero means these teams are not accessible.

**Team:MaxPerTeam**

**Type:** Integer

**Default:** 1000

The maximum number of players on a public freq. Zero means these teams are not accessible

**Team:MaxTeamDifference**

**Type:** Integer

**Default:** 1

How many players difference the balancer should tolerate. Does not apply if a custom balancer module is used.

**Team:PrivFreqStart**

**Type:** Integer

**Default:** 100

**Range:** 0-9999

Freqs above this value are considered private freqs.

**Team:RequiredTeams**

**Type:** Integer

**Default:** 0

The number of teams that the freq manager will keep in memory. Must be at least as high as RequiredTeams.

**Team:SpectatorFrequency**

**Type:** Integer

**Default:** 8025

**Range:** 0-9999

The frequency that spectators are assigned to, by default.

**Toggle:AntiWarpPixels**

**Type:** Integer

Distance Anti-Warp affects other players (in pixels) (note: enemy must also be on radar)

**TurfReward:MinFlags**

**Type:** Integer

**Default:** 1

The minimum number of flags needed to be owned by a freq for that team to be eligible to receive points.

**TurfReward:MinFlagsPercent**

**Type:** Integer

**Default:** 0

The minimum percent of flags needed to be owned by a freq for that team to be eligible to receive points. (ex. 18532 means 18.532%)

**TurfReward:MinPercent**

**Type:** Integer

**Default:** 0

The minimum percent of points needed to be owned by a freq for that team to be eligible to receive points. (ex. 18532 means 18.532%)

**TurfReward:MinPlayersArena**

**Type:** Integer

**Default:** 6

The minimum number of players needed in the arena for anyone to be eligible to receive points.

**TurfReward:MinPlayersTeam**

**Type:** Integer

**Default:** 3

The minimum number of players needed on a team for players on that team to be eligible to

recieve points.

**TurfReward:MinTeams**

**Type:** Integer

**Default:** 2

The minimum number of teams needed in the arena for anyone to be eligible to recieve points.

**TurfReward:MinWeights**

**Type:** Integer

**Default:** 1

The minimum number of weights needed to be owned by a freq for that team to be eligible to recieve points.

**TurfReward:MinWeightsPercent**

**Type:** Integer

**Default:** 0

The minimum percent of weights needed to be owned by a freq for that team to be eligible to recieve points. (ex. 18532 means 18.532%)

**TurfReward:RecoverDings**

**Type:** Integer

**Default:** 1

After losing a flag, the number of dings allowed to pass before a freq loses the chance to recover. 0 means you have no chance of recovery after it dings (to recover, you must recover before any ding occurs), 1 means it is allowed to ding once and you still have a chance to recover (any ding after that you lost chance of full recovery), ...

**TurfReward:RecoverMax**

**Type:** Integer

**Default:** -1

Maximum number of times a flag may be recovered. (-1 means no max)

**TurfReward:RecoverTime**

**Type:** Integer

**Default:** 300

After losing a flag, the time (seconds) allowed to pass before a freq loses the chance to recover.

**TurfReward:RecoveryCutoff**

**Type:** Enumerated

**Default:** TR\_RECOVERY\_DINGS

Style of recovery cutoff to be used. TR\_RECOVERY\_DINGS - recovery cutoff based on RecoverDings. TR\_RECOVERY\_TIME - recovery cutoff based on RecoverTime. TR\_RECOVERY\_DINGS\_AND\_TIME - recovery cutoff based on both RecoverDings and RecoverTime.

**TurfReward:RewardModifier**

**Type:** Integer

**Default:** 200

Modifies the number of points to award. Meaning varies based on reward algorithm being used. For \$REWARD\_STD: jackpot = # players \* RewardModifier

**TurfReward:RewardStyle**

**Type:** Enumerated

**Default:** TR\_STYLE\_DISABLED

The reward algorithm to be used. Built in algorithms include: TR\_STYLE\_DISABLED: disable scoring, TR\_STYLE\_PERIODIC: normal periodic scoring but with the all the extra stats, TR\_STYLE\_STANDARD: see source code documentation (complex formula) + jackpot based on



# players TR\_STYLE\_STD\_BT\_Y: standard + jackpot based on bounty exchanged TR\_STYLE\_FIXED\_PTS: each team gets a fixed # of points based on 1st, 2nd, 3rd,... place TR\_STYLE\_WEIGHTS: number of points to award equals number of weights owned

**TurfReward:SafeRecievePoints**

**Type:** Boolean

**Default:** No

Whether players in safe zones recieve reward points.

**TurfReward:SetWeights**

**Type:** Integer

**Default:** 0

How many weights to set from cfg (16 means you want to specify Weight0 to Weight15). If set to 0, then by default one weight is set with a value of 1.

**TurfReward:SpecRecievePoints**

**Type:** Boolean

**Default:** No

Whether players in spectator mode recieve reward points.

**TurfReward:TimerInitial**

**Type:** Integer

**Default:** 6000

Initial turf\_reward ding timer period.

**TurfReward:TimerInterval**

**Type:** Integer

**Default:** 6000

Subsequent turf\_reward ding timer period.

**TurfReward:WeightCalc**

**Type:** Enumerated

**Default:** TR\_WEIGHT\_DINGS

The method weights are calculated: TR\_WEIGHT\_TIME means each weight stands for one minute (ex: Weight004 is the weight for a flag owned for 4 minutes). TR\_WEIGHT\_DINGS means each weight stands for one ding of ownership (ex: Weight004 is the weight for a flag that was owned during 4 dings).

**Wormhole:GravityBombs**

**Type:** Boolean

Whether a wormhole affects bombs

**Wormhole:SwitchTime**

**Type:** Integer

How often the wormhole switches its destination

## 12.3 Other settings

**General:AllowUnknown**

**File:** passwd.conf

**Type:** Boolean

**Requires module:** auth\_file

**Default:** Yes

Determines whether to allow players not listed in the password file.

**General:RequireAuthenticationToSetPassword**

**File:** passwd.conf

**Type:** Boolean

**Requires module:** auth\_file

**Default:** Yes

If true, you must be authenticated (have used a correct password) according to this module or some other module before using ?local-password to change your local password.

## 12.4 More detail on specific sections

### 12.4.1 Flags

Flags in assf are implemented by the coordination of several modules: **flagcore** implements the actual flag-related pieces of the game protocol, and general state-keeping. The rules for a specific flag game is implemented by a **fg\_something** module, of which two are supplied with the server: **fg\_wz** is a basic warzone-type flag game, where a team has to own all the flags to win. **fg\_turf** is a turf-style game, where the flags are stationary, and points are awarded based on flags owned.

To get one of these games working, you should make sure **flagcore** and the desired flag game module are both loaded. Then attach the desired flag game module to your arena, by listing it in the **Modules:AttachModules** setting. Then configure it with the appropriate settings, all of which can be found in the **Flag** section. At a minimum, for **fg\_wz**, you need to set **Flag:FlagCount**.

### 12.4.2 Energy/inventory viewing

There are two arena settings that control whether players see other player's energy and ship inventory (from spec):

- **Misc:SpecSeeEnergy** This affects what players in spec see. If it's set to **\$SEE\_ALL**, spectators see energy for all players. If it's **\$SEE\_SPEC**, they see energy for only the player they are spectating, and if it's **\$SEE\_NONE**, they don't see any player's energy.
- **Misc:SeeEnergy** This is like the previous setting, but applies to players in ships. **\$SEE\_ALL** and **\$SEE\_NONE** work as before. **\$SEE\_SPEC** isn't allowed here, and a new option is **\$SEE\_TEAM**, which allows everyone to see the energy of their teammates.
- **Misc:SpecSeeExtra** This boolean option determines whether spectators see the extra inventory data for players they are spectating.

In addition, there are two capabilities that override the above settings. **seepd** allows players to see energy/inventory from spec, and **seenrg** allows energy viewing while playing.

## 13 Acknowledgements

Thanks to these people:

- divine.216 for general support, lots of help testing, banner support, and many useful suggestions.
- Mine GO BOOM for lots of bug-finding and suggestions, as well as being the first person besides me to actually contribute code to assf.
- Stag Shot for making sure powerball isn't left out, timer features, and other small contributions.
- GiGaKiLLeR for contributing a turf rewards module.
- Mr. Ekted for technical help and discussions.

- ZippyDan for encouragement and comic relief.
- xalimar for shell accounts and hosting, mostly.
- numpf for design critiques and other criticism.
- Remnant for being the first person to log into ass (besides me, of course), and help testing.
- The rest of the PowerBot chat for friendly conversation and entertainment.
- Dr Brain for being brave enough to try ass on a real live zone, report all the problems he had, and spend time helping me debug them.
- Catid for a bunch of contributions including security and other bug fixes.
- PriitK for helping with Continuum interoperability and also for some ambitious feature suggestions.
- D.A.F. (not a subspace player) for conversations on design and more.