



Phasor 2.0

2.00.07.21

<http://phasor.proboards.com>

2013

Table of Contents

Overview.....	3
Installation.....	4
Setting Up Admins	5
Using Phasor.....	7
Commands.....	9
Map Voting.....	9
Script Management.....	9
Logging.....	9
Alias	10
Admin.....	10
Version.....	11
Miscellaneous	11
Scripts	12
Installing a script	14
Creating scripts	14
Miscellany.....	16
Command line parameters.....	16
Earlyinit.....	16
Acknowledgements.....	17

Overview

Phasor is a server extension for Halo PC/CE, which provides numerous additional features when compared to a vanilla server. Version 2.0 is a complete rewrite of the previous versions of Phasor and as such it is faster, more secure, and simply better. Phasor's features can be split into three broad categories: scripts, commands, and fixes. Phasor supports extensibility via scripts, there are many different scripts written by the community, some examples include: infection, aimbot detection, taxi, chat filter, custom commands, and many more. See the scripting section for more information. Phasor also provides various new commands for server admins, giving them more control over the server. Finally, Phasor makes your server more secure by fixing various defects that are present in the vanilla Halo Dedicated Server.

Long story short, Phasor vastly increases the capabilities of your Halo server.

Installation

Installing Phasor is fairly straight forward, but many people seem to have issues doing it. Please follow these instructions closely before asking us for help.

1. Download Phasor, you can find the latest version at <http://phasorforhalo.com/download>
2. The file you downloaded will be named (similar to) Phasor-02-00-07-21.zip. In order to extract this file you will need some program which supports zip files; I think Windows comes with a suitable program. Go ahead and extract the files into a folder somewhere, for example your Desktop. The rest of these instructions assume you have extracted them to *Desktop/Phasor*.
3. Make sure you have Halo's Dedicated Server installed, if you're using Phasor CE you don't have to do anything because Halo CE comes bundled with it. However, CE users will need to follow step 4, too. Phasor PC users will need to download and install the Dedicated Server manually, you can obtain an update-to-date copy from <http://phasorforhalo.com/download>
 - a. Note: The above link will take you to a downloads page, you should be able to find the server link pretty easily. The file you download will be another zip file, first run *HaloDS.exe* to install the server, then copy the included *haloded.exe* to where you installed the server, overwriting the existing file. By default it will install (on 32-bit systems) to *C:\Program Files\Microsoft Games\Halo Server*.
4. (Phasor CE only) The Halo CE server is, by default, installed into the same directory as the game. This doesn't work with Phasor because it needs to overwrite some of Halo's files, and this will cause your game to stop working. So, to save yourself many headaches, do as follows.
 - a. Create a new folder (in your Halo CE folder) called server.
 - b. Copy *haloceded.exe* **and** your maps folder into your new server folder. You are required to copy yours maps folder, this is a bug with the Halo CE server and I may fix it in a future release.
 - c. This new (server) folder is what you should use for the remainder of the installation.
5. Copy Phasor's files into your server folder. For PC users this is (usually) *C:\Program Files\Microsoft Games\Halo Server* and for CE users it is (usually) *C:\Program Files\Microsoft Games\Halo CE\server*.
 - a. (Phasor PC only) Copy the files that are inside *Desktop/Phasor/PC* into your server folder. **Do not copy the folder itself, only the files/folders contained within it!**

- i. You may be asked to overwrite *strings.dll*, say yes. You should make a backup of the original *strings.dll* before overwriting, of course.
 - b. (Phasor CE only) Copy the files that are inside Desktop/Phasor/CE into your server folder. **Do not copy the folder itself, only the files/folders contained within it!**
 - c. Copy the files and folders contained within Desktop/Phasor/CopyToServer into your server folder. **Do not copy the folder itself, only the files/folders contained within it!**
6. You're nearly done, the last thing you need to do is setup your data files. Halo stores various settings (such as your banlist) inside another folder, and so does Phasor. By default this directory is inside your *My Games* folder. For Halo PC it is *My Games/Halo* and for Halo CE it is *My Games/Halo CE*.
 - a. Copy the files and folders contained within Desktop/Phasor/CopyToData into this folder. **Do not copy the folder itself, only the files/folders contained within it!**
7. You're done. Phasor is ready to use, but you'll probably want to setup scripts and admins, so it's best to keep reading.

The rest of this document will assume you're familiar with operating a normal Halo server. If you're not sure what admins, rcon passwords, cd keys etc. are, then you should consult your good friend Google.

Setting Up Admins

Phasor implements an admin system that is based off of your CD-Key hash. Each time you join a server, this hash (which is unique to you), gets sent to the server. It is used for banning players, and Phasor can also use it to detect admins. When enabled, Phasor's hash-based admin system provides greater security for your server because players need more than just the rcon password to execute commands. It also lets scripts detect admins and many scripts rely on this feature.

Anyone that you've added to your admin list (see below) is considered to be an admin, but you can give different admins different levels of power. This is accomplished via *admin levels*. You can completely customize what each level is allowed to do, and you should use the *RconAccessEditor* to do it, although it requires a separate download, <http://phasorforhalo.com/downloads>.

Phasor also comes with a set of ready-made access levels that will probably suit most people's needs, but if you want to change them you can use the above tool (or simply edit *access.ini*). There are 3 default access levels.

- Level 2 – This is the most restricted level, admins at this level can be thought of as moderators. They are allowed to kick people, but not ban.

They can execute the following commands: *sv_players*, *sv_kick*, *sv_say*, *sv_kill*.

- Level 1 – Admins at this level can do everything level 2 admins can do, but they can also ban people. They can execute the following commands, in addition to level 2 commands: *sv_ban*, *sv_banlist*, *sv_unban*.
- Level 0 – This is effectively “super admin” status; they can execute any command, including those provided by scripts. This is probably the most used level, simply because it integrates a lot better with scripts.

Now that you know what admin levels are, you can finally add some admins. The easiest way to do this is via the *sv_admin_add* command; its usage is described in the commands section.

Using Phasor

Now that you've got your server setup and running Phasor, it's probably time to learn *how* to use Phasor. This section will guide you through creating an *init.txt* file to create a useful mapcycle. It will assume you're familiar with using normal Halo servers.

Let's assume that you have some scripts installed (see Scripting section) and that they are called *script1* and *script2*. You already know how to create a mapcycle, so let's start with the below *init.txt* file.

```
sv_public 1
sv_timelimit 30
sv_rcon_password "test"
sv_name "My New Phasor Server"
sv_mapcycle_add bloodgulch ctf
sv_mapcycle_add timberland slayer
sv_mapcycle_timeout 15
sv_mapcycle_begin
```

But now we want to add scripts to each of the mapcycle options, perhaps *script1* is a gametype script or something? Luckily, it's really easy to get this working – simply modify the *sv_mapcycle_add* command to include the script(s) to load at the end. So, our init file becomes

```
sv_public 1
sv_timelimit 30
sv_rcon_password "test"
sv_name "My New Phasor Server"
sv_mapcycle_add bloodgulch ctf script1
sv_mapcycle_add timberland slayer script1 script2
sv_mapcycle_timeout 15
sv_mapcycle_begin
```

So far so good, but now we decide that no matter what, we want this server to load *script1*. For example, *script1* could be an infection script and we want every game to use it. Do we have to add it to every mapcycle option? No. Phasor now supports persistent scripts that don't get unloaded between games, so we can just make *script1* persistent. There are a few ways to do this: we could put it into the *persistent* folder (which is in the scripts folder) so that Phasor will load it when it starts, or we could use the *sv_script_load* command (see the Commands section) to load it for us. So we change the init file and it now looks like this.

```
sv_public 1
sv_timelimit 30
sv_rcon_password "test"
sv_name "My New Phasor Server"
sv_mapcycle_add bloodgulch ctf
sv_mapcycle_add timberland slayer script2
sv_mapcycle_timeout 15
sv_mapcycle_begin
sv_script_load script1 true
```

Note that we removed *script1* from the *sv_mapcycle_add* command, because it will already be loaded thanks to *sv_script_load*.

Finally, we decide we want the server to use mapvoting instead of mapcycles. The map voting commands are analogous to the map cycle commands, so the changes are pretty simple. The only real difference is that map voting requires each option to be described, so that players know what they're voting for. Again, this is pretty straightforward and so we end up with the following init file (see Commands section for info on the map vote functions).

```
sv_public 1
sv_timelimit 30
sv_rcon_password "test"
sv_name "My New Phasor Server"
sv_mapvote_add bloodgulch ctf "Bloodgulch CTF"
sv_mapvote_add timberland slayer "Timberland Slayer
running script2" script2
sv_mapcycle_timeout 15
sv_mapvote_begin
sv_script_load script1 true
```

The changes are highlighted, and you should note that *sv_mapcycle_timeout* still controls how long to wait after a game, ie how long to let players vote for.

Commands

Here's a list of commands provided by Phasor, with basic descriptions about what they do.

Map Voting

- `sv_mapvote <boolean>`
 - Used to toggle whether or not map voting is active.
- `sv_mapvote_size <integer>`
 - Specifies the number of vote options to show (default: 5).
- `sv_mapvote_add <map> <gametype> <description> [scripts]`
 - Adds an option to the vote list, description is shown to players when they're voting.
- `sv_mapvote_del <index>`
 - Removes an option from the vote list, see `sv_mapvote_list` for indices.
- `sv_mapvote_list`
 - Displays a list of the current vote options.

Script Management

- `sv_script_reload [script]`
 - Reloads the specified script, or all scripts if none specified.
- `sv_script_load <script> [persistent]`
 - Loads the specified script, persistent specifies if the script should be stay loaded between games (default: false)
- `sv_script_unload <script>`
 - Unloads the specified script.
- `sv_script_list`
 - Lists all currently loaded scripts.

Logging

Note: log type can be any of the following: phasor, script, game, rcon.

- `sv_log_name <log type> <new name>`
 - Changes the name of the file where the specified log is saved.
- `sv_log_limit <log type> <max size in kB>`

- Sets the maximum size of a log, once a log reaches the specified size it is moved to to directory specified by `sv_log_move_dir`
- `sv_log_move_dir <directory>`
 - Sets the directory where old logs should be stored.

Alias

- `sv_alias_enable <boolean>`
 - Enables (or disables) alias tracking (default: true)
- `sv_alias_hash <hash or player index>`
 - Searches the alias database for the specified hash, returning the results.
- `sv_alias_search <partial match to find>`
 - Searches the alias database for names matching the specified format.
 - Example: `sv_alias_search oxi%`
 - Finds people with names starting with oxi
 - Example: `sv_alias_search oxide`
 - Finds people named oxide
 - Example: `sv_alias_search %oxi%`
 - Finds people with names containing oxi

Admin

- `sv_admin_add <hash or player index> <auth name> <level>`
 - Adds the specified player (or hash) as an admin who will be identified by their auth name, they will have access at the specified level (0 is, by default, all access)
- `sv_admin_del <admin name>`
 - Removes the specified admin.
- `sv_admin_list`
 - Lists all current admins.
- `sv_admin_cur`
 - Lists all admins currently in the server.
- `sv_admin_reload`
 - Reloads the admin list (admin.txt)
- `sv_admin_commands`
 - Lists the commands you're allowed to execute based on your level.
- `sv_admin_check`
 - Specifies whether or not admins should only be declared as such once gamespy has validated their hash (default: true)

Version

Valid versions (PC): 100, 101, 102, 103, 104, 105, 106, 107, 108, 109

Valid versions (CE): 100, 107, 108, 109

- `sv_version <ver>`
 - Sets the version the server should broadcast on (default: 109)
- `sv_version_check <boolean>`
 - Enables/Disables version checking so that clients with an incorrect Halo version can join (default: false).

Miscellaneous

- `sv_hash_check <boolean>`
 - Specifies whether or not to allow players with invalid hashes should be allowed to join (default: false).
- `sv_kickafk <time in minutes>`
 - Specifies how long to wait before kicking AFK players, specify 0 to disable (default: 0).
- `sv_kill <player>`
 - Kills the specified player.
- `sv_getobject <objid>`
 - Prints the memory address of the specified object (this is only really used by scripters and myself).
- `sv_invis <player> [duration in seconds]`
 - Makes the specified player invisible for the specified duration (in seconds). A duration of 0 (default) gives them active camo until their death.
- `sv_setspeed <player> <speed>`
 - Changes the specified player's speed (default: 1).
- `sv_say <msg>`
 - Sends the specified message to the entire server.
- `sv_gethash <player>`
 - Prints the specified player's hash.

Scripts

Scripts are what really make Phasor useful because they let the server do many highly customized things, things that I simply can't think of, or wouldn't be able to do justice to. The scripting community is active and they have released a pretty broad range of scripts. There are scripts that implement cool new gametypes (infection, race, bases, one in the chamber, the list goes on) and others which provide admins with more control over the server (chat filters, more commands, player control etc) and everything in between. Most scripters release scripts at the Phasor forums, <http://phasor.proboards.com>.

It's also worth noting that Phasor comes with a few scripts when you download it. I did not make them, but the authors have allowed me to include them with Phasor. If you find any issues with them, please check for a more update-to-date version, or let the author (not me) know about the issue. Here's a list of the included scripts, ordered by author:

- Nuggets
 - Random Weapon: *Every time you spawn, you will spawn with random attributes.*
 - Conquest: *Take control of the enemies' base before they take control of yours.*
 - Gravity Gun: *Defy gravity with a plasma pistol.*
 - Helpers: geo, iter, sendconsoletex, table: *Useful functions which other scripters can use.*
- AelitePrime
 - Blocking: *Stop players from going into various (hacky) locations.*
 - Chat Filter: *For those with sensitive ears.*
 - Commands: *Provides a ridiculous amount of additional admin commands.*
 - IP-Alias: *Admin system implemented on IP addresses instead of CD Keys.*
 - Tbag: *Let the server know when you're tbagging someone.*
 - Teleport: *Lets admins teleport around the map via server commands.*
 - UniqueID: *Give every player a unique ID.*
 - AntiBot: *Detect players using aimbots.*
 - Zombie: *Injection.*
- H® Shaft
 - Ping Kick: *Automatically kick those dirty foreigners.*
 - Rock the vote: *Let players control what map gets run.*
 - Kill Messages: *Detailed death information.*
 - Warthog-Pistol: *Shoot a warthog to enter.*
- TelyX

- CustomSpawnWeapons: *Have players spawn with 0 to 4 weapons and custom assign the weapons they spawn with on each map, under each gametype.*

Installing a script

So, you've found a script that you want to use, but how do you install it? The answer is usually pretty straightforward and I will describe the general procedure, but bear in mind that certain scripts will require additional steps and so you should always read their documentation.

1. Download the script. Most scripters release their scripts via a website called pastebin. There's a download button at the top of the page.
2. Unfortunately, pastebin will usually save the script as a text-file, and Phasor only loads scripts that are Lua files. Luckily, the only difference between the two is their extension, so find where you downloaded the script to and rename it to .lua. For example, if I downloaded a script called commands, and it saved it as commands.txt, I'd want to rename it to commands.lua.
 - a. If you don't know how to change a file's extension, please search for information on Google.
 - b. To save time (depending on your internet browser), you can right-click Download and select "Save link as", and save it as a .lua file.
3. Now that you've saved the script as a .lua file, all you need to do is copy it to your Scripts folder. This folder is, by default, located in *My Games/Halo/scripts* for Halo PC, or *My Games/Halo CE/scripts* for Halo CE.
4. Phasor now supports the notion of a persistent script, which isn't unloaded at the end of each game. If you'd like the server to load this script when it starts, and keep it loaded until it closes, then put into *scripts/persistent* instead of simply putting it in the scripts folder. *Note: not all scripts support persistence, so you could consult its documentation first.*
5. The script has been installed. You can load or unload it via *sv_script_load* and *sv_script_unload*, or specify it as an option in your mapcycle. See the examples section for details on how to do this.

Creating scripts

If you want to create scripts then you need to learn quite a few things. First off, you need to learn the *Lua* programming language, which is what you'll use to write scripts. Then you'll need to learn the Phasor specific stuff – how Phasor uses scripts, and how scripts can use Phasor. All of this stuff takes time, and the best way to get started is to head over to the Phasor forums and have a read of the scripting section. Once you're familiar with Lua you might find the following links quite useful, they describe Phasor's scripting interface.

Nuggets has kindly made the following brief introduction to scripting, it has some advice for beginners and lists some of the changes that this version has made to scripts, <http://phasor.proboards.com/thread/735/started-scripting>.

Miscellany

This section will be somewhat less verbose than previous ones, I will assume you're competent with using a Windows PC.

Command line parameters

Phasor lets you change a few options via command-line parameters.

- -mappath <path> Specifies the path to use for location map files, this only works on PC at the moment, sorry.
- -scriptpath <path> Specifies the path to use as your script folder.

Earlyinit

Phasor will load a text file, *earlyinit.txt*, from your server folder when the server loads. This can be used for changing log directories, etc. You can also load persistent scripts from here and they will get access to all of Phasor's load up commands (or just use the persistent folder, same effect).

Acknowledgements

Many people have contributed to Phasor in one way or another, whether with code, ideas, testing, or whatever. This list isn't exhaustive, but thanks to the following: Wizard, Nuggets, chalonic, btc22, sehe, all my testers, and anyone else I've forgotten.

Phasor is released under the terms of the MIT license, reproduced here.

Copyright © 2010–2013 phasorforhalo.com.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Phasor utilizes many libraries built by other people, here are their relevant licences.

Lua

Copyright © 1994–2013 Lua.org, PUC-Rio.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:



The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Curl

COPYRIGHT AND PERMISSION NOTICE Copyright (c) 1996 - 2013, Daniel Stenberg, <daniel@haxx.se>. All rights reserved. Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE. Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.