**Long version of dirtsand build**

"apt-get" has a replacement which is just "apt" I am testing this and if it works I will edit the listing

Yes I built another shard and apt on its own and the terminating -y works so I have edited both files

[/edit]

NOTE: There is a fix part way through this description credited to Hoikas Note this is a TEMPORARY fix at some point we will remove that fix at which point It will no longer be needed.

This is one method of building an Uru shard, it describes a shard that will allow multiple users on a local area network. It requires an internet connection for building but once built it no longer requires that connection.

HARDWARE DESCRIPTION:

At this time the system as described has only been tested with two users and has only been tested with the hardware described.

The main Shard is a 64bit AMD former Vista PC with 4 Gig of memory and a 250 Gbyte Sata Hard Drive a VGA on board graphics and an old VGA monitor this is all salvage. It has a keyboard but no mouse. Keyboard and monitor are repurposed once the shard is running.

I use an old XP notepad as a remote terminal for the shard this is not actually required but it allows me to put the shard out of the way and to run it without a keyboard or monitor once it is up and running.

This notebook is pre-installed with ConText, CoreFTP and Putty.

Note Filezilla is a viable alternative to CoreFTP and Notepad++ an alternative to ConText as a text editor these are preferences not hard options.

I have two workstations both have been tested and used as MOULa access terminals for a number of years.

This is by no means a minimum specification just a description of the hardware I used for testing.

PRE REQUISITES:

Ok I grabbed the latest images of Ubuntu Server 18.04. At this time this is only available as 64 bit and as I am building this on a former 64bit AMD Vista PC this is viable.

I have been assured this will install on 32 bit earlier Ubuntu Servers and Desktop Distros confirmations to follow once tested.

I also specifically tried using Debian as a platform and I will test with other hardware and distros once I have everything tied down as software.

As a location during the build I made a folder in my Download directory called Dirtsand.

Other things it will be useful to grab visit https://github.com/H-uru/moul-scripts click the green Clone or Download button and download the zip file and unpack this. This is one of the things I put in that Download folder.

Also go to http://files.mystler.eu/Client.zip download and unpack a copy of this. This is Mystlers client for Destiny.

Install a clean copy of Myst Online Uru Live (MOULa) link to https://mystonline.com/en/play/ and click the link to Cyan Worlds (Full Install). While it is nice to have a login to

MOULa strictly speaking you will probably not need it. I logged in once just to make sure I had the latest updates.

I made a copy of that Uru Live folder to a portable drive and burned a copy to DVD this will save you download time if you end up installing multiple times trying to understand the method described here.

I started to build my shard and had real problems till I spotted you were no longer allowed to use upper case letters in the computer name and if you tried to tab to the next field it took it that you were finished with that page. I solved that by changing all the upper case to lower case (go figure).

For the purposes of clarity in this document I am using the following users and passwords and you can use a replace to personalise the document.

Server name floatingpencil

Server user Sidney Milkman

Login sid

User password milkbottle

User dirtsand password soilgrit

Dirtsand database password laundrymarker

I built the server with the name flotingpencil with a user Sidney Milkman with a login user name sid and a password of milkbottle (these are sample names not actual I am not that stupid).

If you are re-installing it is worth noting that some distros do not automatically clear down a hard drive and it may be worth visiting the command.

[code]

$fdisk /dev/sda

[/code]

I had issues moving between Debian and Ubuntu and it was not always clear that command allows you to insure the partition table is created fresh when you do a reinstall.

So the server is built. By default, the Ubuntu server version installs with ssh automatically and I can reboot switch off the monitor and then log in remotely from a small local workstation using putty. It is just worth noting that under Debian I had to manually add an ssh server.

Although everything can actually be run from the command line on the fresh server there is something later on that will be easier if you are using a putty connection.

Before I run the apt-get install line provided by Deledrius I do the flowing two commands sudo obviously needs my milkbottle password after the first and that lasts for the duration of the session.

The $ symbol shows these are Linux shell command line entries.

GETTING THE TOOLS:

[code]

$ sudo apt update

$ sudo apt upgrade -y

[/code]

I don't know if these are strictly necessary but I figure I will be told if this is wrong. Even though this is a recent download of Ubuntu there are still some upgrades that appear to download and upgrade.

Following the instructions provided by Deledrius:

[code]

$ sudo apt install build-essential git cmake libpq-dev libssl-dev libreadline-dev zlib1g-dev postgresql postgresql-contrib apache2 php7.2 -y

[/code]

I get a message saying git is already the latest version but I am not worried about this as I have seen this many times before it does no harm to leave it in the command. I have added postgresql and postgresql-contrib because after multiple re-installs to prepare this document I have realised I will need these later on in the installation. So we might as well add these here.

The apache2 has been added because I wanted a quick way to see if the reboot had completed on those times it was used and php7.2 because at some point if I want to follow the example found in Mystlers Destiny shard I will need to have status.php added.

Neither of these is a requirement and if you are putting this on a drive where space is an issue feel free to drop them.

This will take a while to load and at some points it will appear to have stalled I assure you there are times it waits for a while particularly during the man.db data grab. Be patient it will complete eventually.

We are now going to build a new user that will be the dirtsand user and give that new user a home directory that will reside in a the /opt folder the prefix forward slash indicates that this is path starts at the drive root.

[code]

$ sudo useradd dirtsand -d /opt/dirtsand -m -s /bin/bash -g sudo

[/code]

Here we create a user called dirtsand the -d option creates and sets that home directory as /opt/dirtsand, -m grabs the default settings for a start-up, the -s sets the shell prompt to the /bin/bash type. The -g option adds dirtsand to a user group, in this case the sudoers this allows the dirtsand account to use the sudo prefix and gain super user access, the -p option that was in the original version of the load used to add a password is no longer valid in this Linux distro I am using here.

The user is created but the password is not. With this new user created I use the following to add a password to our new dirtsand account.

[code]

$ sudo passwd dirtsand

[/code]

Then follow the prompts and add a password soilgrit for this new dirtsand account.

Your work done enter the following.

[code]

$ sudo reboot

[/code]

BUILDING THE CODE:

This reboots the server and I log in from my slave workstation using putty as dirtsand with the new password soilgrit.

Logged in use the Linux command pwd (print working directory). This confirms that the dirtsand login has a home directory of /opt/dirtsand and that directory is now our working directory.

Deladrius suggested creating a sub-directory in that home directory in this case "/opt/dirtsand" called development and then moving to make this the working directory. I have dropped this option for tidiness. Plus I reinstall multiple times and in a server that is not used for anything else it is a step in path complexity that is just not required.

[code]

$ git clone https://github.com/zrax/string\_theory.git

[/code]

Here we are making a clone copy of the sub-directory from the github called string\_theory this will sit in the /opt/dirtsand directory.

[code]

$ git clone https://github.com/H-uru/dirtsand.git

[/code]

Here we are making a clone copy of the sub-directory from the github called dirtsand this will also sit in the /opt/dirtsand directory.

[code]

$ mkdir string\_theory/build && mkdir dirtsand/build

[/code]

This adds sub-directories called build in each of these cloned directories the double ampersand links two commands into a single command line that will stop if one fails.

[code]

$ mkdir ages

$ mkdir data

$ mkdir authdata

$ mkdir SDL

[/code]

These other four directories will be required later.

[code]

$ cd string\_theory/build

[/code]

Here we are changing directory so our working directory is the newly created build directory in the string\_theory sub-directory.

[code]

$ cmake ..

[/code]

Here we run the cmake command using as its source the stuff it finds in the parent folder of our current working directory.

Note there is a space between the command and the two periods.

[code]

$ sudo make install

[/code]

The commands cmake and make install is a two-step compilation of code, so here we do the second stage which requires the “make” command this needs to be run as sudo because it makes changes to the main directory structure of the server. So you will be asked for your dirtsand password so in our sample case this would be soilgrit.

You are asked for the dirtsand password because this is the first time you will have used sudo with the dirtsand login.

Before we proceeded we are going to do a bugfix provided by Hoikas this will be subject to a full repair later but for the time being you need the following actions.

[code]

$cd /opt/dirtsand/dirtsand/AuthServ

$nano AuthDaemon.cpp

[/code]

You need to navigate to line 33 it should look like

std::unordered\_map<ST::string, SDL::State, ST::hash\_i, ST::equal\_i> s\_globalStates;

You need to edit it to remove the \_i in the middle so it reads

std::unordered\_map<ST::string, SDL::State, ST::hash, ST::equal\_i> s\_globalStates;

[Ctrl][x] y [Enter] will replace the existing file with the corrected one.

Now we make the other build folder the working directory and do the same two stages this time the cmake command has a prefix.

[code]

$ cd /opt/dirtsand/dirtsand/build

$ cmake -DCMAKE\_INSTALL\_PREFIX = /opt/dirtsand ..

$ sudo make install

[/code]

Note there is a space between the cmake and the -DCMAKE and another one between /opt/dirtsand and the double period at the end but no spaces either side of the = in the middle of the line.

SETTING UP YOUR SERVER:

You will need a working Postgres server which DIRTSAND can use to store its data.

This was the reason for the postgresql additions added to the first apt-get install.

For the default installation, the provided scripts will create a dirtsand database and set its ownership to a 'dirtsand' database user, which can directly map the system dirtsand user created in step 1 of the "BUILDING THE CODE:" instructions above.

For better security, it is recommended to use a password (as shown in the steps below) we will use laundrymarker as a placeholder for the password you designate, which can be configured in the server settings as described in the "configure dirtsand" step.

SET UP THE POSTGRESQL USER:

[code]

$ sudo -u postgres psql -d template1

[/code]

This accesses the postgresql database software using the default “template1” at this point your prompt will change to the “template1=#” you will see in the following lines, this should inform you that you are issuing postgresql commands and not native Linux commands.

Note: These database commands are terminated with a semi-colon this is the normal SQL end of line marker.

Should you forget the semicolon you will not get the response, just a new line with the database prompt just enter the semicolon on its own and press enter. The command should then either complete and respond or give you some sort of error message probably because you missed out an equals or some speech marks. Just correct and re-enter.

[code]

template1=# CREATE USER dirtsand WITH PASSWORD 'laundrymarker';

[/code]

You are going to give this a password keep the single quotes try to give this a password that has not been used previously laundrymarker is just a placeholder for your password. Make a temporary note of the password you are going to need this later. If you get this right you should get the response

CREATE ROLE

If you do not get the response then you probably missed the semicolon just enter it on the empty line and the instruction should complete.

[code]

template1=# CREATE DATABASE dirtsand WITH TEMPLATE = template0 ENCODING = 'UTF8';

[/code]

Again it is worth noting SQL command reserved words are all in upper case and I remind you the end of line semicolon is required. You can sometimes get away with mixing the case in SQL commands but it is good practice not to do this. Again if you have this correct you will get a response.

CREATE DATABASE

[code]

template1=# ALTER DATABASE dirtsand OWNER TO dirtsand;

[/code]

Here we are changing the designated ownership of the database note this is not the Linux user dirtsand but the database user dirtsand. Note there are no speech marks just the semicolon. This time the response should be.

ALTER DATABASE

[code]

template1=# \q

[/code]

OK this is probably the only SQL command not terminated with a semi-colon this is the “exit” think of it as shorthand for backslash quit.

Install the UUID functionality:

This is provided by your OS distribution under Ubuntu, you simply install the postgresql-contrib package remember we did this at the start. We are now going to log back into postgresql but this time as our new user dirtsand.

[code]

$ sudo -u postgres psql -d dirtsand

[/code]

This time the postgresql prompt changes to “dirtsand=#”.

[code]

dirtsand=# CREATE EXTENSION "uuid-ossp";

[/code]

This is a one shot command to add an extension to the database. This gives the response.

CREATE EXTENSION

[code]

dirtsand=# \q

[/code]

Exit again but this time you are exiting as the dirtsand user not the template1 user.

[code]

$cd /opt/dirtsand/dirtsand

[/code]

Move to the correct directory this puts you in the right location to call the db sub directory. The pwd directory should show you are sitting in the /opt/dirtsand/dirtsand folder.

We are now going to Set up the dirtsand database. $ prompt would suggest it is a Linux command and the psql looks like a postgresql command but we have been told to log out with the \q so you may wonder what is the is the user login if it is postgresql and where is the closing semi-colon . Confusing yes?

[code]

$ psql -d dirtsand < db/dbinit.sql

$ psql -d dirtsand < db/functions.sql

[/code]

In reality this is pair of Linux commands that call postgresql database directly with psql with an argument that is a text file with an extension of .sql these two commands effectively pre-stuff the database you will generate a password error don’t worry. The left pointing less than symbol indicates what follows to the right of the symbol is being fed into whatever is left of the symbol.

Expect to see a load of stuff scroll up your screen this is the response lines that are generated by postgresql.

Do not be surprised by the error message “Error: language “plpgsql” already exists” when you run the second line I believe this is a failsafe because there is a check in the

functions.sql file. Expect a password error this is to be expected because the internal users’ password has not been set. If there were no other errors, your database should be ready to roll.

CONFIGURE DIRTSAND:

A sample dirtsand.ini has been provided in the root of the dirtsand sources. We can copy this with a new name to our install directory and then edit the fields we need.

Specifically, you will need to adjust the server addresses and the RC4 keys. If you have dirtsand installed to somewhere other than /opt/dirtsand, you will also need to point the configuration to the right paths too. At this time the plan is to keep things simple so that is not something to worry about.

[code]

$cd /opt/dirtsand/dirtsand

$ cp dirtsand.sample.ini dirtsand.ini

[/code]

The first command makes sure we are in the correct working directory.

The command previously needed a change of ownership but we are using a dirtsand login so we have ownership by default. This bog-standard Linux shell command that says find and make a copy of dirtsand.sample.ini and name it dirtsand.ini.

You now need to generate the RC4 keys, you can simply run the keygen command from within the dirtsand interactive console: You need your working directory to be /opt/dirtsand/dirtsand because the interactive console is in the bin sub-directory of that folder.

[code]

$ bin/dirtsand

[/code]

This runs that interactive console.

Hitting enter spits out some information about missing data that has a hold cursor so you can read error messages. This is to be expected because you have not made the settings to prevent these messages hitting enter the second time changes to the interactive prompt. In my case it gives a new line that a “ds-918>” prompt this appears to be the prompt for the interactive console. ds-902> was the example provided by Deledrius I believe at this time the number is a version thing.

[code]

ds-918> keygen new

[/code]

Here you get the following response line with a growing line of waiting dots. The number of trailing dots is obviously dependent on the speed of your server.

Generating new server keys . . . This will take a while . . .

Eventually this then spits out something like the following.

--------------------

Server keys: (dirtsand.ini)

Key.Auth.N = 4142zYQw2nEw24VdMTgzHZLwh2MMOWWJpWro7cg+9uNpit3klrTPjR35gNq9Ic4VbFKs0rDuWnJSg/2kq0q+6w==

Key.Auth.K = 2c1AMtmiZIZFgop4nnisWxAS4ggSATbAV57U0gVfvL3RQ/k+m2S5pTi79ArG+d9pruUuz5PInaqnkDIKM5G2Iw==

Key.Game.N = yEfFVgixz3T1FCp6MXBfUBPV2wGuuzllidaf6AExkdb6oXZOaEsuD4Q+drsvCIZvaw/GGKJPu37LzRdEd4D+Yw==

Key.Game.K = +iG78dLDxnqVGip3H3hFroVsrpS8TENr5AKkjGc5VVzinshIcPb6J8WjD3//u4PrE+rFxYT4SBG07Kl2i4uLfw==

Key.Gate.N = 96WqKSdYJJ/pslNX4CeQ9PAG54XK8ZxRSUEYypTRT2xs98oBe8Sh5enM7AVWkLuNriqL09FTEPgWis/iebKmlw==

Key.Gate.K = 0JtHtW+oYsQ6GOtUtXgmUQiXE9k+C1qMW42Qf6El/+qR2x7YdYYM//jiwRzKiuN/yFhRt/1Wwd1N4bqBgUhwOw==

--------------------

Client keys: (server.ini)

Server.Auth.N "4142zYQw2nEw24VdMTgzHZLwh2MMOWWJpWro7cg+9uNpit3klrTPjR35gNq9Ic4VbFKs0rDuWnJSg/2kq0q+6w=="

Server.Auth.X "GB1q6/ZCuLdfU0ek5zrIwiL1H2198YGGZip+7q3h6FghwnP7ONNyEdZQKeYzota7PEuY4+P52uLTopcoMv0ZPg=="

Server.Game.N "yEfFVgixz3T1FCp6MXBfUBPV2wGuuzllidaf6AExkdb6oXZOaEsuD4Q+drsvCIZvaw/GGKJPu37LzRdEd4D+Yw=="

Server.Game.X "OUP3vLEyWZG/Nw0CARydnVVxTUg+RA1XeYqlV2iJoYY/Qkg4um6reEbTLwIL17+siRWiqUZUu0o2EhA0hPHS3g=="

Server.Gate.N "96WqKSdYJJ/pslNX4CeQ9PAG54XK8ZxRSUEYypTRT2xs98oBe8Sh5enM7AVWkLuNriqL09FTEPgWis/iebKmlw=="

Server.Gate.X "dR49pO8vWFdqnCLSKDi9jQeyuVDBzCFquna+joruueNthSLhOvZCDMgWQL11/nzfvxNVuanl4+UKsz+42Ub6vA=="

--------------------

You now need to quit the interactive console with the following

[code]

ds-918> quit

[/code]

Remember I suggested running this from putty on a small local workstation an advantage that comes with putty is that if you highlight the created keys they are automatically added to the clipboard and you can paste them into a dummy text file for later.

Cut and paste in Linux command line editors is a pain I choose to use CoreFTP to copy dirtsand.ini to my workstation and edit dirtsand.ini there.

Then I opened the new file “dirtsand.ini” for editing, my editor of choice happens to be CONText.

Take your first block of server Keys (the six lines that start with the word Key) and paste those into the start of your dirtsand.ini

file replacing the dummy that look like the following

#Key.Auth.N = ...

#Key.Auth.K = ...

#Key.Game.N = ...

#Key.Game.K = ...

#Key.Gate.N = ...

#Key.Gate.K = ...

Scroll down and look for the following section you need to change these ip addresses to be the ip address of your server.

# EXTERNAL server addresses -- should probably point to this server for now

File.Host = 123.45.67.89

Auth.Host = 123.45.67.89

Game.Host = 123.45.67.89

I made the mistake of thinking “should probably point to this server for now” meant leave this as 123.45.67.89 when you type it out you realise this is 123456789 Doh!

You need to change these addresses so that they match your shards ip address. So in my case it ends up looking like this because my server ip is 192.168.0.20

# EXTERNAL server addresses -- should probably point to this server for now

File.Host = 192.168.0.20

Auth.Host = 192.168.0.20

Game.Host = 192.168.0.20

While you are editing the file scroll to the bottom and where it says:

Db.Password = MySuperSecretPassword

Here you need to change this so that it has your Dirtsand Database password the example we are using in this document would make this laundrymarker. This is whatever password you made inside postgresql.

Remember if you make a complete hash of any of this you can delete “dirtsand.ini” and make a new clean copy using the “dirtsand.sample.ini” as a master.

You can now take this modified dirtsand.ini and FTP transfer it back into /opt/dirtsand/dirtsand overwriting your existing dirtsand.ini.

Now we need to set up our client:

If you remember at the start, I suggested a clean install of MOULa by default this is dumped into C:>Program Files (x86) in a folder called Uru Live My method involves copying this Uru Live folder to the C:> root this makes it easier for the addition of any data you might create with Korman. You can leave it where it is and modify it there but making a copy also makes it easier if you need to reinstall.

In my case I rename this new folder Enobmort. Our Uru Live remains as is and will still work for Uru Live we are now editing a copy.

Go to your unpacked Destiny and copy the files not in folders into that duplicate Uru Live folder remember mine is C:> Enobmort

Allow it to replace any files with duplicate names.

We are now going to edit server.ini we find in that folder.

This is the copy of the Destiny folder server.ini we will have copied from the Destiny folder we need to edit this.

[code]

Server.Auth.N "z5t8bJyN8xhDbVEBqXAr1S66haK9wEUzbWVLfYMW5NG7F6tN63iDUhCzpFqhtBxlMn4sqSQxn+k2H/r/cEpUVw=="

Server.Auth.X "E/0przoOD8V0pEBX038xZDAIW7gl/zwo8yy5DhzBT43OxqJd2DMzwaNtvkSNuJpYPuKHpNuBSjTY0GUDDGvPRg=="

Server.Game.N "0tw8B8UMuaeMqd+gp5fpwqCkNnLojV7MrIlt0Ef6ZgzjKA57U2J+VOI1E2AZ4SBrRo8ukQmsaodxzfYCDcccsw=="

Server.Game.X "YLbEg7b9nXaPcqn4qlWSMu+pFIx3gZPn2fFOsseQmwbfneiwB8lAlqXvsuwszylYt+IlIcWf8PxJ0qnFV/RHOQ=="

Server.Gate.N "4NuifLYkTghXcY1j7+wND8N/SJHDwBIuPeqKAqIJtTQqLj53JdvqdancI+EIgz9mFEQSfWigNE/IfzyoJvCyqw=="

Server.Gate.X "2NpEoop5ECST7vsNvmHY4ChRL+gjXofpzFmh0zB8D/JAO3y8uyHjnVWjXiNJL8PFy2R94hNiWSIdehpDLGTqLA=="

Server.Gate.Host "mystler.eu"

Server.Auth.Host "mystler.eu"

Server.Auth.G "41"

Server.Game.G "73"

Server.Gate.G "4"

Server.DispName "DestinyURU"

Server.Status "https://destiny.mystler.eu/status.php"

Server.Signup https://destiny.mystler.eu

[/code]

Your second set of keys can be placed directly into the client's server.ini replacing the Destiny Server Keys

You also need to change

Server.Gate.Host "mystler.eu"

Server.Auth.Host "mystler.eu"

You need to make it look like this so that it contains your shard ip address so in my case it is

Server.Gate.Host 192.168.0.20

Server.Auth.Host 192.168.0.20

Change the last three lines to look like this you can have whatever you want as a display name and until you have status or signup just remove the pointers to Mystlers shard lets face it at this time pointing to his shard is stupid.

Server.DispName "EnobmortURU"

Server.Status

Server.Signup

Check your new folder for a Python.pak file I do not have one and if you did a recent download you probably don’t have one either but if you do delete that file it is not needed.

Back at the start you downloaded and unpacked the moul-scripts from the github. We need to drag copy the dat, SDL and Python folders from that github download into our new directory replacing any files it finds with a name clash.

You are now going to go to that folder and edit server.ini you find in that folder.

So, my server.ini now looks like this.

[code]

Server.Auth.N "4142zYQw2nEw24VdMTgzHZLwh2MMOWWJpWro7cg+9uNpit3klrTPjR35gNq9Ic4VbFKs0rDuWnJSg/2kq0q+6w=="

Server.Auth.X "GB1q6/ZCuLdfU0ek5zrIwiL1H2198YGGZip+7q3h6FghwnP7ONNyEdZQKeYzota7PEuY4+P52uLTopcoMv0ZPg=="

Server.Game.N "yEfFVgixz3T1FCp6MXBfUBPV2wGuuzllidaf6AExkdb6oXZOaEsuD4Q+drsvCIZvaw/GGKJPu37LzRdEd4D+Yw=="

Server.Game.X "OUP3vLEyWZG/Nw0CARydnVVxTUg+RA1XeYqlV2iJoYY/Qkg4um6reEbTLwIL17+siRWiqUZUu0o2EhA0hPHS3g=="

Server.Gate.N "96WqKSdYJJ/pslNX4CeQ9PAG54XK8ZxRSUEYypTRT2xs98oBe8Sh5enM7AVWkLuNriqL09FTEPgWis/iebKmlw=="

Server.Gate.X "dR49pO8vWFdqnCLSKDi9jQeyuVDBzCFquna+joruueNthSLhOvZCDMgWQL11/nzfvxNVuanl4+UKsz+42Ub6vA=="

Server.Gate.Host 192.168.0.20

Server.Auth.Host 192.168.0.20

Server.Auth.G "41"

Server.Game.G "73"

Server.Gate.G "4"

Server.DispName "EnobmortURU"

Server.Status

Server.Signup

[/code]

Obviously your ip address could well be different. But might be the same.

Note I changed the Server.DispName to Enobmort.URU were this actually displays I will add here when I figure it out lol.

Notice I have removed the server.status and server.signup information if you want these because you are going to make your shard public you are going to need to make the required files available and that is another reason we installed apache2 and php7.2 back at the start in a little bit of future proofing. For now these remain blank.

You will at least need the folders indicated in the path statements so let us make sure we are in the right folder this should be /opt/dirtsand and you need to make four new sub directories. But if you recall we did make those earlier. If you did not do that now

[code]

$cd /opt/dirtsand

$mkdir data

$mkdir authdata

$mkdir SDL

$mkdir ages

[/code]

At this time we are working without a data server this is an added step we will add at a later point for now it is a level of complexity we can avoid.

I make a copy of my dat folder in my C:> Enobmort folder and rename that ages and then using CoreFTP I copy the two directories ages and SDL into /opt/dirtsand overwriting those folders I want to be logged into CoreFTP using my dirtsand soilgrit login to preserve the rights.

RUN THE SERVER:

Assuming everything else went smoothly, you should now be able to start your server and connect to it! You'll have to create an account first, which can be done from the console:

Start by making sure you are in the right folder. Notice there is no / before the bin/dirtsand and you will get some messages about stuff not found don’t worry it is just because this will be the first time you have run this with a password in place.

The Number after ds- appears to be a version and may be different.

[code]

$cd /opt/dirtsand/dirtsand

$bin/dirtsand dirtsand.ini

ds-918> addacct <username> <password>

[/code]

Avoid using account names and user names with the apostrophe ‘ because that is the key used to open the Plasma Console in due course you may be able to fix this but for the time being stick to letters and numbers and no spaces.

I took the opportunity to make two accounts I am logging in from two machines this is not the avatar but the account this is like when you log into MOULa it asks for a login and after that you can make a few avatars but you want multiple avatars in game simultaneously each one must be on a different account.

[edit]

Apparent finished structure (subject to revision and addition of more detail) in response to points raised by Korovev later in this thread.

I wish to thank Korovev for drawing attention to this omision

[code]

root

\opt

dirtsand

ages (unencrypted copied in from Destiny duplicated in my client folder but called dat in that directory)

SDL (unencrypted copied in from Destiny duplicated in my client folder as is)

authdata (empty)

data (empty)

string\_theory (cloned stuff from git most in the following sub-directories plus 4 files )

.git

build (results of cmake ..)

cmake

doc

include

src

test

dirtsand (cloned stuff from git most in sub directories plus 16 other files on of which is your modified dirtsand.ini)

.git

AuthServ (contains file Hoikas tells you to edit AuthDaemon.cpp)

bin

build (results of cmake -DCMAKE\_INSTALL\_PREFIX = /opt/dirtsand .. (Note:after Hoikas suggested modification of root/opt/dirtsand/dirtsand/AuthServ/AuthDaemon.cpp)

cmake

db (contains .sql files used to pre stuff postgresql database)

FileServ

GameServ

NetIO

PlasMOUL

SDL

Types

[/code]

[/edit]

If you cannot connect for some reason, make sure you copied the keys and server addresses correctly into the client's server.ini and its corresponding set in dirtsand.ini on the shard.

Note I actually use my little notebook to putty log onto my built shard as dirtsand I change the directory so that my working directory is /opt/dirtsand/dirtsand before I run the shard I run this with bin/dirtsand dirtsand.ini

The dirtsand keys generated by a previous install appear to work with a fresh install as long as you use the pair of Server & Client.

Leave this running go to your client PC and right drag plClient.exe from your working directory onto your desktop and create a shortcut, rename this shortcut in my case Enobmort right click select properties on the end of the target line add a space followed by -LocalData include the hyphen capitalized as is with no space.

Apply and try using the useracct (User Account) login you created.

[HEADS UP]

Under no condition log in and create an Avatar UNTIL you have your SDL files on Both your Client and Server FAILURE to do this will screw your login. The couple of times I have been stupid enough to do this I ended up Rebuilding my Shard.

[/HEADS UP]