MEDIA TEMPLE AMAZON S3 BACKUP FOR DEDICATED VIRTUAL SERVERS

A comprehensive guide to installing, configuring, and automating Media Temple backups to Amazon S3.

JAY D. ROGERS

ServerSideUp

Media Temple Amazon S3 Backup for DV (this works on other Linux servers too)

I’ve been using Media Temple for hosting and overall it has been a very pleasant experience. There are some things that are a little inconvenient, especially things like no native support for a Media Temple Amazon S3 backup. You can’t blame Media Temple for that though… it’s mainly a Plesk thing. Fortunately enough, Plesk stays enough out of your way where you can set this up on your own. We can utilize free tools from Amazon to automate backups from MediaTemple DV 4.5 servers.

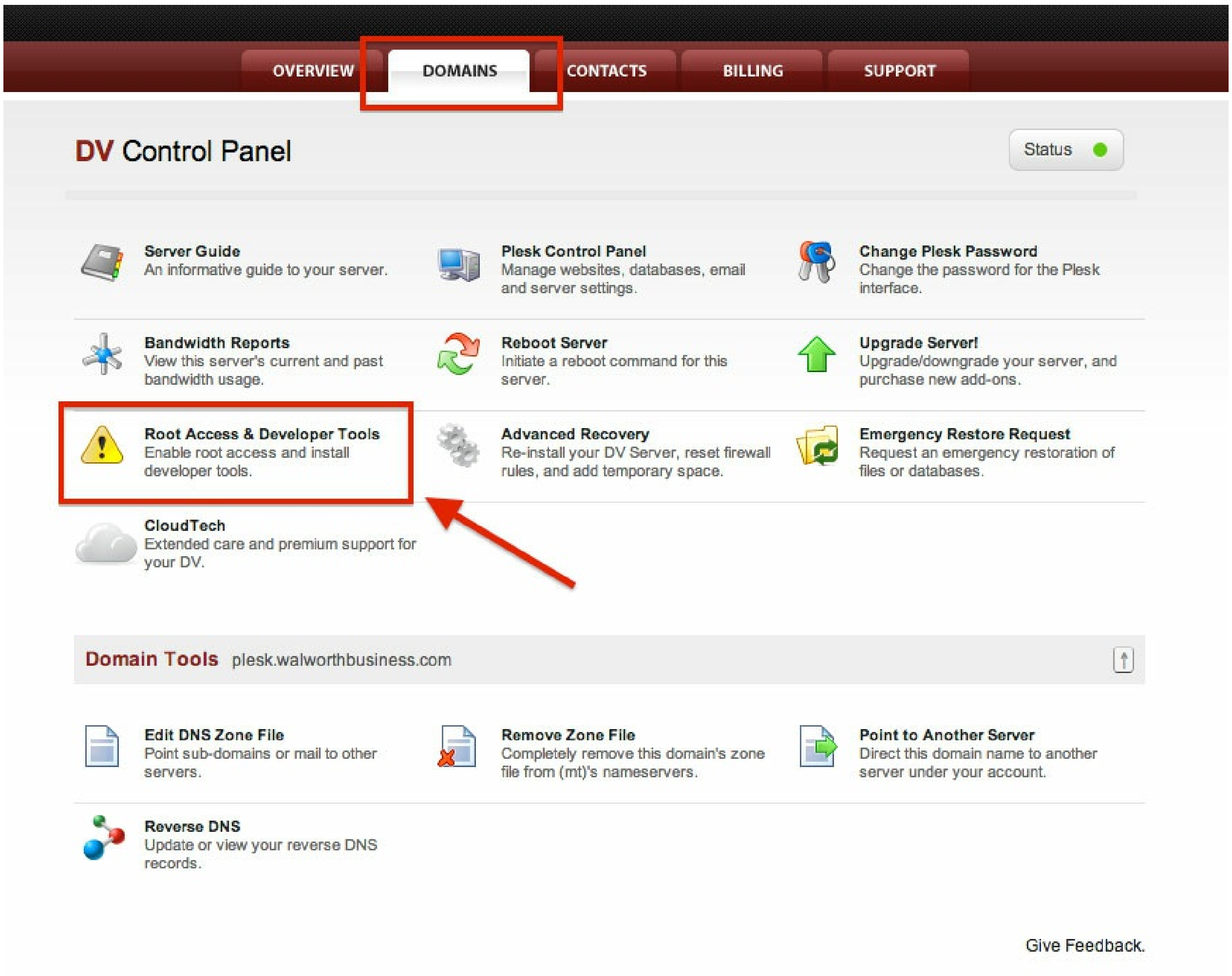
```
**Disclaimer**

Media Temple recently has shifted their policies to frown upon roo
```

Prepare The Server

Enable Root Access

Now since I've got the legal mumbo-jumbo taken care of, lets put the rubber to the asphalt by making sure that you have root access. We will be making changes to the system that require higher permissions, so if you are not doing this with the root account — we won't be getting too far. By default, new Dedicated Virtual servers are shipped with root access disabled. You can do this in the Account Center at Media Temple.

Now since I've got the legal mumbo-jumbo taken care of, lets put the rubber to the asphalt by making sure that you have root access. We will be making changes to the system that require higher permissions, so if you are not doing this with the root account — we won't be getting too far. By default, new Dedicated Virtual servers are shipped with root access disabled. You can do this in the Account Center at Media Temple.



DV Control Panel

Server Guide An informative guide to your server.

Plesk Control Panel Manage websites, databases, email and server settings.

Change Plesk Password Change the password for the Plesk interface.

Bandwidth Reports View this server's current and past bandwidth usage.

Reboot Server Initiate a reboot command for this server.

Upgrade Server! Upgrade/downgrade your server, and purchase new add-ons.

Root Access & Developer Tools Enable root access and install developer tools.

Advanced Recovery Re-install your DV Server, reset firewall rules, and add temporary space.

Emergency Restore Request Request an emergency restoration of files or databases.

CloudTech Extended care and premium support for your DV.

Domain Tools plesk.walworthbusiness.com

Edit DNS Zone File Point sub-domains or mail to other servers.

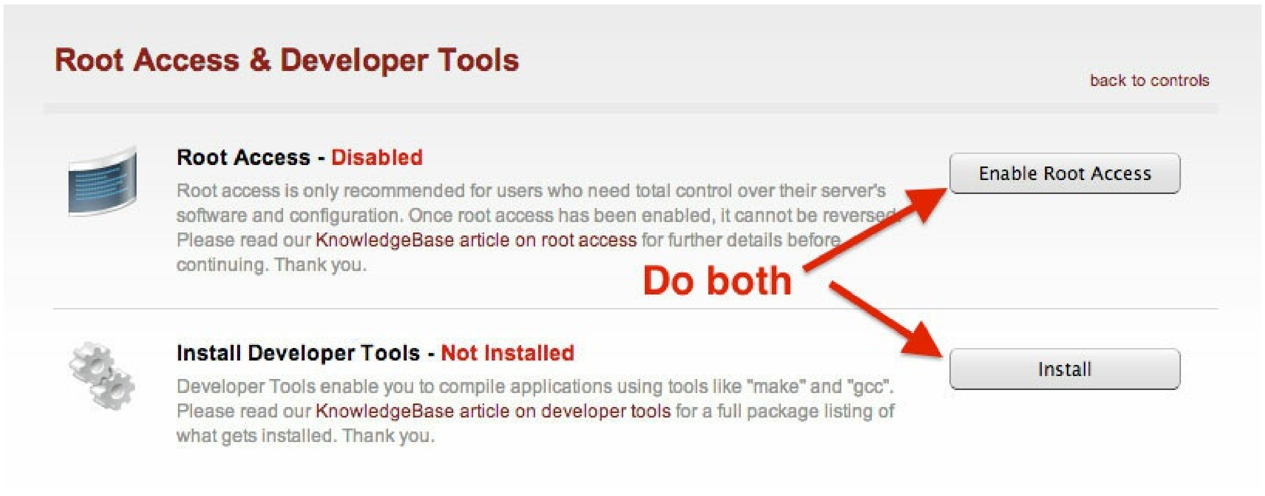
Remove Zone File Completely remove this domain's zone file from (mt)'s nameservers.

Point to Another Server Direct this domain name to another server under your account.

Reverse DNS Update or view your reverse DNS records.

Give Feedback.

When you click on that, you will need to configure BOTH items. This means you will have to **enable root access** and install the **Developer Tools**.


Root Access & Developer Tools

back to controls

Root Access - Disabled

Root access is only recommended for users who need total control over their server's software and configuration. Once root access has been enabled, it cannot be reversed. Please read our KnowledgeBase article on root access for further details before continuing. Thank you.

Install Developer Tools - Not Installed

Developer Tools enable you to compile applications using tools like "make" and "gcc". Please read our KnowledgeBase article on developer tools for a full package listing of what gets installed. Thank you.

This may take a minute or two for this to complete. That's okay, it is a perfect amount of time to grab a coffee/beer and come back to a screen with green text saying "Enabled" and "Installed".

Download and Install Amazon AWS CLI Tools

Now that we have the correct access and libraries to do some compiling, we need to download Amazon AWS CLI Tools (Amazon Command Line Interface Tools). These tools will allow us to connect to Amazon's services without installing all of this weird poo poo to get connected (been there, done that). Servers generally do not like poo poo so it is important that these tools are officially supported by Amazon and they do not make Plesk crap itself.

Before we dive in and start running Amazon commands, to the date of this post these tools do require Python 2.6 or higher. To do that, simply run the `python --version` command to see what version you are running. It should echo out the version on the line below.

Check Python Version (must be 2.6 or higher)

Below you can see the output of the `python --version` command:

```
[root@myserver ~]# python --version
Python 2.6.6
```

Sweet, it looks like I am good to go on this server. Now we have to download the tools… OH WAIT! *There's more!* Remember my line above where I said we do not have to download a bunch of weird poo poo? We do have to download something called pip (which is a package manager for Python), but trust me — this is very safe based off of the other methods that I have seen. Pip will simplify the installation process for us.

Install pip (Python Package Manager)

The command below will download the latest version of `pip` and execute the installation:

```
wget --no-check-certificate https://raw.github.com/pypa/pip/master
```

Install Amazon AWS CLI

Sweet! We have `pip` installed on our server, so lets use `pip` to install Amazon's CLI Tools:

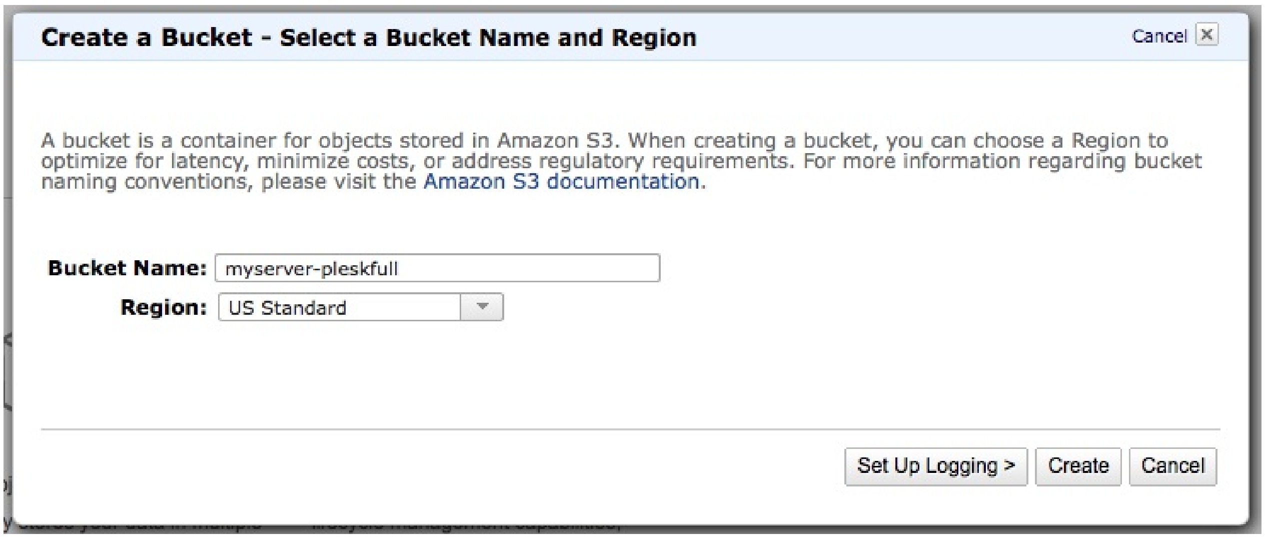
```
pip install awscli
```

Well, that was quite simple… check for errors, but if everything looks good — we are now ready to prepare our connection to Amazon S3.

Configure Amazon Access

Before we start hacking away to get this server connected, we need to make sure that we have S3 buckets created and the right credentials to run a backup. Log into Amazon AWS. Once you are logged in, then proceed to click on *S3* then you should be able create your Amazon S3 bucket using the on-screen prompts.

In this example, I will be using a bucket named `myserver-pleskfull`, but your bucket name must be original and not conflict with any other bucket names on Amazon S3. Whatever you name this bucket, we will configure full server Plesk backups to go into this bucket. Make sure you change my code to whatever your bucket name is as you progress through the book.



Create a Bucket - Select a Bucket Name and Region

A bucket is a container for objects stored in Amazon S3. When creating a bucket, you can choose a Region to optimize for latency, minimize costs, or address regulatory requirements. For more information regarding bucket naming conventions, please visit the Amazon S3 documentation.

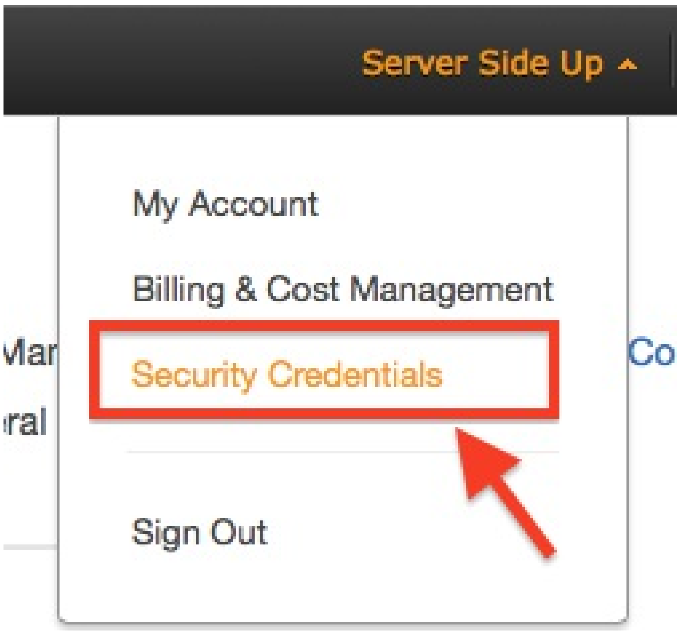
Bucket Name: myserver-pleskfull

Region: US Standard

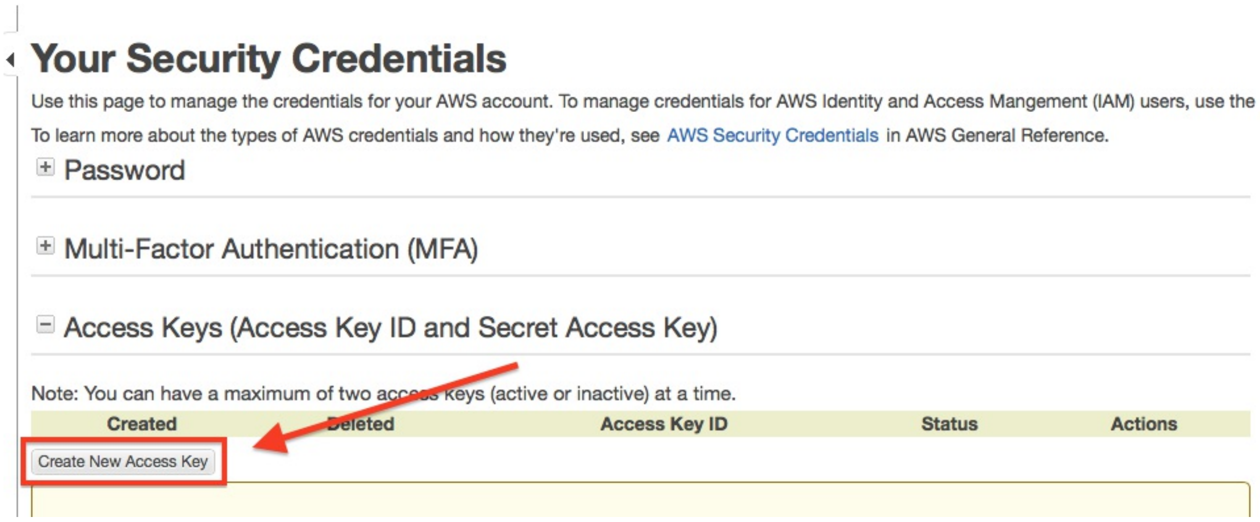
```
***Best Practice Note***
If this Amazon account is only going to be accessed by one user, o
```

Create Access Keys

When you are logged into the AWS Console, click on your name and then choose security credentials.



Then choose *Create New Access Key*:



Your Security Credentials

Use this page to manage the credentials for your AWS account. To manage credentials for AWS Identity and Access Mangement (IAM) users, use the

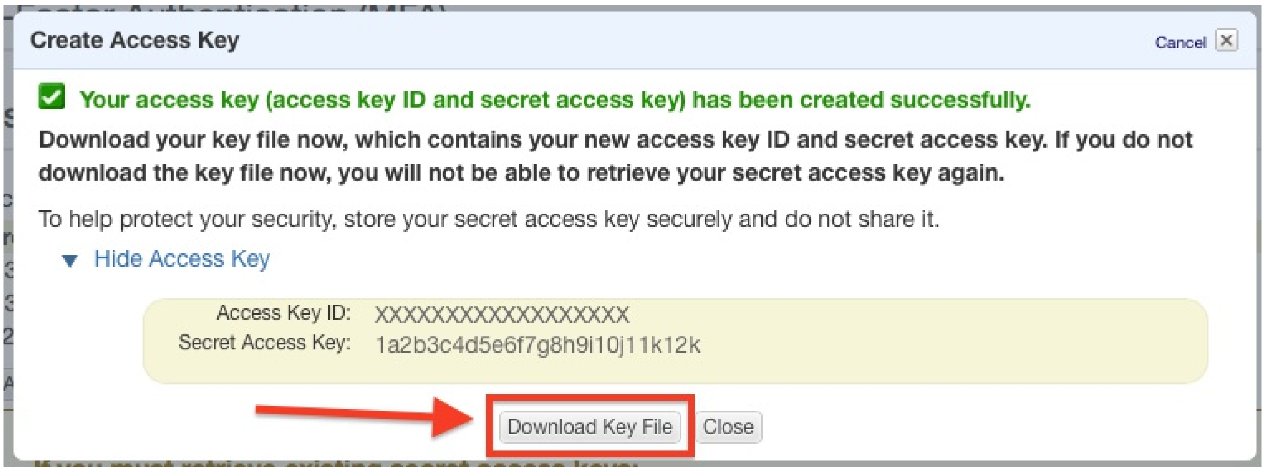
To learn more about the types of AWS credentials and how they're used, see AWS Security Credentials in AWS General Reference.

Note: You can have a maximum of two access keys (active or inactive) at a time.

Created	Deleted	Access Key ID	Status	Actions

Password
Multi-Factor Authentication (MFA)
Access Keys (Access Key ID and Secret Access Key)
Create New Access Key

When you create the keys, be sure to download them — because you will NEVER see them again otherwise!


Create Access Key

Your access key (access key ID and secret access key) has been created successfully.

Download your key file now, which contains your new access key ID and secret access key. If you do not download the key file now, you will not be able to retrieve your secret access key again.

To help protect your security, store your secret access key securely and do not share it.

Access Key ID:	XXXXXXXXXXXXXXXXXX
Secret Access Key:	1a2b3c4d5e6f7g8h9i10j11k12k

Once you have that downloaded, run `aws configure` you will be prompted to enter your credential information. Here is the output of the `aws configure` command:

```
[root@myserver ~]# aws configure
AWS Access Key ID [None]: XXXXXXXXXXXXXXXXXXXX
AWS Secret Access Key [None]: 1a2b3c4d5e6f7g8h9i10j11k12k
Default region name [None]: us-east-1
Default output format [None]:<LEAVE BLANK -- JUST PRESS ENTER>
```

Important Note for Default Region

For the default region, I chose us-east-1, but you can choose whatever region you would like by seeing the list from Amazon here. For what we are doing in the scope of this article, the regions are irrelevant to Amazon S3 but if you do any management of other AWS regions — it may be very important to set this correctly. You can also see that I left *Default output format [None]:* blank. This will default to JSON. You can read more about your options here.

Test Amazon Access

Now that we have our authentication configured, you should now be able to test it out. Create some test files in your home folder on the server you will be backing up:

```
[root@myserver ~]# mkdir -p test-parent-folder/
[root@myserver ~]# touch testfolder/test1.txt
```

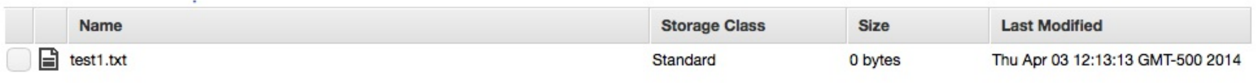
If you run `ll` or `ls -l` you should see your folder there. Now lets run the command to move it up to our S3 Bucket. Remember the bucket name that we chose from above is `myserver-pleskfull`. The syntax for the `aws` command is the following:

```
aws [options] <command> <subcommand> [parameters]
```

Using the syntax above (advanced syntax can be found here), we can now run our move command and watch its results:

```
[root@myserver ~]# aws s3 mv testfolder/ s3://myserver-pleskfull -
move: testfolder/test1.txt to s3://myserver-pleskfull/test1.txt
```

Now when you check the bucket in the Amazon Console, you can see that it appears (notice how it copied just the file, not the folder):



	Name	Storage Class	Size	Last Modified
	test1.txt	Standard	0 bytes	Thu Apr 03 12:13:13 GMT-500 2014

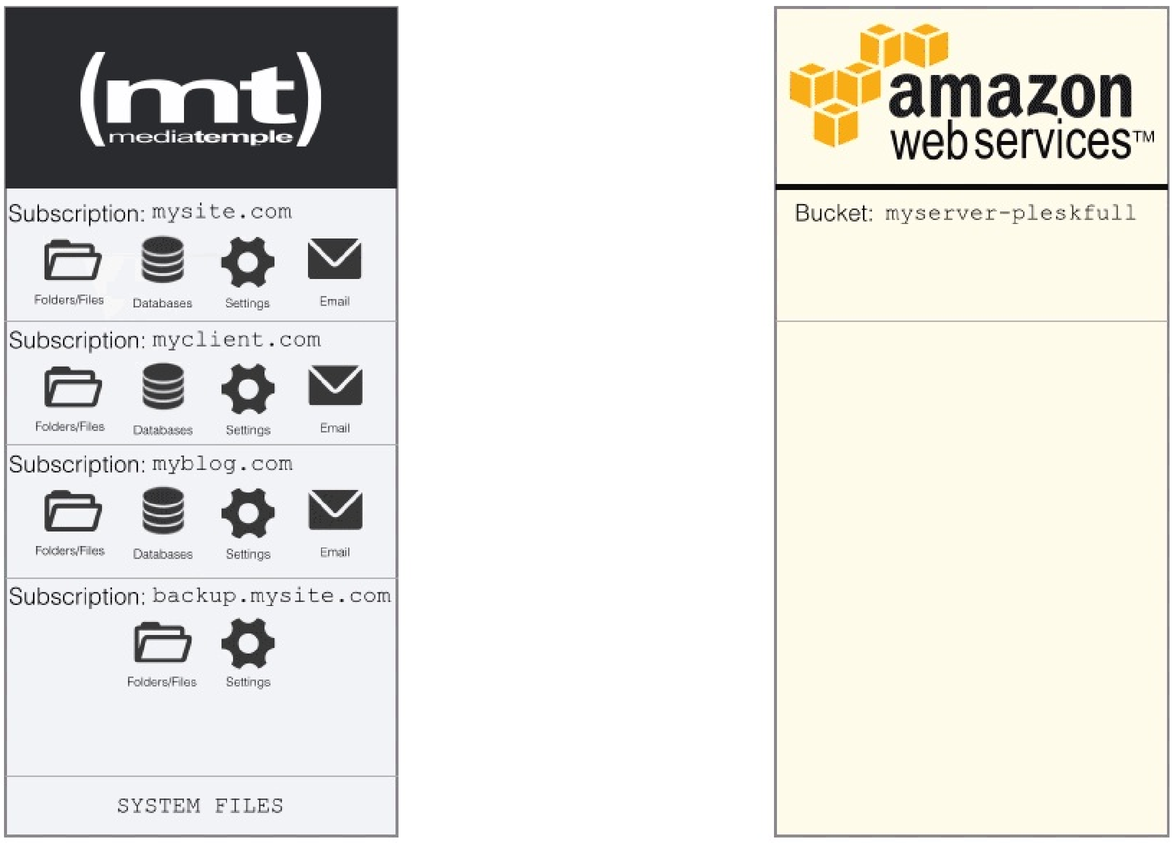
You can also check via command line:

```
[root@myserver ~]# aws s3 ls s3://myserver-pleskfull-full --recurs
2014-04-03 12:13:13          0 test1.txt
```

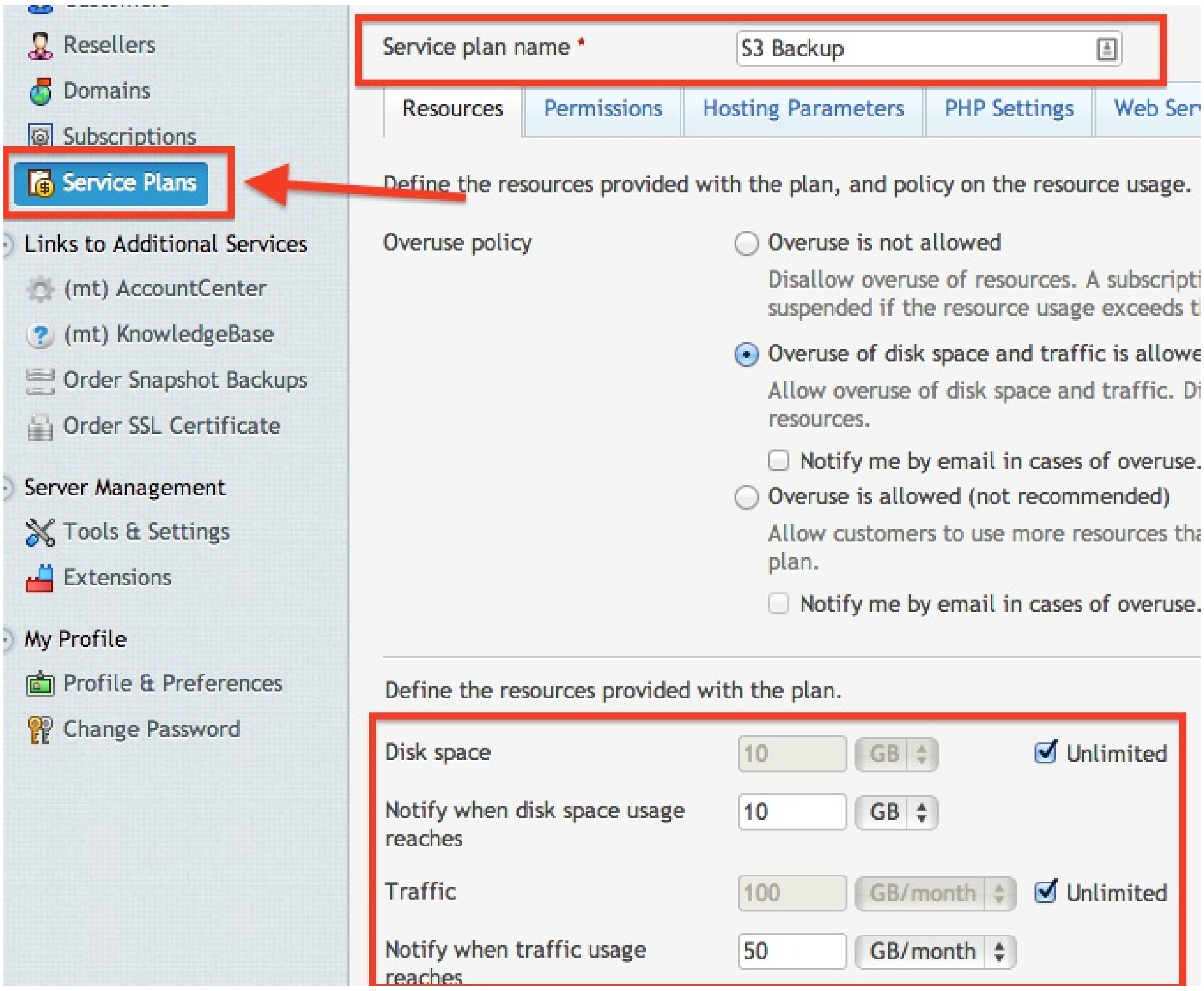
Configure Plesk for Full Backup

So this is going to be interesting on how this works… We are going to use the native Plesk Backup Manager, but we are going to choose the FTP solution and the server will essentially FTP the backup file to itself. Sounds very strange, but there are a few reasons why I did it this way which we will discover as we go through the set up.

Here is an image that essentially demonstrates our full server backup:



Now since we have the logic down, we have to create a Subscription to store our backups. But before we do that, we need to make sure that we have a Plan that will allow us to do that. Plesk would get quite emotional if we tried to store our 5GB backup in a plan that only allowed 1GB. To do this, select *Service Plans*, choose *Add New Plan*. Set the name to *S3 Backup* and make sure that you allow Unlimited storage a subscription and Traffic (you can choose whatever subdomain that you want). It would also be a wise choice to set the notifications to make sure that your subscription does not get out of control. Set these values to whatever you think suits your server best. Since our backups will be *MOVED* (not copy) to Amazon S3, we don't have to worry about the subscription keeping an archive of backups.



Service plan name * S3 Backup

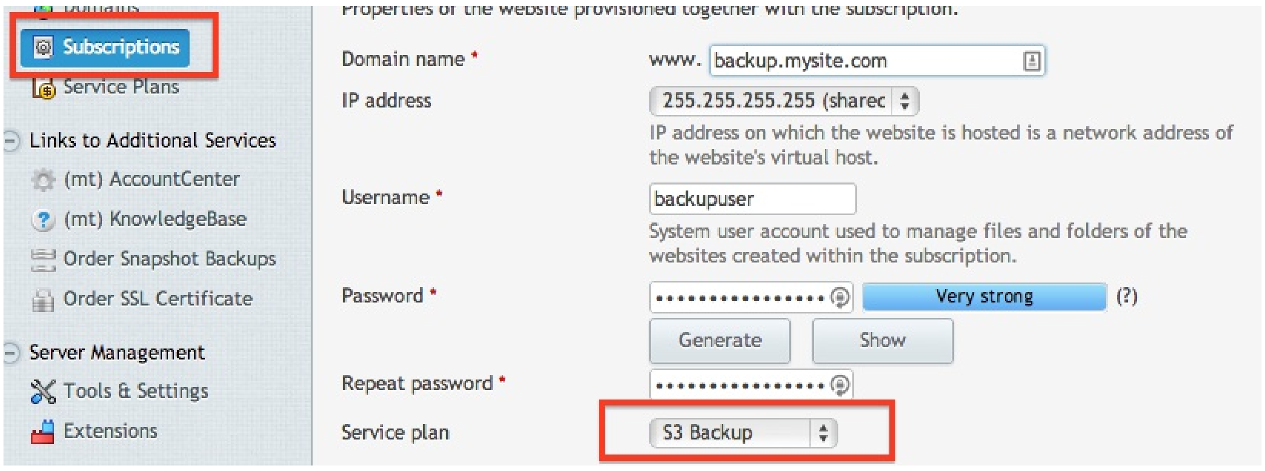
Define the resources provided with the plan, and policy on the resource usage.

Overuse policy
- ○ Overuse is not allowed
  Disallow overuse of resources. A subscripti suspended if the resource usage exceeds t
- ● Overuse of disk space and traffic is allowe
  Allow overuse of disk space and traffic. Di resources.
  - ☐ Notify me by email in cases of overuse.
- ○ Overuse is allowed (not recommended)
  Allow customers to use more resources tha plan.
  - ☐ Notify me by email in cases of overuse.

Define the resources provided with the plan.

Disk space	10	GB	☑ Unlimited
Notify when disk space usage reaches	10	GB	
Traffic	100	GB/month	☑ Unlimited
Notify when traffic usage reaches	50	GB/month	

Once you have the Service Plan created, it’s now time to create the Subscription. You can use any domain that you would like, but it just has to **exist and have the DNS properly pointed to your server’s IP address**. Once you have your domain configured, create the Subscription and assign it to the new plan that we created.



Properties of the website provisioned together with the subscription.

Domain name * www. backup.mysite.com

IP address 255.255.255.255 (sharec

IP address on which the website is hosted is a network address of the website's virtual host.

Username * backupuser

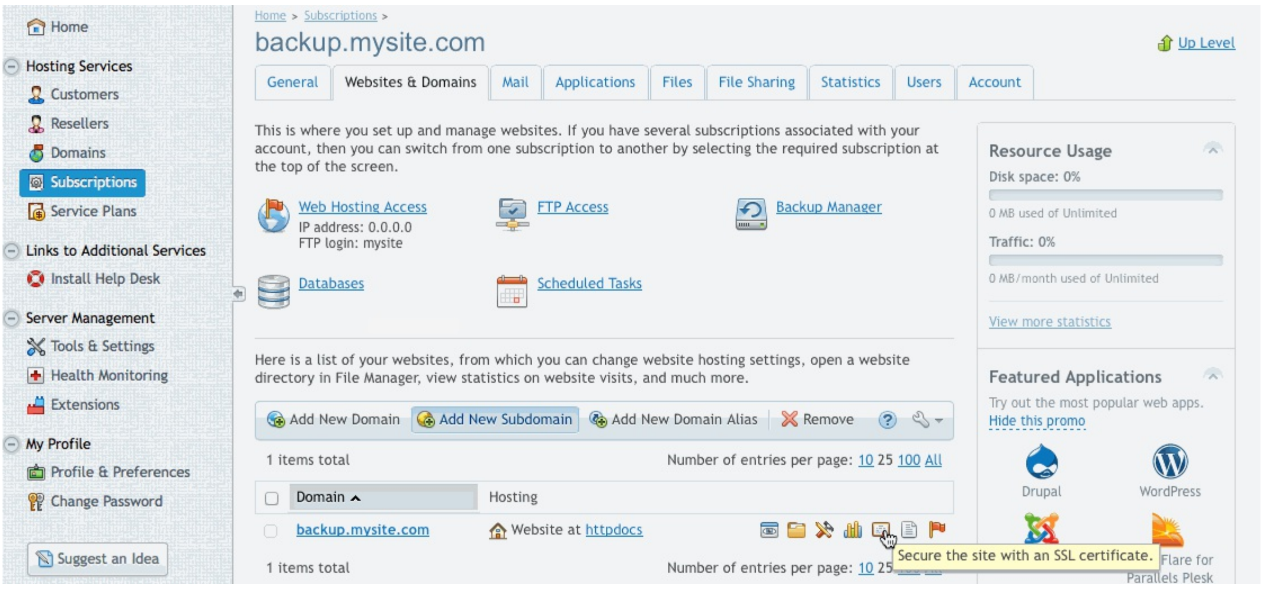
System user account used to manage files and folders of the websites created within the subscription.

Password * ••••••••••••••••

Repeat password * ••••••••••••••••

Service plan S3 Backup

Once the subscription is created, let’s lock down that subscription so we don’t have any data leaks. We won’t be needing any web hosting, only FTP:


Home > Subscriptions >

backup.mysite.com

Up Level

General | Websites & Domains | Mail | Applications | Files | File Sharing | Statistics | Users | Account

This is where you set up and manage websites. If you have several subscriptions associated with your account, then you can switch from one subscription to another by selecting the required subscription at the top of the screen.

Web Hosting Access
IP address: 0.0.0.0
FTP login: mysite

FTP Access

Backup Manager

Databases

Scheduled Tasks

Here is a list of your websites, from which you can change website hosting settings, open a website directory in File Manager, view statistics on website visits, and much more.

Add New Domain | Add New Subdomain | Add New Domain Alias | Remove

1 items total — Number of entries per page: 10 25 100 All

Domain	Hosting
backup.mysite.com	Website at httpdocs

Secure the site with an SSL certificate.

1 items total — Number of entries per page: 10 25

Resource Usage
Disk space: 0%
0 MB used of Unlimited
Traffic: 0%
0 MB/month used of Unlimited
View more statistics

Featured Applications
Try out the most popular web apps.
Hide this promo

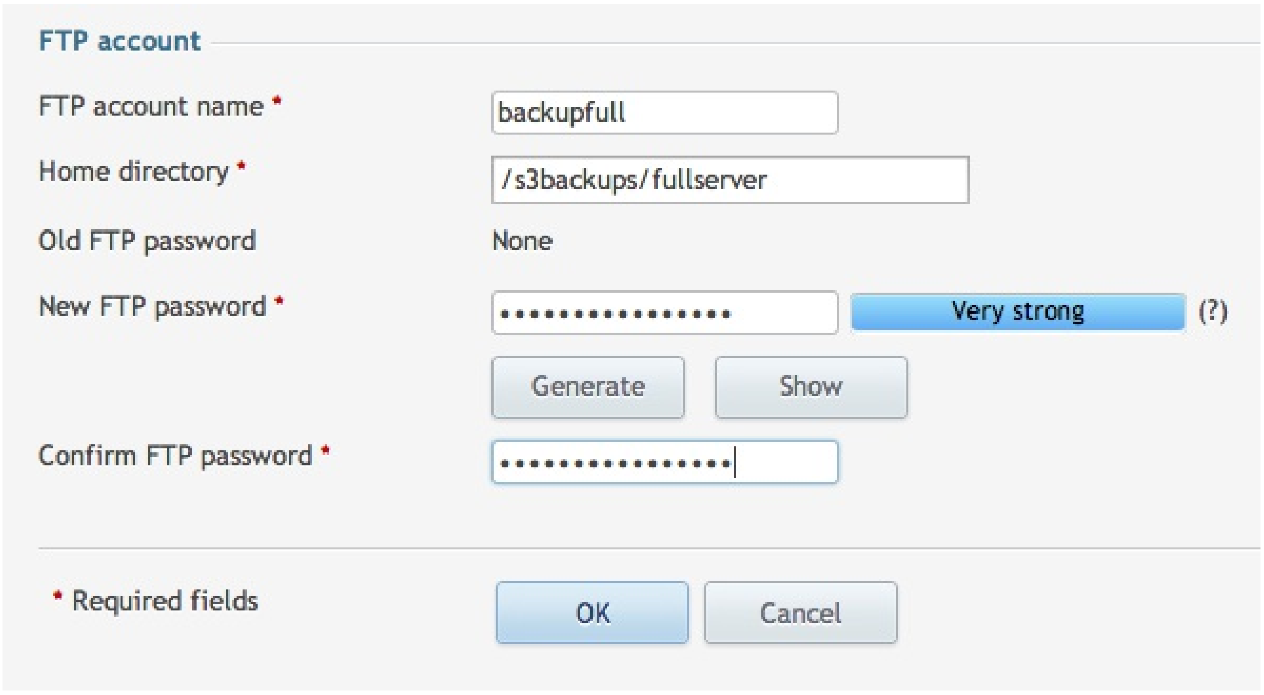
Now that we have the web hosting disabled, we can create our directories for our backup (be sure to change the path to the domain name that you chose):

```
mkdir -p /var/www/vhosts/backup.mysite.com/s3backups/fullserver
```

Before we get too excited, we need to make sure that our `backupuser` has ownership of the `s3backups` folder.

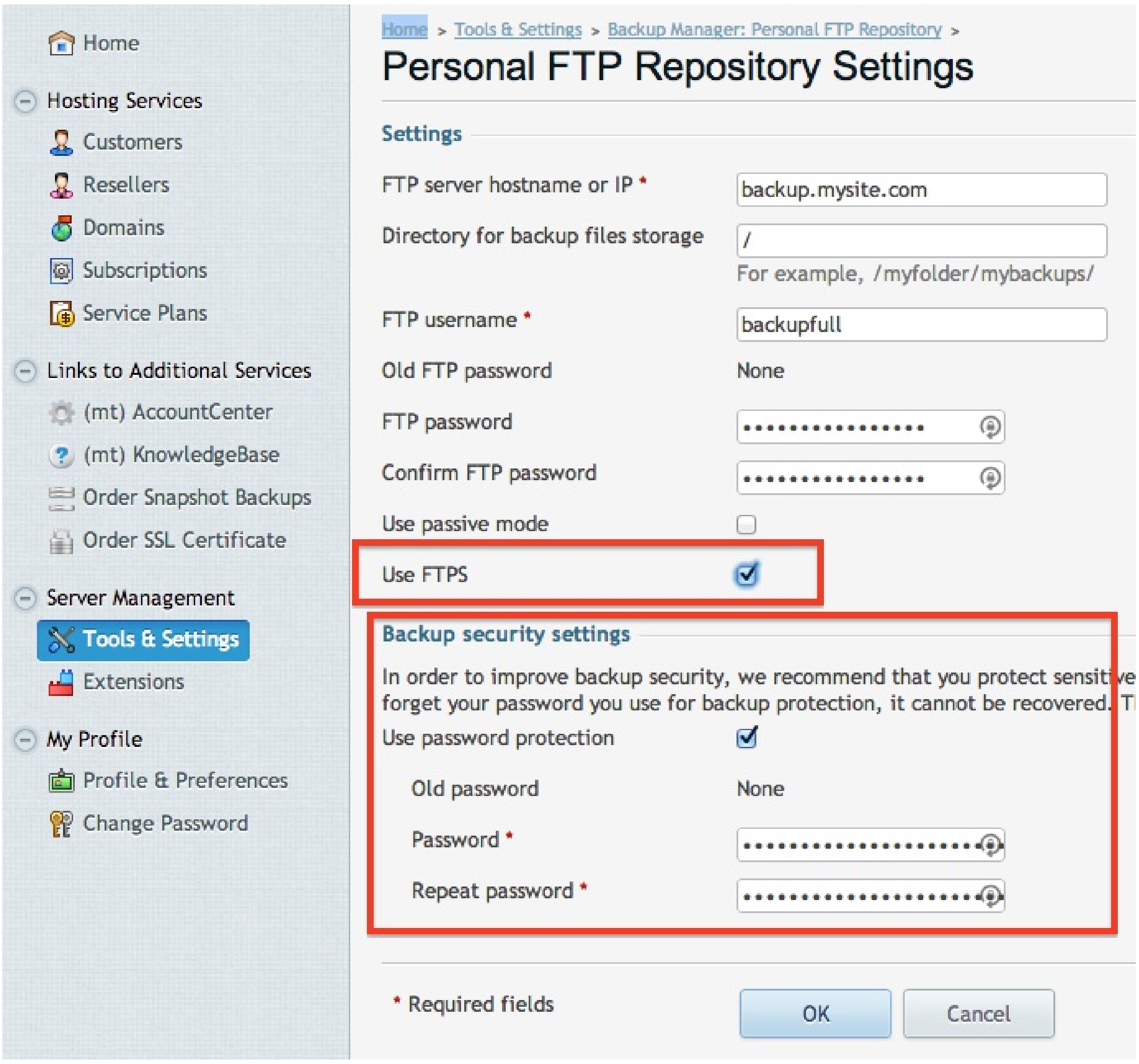
```
chown -R backupuser:psacln /var/www/vhosts/backup.mysite.com/s3bac
```

This directory that we just created is what we will be using to store our full server backups. We now need to create an FTP account that will have access only to this directory.


FTP account

FTP account name *	backupfull
Home directory *	/s3backups/fullserver
Old FTP password	None
New FTP password *	•••••••••••••••• Very strong (?)
Confirm FTP password *	••••••••••••••••

The user is now configured, but we need to tell our server where to run the backup. Go to *Tools & Settings* and then choose *Personal FTP Settings*. Here is where you can enter in the information that we just created. When you are filling this out, be sure to select **FTPS**. Even though we are sending it to ourselves, we want to make sure that we use **FTPS** to secure the transfer. Also, it is a wise idea to create an archive password. This is another step that will prevent unauthorized hands from getting on your data. **Be sure to save these passwords in a safe place!**:



Personal FTP Repository Settings

Settings

FTP server hostname or IP *	backup.mysite.com
Directory for backup files storage	/ For example, /myfolder/mybackups/
FTP username *	backupfull
Old FTP password	None
FTP password	••••••••••••••••
Confirm FTP password	••••••••••••••••
Use passive mode	☐
Use FTPS	✓

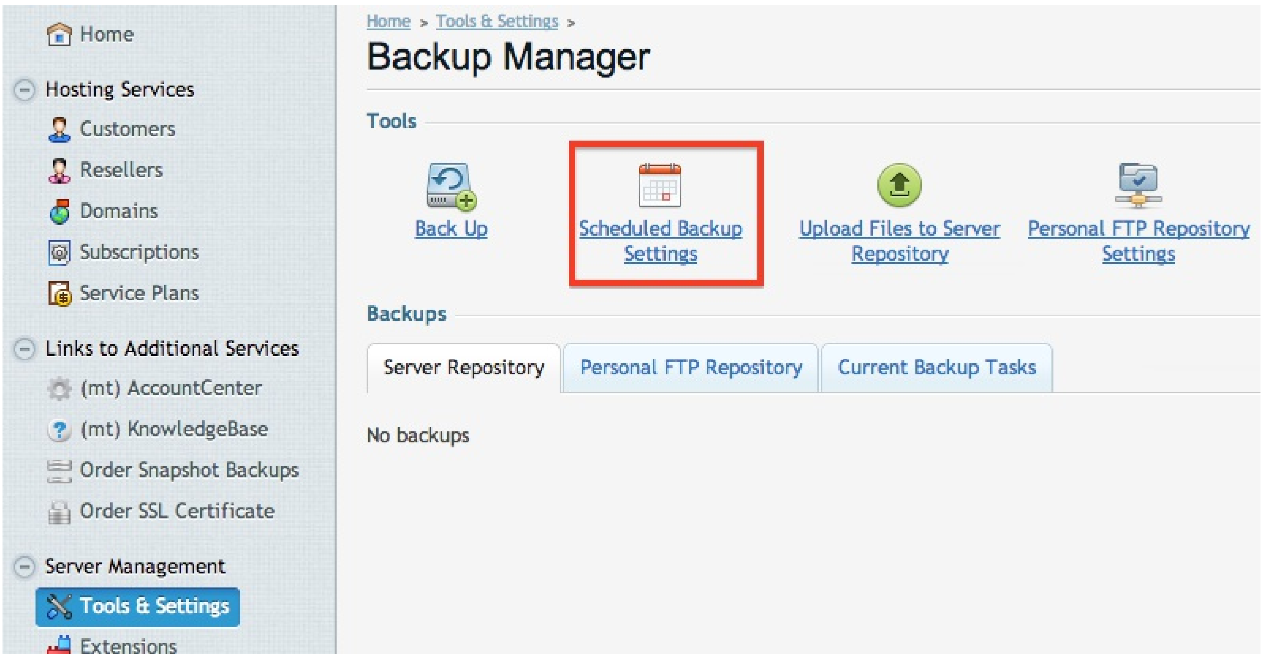
Backup security settings

In order to improve backup security, we recommend that you protect sensitive
forget your password you use for backup protection, it cannot be recovered. T

Use password protection	✓
Old password	None
Password *	••••••••••••••••••••
Repeat password *	••••••••••••••••••••

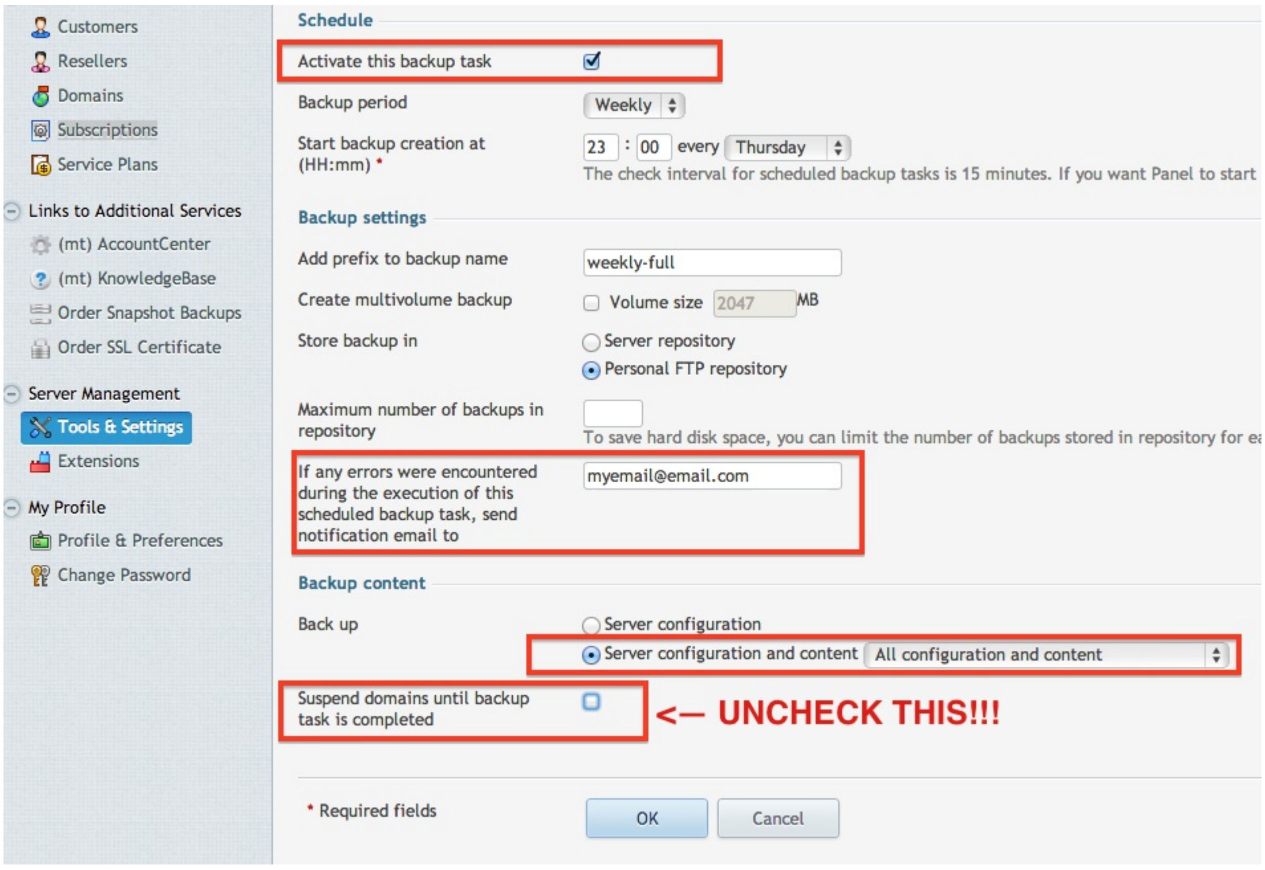
* Required fields

With the FTP settings set, we can now schedule our backup to run. Choosing the frequency of the backup is all up to you. Yes, Amazon is only pennies per GB, but it can add up fast if you are running large backups frequently. Go back to the *Backup Manager* then choose *Scheduled Backup Settings*.



No backups

When you are configuring your settings, be sure that you click Activate at the top (many people forget to select that). Fill out the other fields anyway you want, but you can put in your email for notifications upon any errors (very good idea). Also, make sure that you select *All Configuration and Content* and make sure you uncheck the domain suspension. This will disable your sites while the backup runs. Only in special cases would you want that to happen.



Schedule

Activate this backup task	☑
Backup period	Weekly
Start backup creation at (HH:mm) *	23 : 00 every Thursday The check interval for scheduled backup tasks is 15 minutes. If you want Panel to start

Backup settings

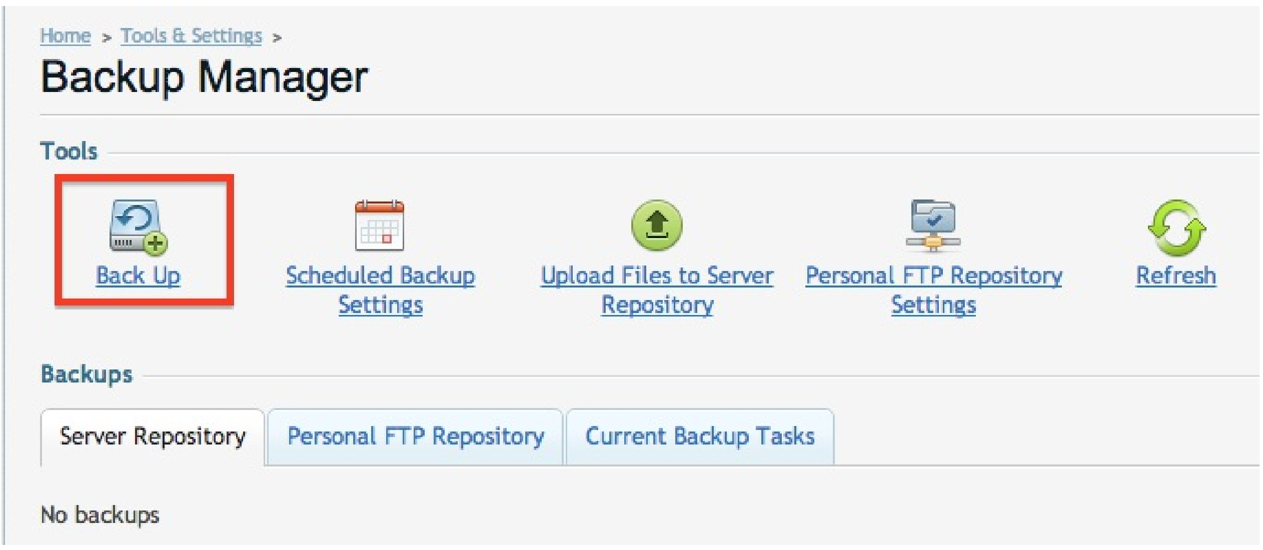
Add prefix to backup name	weekly-full
Create multivolume backup	☐ Volume size 2047 MB
Store backup in	○ Server repository ● Personal FTP repository
Maximum number of backups in repository	 To save hard disk space, you can limit the number of backups stored in repository for e
If any errors were encountered during the execution of this scheduled backup task, send notification email to	myemail@email.com

Backup content

Back up	○ Server configuration ● Server configuration and content All configuration and content
Suspend domains until backup task is completed	☐

Testing The Backup

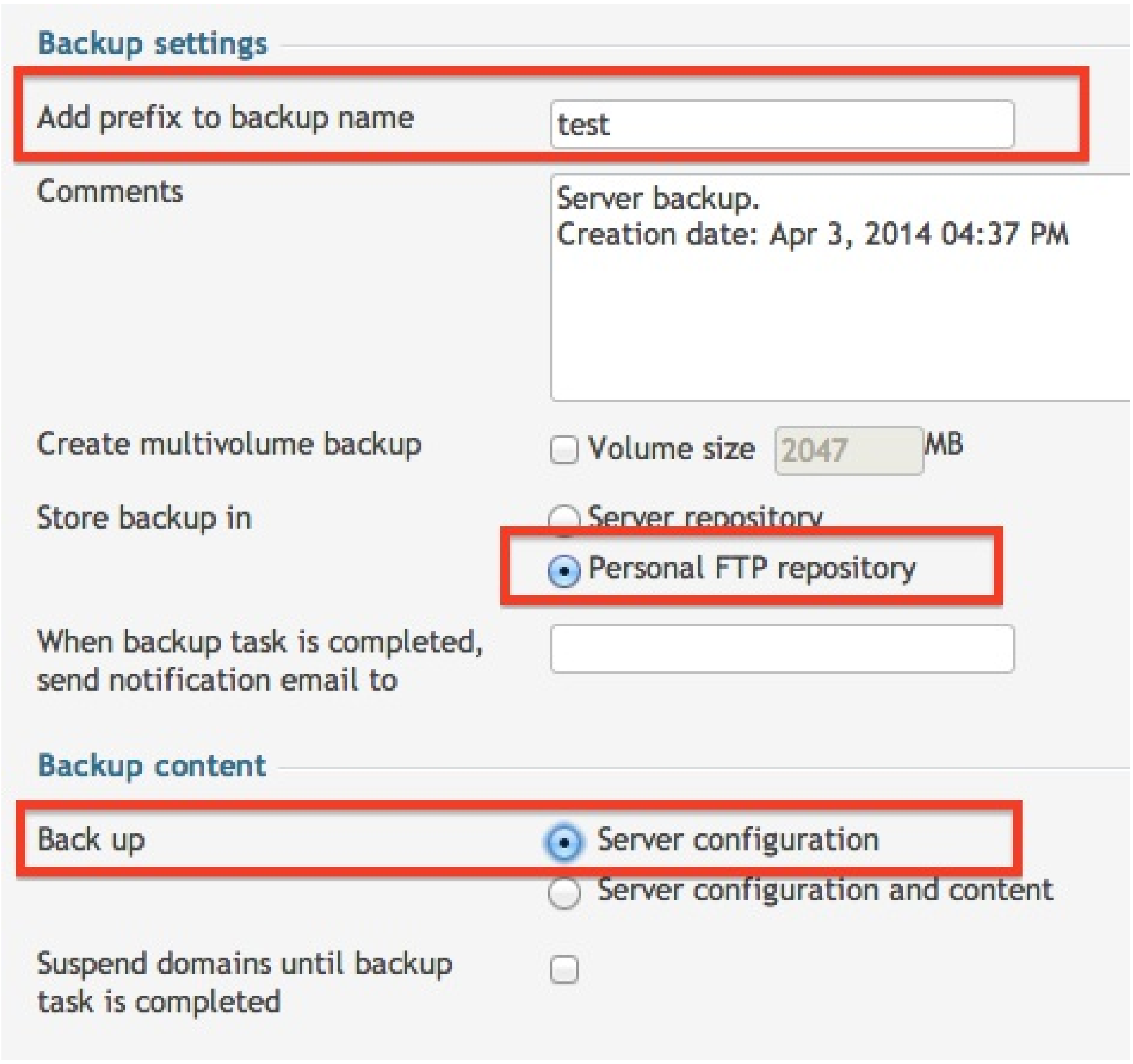
We're almost there! Now we can test it by choosing *Backup* from the Backup Manager.



Home > Tools & Settings >
Backup Manager
Tools
Back Up
Scheduled Backup Settings
Upload Files to Server Repository
Personal FTP Repository Settings
Refresh
Backups
Server Repository
Personal FTP Repository
Current Backup Tasks

No backups

Here is where we can run our first test. Instead of backing up everything, lets just back up the server configuration. This will save A LOT of time in testing.



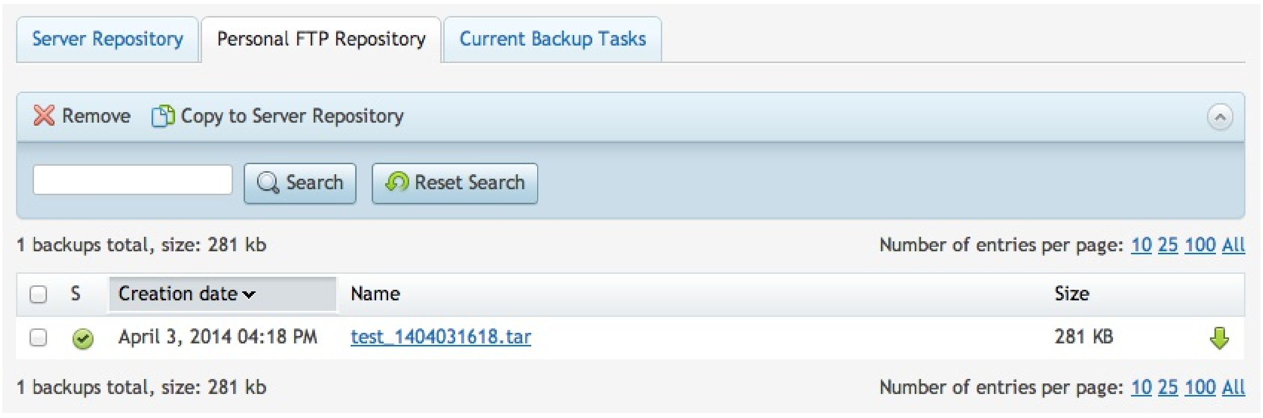
Backup settings

Add prefix to backup name	test
Comments	Server backup. Creation date: Apr 3, 2014 04:37 PM
Create multivolume backup	☐ Volume size 2047 MB
Store backup in	○ Server repository ● Personal FTP repository
When backup task is completed, send notification email to	

Backup content

Back up	● Server configuration ○ Server configuration and content
Suspend domains until backup task is completed	☐

It will start to run a backup. This is your cue to go grab another coffee/beer while it finishes. Once it is complete, you will see it appear in the Personal FTP tab:



1 backups total, size: 281 kb   Number of entries per page: 10 25 100 All

	S	Creation date	Name	Size
	✓	April 3, 2014 04:18 PM	test_1404031618.tar	281 KB

1 backups total, size: 281 kb   Number of entries per page: 10 25 100 All

Since that backup is done, lets see where it is at. We can check by running the `ll` or `ls` command:

```
[root@myserver ~]# ll /var/www/vhosts/backup.mysite.com/s3backups/
total 284
-rw-r--r-- 1 backupuser psacln 287916 Apr  3 16:18 test_1404031618
```

Now since the file is there, lets move it to Amazon S3. This is going to be the same command as our test from before, but we need to change the directory to the *fullserver* directory. You will see its output as it moves the file.

```
[root@myserver ~]# aws s3 mv /var/www/vhosts/backup.mysite.com/s3b

move: ../var/www/vhosts/backup.mysite.com/s3backups/fullserver/tes
```

We can now list out the directory on the Amazon S3 bucket to confirm our file made it safely. When I run the command, you can see it shows our test text file as well:

```
[root@myserver ~]# aws s3 ls s3://myserver-pleskfull --recursive
2014-04-03 12:13:13          0 test1.txt
2014-04-03 16:47:19     287916 test_1404031618.tar
```

Automating the Process

We have made it this far, but it would be really pointless if this was not automated. Fortunately, Linux allows us to run a script every hour to check if there is a new backup. If there is a backup, then it will then execute the move to Amazon S3. Our script will be located in `/etc/cron.hourly/`. Create the file using `nano` and **be sure to NOT have .sh at the end of the filename** (otherwise it will not execute).

```
nano /etc/cron.hourly/aws-s3move
```

Here is the code that we can use to check our directory (online version here):

```
#! /bin/bash
# Author: Jay Rogers - jay@521dimensions.com

#IMPORTANT -- CONFIGURATION VARIABLES
BACKUPDOMAIN=backup.mydomain.com
SERVERBUCKETNAME=myserver-pleskfull

#Check to see if there are any SERVER backups to move. If so, move
if [ "$(find /var/www/vhosts/$BACKUPDOMAIN/s3backups/fullserver/ -
        #Make sure that CRON is able to find the AWS Configuration
        export AWS_CONFIG_FILE=/root/.aws/config
        #Execute S3 moved to Bucket configured above
        aws s3 mv /var/www/vhosts/$BACKUPDOMAIN/s3backups/fullserv
        exit 0
else
        #If nothing exists in any of these folders, then do nothin
        exit 0
fi
```

Write the file out with `nano` and then set the file permissions so our script can be executed:

```
chmod 755 /etc/cron.hourly/aws-s3move
```

Since our script is now executable, it is now ready for a test run. **Run another backup** (like what we did before — just the server configuration). Only this time when it completes, we will run our script. When you run it via BASH, you should see the results below:

```
[root@myserver ~]# bash /etc/cron.hourly/aws-s3move

move: ../var/www/vhosts/backup.mysite.com/s3backups/fullserver/tes
```

Restoring From A Backup

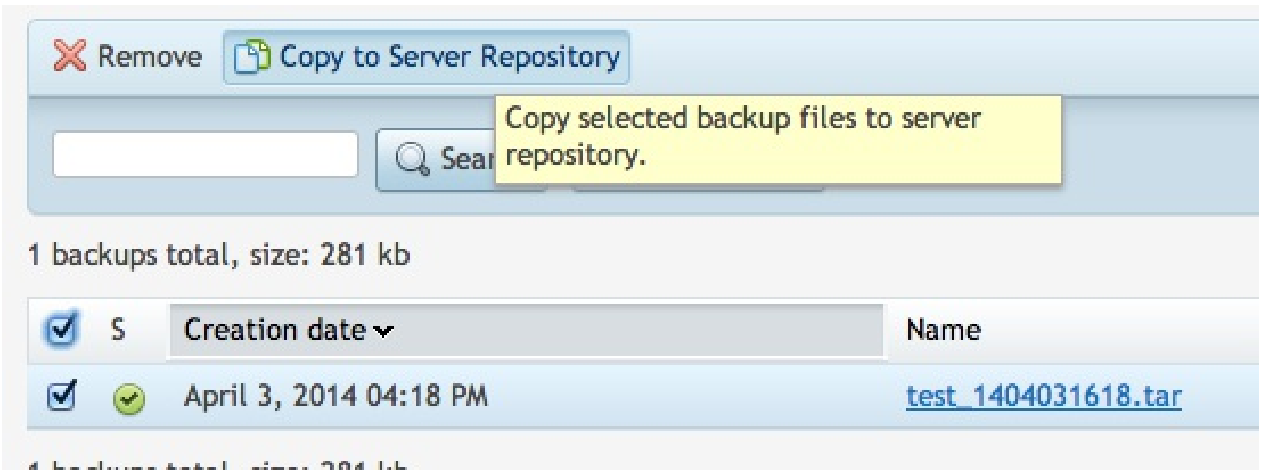
The most important piece in a backup process is the restoration. For most people, it's usually the least tested piece of the process. To get this to work, we are just going to copy the file that we want back into our FTP folder. Make sure you change `backup-file.tar` to your actual file name that you want.

```
aws s3 cp s3://myserver-pleskfull/test_1404031618.tar /var/www/vho
```

We will get the results below:

```
[root@myserver ~]# aws s3 cp s3://myserver-pleskfull/test_14040316

download: s3://myserver-pleskfull/test_1404031618.tar to ../var/ww
```

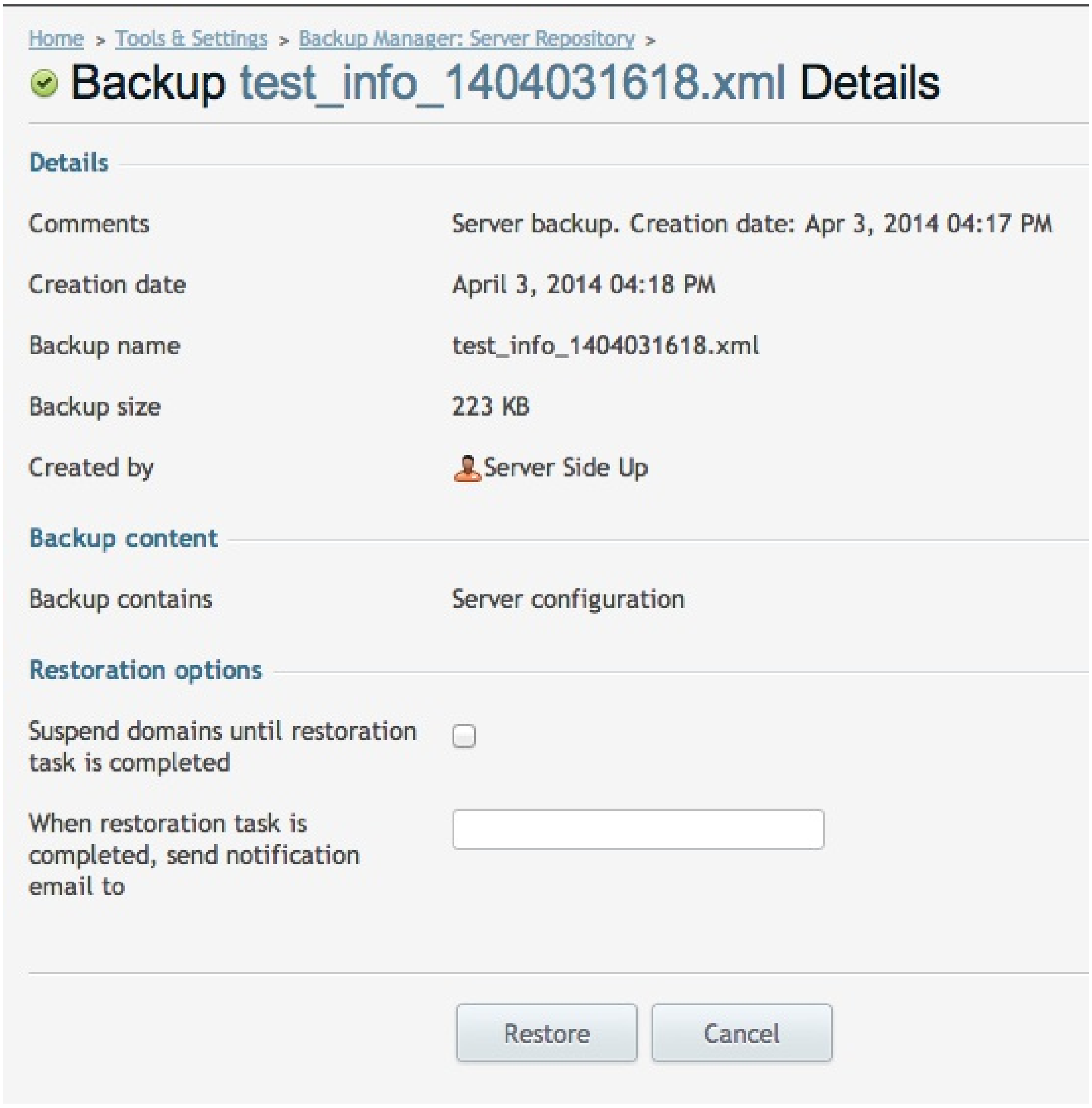
Once you have it in the FTP folder, you should now be able to copy it to your server repository. (It will ask you for your restore password, so I hope you wrote that down!):


Remove
Copy to Server Repository
Copy selected backup files to server repository.
Sear

1 backups total, size: 281 kb

	S	Creation date	Name
✓	✓	April 3, 2014 04:18 PM	test_1404031618.tar

Then once it is in your server repo, you can click on that file to initiate the restore:


Home > Tools & Settings > Backup Manager: Server Repository >

Backup test_info_1404031618.xml Details

Details

Comments	Server backup. Creation date: Apr 3, 2014 04:17 PM
Creation date	April 3, 2014 04:18 PM
Backup name	test_info_1404031618.xml
Backup size	223 KB
Created by	Server Side Up

Backup content

Backup contains	Server configuration

Restoration options

Suspend domains until restoration task is completed	☐
When restoration task is completed, send notification email to	

Taking It A Step Further

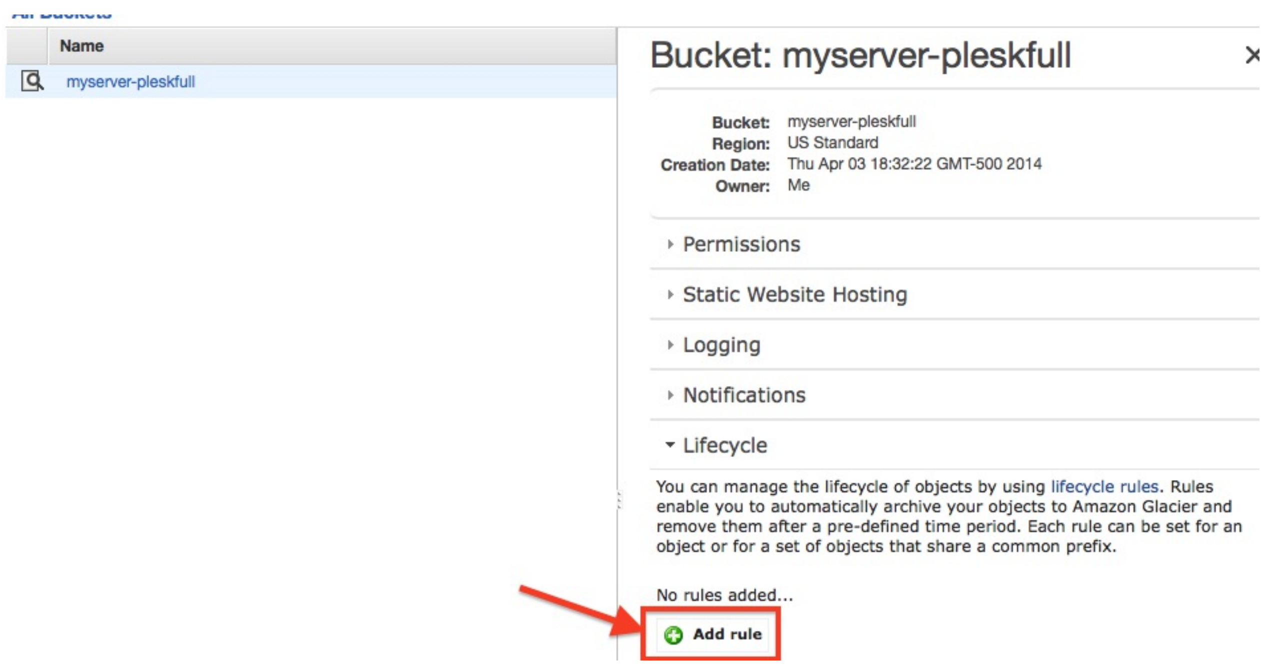
If ou are interested in taking this backup process a step further, you can easily configure the following:

1. Configure Retention Policys (to save on Amazon Costs)
2. Configure Backups for Specific Subcriptions (Vhosts)

Configure Retention Policies

To prevent from paying a butt-load in Amazon S3 charges, it is a wise idea to configure retention policy. You can automatically have Amazon S3 delete the file after X amount of days or you can archive it to Amazon Glacier. The thing that you have to plan for on Amazon Glacier, is that when you want your file for restoration, it takes Glacier 4 hours to get your file ready before you can download it. That could be a disaster if that is a very important backup file that you need RIGHT AWAY.

Log into your Amazon AWS Console and then select the bucket. Under *Properties* you will find an accordion menu that says *Lifecycle*. Click on that then press *Add rule*:



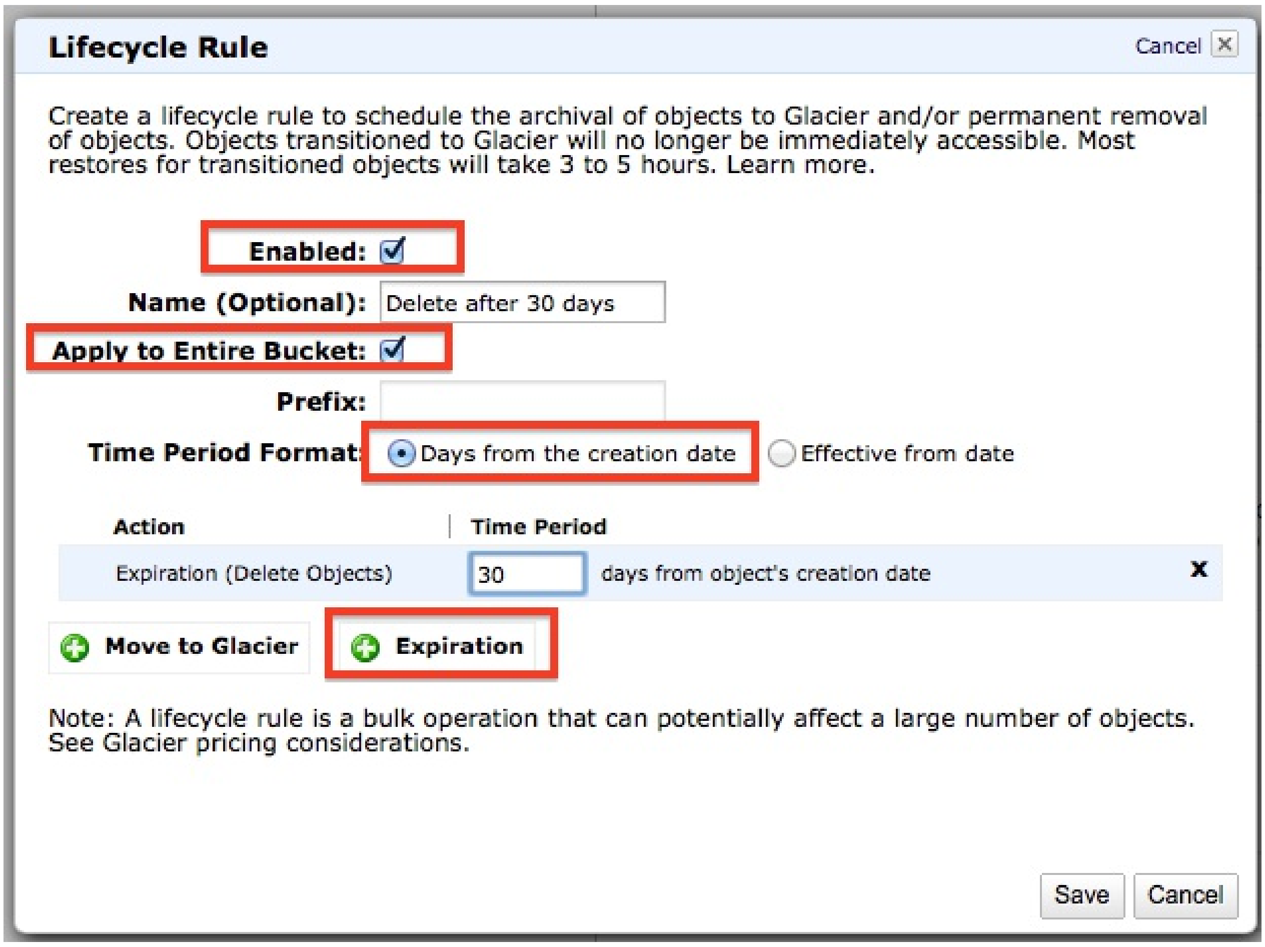
Bucket: myserver-pleskfull
Region: US Standard
Creation Date: Thu Apr 03 18:32:22 GMT-500 2014
Owner: Me

You can manage the lifecycle of objects by using lifecycle rules. Rules enable you to automatically archive your objects to Amazon Glacier and remove them after a pre-defined time period. Each rule can be set for an object or for a set of objects that share a common prefix.

No rules added...

.

Then you can configure the bucket to delete any objects X amount of days from its creation date.



Lifecycle Rule Cancel ✕

Create a lifecycle rule to schedule the archival of objects to Glacier and/or permanent removal of objects. Objects transitioned to Glacier will no longer be immediately accessible. Most restores for transitioned objects will take 3 to 5 hours. Learn more.

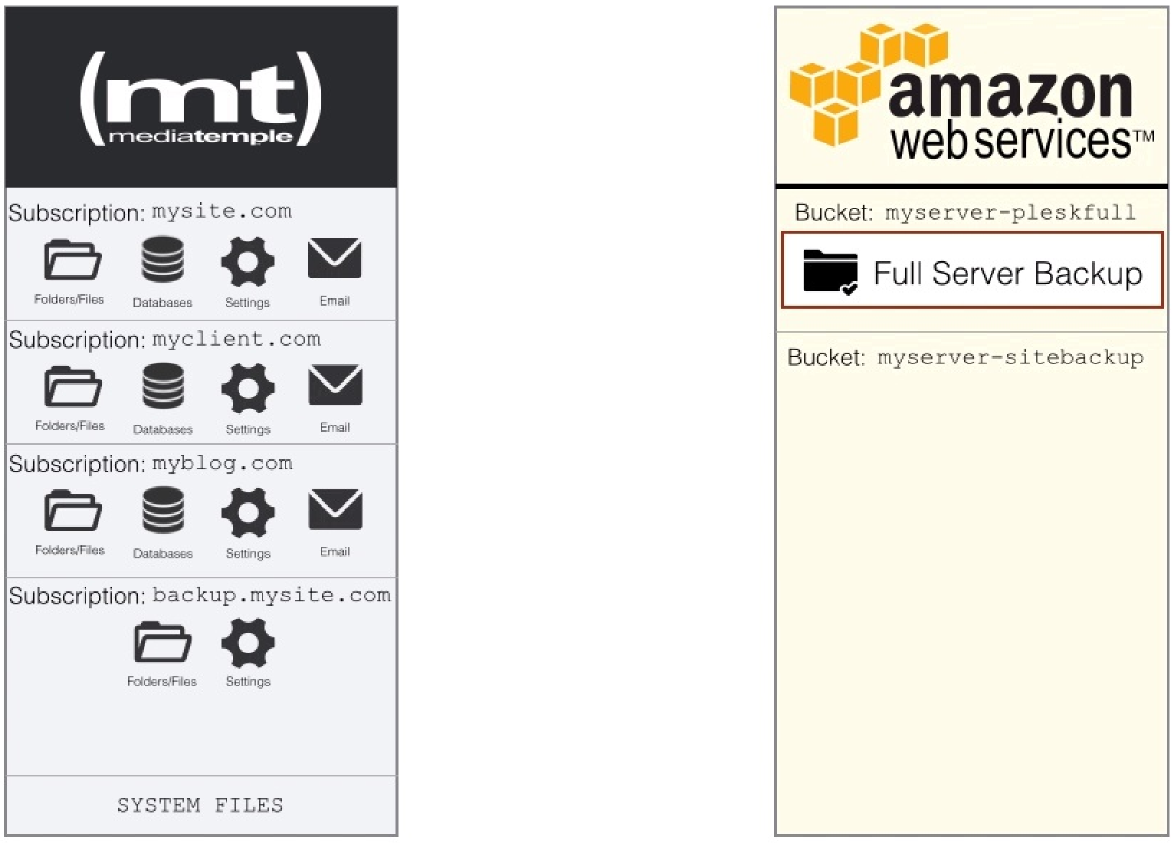
Enabled:	✓
Name (Optional):	Delete after 30 days
Apply to Entire Bucket:	✓
Prefix:	
Time Period Format:	[● Days from the creation date] ○ Effective from date

Action	Time Period
Expiration (Delete Objects)	30 days from object's creation date

Note: A lifecycle rule is a bulk operation that can potentially affect a large number of objects. See Glacier pricing considerations.

Configure Subscription Specific (Vhost) Backups

You can also store single site backups in a separate bucket. Why would we want to do that? A separate bucket makes sense because we can run different backup frequencies and also apply different retention policies. Here is how it would work:



To configure the backups, we would have to create another folder for our site backups:

```
mkdir -p /var/www/vhosts/backup.mysite.com/s3backups/sites/
```

Be sure to set the right permissions after that:

```
chown -R backupuser:psacln /var/www/vhosts/backup.mysite.com/s3bac
```

Now you can go into the backup.mysite.com subscription and create specific FTP users for your sites. Once you have those users created, you can then go into the subscription that you want to backup and configure the Backup Manager (just like how we did it for our Full Server backup). We will also have to modify our script in the cron.hourly:

```
nano /etc/cron.hourly/aws-s3move
```

Then update the script to this. Be sure to configure the options at the top! (Online Version available here)

```
#! /bin/bash
# Author: Jay Rogers - jay@521dimensions.com

#IMPORTANT -- CONFIGURATION VARIABLES
BACKUPDOMAIN=backup.mydomain.com
SITESBUCKETNAME=myserver-plesksites
SERVERBUCKETNAME=myserver-pleskfull

#Check to see if there are any individual SITES to back up. If so,
if [ "$(find /var/www/vhosts/$BACKUPDOMAIN/s3backups/sites/ -name
        #Make sure that CRON is able to find the AWS Configuration
        export AWS_CONFIG_FILE=/root/.aws/config
        #Execute S3 moved to Bucket configured above
        aws s3 mv /var/www/vhosts/$BACKUPDOMAIN/s3backups/sites/ s
fi

#Check to see if there are any SERVER backups to move. If so, move
if [ "$(find /var/www/vhosts/$BACKUPDOMAIN/s3backups/fullserver/ -
        #Make sure that CRON is able to find the AWS Configuration
        export AWS_CONFIG_FILE=/root/.aws/config
        #Execute S3 moved to Bucket configured above
        aws s3 mv /var/www/vhosts/$BACKUPDOMAIN/s3backups/fullserv
        exit 0
else
        #If nothing exists in any of these folders, then do nothin
        exit 0
fi
```

No matter how many other individual site backups you add, this script does not need to be modified. It grabs recursively from s3backups/sites/ and will upload them to Amazon S3.

Getting Help

Hopefully this all makes sense and works as well for you as it does for me. Feel free to hit me up on Twitter or leave a comment below if you have any thoughts/feedback/questions on anything!

Original Article: http://serversideup.net/media-temple-amazon-s3-backup-and-other-linux-servers/

Twitter: @jaydrogers

Table of Contents