FUJITSU Cloud Service S5

Setup and Configure an Additional Disk with CentOS Servers

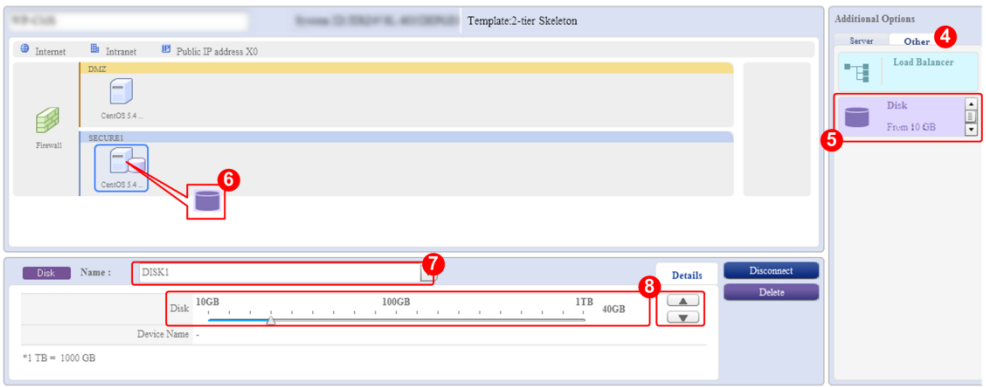
This guide details the steps required to install and configure an additional disk with a CentOS server on the FUJITSU Cloud Service S5 and how to use that disk from the CentOS server.

Assumptions

- User Account, Certificate and connectivity to the FUJITSU Cloud Service S5.
- Familiarity with the FUJITSU Cloud Service S5 Portal basic configuration and administration tasks.
- Familiarity with the FUJITSU Cloud Service S5 Portal System Manager and Design Studio.
- SSH Client connectivity software is installed
- Familiarity with Linux / CentOS configuration and administration tasks.
- Basic familiarity with a Linux text editor such as vi

Installation and Configuration of an Additional Disk on the FUJITSU Cloud Service S5

1. Using the FUJITSU Cloud Service S5 Portal System Manager select the virtual system to modify.
2. Click **Reconfigure**.
3. Identify the CentOS server to modify and where additional disk space is required.
4. Click on the **Other** tab in the **Additional Options** section.
5. Select a **Disk**.
6. Holding the left mouse key drag the disk icon onto the CentOS server where additional space is required.
7. Type a suitable name for the disk in the **Name:** text box or accept the default in the disk **Details** section.
8. Select the disk size using the slider or the up / down arrows in the disk **Details** section.



9. Click **Next>** and follow through to **Final Confirmation** remembering to check **I Agree to the Terms of Service** as appropriate.

Adding a new Disk to a CentOS Server

1. Create a VPN Connection and connect via SSH (for example PuTTY)
2. Login with appropriate permissions to configure, partition, format, and mount a new disk device.
3. List the hard disk partition table(s) on the server by typing **fdisk –l** followed by the **ENTER** key. In this example the new disk being added is 40GB in size,

```
[root@localhost ~]# fdisk –l

Disk /dev/xvda: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/xvda1   *           1        1305    10482381   83  Linux

Disk /dev/xvdb: 42.9 GB, 42949672960 bytes
255 heads, 63 sectors/track, 5221 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/xvdb doesn't contain a valid partition table

Disk /dev/xvdc: 0 MB, 360448 bytes
255 heads, 63 sectors/track, 0 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/xvdc doesn't contain a valid partition table
[root@localhost ~]#
```

4. Identify the ~40GB disk. Looking at the partition list generated by the previous step, we see that in this example, there are three disks named /dev/xvda, /dev/xvdb and /dev/xvdc. We require a new and therefore unpartitioned disk that is 40GB. The device name matching these criteria is **/dev/xvdb**,

*Disk **/dev/xvdb**: 42.9 GB, 42949672960 bytes*

5. The device name of the hard disk to the partitioned, formatted and mounted based on the output of this example is **/dev/xvdb**. Remember your output may vary. Next step is to create a primary partition on the device by typing the following **fdisk <device-name>**. An example of this is provided next,

```
[root@localhost ~]# fdisk /dev/xvdb
Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
Building a new DOS disklabel. Changes will remain in memory only,
until you decide to write them. After that, of course, the previous
content won't be recoverable.

The number of cylinders for this disk is set to 5221.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)
Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

Command (m for help):
```

6. Type **n** (add a new partition) followed by the **ENTER** key to add a new partition table,

```
Command (m for help): n

Command action
   e   extended
   p   primary partition (1-4)
```

7. Create a primary partition. Type **p** followed by the **ENTER** key.
8. Enter **1** for the Partition number and then press the **ENTER** key again to accept the default settings for First Parition,

```
Partition number (1-4): 1
First cylinder (1-5221, default 1): 1
Last cylinder or +size or +sizeM or +sizeK (1-5221, default 5221):
Using default value 5221
```

9. Press enter to accept default for Last Cylinder
10. Typing **w** (write table to disk and exit) followed by **ENTER** will write the changes, and exit the program.

```
Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
[root@localhost ~]#
```

11. Confirm a Linux partition, by retyping the **fdisk** command but this time choose the **p** option (print the partition table) and then **q** to quit.

```
[root@localhost ~]# fdisk /dev/xvdb

The number of cylinders for this disk is set to 5221.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other Oss
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help): p

Disk /dev/xvdb: 42.9 GB, 42949672960 bytes
255 heads, 63 sectors/track, 5221 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/xvdb1               1        5221    41937651   83  Linux

Command (m for help): q
```

12. The next step is to format the new partition. This is done using the **mkfs -t ext3 <device-name>** command (remember the device in this example is **/dev/xvdb**).

```
[root@localhost ~]# mkfs -t ext3 /dev/xvdb
mke2fs 1.39 (29-May-2006)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
5242880 inodes, 10484412 blocks
524220 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=4294967296
320 block groups
32768 blocks per group, 32768 fragments per group
16384 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
        4096000, 7962624

Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 30 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
[root@localhost ~]#
```

13. Create mount point and set up automatic mount when system start-up. The command to create a mount point is **mkdir <path>**. In this example the new disk will be used to install MySQL so the mount point will be under **mysql**,

```
[root@localhost /]# mkdir /home/mysql
```

Add the following as a new last line into the **fstab** configuration. The **vi** editor is used for this example,

```
/dev/xvdb                /home/mysql     ext3        defaults    0           1
```

```
[root@localhost /]# vi /etc/fstab
```

NOTE: Your device name and mount directory path may vary from the example above.

14. Finally mount the new disk,

```
[root@localhost /]# mount /home/mysql
```

A new disk is now installed, configured, partitioned, formatted, mounted and is ready to use on the CentOS server.

Contact

Fujitsu Global Cloud Team
FUJITSU
E-mail: cloud_gsd@au.fujitsu.com
Website: au.fujitsu.com