Usefull CMDs

- Linux
  - Disk / Space CMD
  - Mount network share
  - Vault 2 issues with Nginx Proxy not working
  - □□ Sharing a Folder on Debian Linux via Samba
  - □□ Mounting a Network Share for Docker on Debian
- PowerShell / CMD
  - Exchange Powershell CMDs
  - Unlock Account
  - CMD - Free Diskspace

Linux

Linux

Disk / Space CMD

Disk / Space CMD

Will give the usage of the user you are logged in as

```
sudo du -h --max-depth=1
```

The `df` command primarily checks disk usage on a mounted filesystem. If you don't include a file name, the output shows the space available on all currently mounted filesystems. Disk space is shown in 1K blocks by default:

```
df -h
```

```
root@vault2:/var/lib/docker/containers# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            6.8G     0  6.8G   0% /dev
tmpfs           1.4G  4.2M  1.4G   1% /run
/dev/sda1       294G   37G  242G  14% /
tmpfs           6.9G     0  6.9G   0% /dev/shm
tmpfs           5.0M     0  5.0M   0% /run/lock
tmpfs           6.9G     0  6.9G   0% /tmp
/dev/sdb1       7.0T  6.1T  949G  87% /srv/dev-disk-by-uuid-2bb191d0-15da-4
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/f099437856c9
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/77019b2cf7da
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/b18c38072d36
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/7611c2eedffb
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/8986cce68347
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/b547157cff41
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/bb569b4df2e2
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/8e683b0cf25e
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/4d099d427617
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/b51bef5c2993
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/cbeb182e49d6
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/0cad57c062a9
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/beb11364d559
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/5e3a18a98185
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/42922fff423b
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/397a4784bc7b
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/509cb8f1d576
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/1a0c94fcec26
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/eba4949c04b0
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/98c292d34a42
overlay         294G   37G  242G  14% /var/lib/docker/overlay2/48adf765dc81
root@vault2:/var/lib/docker/containers# ^C
```

List All Block Devices in Linux

lsblk

```
lsblk
```

```
root@vault2:/var/lib/docker/containers# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda      8:0    0  350G  0 disk
├─sda1   8:1    0  299G  0 part /
├─sda2   8:2    0    1K  0 part
└─sda5   8:5    0  975M  0 part [SWAP]
sdb      8:16   0    7T  0 disk
└─sdb1   8:17   0    7T  0 part /srv/dev-disk-by-uuid-2bb191d0-15da-4ffe-8266-0d1b2133ec7c
sr0     11:0    1 1024M  0 rom
```

```
root@dbt3ch /home# ls
Docker  docker.zip
root@dbt3ch /home# ls
Docker  docker.zip
root@dbt3ch /home# scp docker.zip root@192.168.0.41:/home
```

Linux

Mount network share

```
sudo mount.cifs //192.168.1.252/5tb /5TB -o user=USER,pass=PASS
```

Install CIFS

```
sudo apt install cifs-utils
```

```
mount.cifs //192.168.1.251/hel /mnt/share/NAS/hydra -o user=USER,pass=PASS
```

Linux

Vault 2 issues with Nginx Proxy not working

Check logs in Portainer - bind port already used (80)

```
lsof -i tcp:80
```

This will show whats running on that port

```
fuser -k 80/tcp
```

This should kill the process, but if not do the below (based on the COMMAND Name)

```
systemctl stop haproxy
```

Linux

? Sharing a Folder on Debian Linux via Samba

Last Updated: 2025-06-01
Applies To: Debian 10+, Ubuntu, other Debian-based distributions
Use Case: Share a local folder over the network for access from Windows, macOS, or Linux machines.

? Prerequisites

- Debian-based Linux system
- `sudo` privileges
- Network connectivity to target devices

? Step 1: Install Samba

Update the system and install Samba:

bash
```
sudo apt update
sudo apt install samba
```

? Step 2: Create a Shared Directory

Choose or create the folder to share. Example:

bash
```
sudo mkdir -p /srv/share
sudo chown nobody:nogroup /srv/share
sudo chmod 0775 /srv/share
```

This creates a public share folder with loose permissions (for anonymous access).

?? Step 3: Configure Samba

Edit the Samba configuration file:

bash
```
sudo nano /etc/samba/smb.conf
```
Add the following to the bottom of the file:

ini
```
[PublicShare]
   path = /srv/share
   browseable = yes
   read only = no
   guest ok = yes
   force user = nobody
```
> ◫ *Tip:* `force user = nobody` ensures files are written as the `nobody` user, preventing permission issues.

? Step 4: Restart Samba Services

bash
```
sudo systemctl restart smbd
```

? Step 5: Open Firewall (If Enabled)

For systems using `ufw` (Uncomplicated Firewall):

bash
```
sudo ufw allow 'Samba'
```
Or manually:

bash
```
sudo ufw allow 137,138/udp
sudo ufw allow 139,445/tcp
```

? Step 6: Access the Share

From a Windows or Linux machine:

- Open File Explorer or file manager.
- Go to: `\\<your-debian-ip>\PublicShare`

You should see the shared folder contents.

? Optional: Secure with Username/Password

To restrict access:

1. Disable guest access:
   ini
   `guest ok = no`
2. Create a Samba user:
   bash
   `sudo smbpasswd -a <your-linux-user>`
3. Restart Samba:
   bash
   `sudo systemctl restart smbd`

You'll now need a username/password to access the share.

? Troubleshooting

Problem	Fix
Can't connect to share	Check firewall, Samba config syntax, and network IPs
Files not writable	Check folder permissions and `force user` setting
Authentication issues	Ensure Samba user exists and password is set via `smbpasswd`

? Related

- `/etc/samba/smb.conf`: Samba main configuration file
- `sudo testparm`: Validates Samba config
- `sudo smbstatus`: Shows active connections

Linux

? Mounting a Network Share for Docker on Debian

? Overview

This guide shows how to mount a CIFS/SMB network share on a Debian-based system and use it as a volume in Docker Compose. Useful when your Docker container needs access to shared files hosted on another machine (e.g., a NAS, Windows PC, or another Linux box).

?? Step 1: Install CIFS Utilities

```
sudo apt update
sudo apt install cifs-utils
```

? Step 2: Create a Mount Point

```
sudo mkdir -p /mnt/downloads
```

? Step 3: Mount the Share

```
sudo mount -t cifs //LORGAR/Downloads /mnt/downloads -o
username=slitzer,password=blink182,vers=3.0
```

Replace:

- `//LORGAR/Downloads` — with the remote share path
- `username` and `password` — with valid credentials
- `vers=3.0` — adjust based on your SMB version (`2.0`, `3.1.1`, etc.)

? Optional: Persistent Mount via fstab

Edit `/etc/fstab`:

```
//LORGAR/Downloads /mnt/downloads cifs
username=slitzer,password=blink182,vers=3.0,uid=1000,gid=1000 0 0
```

Using a secure credentials file:

```
# /root/.smbcredentials
username=slitzer
password=blink182
```

Then in `/etc/fstab`:

```
//LORGAR/Downloads /mnt/downloads cifs
credentials=/root/.smbcredentials,vers=3.0,uid=1000,gid=1000 0 0
```

Don't forget: `chmod 600 /root/.smbcredentials`

? Step 4: Use in Docker Compose

Update your `docker-compose.yml`:

```
version: '3'
services:
  myapp:
    image: myapp/image
    volumes:
      - /mnt/downloads:/app/shared
```

This maps the mounted network share inside the container.

? Troubleshooting

- Check if mounted: `mount | grep downloads` or `df -h`
- Check permissions: `ls -la /mnt/downloads`
- Logs: `dmesg`, `journalctl -xe`

? TL;DR

```
sudo apt install cifs-utils
sudo mkdir -p /mnt/downloads
sudo mount -t cifs //REMOTE_HOST/Share /mnt/downloads -o username=...,password=...,vers=3.0
```

Then reference `/mnt/downloads` in your Docker volumes.

□□ **Pro Tip:** Avoid hardcoding passwords by using credential files, especially for systems in production.

PowerShell / CMD

Exchange Powershell CMDs

Step 1: Start PowerShell

Using any of these methods, or any other you may know of:

WinKey + R (Run Dialog): powershell.exe

Start Menu -> type 'Powershell', click it

Navigate to C:\Windows\System32\WindowsPowerShell\V1.0\powershell.exe

Step 2: (optional/dependent) Connect to the AD Sync Server

If you're running PowerShell on the Server where AD Connect is running, skip this step.

In the command window run the following command, replacing [SERVERNAME] with the name of the server you need to connect to (you may not be able to do this is PSRemoting is not enabled on the remote server):

```
Enter-PSSession -ComputerName [SERVERNAME]
```

Step 3: Import the ADSync Module

Run the following command:

```
Import-Module ADSync
```

Step 4: Run the Sync Command

Run the following Command(s):

For a Delta Sync (most common, and used for most situations):

```
Start-ADSyncSyncCycle -PolicyType Delta
```

For a Full Sync (only necessary in some situations):

```
Start-ADSyncSyncCycle -PolicyType Initial
```

Step 5: (Optional/Dependent) Exit PSSession

If you used the Enter-PSSession command, you must exit the session or it will remain open even after terminating the PS Host. You can do so by simply using the command: Exit

PowerShell / CMD

Unlock Account

unlock CMD account (codeblue)

```
NET USER codeblue /ACTIVE:YES
```

CMD - Free Diskspace

Gives free diskspace, change drive letter as needed

> powershell -command "(Get-PSDrive C).Free/1GB"