Linux

Common issues

Shared drive is not mounted even though it's setup in fstab

Usually it's coupled with this error message when starting or shutting down the system:

`Dependency failed for Remote File Systems`

The issue is that when the system tries to mount the shared drive, the network is not up yet. To fix it, run the following command:

`sudo systemctl enable systemd-networkd-wait-online`

Commands

Extend partition size

> If the partition size was changed in Proxmox, then after the resize on the UI, restart the VM

1. Get PV (Physical Volume) name (Ex: /dev/sda1)

   ```
   sudo pvs
   ```

2. Resize the PV

   ```
   sudo pvresize /dev/sda1
   ```

3. Get root logical volume name (Filesystem value of / row; for example: /dev/mapper/ubuntu--vg-root)

   ```
   df -h
   ```

4. Expand logical volume

   ```
   sudo lvextend -r -l +100%FREE /dev/mapper/ubuntu--vg-root
   ```

Source:

- Can't resize a partition using resize2fs

Create sudo user

1. First create the user: `adduser **sammy**`
2. Add user to the *sudo* group: `usermod -aG sudo **sammy**`
3. Then either logout and log back in, or switch to the user: `su - **sammy**`

Source:

- How To Create A New Sudo Enabled User on Ubuntu 22.04

Replace text using sed

- ```
  sed -i 's/FIND_TEXT/REPLACE_TEXT/' file.txt
  ```
  ```

Setup Access Control Lists (ACL)

ACL can be used to provide additional permissions to the file system.

If not installed run the following command: `sudo apt install acl`

There are two commands: **setfacl** to set acl and **getfacl** to get acl permissions

To get acl of a folder:

`getfacl ~/test`

it returns with a similar result:

```
# file: test/
# owner: user
# group: user
user::rwx
group::r-x
other::r-x
```

To set the acl for a user:

`setfacl -m "u:user:permissions" /path/to/file`

For example:

`setfacl -m u:user:rwx test/`

But if you want to change the permission recursively for a user:

`setfacl -Rm u:user:rwx test`

To change the permission recursively for a group:

`setfacl -Rm g:group:rwx test`

To **remove** all the acl permissions:

`setfacl -b path/to/file`

Securing SSH with public keys

Securing keys with YubiKey

Make sure you use on both your client and server at least version 8.2 of OpenSSH, because previous versions do not support 2FA with hardware security keys:

`ssh -V`

Check if OpenSSH can generate U2F keys:

`ssh-keygen --help`

and you should see something like that:

`[-t dsa | ecdsa | ecdsa-sk | ed25519 | ed25519-sk | rsa]`

The ecdsa-sk and ed25519-sk types are the only valid ones for 2FA enabled SSH key pairs. The ‘-sk’ part stands for ‘security key’.

To generate SSH key-pair:

`ssh-keygen -t ed25519-sk -f ~/.ssh/securitykey`

> If you get this or something similar error while trying to generate a key: "error while loading shared libraries: libfido2.so.1", you have to install the "libfido2" dependency on your system.

It’ll ask you to touch your YubiKey:

> Generating public/private ed25519-sk key pair.
> You may need to touch your authenticator to authorize key generation.

Just touch the metal circle and it’ll bind the SSH key pair to your YubiKey.

When it says “Enter passphrase (empty for no passphrase)”, you can just press enter to leave it empty.

> Beware "ed25519-sk" is only supported from version 5.2.3, so if you have a YubiKey with an earlier firmware, use ecdsa-sk

> To see which firmware version is on your YubiKey:
> https://www.yubico.com/support/download/yubikey-manager

Two keys are generated under ~/.ssh

- securitykey (private key, make sure to never share it)
- securitykey.pub (public key that you have to copy to your server)

To copy the public security key to the server you can either use:

`ssh-copy-id -i ~/.ssh/securitykey.pub user@server`

or

login to the server and create the `~/.ssh/authorized_keys` file if it doesn't exist, then copy and paste the content from your `securitykey.pub` into the `authorized_keys` file.

Finally to connect to the server:

`ssh -i securitykey user@server`

If it works, then you can make the **server more secure by disabling password authentication** .

Open `/etc/ssh/sshd_config` and find this section, comment it back and set it to no:

```
# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication no
```

Then run:

`service ssh restart`

Securing keys with Trezor Model T

The steps are basically the same just like with YubiKey, except in this case `ed25519-sk` can be used as well.

Source:

- https://rameerez.com/how-to-use-yubikey-to-log-in-via-ssh-to-server/

- https://cryptsus.com/blog/how-to-configure-openssh-with-yubikey-security-keys-u2f-otp-authentication-ed25519-sk-ecdsa-sk-on-ubuntu-18.04.html
- https://stackoverflow.com/questions/20898384/disable-password-authentication-for-ssh