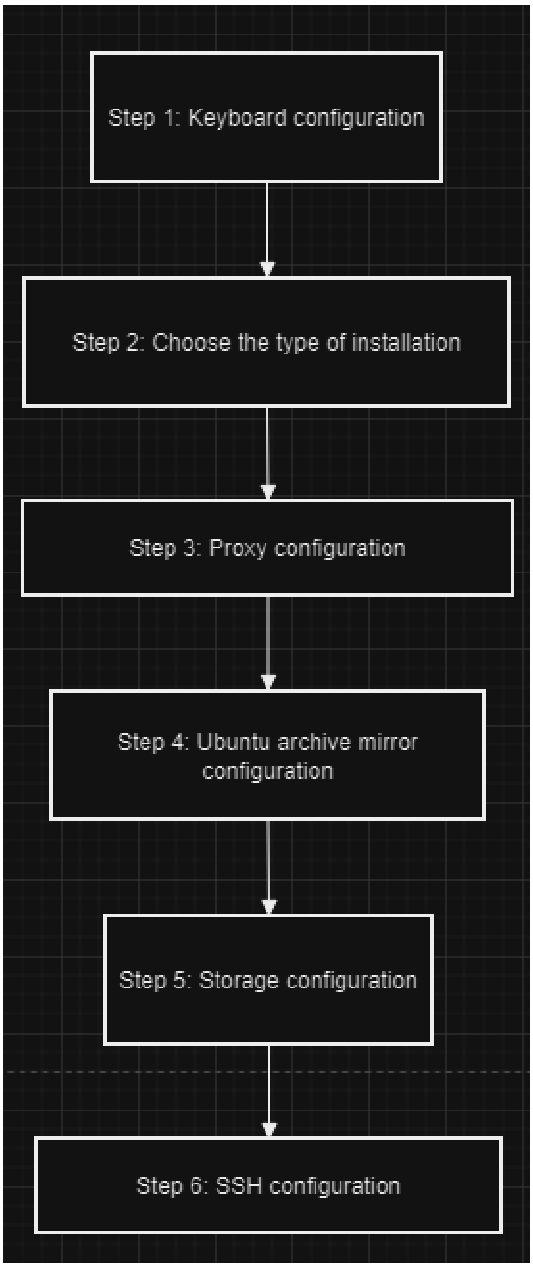
UNIVERSITY OF SCIENCE & TECHNOLOGY OF HANOI

REPORT: LINUX INSTALLATION AND DATA SPACE PREPARATION

Full name	Student ID
Nguyễn Việt Anh	BA12-009
Phạm Phú Hưng	BA12-081
Đào Ngọc Tùng	BA12-185
Nguyễn Tiến Ngọc	BA12-140
Trương Quang Huy	BA12-087
Nguyễn Khánh Duy	BA12-063
Phạm Tùng Anh	BA12-010

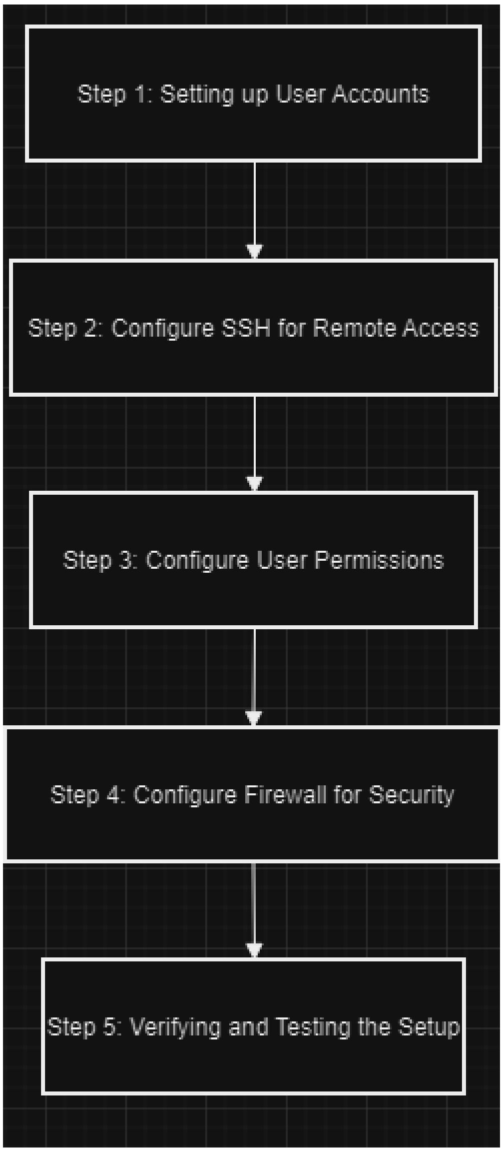
I. Overview Flow

Installation



Step 1: Keyboard configuration
Step 2: Choose the type of installation
Step 3: Proxy configuration
Step 4: Ubuntu archive mirror configuration
Step 5: Storage configuration
Step 6: SSH configuration

Configuration

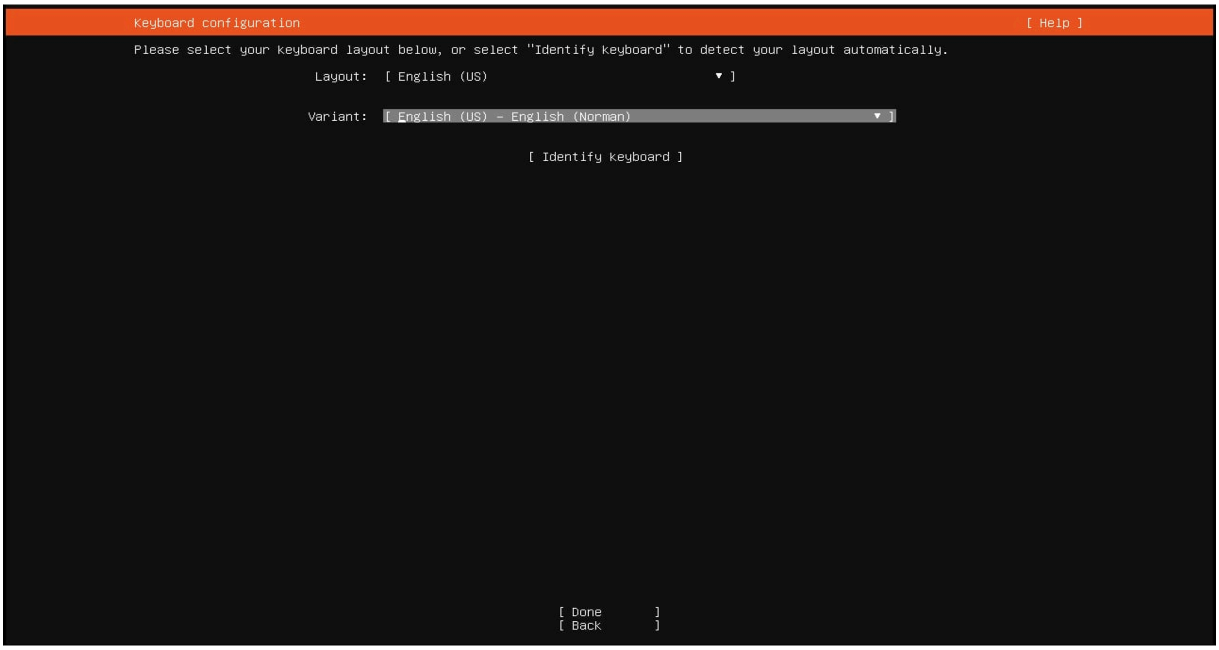


Step 1: Setting up User Accounts
Step 2: Configure SSH for Remote Access
Step 3: Configure User Permissions
Step 4: Configure Firewall for Security
Step 5: Verifying and Testing the Setup

II. Installation

We select Ubuntu Server 22.04.5 LTS

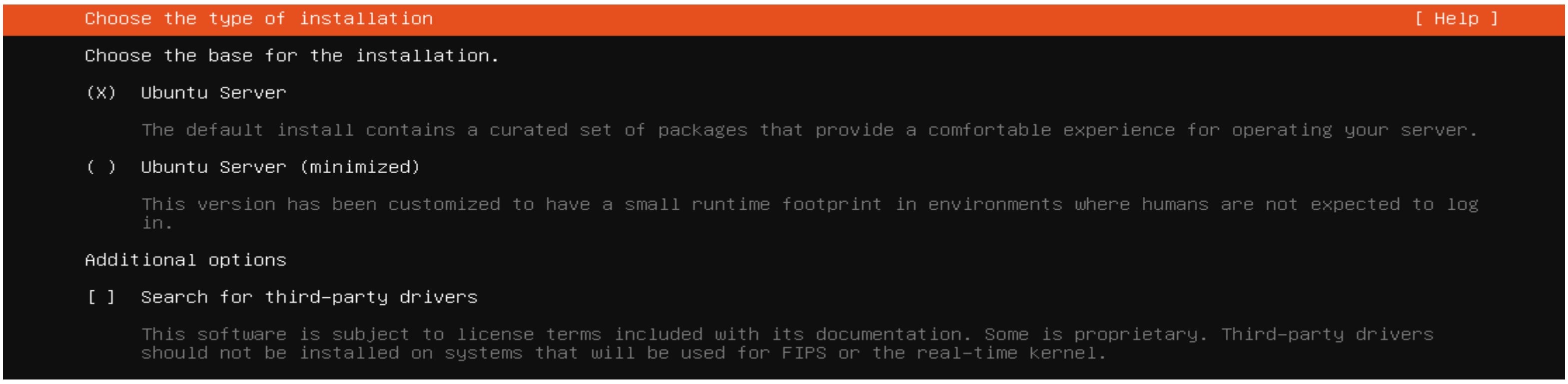
Step 1: Keyboard configuration



```
Keyboard configuration                                                                  [ Help ]
Please select your keyboard layout below, or select "Identify keyboard" to detect your layout automatically.
                  Layout:  [ English (US)                        ▼ ]

                 Variant:  [ English (US) - English (Norman)                         ▼ ]

                                        [ Identify keyboard ]

                                            [ Done       ]
                                            [ Back       ]
```

- This section allows us to choose a language that the server will use to display
- **[Layout]** allows us to choose a keyboard layout and [variant] enables users to select a keyboard layout more specific to their religion.
- **[Identify keyboard]** option will automatically detect user language

Step 2: Choose the type of installation



```
Choose the type of installation                                                                  [ Help ]
Choose the base for the installation.
(X)  Ubuntu Server
     The default install contains a curated set of packages that provide a comfortable experience for operating your server.
( )  Ubuntu Server (minimized)
     This version has been customized to have a small runtime footprint in environments where humans are not expected to log
     in.
Additional options
[ ]  Search for third-party drivers
     This software is subject to license terms included with its documentation. Some is proprietary. Third-party drivers
     should not be installed on systems that will be used for FIPS or the real-time kernel.
```

- **Reason:** The type of installation determines which system components and packages are installed on the server. Choosing the appropriate option ensures that your system has the necessary tools for the intended purpose. If this step is not properly configured, the server might be missing essential packages or may be overburdened with unnecessary components.
- **Method:**
  1. **Select Ubuntu Server (Full):**

     This option provides a comprehensive set of curated packages necessary for operating a server. It is designed to give a comfortable experience with all core server functionality pre-installed.

2. **Alternative Option (Minimized):**

   If a minimal server environment is needed, users can choose **Ubuntu Server (minimized)**. This version provides a lightweight footprint with a smaller runtime environment, ideal for environments where human interaction with the system is rare (e.g., virtualized or automated tasks).

3. **Optional - Search for Third-Party Drivers:**

   Optionally, the installer can search for third-party drivers if the server hardware requires additional proprietary drivers for networking, storage, or graphics cards.

- **Method Description:**

1. Ubuntu Server vs Ubuntu Server(minimized) checkbox:

   The minimized version is a reduced set of pre-installed software compared to the normal server installation.

   We download both versions and compare:

   ```
   apt list --installed > /tmp/ubuntu-2204-minimized-apps.txt
   ```
   Contents of /tmp/ubuntu-2204-minimized-apps.txt (420 packages)

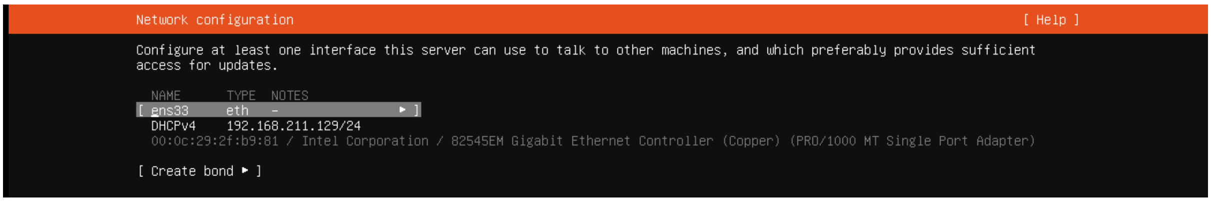
   ```
   apt list --installed > /tmp/ubuntu-2204-full-apps.txt
   ```
   Contents of /tmp/ubuntu-2204-full-apps.txt (606 packages)

2. The "Search for third-party drivers" checkbox does two things:

   It installs `ubuntu-restricted-addons` much later on in the main install process.

   It asks the Additional Drivers program to enable any drivers that can be automatically installed.

   (The message beneath mentions that these drivers may be **proprietary**, meaning that they are not open-source and may come with **license terms**.)

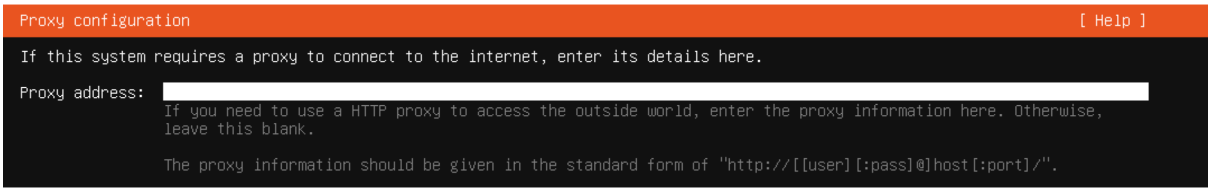


```
Network configuration                                                              [ Help ]

Configure at least one interface this server can use to talk to other machines, and which preferably provides sufficient
access for updates.

  NAME      TYPE  NOTES
[ ens33     eth   -                    ▸ ]
  DHCPv4    192.168.211.129/24
  00:0c:29:2f:b9:81 / Intel Corporation / 82545EM Gigabit Ethernet Controller (Copper) (PRO/1000 MT Single Port Adapter)

[ Create bond ▸ ]
```

- In this section, Users have to select an interface for the server to connect to the Internet. Because I use this virtual machine and doesn’t enable it to use wlan interface so it only has access to eth interface.

- **[Create Bone] Bonding**, also called **port trunking** or **link aggregation** means combining several network interfaces (NICs) to a single link, providing either high-availability, load-balancing, maximum throughput, or a combination of these. Because our laptops have only 1 NIC so we'll leave it blank.

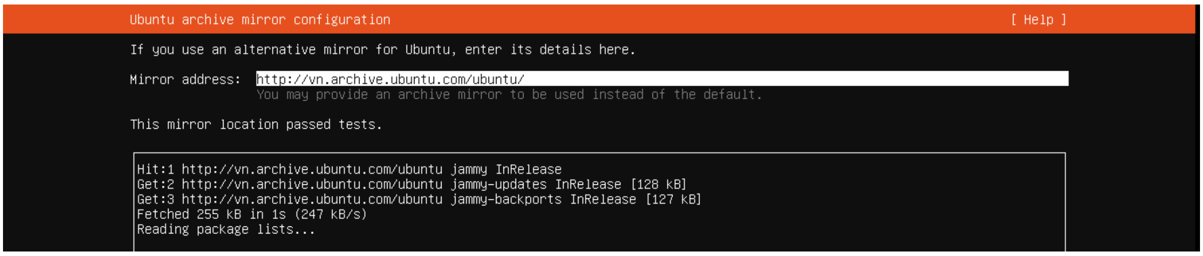
Step 3: Proxy configuration



```
Proxy configuration                                                                          [ Help ]

If this system requires a proxy to connect to the internet, enter its details here.

Proxy address:
                 If you need to use a HTTP proxy to access the outside world, enter the proxy information here. Otherwise,
                 leave this blank.

                 The proxy information should be given in the standard form of "http://[[user][:pass]@]host[:port]/".
```

- A **proxy server** is a server application that acts as an intermediary between a client requesting a resource and the server providing that resource. It improves privacy, security, and possibly performance in the process.
- We don't connect to any proxy server to send traffic to the Internet so we'll leave it blank

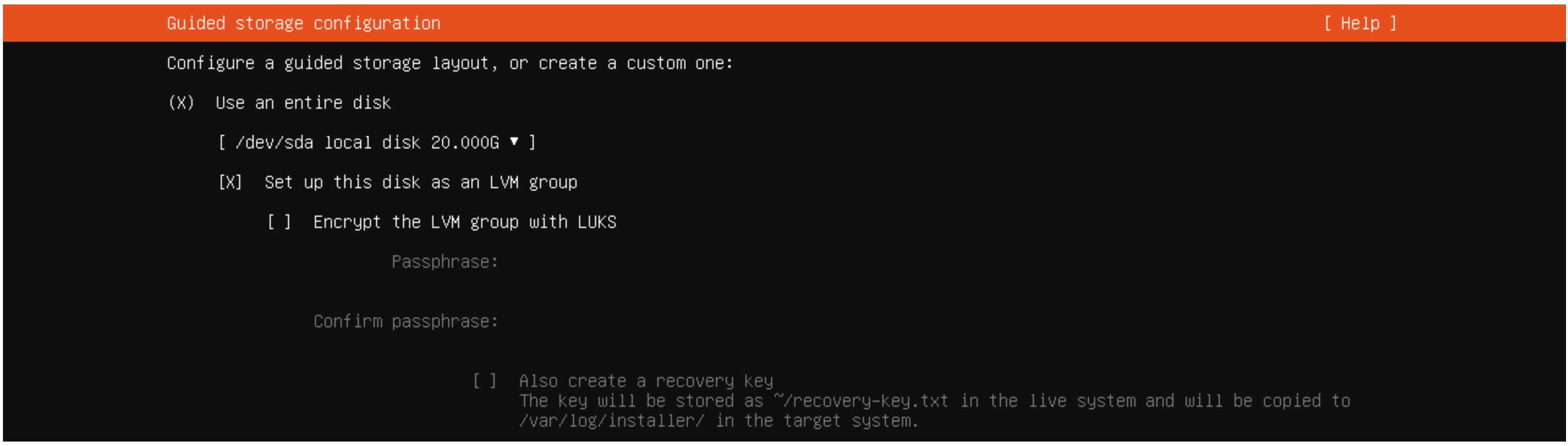
Step 4: Ubuntu archive mirror configuration



```
Ubuntu archive mirror configuration                                                  [ Help ]

If you use an alternative mirror for Ubuntu, enter its details here.

Mirror address:  http://vn.archive.ubuntu.com/ubuntu/
                 You may provide an archive mirror to be used instead of the default.

This mirror location passed tests.

Hit:1 http://vn.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://vn.archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]
Get:3 http://vn.archive.ubuntu.com/ubuntu jammy-backports InRelease [127 kB]
Fetched 255 kB in 1s (247 kB/s)
Reading package lists...
```

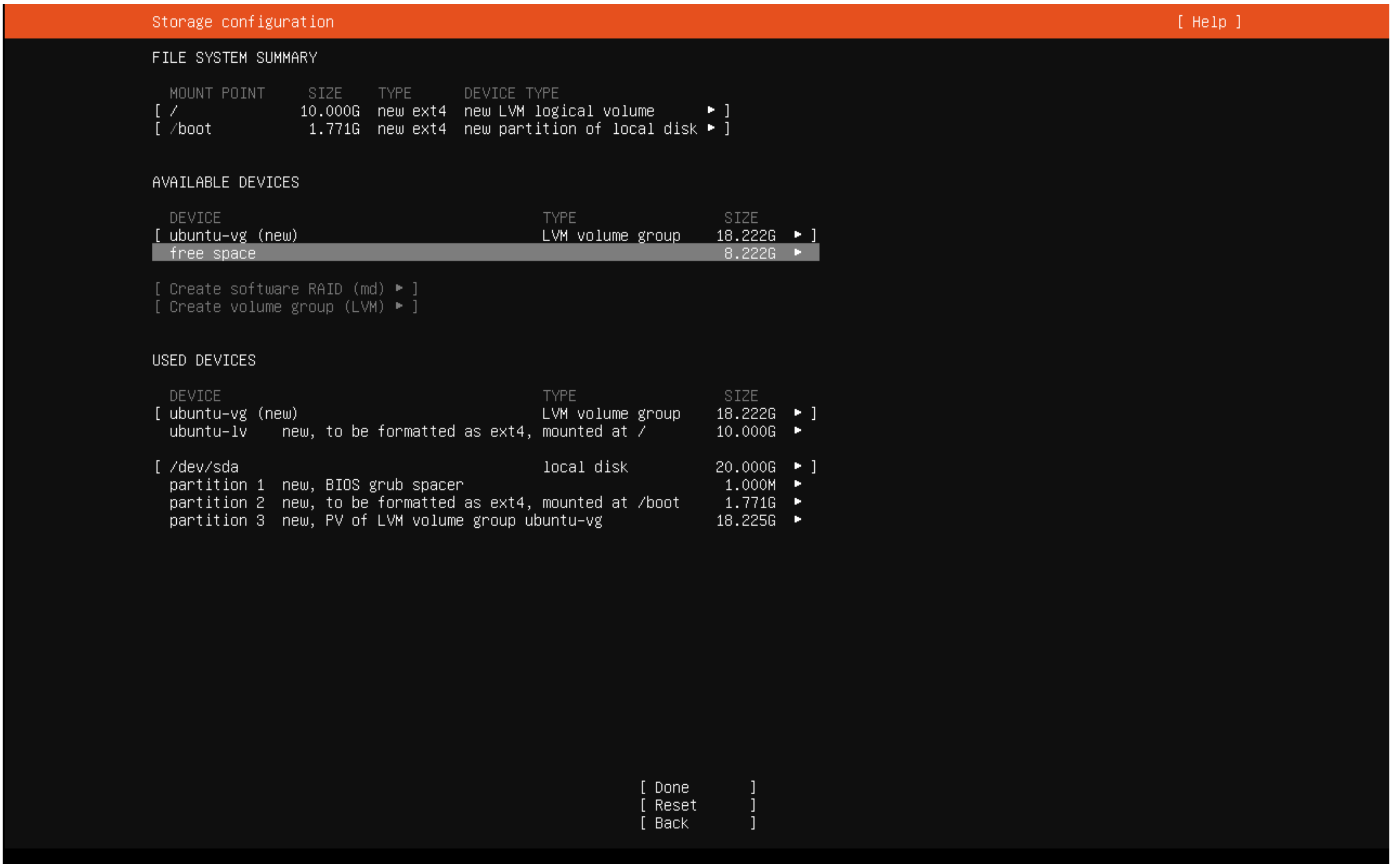
- A mirror site is a website or set of files on a computer server that has been copied to another computer server so that the site or files are available from more than one place. A mirror site has its own URL but is otherwise identical to the principal site.
- Mirror address is where distributes software packages,When you run `sudo apt update` or `sudo apt install package-name`, the package manager (APT) fetches the necessary files from a mirror site.
- There are two types of mirrors:
  - Country mirrors (e.g. nl.archive.ubuntu.com/nl.releases.ubuntu.com) (this's what we use)
  - Normal mirrors (reachable via their own hostname)

Step 5: Storage configuration



```
Guided storage configuration                                                             [ Help ]
Configure a guided storage layout, or create a custom one:

(X)  Use an entire disk

     [ /dev/sda local disk 20.000G ▾ ]

     [X]  Set up this disk as an LVM group

          [ ]  Encrypt the LVM group with LUKS

                       Passphrase:


                  Confirm passphrase:


                              [ ]  Also create a recovery key
                                   The key will be stored as ~/recovery-key.txt in the live system and will be copied to
                                   /var/log/installer/ in the target system.
```

- Beside the classic disk storage layout using partitions, Linux operating systems include the possibility to use the logical volume manager (LVM), that allows to create logical volumes out of one or multiple physical fixed disks. LVM volumes can be created on both software RAID partitions and standard partitions. Such volumes can be extended, giving greater flexibility to systems as requirements change.
- When installing Ubuntu Server, we choose to use LVM by selecting the Set up this disk as an LVM group checkbox of the 'Use an entire disk'.



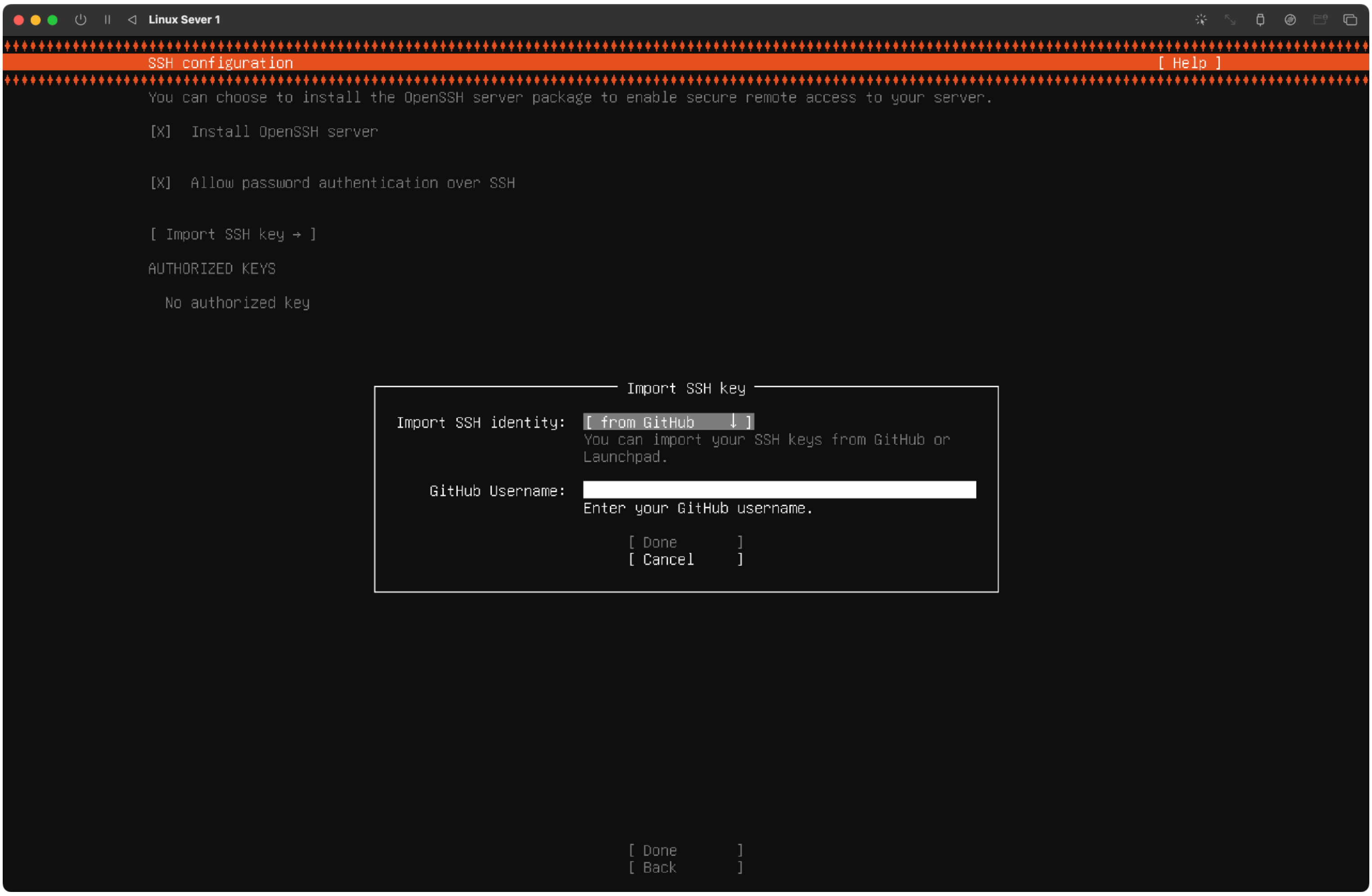
```
Storage configuration                                                                                  [ Help ]
FILE SYSTEM SUMMARY

  MOUNT POINT     SIZE    TYPE      DEVICE TYPE
[ /              10.000G  new ext4  new LVM logical volume      ▸ ]
[ /boot           1.771G  new ext4  new partition of local disk ▸ ]

AVAILABLE DEVICES

  DEVICE                                     TYPE                SIZE
[ ubuntu-vg (new)                            LVM volume group    18.222G  ▸ ]
  free space                                                      8.222G  ▸

[ Create software RAID (md) ▸ ]
[ Create volume group (LVM) ▸ ]

USED DEVICES

  DEVICE                                     TYPE                SIZE
[ ubuntu-vg (new)                            LVM volume group    18.222G  ▸ ]
  ubuntu-lv    new, to be formatted as ext4, mounted at /        10.000G  ▸

[ /dev/sda                                   local disk          20.000G  ▸ ]
  partition 1  new, BIOS grub spacer                              1.000M  ▸
  partition 2  new, to be formatted as ext4, mounted at /boot     1.771G  ▸
  partition 3  new, PV of LVM volume group ubuntu-vg             18.225G  ▸

                                                         [ Done       ]
                                                         [ Reset      ]
                                                         [ Back       ]
```

In an LVM context:

- A **physical volume (PV)** is a physical fixed disk, disk partition, or software RAID partition formatted as LVM PV. In the screenshot above, there is one PV, corresponding to /dev/sda3 (the first two partitions on the disk being used for booting the system (future mount points /boot).
- A **volume group (VG)** is made from one or more physical volumes. After having been created, it can be extended by adding more PVs (cf. Extending an LVM group). A VG is like a virtual disk drive, from which one or more logical volumes are carved. The screenshot shows that the setup program will create an LVM volume group, called ubuntu-vg (made from /dev/sda3).
- A **logical volume (LV)** is similar to a partition in a non-LVM system. A LV is formatted with the desired file system (EXT3, EXT4, XFS, JFS, etc) and is then available for mounting and data storage. In our case, the logical volume, called ubuntu-lv, corresponds to the 10/18.222 G of the volume group (virtual disk) ubuntu-vg, i.e. to

the physical volume /dev/sda3. It will be formatted using the EXT4 filesystem and mounted at /.

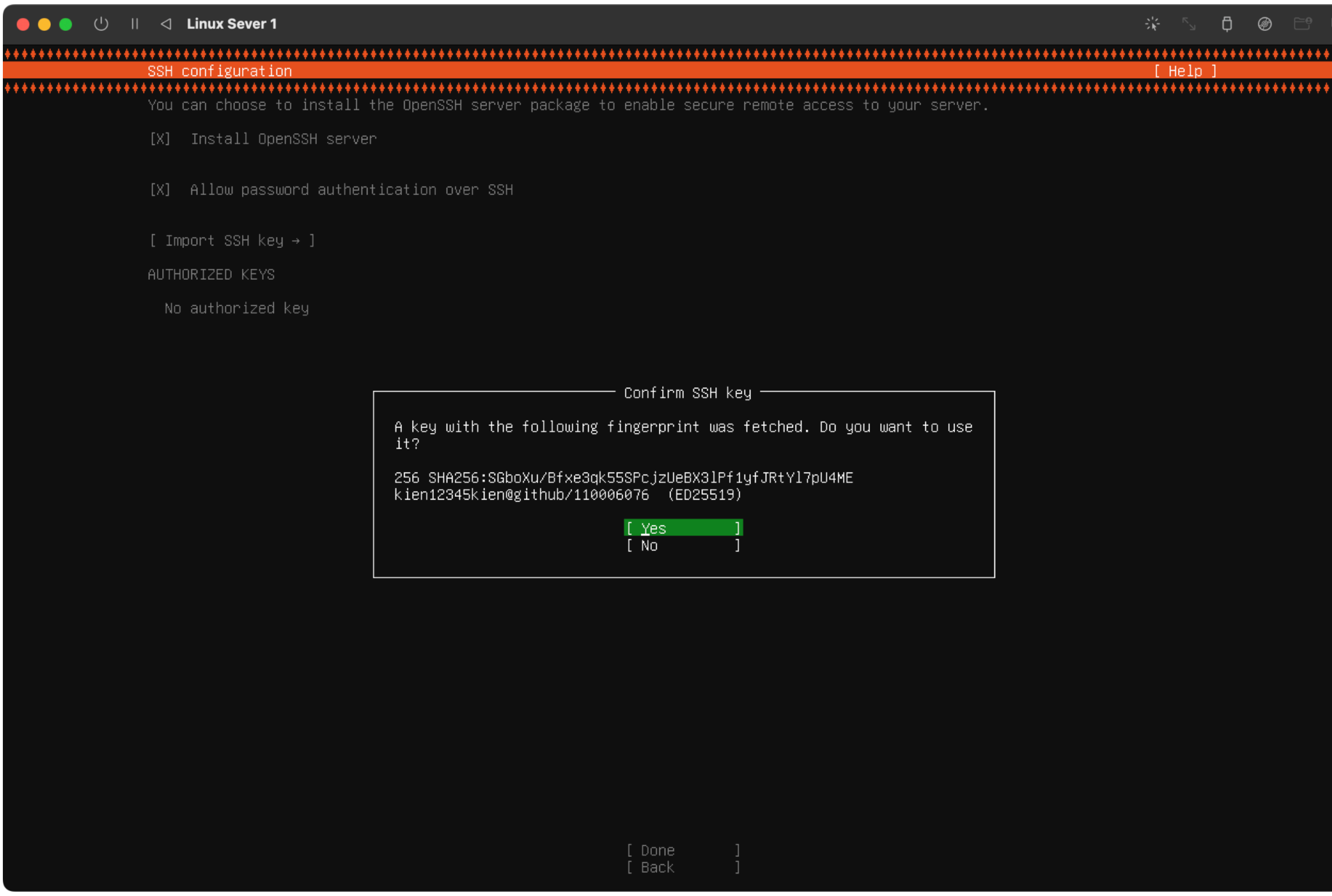
Step 6: SSH configuration



```
SSH configuration                                                                    [ Help ]
You can choose to install the OpenSSH server package to enable secure remote access to your server.

[X]  Install OpenSSH server

[X]  Allow password authentication over SSH

[ Import SSH key → ]

AUTHORIZED KEYS

  No authorized key

                       ──────────────── Import SSH key ────────────────
                        Import SSH identity:  [ from GitHub      ↓ ]
                                              You can import your SSH keys from GitHub or
                                              Launchpad.

                            GitHub Username:  
                                              Enter your GitHub username.

                                                   [ Done       ]
                                                   [ Cancel     ]

                                                   [ Done       ]
                                                   [ Back       ]
```

The image shows an SSH configuration screen for a Linux Server. The current screen allows the user to:

1. Install the OpenSSH server: This option is checked and means you will install the OpenSSH server package. OpenSSH is a set of tools.
2. Allow password authentication over SSH(Enable password authentication over SSH).
3. Import SSH identity(enter SSH): Key SSH from Github (code hosting platforms) for secure authentication. This feature makes it easy to fetch and use an SSH key stored on GitHub.
4. Authorized key: Currently, no authorized keys are listed ("No authorized key"). This section displays any SSH keys that have been added and authorized for access to the server.



```
SSH configuration                                                                    [ Help ]
        You can choose to install the OpenSSH server package to enable secure remote access to your server.

        [X]  Install OpenSSH server

        [X]  Allow password authentication over SSH

        [ Import SSH key → ]

        AUTHORIZED KEYS

          No authorized key

                     ┌──────────────────────── Confirm SSH key ────────────────────────┐
                     │                                                                 │
                     │  A key with the following fingerprint was fetched. Do you want to use │
                     │  it?                                                            │
                     │                                                                 │
                     │  256 SHA256:SGboXu/Bfxe3qk55SPcjzUeBX3lPf1yfJRtYl7pU4ME          │
                     │  kien12345kien@github/110006076  (ED25519)                      │
                     │                                                                 │
                     │                         [ Yes          ]                        │
                     │                         [ No           ]                        │
                     │                                                                 │
                     └─────────────────────────────────────────────────────────────────┘

                                             [ Done         ]
                                             [ Back         ]
```

Confirm SSH key:

- The system has successfully fetched an SSH key associated with the provided GitHub username. The key's fingerprint is displayed for verification.

Fingerprint:

- The fingerprint of the SSH key is 256 `SHA256:SGboXu/Bfxe3qk5SSP....`. This is a unique identifier of the key to ensure that the correct one is being imported.

GitHub account:

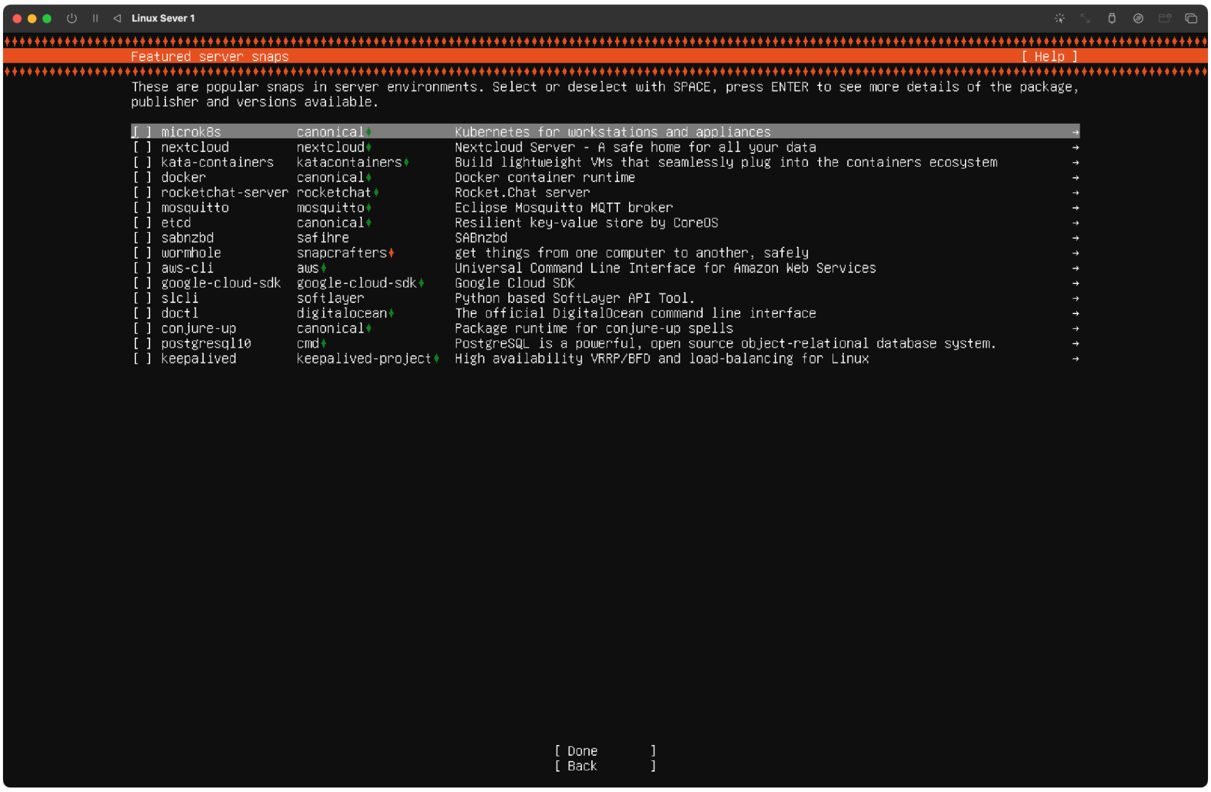
- The key is fetched from `kien12345kien@github/11000676`. This shows the GitHub account and a unique identifier for the key.

ED25519:

- This indicates that the key is of type ED25519, which is a modern, high-security algorithm for SSH keys.

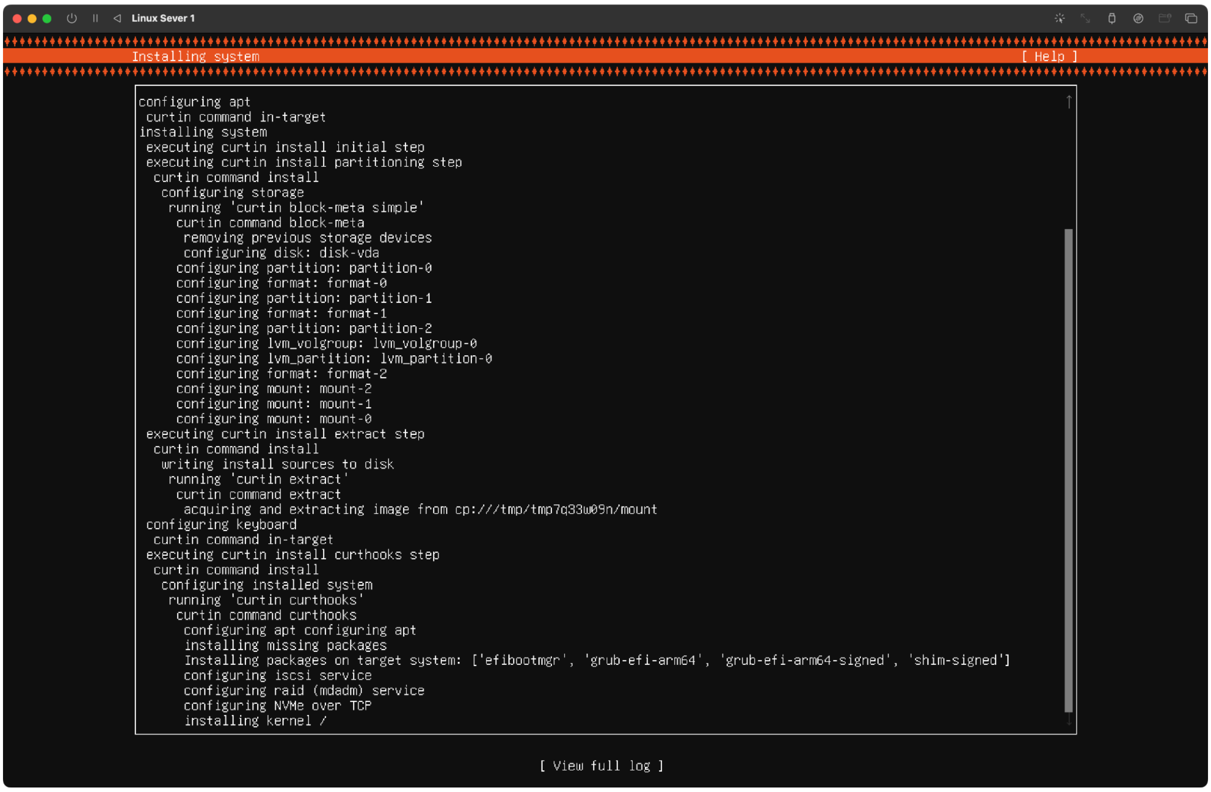
Confirmation options:

- **Yes**: If you choose "Yes," the SSH key will be imported and added to the list of authorized keys for SSH access to the server.
- **No**: If you choose "No," the import will be canceled.



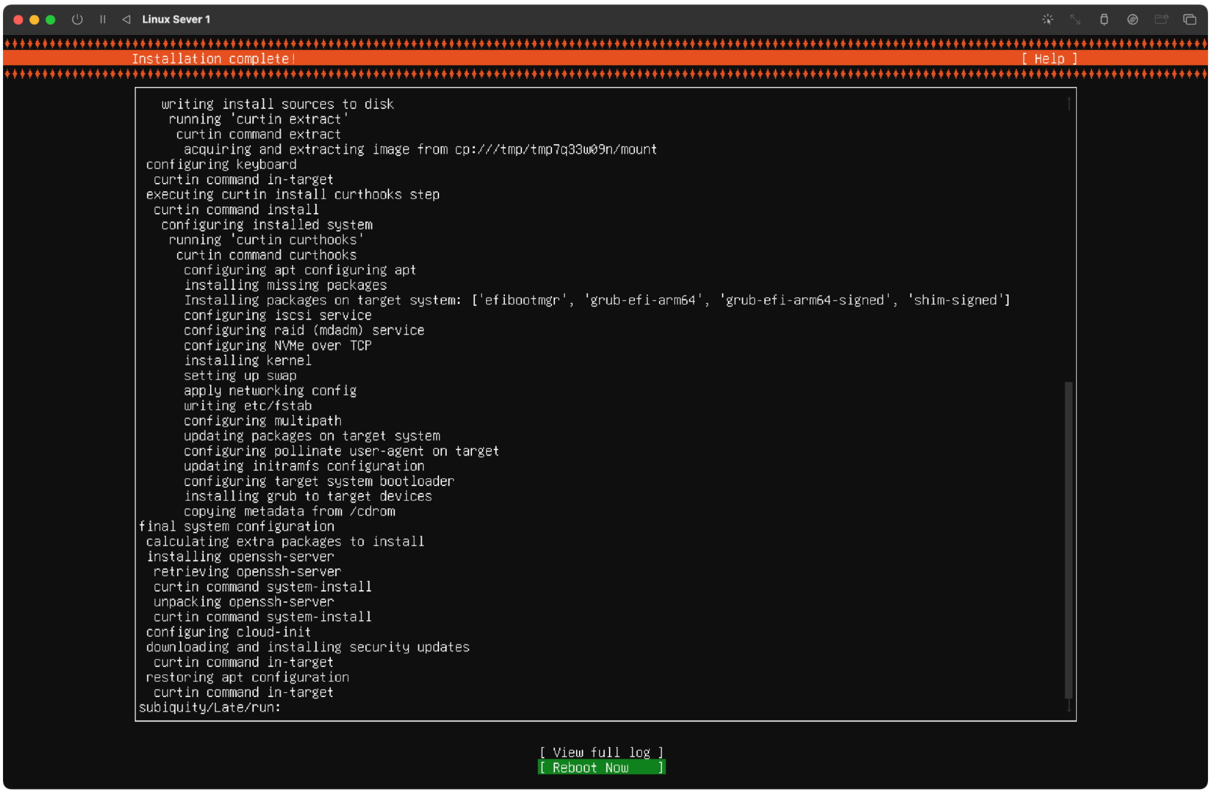
```
  Featured server snaps                                                                                  [ Help ]
  These are popular snaps in server environments. Select or deselect with SPACE, press ENTER to see more details of the package,
  publisher and versions available.

  [ ] microk8s          canonical✓          Kubernetes for workstations and appliances                         →
  [ ] nextcloud         nextcloud✓          Nextcloud Server - A safe home for all your data                   →
  [ ] kata-containers   katacontainers✓     Build lightweight VMs that seamlessly plug into the containers ecosystem →
  [ ] docker            canonical✓          Docker container runtime                                           →
  [ ] rocketchat-server rocketchat✓         Rocket.Chat server                                                 →
  [ ] mosquitto         mosquitto✓          Eclipse Mosquitto MQTT broker                                      →
  [ ] etcd              canonical✓          Resilient key-value store by CoreOS                                →
  [ ] sabnzbd           safihre             SABnzbd                                                            →
  [ ] wormhole          snapcrafters✪       get things from one computer to another, safely                    →
  [ ] aws-cli           aws✓                Universal Command Line Interface for Amazon Web Services           →
  [ ] google-cloud-sdk  google-cloud-sdk✓   Google Cloud SDK                                                   →
  [ ] slcli             softlayer           Python based SoftLayer API Tool.                                   →
  [ ] doctl             digitalocean✓       The official DigitalOcean command line interface                   →
  [ ] conjure-up        canonical✓          Package runtime for conjure-up spells                              →
  [ ] postgresql10      cmd✓                PostgreSQL is a powerful, open source object-relational database system. →
  [ ] keepalived        keepalived-project✓ High availability VRRP/BFD and load-balancing for Linux            →

                                              [ Done       ]
                                              [ Back       ]
```

The image shows a selection menu for **Featured server snaps** in a Linux environment. Snaps are containerized software packages that can be installed on Linux servers to provide additional functionality. Each snap is associated with a publisher and contains a brief description of its purpose.



```
configuring apt
 curtin command in-target
installing system
 executing curtin install initial step
 executing curtin install partitioning step
  curtin command install
   configuring storage
    running 'curtin block-meta simple'
     curtin command block-meta
      removing previous storage devices
      configuring disk: disk-vda
     configuring partition: partition-0
     configuring format: format-0
     configuring partition: partition-1
     configuring format: format-1
     configuring partition: partition-2
     configuring lvm_volgroup: lvm_volgroup-0
     configuring lvm_partition: lvm_partition-0
     configuring format: format-2
     configuring mount: mount-2
     configuring mount: mount-1
     configuring mount: mount-0
 executing curtin install extract step
  curtin command install
   writing install sources to disk
    running 'curtin extract'
     curtin command extract
      acquiring and extracting image from cp:///tmp/tmp7q33w09n/mount
 configuring keyboard
  curtin command in-target
 executing curtin install curthooks step
  curtin command install
   configuring installed system
    running 'curtin curthooks'
     curtin command curthooks
      configuring apt configuring apt
      installing missing packages
      Installing packages on target system: ['efibootmgr', 'grub-efi-arm64', 'grub-efi-arm64-signed', 'shim-signed']
      configuring iscsi service
      configuring raid (mdadm) service
      configuring NVMe over TCP
      installing kernel /
```

The image shows a detailed system installation process on a Linux server.



```
   writing install sources to disk
    running 'curtin extract'
     curtin command extract
      acquiring and extracting image from cp:///tmp/tmp7q33w09n/mount
 configuring keyboard
  curtin command in-target
 executing curtin install curthooks step
  curtin command install
   configuring installed system
    running 'curtin curthooks'
     curtin command curthooks
      configuring apt configuring apt
      installing missing packages
      Installing packages on target system: ['efibootmgr', 'grub-efi-arm64', 'grub-efi-arm64-signed', 'shim-signed']
      configuring iscsi service
      configuring raid (mdadm) service
      configuring NVMe over TCP
      installing kernel
      setting up swap
      apply networking config
      writing etc/fstab
      configuring multipath
      updating packages on target system
      configuring pollinate user-agent on target
      updating initramfs configuration
      configuring target system bootloader
      installing grub to target devices
      copying metadata from /cdrom
final system configuration
 calculating extra packages to install
 installing openssh-server
  retrieving openssh-server
  curtin command system-install
  unpacking openssh-server
  curtin command system-install
 configuring cloud-init
 downloading and installing security updates
  curtin command in-target
 restoring apt configuration
  curtin command in-target
subiquity/Late/run:
```

The above image shows the final stages of a Linux system installation, with the message **Installation complete!** at the top.

III. Configuration

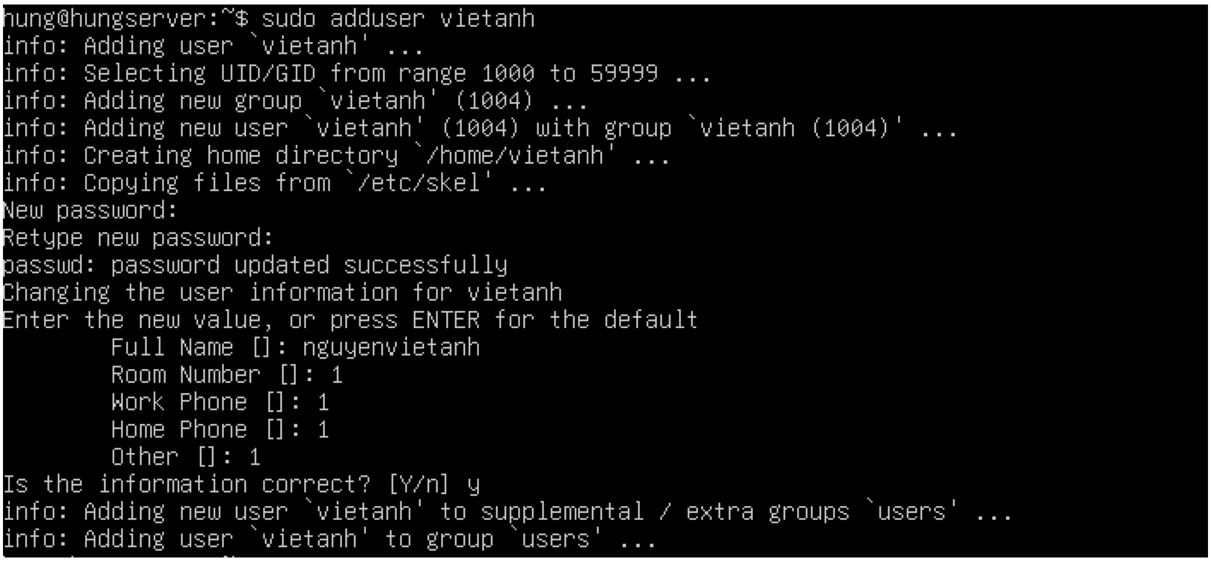
Step 1: Setting up User Accounts

- Creating user accounts for each group member on the server.
- Each group member needs a unique account to securely access and interact with the server. Without separate accounts, tracking activities or managing permissions would become difficult and insecure.

1. Use the `adduser` command to create accounts for each member of the group.
2. Set a password for each user after creating the account.

Commands:

```
sudo adduser username
sudo passwd username
```



```
hung@hungserver:~$ sudo adduser vietanh
info: Adding user `vietanh' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `vietanh' (1004) ...
info: Adding new user `vietanh' (1004) with group `vietanh (1004)' ...
info: Creating home directory `/home/vietanh' ...
info: Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for vietanh
Enter the new value, or press ENTER for the default
        Full Name []: nguyenvietanh
        Room Number []: 1
        Work Phone []: 1
        Home Phone []: 1
        Other []: 1
Is the information correct? [Y/n] y
info: Adding new user `vietanh' to supplemental / extra groups `users' ...
info: Adding user `vietanh' to group `users' ...
```

The `adduser` command creates a new user and assigns them a home directory. The `passwd` command sets the password for the user, ensuring each member has secure access to the server. Creating separate accounts for each user allows proper assignment of privileges and responsibilities.

Step 2: Configure SSH for Remote Access

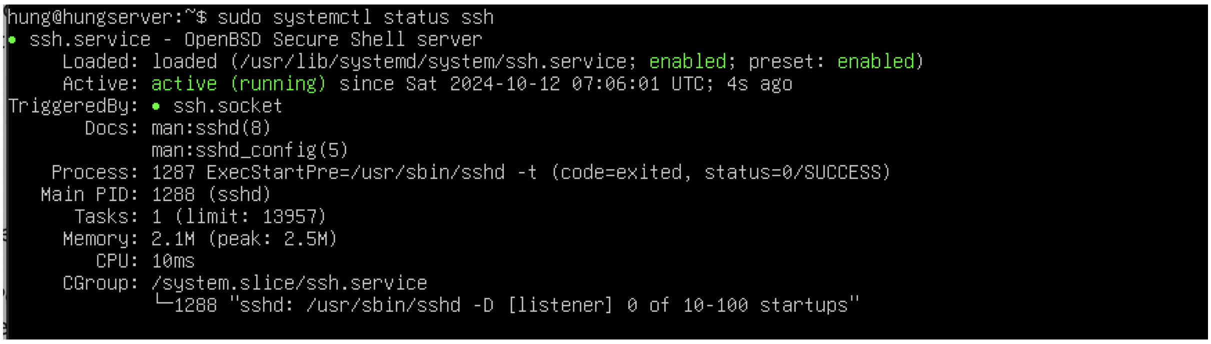
- Enabling SSH for secure remote access.

-SSH (Secure Shell) allows group members to remotely connect to the server. Without SSH, members would need physical access to the machine or use insecure alternatives to interact with the server.

1. Install the OpenSSH server if it’s not already installed.
2. Enable SSH to start automatically upon server boot.
3. Start the SSH service to allow immediate remote connections.

Commands:

sudo apt update
sudo apt install openssh-server
sudo systemctl enable ssh
sudo systemctl start ssh
sudo systemctl status ssh



```
hung@hungserver:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/usr/lib/systemd/system/ssh.service; enabled; preset: enabled)
     Active: active (running) since Sat 2024-10-12 07:06:01 UTC; 4s ago
TriggeredBy: ● ssh.socket
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 1287 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 1288 (sshd)
      Tasks: 1 (limit: 13957)
     Memory: 2.1M (peak: 2.5M)
        CPU: 10ms
     CGroup: /system.slice/ssh.service
             └─1288 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"
```

- These commands ensure that SSH is installed, running, and set to start automatically on boot. This allows all group members to connect securely to the server remotely. It’s important to check the status of SSH to ensure the service is running correctly.

Step 3: Configure User Permissions

- Setting appropriate permissions for directories and files for each user.
- Permissions ensure that each user has access only to the files and directories they are authorized to use. Without configuring permissions, users may access, modify, or delete files they shouldn't.

1. Use `chown` to assign ownership of directories and files to specific users.
2. Use `chmod` to configure read, write, and execute permissions for users, groups, and others.

Commands:

sudo chown username:username /path/to/directory
sudo chmod 755 /path/to/directory

- **Method Description**:

The `chown` command changes the ownership of a file or directory, ensuring that only the intended user has full control over it. The `chmod` command sets the permissions for users and groups. This ensures proper access control, protecting sensitive files and resources from unauthorized users.

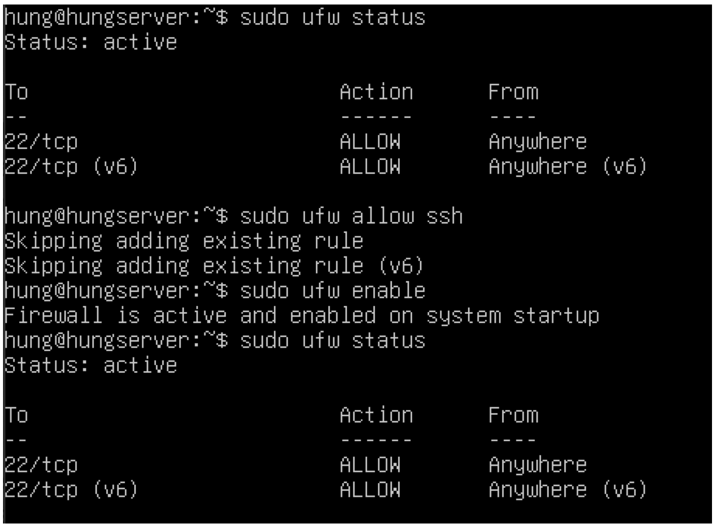
Step 4: Configure Firewall for Security

- Setting up the firewall to secure the server.
- A firewall restricts unauthorized access to the server by controlling which ports are open. Without a firewall, the server could be vulnerable to network attacks.

1. Enable the `ufw` (Uncomplicated Firewall) and allow SSH connections.
2. Enable the firewall to block all other unauthorized traffic.

Commands:

```
sudo ufw allow ssh
sudo ufw enable
sudo ufw status
```



```
hung@hungserver:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)

hung@hungserver:~$ sudo ufw allow ssh
Skipping adding existing rule
Skipping adding existing rule (v6)
hung@hungserver:~$ sudo ufw enable
Firewall is active and enabled on system startup
hung@hungserver:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
```

- Method Description:

The `ufw` firewall ensures that only authorized connections, such as SSH, are allowed to the server. The firewall blocks all other traffic, making the server more secure. Running the status command verifies that the firewall is active and configured correctly.

Step 5: Verifying and Testing the Setup

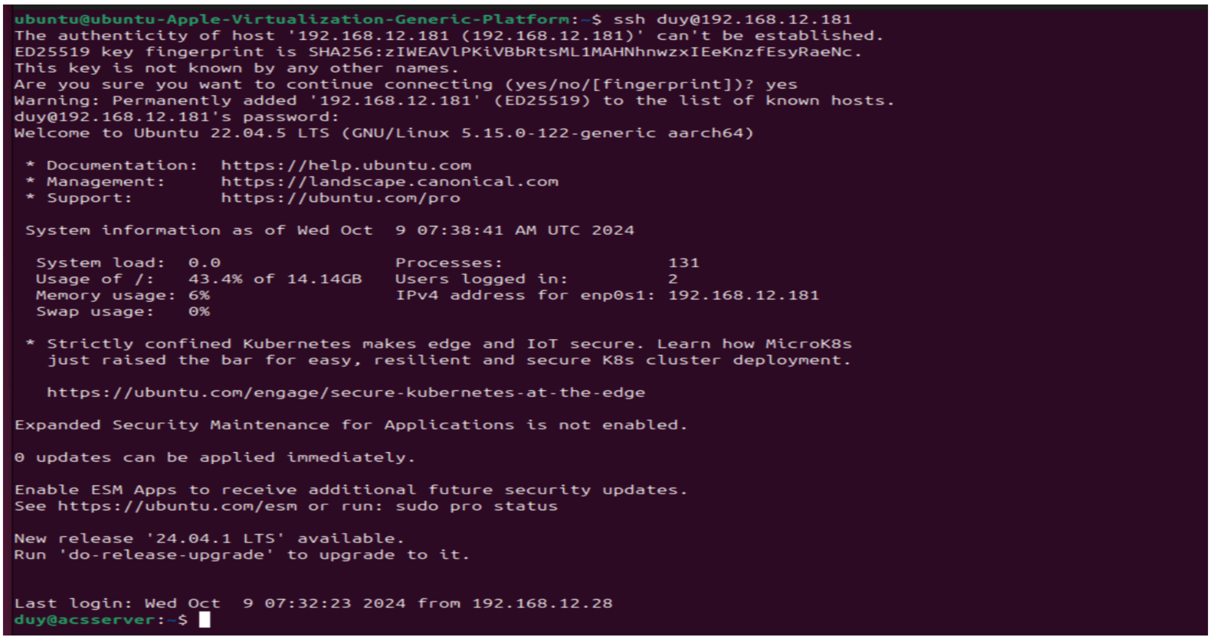
- Testing the configurations to ensure the server is correctly set up.
- After configuration, it's critical to verify that everything is working as intended. Without testing, configuration errors may go unnoticed, causing issues with access or security.

1. Test SSH Access: Ensure that all group members can connect to the server using SSH.
2. Test User Permissions: Verify that users can access only the files and directories they should have access to.

3. Test Firewall Rules: Ensure that the firewall is blocking unauthorized traffic while allowing authorized connections (e.g., SSH).

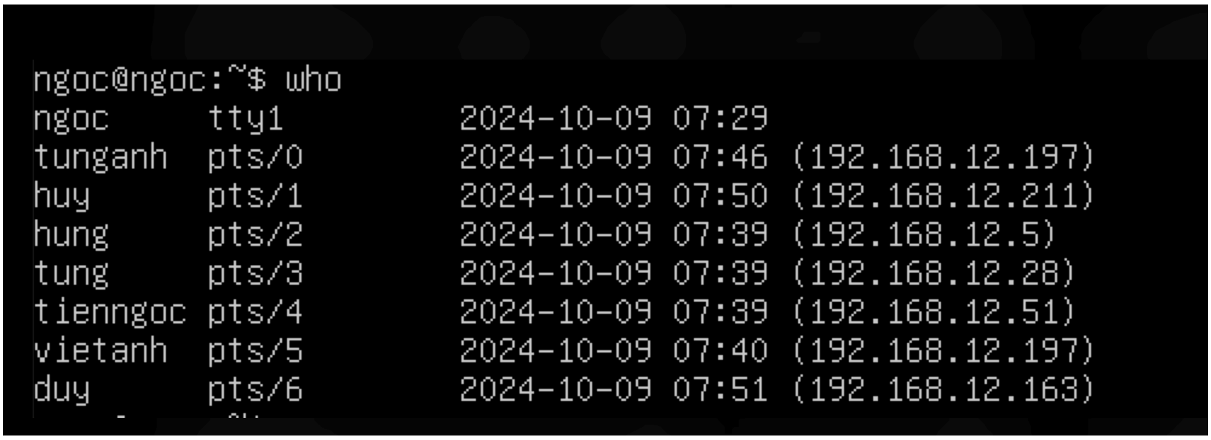
Commands:

ssh username@server-ip



```
ubuntu@ubuntu-Apple-Virtualization-Generic-Platform:~$ ssh duy@192.168.12.181
The authenticity of host '192.168.12.181 (192.168.12.181)' can't be established.
ED25519 key fingerprint is SHA256:zIWEAVlPKiVBbRtsML1MAHNhnwzxIEeKnzfEsyRaeNc.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.12.181' (ED25519) to the list of known hosts.
duy@192.168.12.181's password:
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 5.15.0-122-generic aarch64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Wed Oct  9 07:38:41 AM UTC 2024

  System load:  0.0               Processes:             131
  Usage of /:   43.4% of 14.14GB   Users logged in:       2
  Memory usage: 6%                IPv4 address for enp0s1: 192.168.12.181
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

New release '24.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Wed Oct  9 07:32:23 2024 from 192.168.12.28
duy@acsserver:~$ 
```

- Testing SSH access ensures that users can connect securely to the server. Verifying file permissions and firewall settings confirms that the server is properly secured and that group members have the correct level of access.



```
ngoc@ngoc:~$ who
ngoc      tty1         2024-10-09 07:29
tunganh   pts/0        2024-10-09 07:46 (192.168.12.197)
huy       pts/1        2024-10-09 07:50 (192.168.12.211)
hung      pts/2        2024-10-09 07:39 (192.168.12.5)
tung      pts/3        2024-10-09 07:39 (192.168.12.28)
tienngoc  pts/4        2024-10-09 07:39 (192.168.12.51)
vietanh   pts/5        2024-10-09 07:40 (192.168.12.197)
duy       pts/6        2024-10-09 07:51 (192.168.12.163)
```