



**National University**  
Of Computer and Emerging Sciences

## **Assignment 1**

**Name – Muhammad Taha**

**Roll NO – 23P-0559**

**SECTION – BCS(4A)**

**Subject – OPERATING SYSTEM- LAB**

# Introduction to Gentoo Linux

Gentoo Linux is a source-based distribution, meaning users compile software from source instead of using pre-compiled binaries. This allows for a highly customizable system tailored to specific hardware and needs. It is known for its flexibility, performance and deep learning experience making it a great choice for advanced Linux users.

## Key Features and Benefits

One of the biggest benefits of Gentoo is customization users have full control over what gets installed from the kernel to individual software features. Because everything is compiled specifically for the hardware performance is often better than binary-based distributions making it ideal for performance-critical tasks. Gentoo also follows a rolling-release model meaning it stays up-to-date without requiring major version upgrades. Another major advantage is the learning experience as installing and configuring Gentoo requires a deep understanding of Linux making it ideal for those who want to explore system internals.

## Source-Based vs Binary-Based Distributions

Unlike binary distributions which provide pre-compiled software source-based distributions like Gentoo require users to compile their programs. This has several advantages: optimized software (compiled for specific hardware for better efficiency), flexibility (users can enable or disable features using USE flags), and access to the latest software (often before it appears in binary distributions). However, compiling software takes time and requires technical knowledge.

## Comparison with Other Distributions

### Gentoo vs Ubuntu

- Ubuntu Advantages: User-friendly pre-packaged binaries, large community support.
- Ubuntu Disadvantages: Less customizable, comes with pre-configured settings that may not be ideal for everyone.
- Main Difference: Gentoo is more customizable and optimized, while Ubuntu is easier to set up and use.

### Gentoo vs Arch Linux

- Arch Advantages: Rolling-release model, minimalist approach, extensive Arch Wiki documentation.
- Arch Disadvantages: Requires manual installation and configuration.
- Main Difference: Both are rolling-release distributions, but Gentoo focuses on source-based compilation for optimization, while Arch is binary-based and more user-friendly.

## Challenges of Using Gentoo

While Gentoo offers great flexibility and performance it requires more time and effort to install and maintain. Compiling software can take a long time, and users need to have a good understanding of Linux to configure their system properly. This makes Gentoo less suitable for beginners or those looking for a simple setup.

## Gentoo Dual Boot Installation

This step-by-step guide explains how to install Gentoo Linux using the Live GUI environment alongside another operating system in a dual-boot setup. The Live GUI provides a graphical interface, making the installation process smooth and user-friendly.

### Step 1: Download Required Files

#### 1.1 Get the Gentoo LiveGUI USB Image

Download the Gentoo LiveGUI image from the official website: [Gentoo Downloads](#)

Choose the correct version for your system:

- **AMD64 (x86\_64)** → For most modern Intel and AMD processors (64-bit PCs and laptops).
- **ARM64 (aarch64)** → For ARM-based devices like Raspberry Pi, Apple M1/M2.

#### 1.2 Create a Bootable USB

1. Download **Rufus**.
2. Insert a **USB (at least 8GB)** into your system.
3. Open **Rufus** and:
  - Select the downloaded LiveGUI ISO file.
  - Set **Partition Scheme** → GPT.
  - Set **Target System** → BIOS or UEFI.
  - Choose **File System** → FAT32.
4. Click Start. When prompted, select DD Image Mode (recommended for Gentoo).

### Step 2: Boot from USB

1. Restart your PC and enter the Boot Menu:
  - Dell → Press F12

- HP → Press F9
  - Other brands may use Esc, F2, or Del
2. Select your USB drive and boot into Gentoo Live GUI.
  3. If the Boot Menu doesn't appear:
    - Enter **BIOS/UEFI** (F2 or Del at startup).
    - Change **Boot Priority** to USB first.
    - Disable **Secure Boot**.

### Step 3: Prepare for Installation

1. Once booted into Gentoo Live GUI, connect to Wi-Fi and set Date & Time.
2. Open Terminal and gain root access:
3. **sudo su** # Enter superuser mode
4. **passwd** # Set root password
5. Check available disks:
6. **lsblk**

### Step 4: Partition the Drive

Open cfdisk and select your disk (e.g., /dev/sda):

**cfdisk /dev/sda**

Create the following partitions:

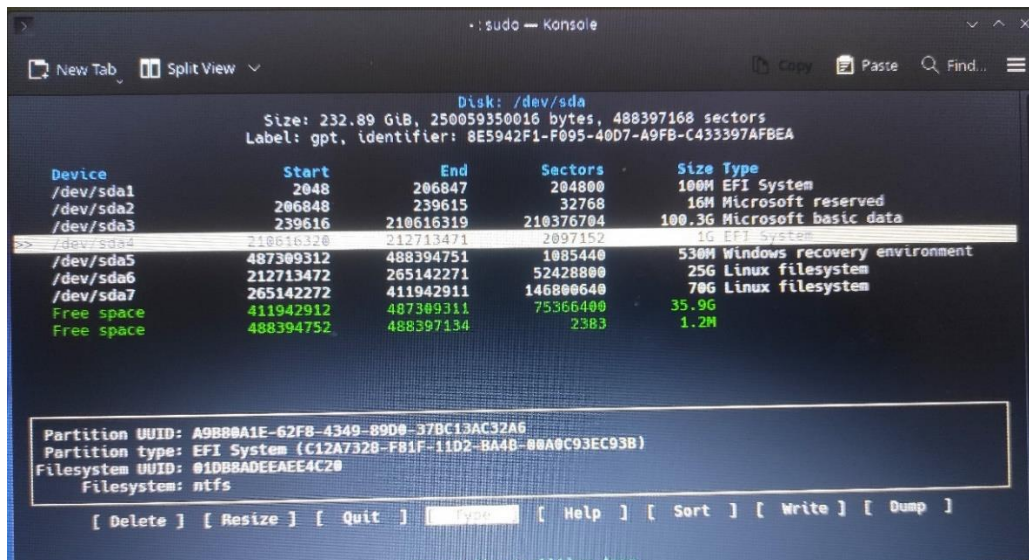
**EFI Partition** (512MB - 1GB, Type: EFI System) → Stores bootloader.

**Root Partition** (25GB+ for Gentoo, Type: Linux Filesystem).

**Swap Partition** (Optional, 4GB+ recommended, Type: Linux Swap).

Save changes and exit.

Note: create swap partion in case your ram is less then 8 gb.



## Step 5: Format and Mount Partitions

### 5.1 Format Partitions

**mkfs.vfat -F32 /dev/sdax** # Format the EFI Partition with FAT32 (UEFI requires this format).

**mkfs.ext4 /dev/sdax** # Format the Root Partition with EXT4 (stable and widely used).

**mkswap /dev/sdax** # Format the Swap Partition (if created).

### 5.2 Mount Partitions

**mount /dev/sda2 /mnt/gentoo** # Mount the root partition at /mnt/gentoo.

**mkdir -p /mnt/gentoo/boot** # Create a directory for the boot partition.

**mount /dev/sda1 /mnt/gentoo/boot** # Mount the EFI partition inside /boot.

**swapon /dev/sda3** # Enable the swap partition (if created).

This x is depend on your partition number of each disk.

- The root partition is mounted at /mnt/gentoo, where we will install the system.
- The boot partition is mounted at /mnt/gentoo/boot, where the bootloader files will be stored.
- swapon enables swap memory to help if RAM usage gets high.

## Step 6: Download and Extract Stage 3 Tarball

The Stage 3 tarball is a pre-compiled base system that includes essential files and tools to set up Gentoo Linux.

**cd /mnt/gentoo** #Go into the folder of the /mnt/gentoo

1. Open Firefox and download the latest Stage 3 tarball: [Gentoo Stage 3](#)
2. **mv /home/gentoo/Downloads/stage-3 + TAB(key) to complete it .tar.xz ./**
3. Extract it:
4. **tar xpvf stage3-\*.tar.xz --xattrs-include='.\*' --numeric-owner**
  - **tar xpvf** → Extracts the archive while maintaining file permissions (p), verbose output (v), and file (f).
  - **--xattrs-include='.\*'** → Ensures extended file attributes are included.
  - **--numeric-owner** → Preserves the correct file ownership.

Once extracted you now have a minimal Gentoo base system ready to configure.

## Step 7: Configure Portage

Portage is Gentoo's package management system, which allows you to install and update software.

**emerge-webrsync # Sync Portage Tree**

- Downloads and updates the latest Portage tree from Gentoo's servers.

**emerge --sync && emerge -uDN @world # Update system**

- **emerge --sync** → Synchronizes the package database.
- **emerge -uDN @world** → Updates all installed packages to their latest versions.

## Step 8: Set Timezone and Locale

Setting the correct timezone and locale ensures that your system displays the correct time and supports your preferred language.

**echo "UTC" > /etc/timezone**

**emerge --config sys-libs/timezone-data**

- **echo "UTC"** → Sets the system time to Coordinated Universal Time (UTC).
- **emerge --config sys-libs/timezone-data** → Applies the new timezone settings.

**nano /etc/locale.gen # Uncomment 'en\_US.UTF-8 UTF-8'**

- Open the locale configuration file.

- Uncomment (remove # from) the following line:

Then, generate the locale:

**locale-gen**

**eselect locale set en\_US.utf8**

- locale-gen → Generates the selected locales.
- eselect locale set en\_US.utf8 → Sets the system's default locale to US English UTF-8.

## Step 9: Install and Configure Kernel

The Linux kernel is the core of the operating system. Gentoo allows you to compile it from source, ensuring maximum optimization for your hardware.

**emerge -v =sys-kernel/gentoo-sources-6.12.16**

- Installs the latest Gentoo-sources kernel (adjust the version if needed).

**cd /usr/src/linux-version-gentoo**                      #in place of version we enter actual version of linux

Compile and install the kernel.

**make menuconfig**

# Opens a menu-based configuration for enabling/disabling kernel features.

**make -j\$(nproc)**                                      # Compiles the kernel using all available CPU cores.

**make modules\_install**                              # Installs the necessary kernel modules.

**make install**                                      # Installs the compiled kernel to /boot.

The process of compiling the Linux kernel in detailed is already explained in detail in Assignment 2.

## Step 10: Configure fstab

The fstab file defines how partitions are mounted on startup.

**nano /etc/fstab**

Add the following lines (modify if needed):

Example:

/dev/sdax /      ext4   defaults   0 1

/dev/sdax /boot   vfat   defaults   0 2

```
/dev/sdax none swap sw 0 0
```

- /dev/sda2 → Root partition, formatted as ext4.
- /dev/sda1 → Boot partition, formatted as vfat (for UEFI).
- /dev/sda3 → Swap partition (if created).

Save and exit (CTRL + X → Y → Enter).

## Step 11: Configure Network

To enable networking, we need to set the hostname and install DHCP.

```
echo "gentoo" > /etc/hostname
```

- Sets the hostname to "gentoo" (change it if needed).

```
emerge --ask net-misc/dhcpd
```

```
rc-update add dhcpd default
```

- Installs dhcpd (Dynamic Host Configuration Protocol client) to obtain an IP address.
- Adds dhcpd to startup services.

## Step 12: Install Bootloader (GRUB for Dual Boot)

To boot into Gentoo, we need to install a bootloader (GRUB).

```
emerge sys-boot/grub
```

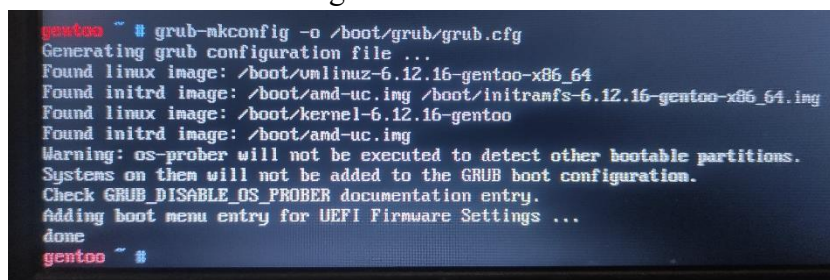
- Installs GRUB bootloader.

If using UEFI:

```
grub-install --target=x86_64-efi --efi-directory=/boot
```

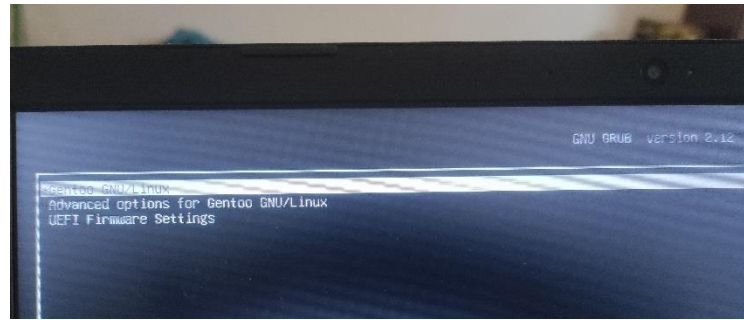
```
grub-mkconfig -o /boot/grub/grub.cfg
```

- Installs GRUB for UEFI mode.
- Generates a GRUB configuration file.



```
gentoo ~ # grub-mkconfig -o /boot/grub/grub.cfg
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-6.12.16-gentoo-x86_64
Found initrd image: /boot/amd-uc.img /boot/initramfs-6.12.16-gentoo-x86_64.img
Found linux image: /boot/kernel-6.12.16-gentoo
Found initrd image: /boot/amd-uc.img
Warning: os-prober will not be executed to detect other bootable partitions.
Systems on them will not be added to the GRUB boot configuration.
Check GRUB_DISABLE_OS_PROBER documentation entry.
Adding boot menu entry for UEFI Firmware Settings ...
done
gentoo ~ #
```





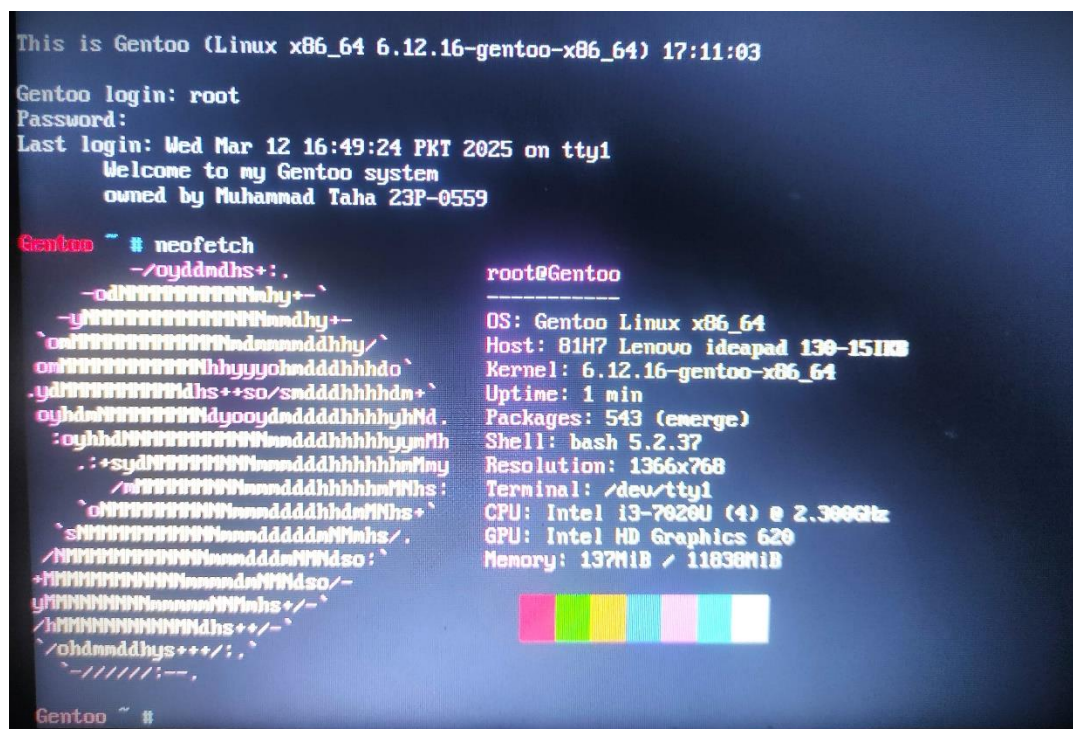
## Step 13: Finishing Up and Rebooting

```
exit # Exit chroot
```

```
umount -R /mnt/gentoo # Unmount partitions
```

```
reboot # Restart system
```

Remove the USB and boot into your newly installed Gentoo system!



Add user:

```
Gentoo ~ # useradd -m -G users,wheel,audio,video -s /bin/bash m_taha
Creating mailbox file: No such file or directory
Gentoo ~ # passwd 1234
passwd: user '1234' does not exist
Gentoo ~ # passwd m_taha

You can now choose the new password or passphrase.

A valid password should be a mix of upper and lower case letters, digits, and
other characters. You can use a password containing at least 7 characters
from all of these classes, or a password containing at least 8 characters
from just 3 of these 4 classes.
An upper case letter that begins the password and a digit that ends it do not
count towards the number of character classes used.

A passphrase should be of at least 3 words, 11 to 72 characters long, and
contain enough different characters.

Alternatively, if no one else can see your terminal now, you can pick this as
your password: "compel6clear_spring".

Enter new password:
Weak password: too short.
Re-type new password:
passwd: password updated successfully
Gentoo ~ #
```

Connect wifi:

```
... and directory index is up-to-date.

* IMPORTANT: 9 news items need reading for repository 'gentoo'.
* Use eselect news read to view new items.

Gentoo ~ # iw dev wlp2s0 scan | grep SSID
SSID: Room113
SSID: Stormfiber 01
SSID: Barg5k2k
SSID: FTCL 5G

Gentoo ~ # emerge --ask net-wireless/wpa_supplicant net-wireless/iw net-nisc/dhccpd
rc

Exiting on signal 2

Gentoo ~ # wpa_passphrase "Stormfiber 01" "03348416324" > /etc/wpa_supplicant/wpa_supplicant.conf
Gentoo ~ # wpa_supplicant -B -i wlp2s0 -c /etc/wpa_supplicant/wpa_supplicant.conf
Successfully initialized wpa_supplicant
Gentoo ~ # dhccpd wlp2s0
Sending commands to dhccpd process
Gentoo ~ # rc-update add dhccpd default
rc-update: dhccpd already installed in runlevel 'default': skipping
Gentoo ~ # rc-update add wpa_supplicant default
rc service wpa_supplicant added to runlevel default
Gentoo ~ #
```

## Troubleshooting & Issues Faced

### 1. OS Prober Not Detecting Other OS

#### Why does this happen?

When you install GRUB (the bootloader) on Gentoo for dual boot, it should detect all operating systems installed on your system. However, sometimes GRUB does not find the other OS (like Windows or another Linux distro). This usually happens because:

1. **OS Prober is missing** – GRUB needs a tool called os-prober to search for other operating systems. If it's not installed, GRUB won't detect them.
2. **OS Prober is disabled** – Some newer versions of GRUB disable OS Prober by default for security reasons.

#### How to fix it?

To make GRUB detect all installed OS:

1. **Install OS Prober** (if it's missing) by running:

`emerge os-prober`

2. **Regenerate the GRUB configuration** so it detects all OS again:

`grub-mkconfig -o /boot/grub/grub.cfg`

3. **(Optional) Enable OS Prober manually** if it's still not working:

Open the GRUB configuration file:

`nano /etc/default/grub`

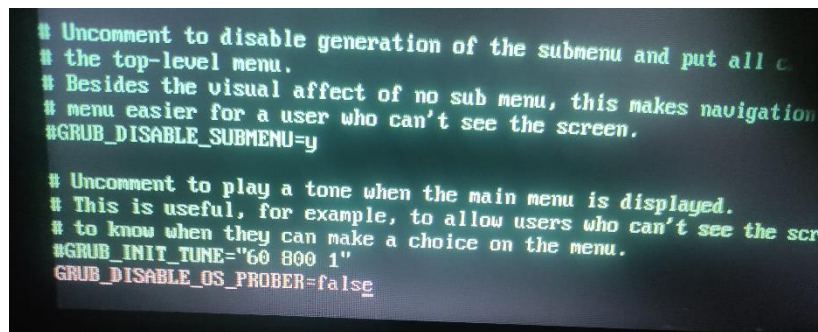
Add this line at the end:

`GRUB_DISABLE_OS_PROBER=false`

Save the file and update GRUB again using:

`grub-mkconfig -o /boot/grub/grub.cfg`

restart



## 2. Kernel Panic on First Boot

### Why does this happen?

Kernel panic occurs when the Linux kernel **fails to start** properly. This can happen for several reasons:

1. **Kernel not compiled correctly** – If the kernel is missing essential drivers (like filesystem drivers), the system won't boot.
2. **Wrong root partition in fstab** – If the system cannot find the correct root partition, it won't know where to load the OS.
3. **Missing framebuffer support** – If the display drivers are missing, the system may fail to boot properly, showing a black screen or panic error.

### How to Fix It?

#### 1. Check Kernel Compilation

If your kernel is missing important features, you need to **recompile it** with the correct settings:

**make menuconfig** # Configure kernel settings

- Navigate to **Device Drivers** → **Graphics Support**
- Make sure these options are enabled:

[ \* ] Support for frame buffer devices (CONFIG\_FB)

[ \* ] EFI-based Framebuffer Support (CONFIG\_FB\_EFI)

[ \* ] Simple framebuffer support (CONFIG\_FB\_SIMPLE)

Then, recompile the kernel and reinstall it:

**make -j\$(nproc)** # Compile kernel using all CPU cores

**make modules\_install**

**make install**

### 3. Manual Copying of Kernel Files (Due to LILO Not Installed)

#### What is this issue?

When you compile the Linux kernel in Gentoo, it creates a bootable kernel file called bzImage inside the directory:

**/usr/src/linux/arch/x86/boot/bzImage**

However, if the LILO bootloader is not installed, the kernel is not automatically copied to the /boot directory, meaning the system won't boot properly.

#### What is LILO?

LILO (Linux Loader) is an old bootloader used in some Linux systems, but most modern systems use GRUB instead.

- LILO does not automatically detect new kernels when they are installed or updated.
- Without LILO, you have to manually copy the kernel file to /boot so that GRUB can find and use it.

#### How to Fix It?

Since LILO is missing, you need to manually copy the bzImage file to the /boot directory so that GRUB can load it at boot time.

##### 1. Navigate to the kernel directory:

**cd /usr/src/linux**

##### 2. Copy the compiled kernel to the boot partition:

**cp arch/x86/boot/bzImage /boot/vmlinuz-6.12.6-gentoo**

**cp System.map /boot/System.map-6.12.6-gentoo**

**cp .config /boot/config-6.12.6-gentoo**

- This moves the **bzImage** file to /boot and renames it as vmlinuz, which is a common name for Linux kernels.
- GRUB expects the kernel file to be inside /boot, so this step ensures GRUB can load it properly.

**3. Update GRUB to detect the new kernel:**

**grub-mkconfig -o /boot/grub/grub.cfg**

This command tells GRUB to scan for available kernels and update its boot menu.

**reboot**

## **Kernel Compilation Not Using All CPU Cores:**

### **Why Does This Happen?**

By default, the **make** command may only use **one CPU core**, making compilation very slow. In some cases, you may encounter errors like **make[2,3]**, which occurs when the CPU usage is maxed out due to insufficient resource allocation for compilation. Additionally, if the **-j** value is set too low (for example, **-j1**), the compilation will take a long time to execute, as it is only using one core.

### **How to Fix It?**

Use multi-core compilation with:

**make -j\$(nproc)**

This automatically detects and uses all available CPU cores.

### **How to Manually Set -j Value?**

- **Recommended formula:**

CPU cores × 1.5

- Example:

**4-core CPU** → Use **-j6**

**8-core CPU** → Use **-j12**

This ensures faster compilation without overloading the system and helps prevent errors like **make[2,3]**. If you encounter issues, you can adjust the **-j** value to better match your system's resources.