

Gentoo Linux Installation Report

1. Introduction

I'm writing about how I installed Gentoo Linux step by step using the LiveGUI ISO. I used OpenRC, the ext4 filesystem, and genkernel to compile the kernel. Doing this manually helped me understand Linux deeply, including partitioning, package management, kernel setup, and bootloader configuration.

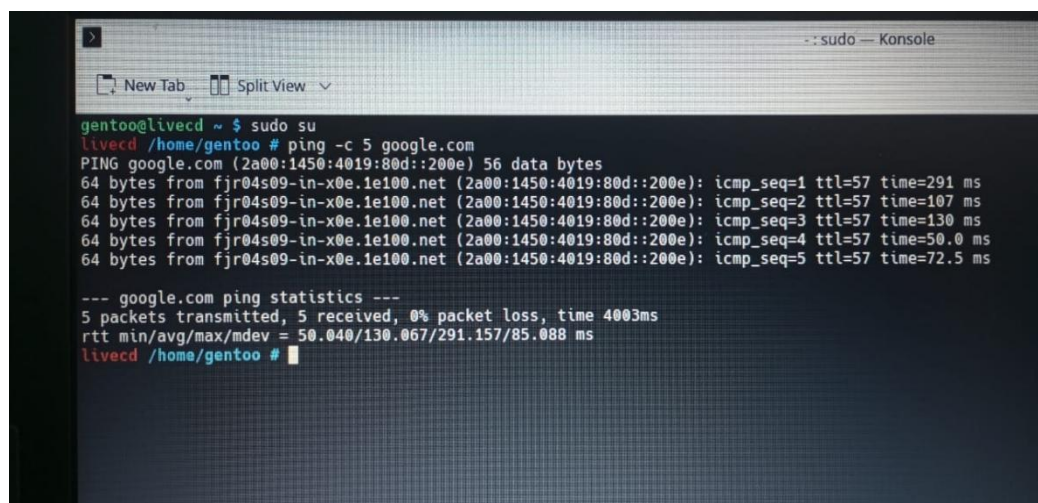
2. System Specifications

- **Laptop Model:** *Lenovo Ideapad 130-15IKB*
- **Processor:** *(Intel Core i3-7020U (2 cores, 4 threads) @ 2.30 GHz)*
- **RAM:** *12 GB*
- **Storage:** *232 GB*

3. Installation Steps

Step 1: Boot the LiveGUI Environment

- Started with making the USB bootable through rufus and then booted into Gentoo LiveGUI environment through it.
- Verified the internet connection using “**ping google.com**”.

A screenshot of a terminal window titled "sudo - Konsole". The terminal shows a user at the "gentoo@livecd ~" prompt running "sudo su" to become root. Then, the user runs "ping -c 5 google.com". The output shows five successful ping requests to google.com (2a00:1450:4019:80d::200e) with varying response times. Finally, the user runs "ping -t google.com" to get statistics, showing 5 packets transmitted, 5 received, 0% packet loss, and a total time of 4003ms.

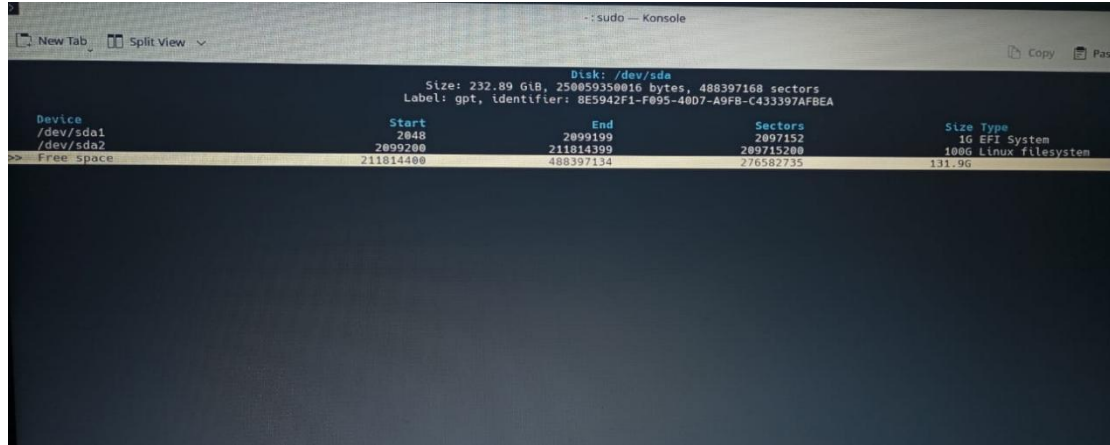
```
gentoo@livecd ~ $ sudo su
livecd /home/gentoo # ping -c 5 google.com
PING google.com (2a00:1450:4019:80d::200e) 56 data bytes
64 bytes from fjr04s09-in-x0e.1e100.net (2a00:1450:4019:80d::200e): icmp_seq=1 ttl=57 time=291 ms
64 bytes from fjr04s09-in-x0e.1e100.net (2a00:1450:4019:80d::200e): icmp_seq=2 ttl=57 time=107 ms
64 bytes from fjr04s09-in-x0e.1e100.net (2a00:1450:4019:80d::200e): icmp_seq=3 ttl=57 time=130 ms
64 bytes from fjr04s09-in-x0e.1e100.net (2a00:1450:4019:80d::200e): icmp_seq=4 ttl=57 time=50.0 ms
64 bytes from fjr04s09-in-x0e.1e100.net (2a00:1450:4019:80d::200e): icmp_seq=5 ttl=57 time=72.5 ms

--- google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4003ms
rtt min/avg/max/mdev = 50.040/130.067/291.157/85.088 ms
livecd /home/gentoo #
```

Step 2: Disk Partitioning

Since I was installing Gentoo on bare metal, I did not create a swap partition. Instead, I configured only:

- EFI Partition (1GB, EFI System) → FAT32
- Root Partition (100G, Linux Filesystem) → ext4



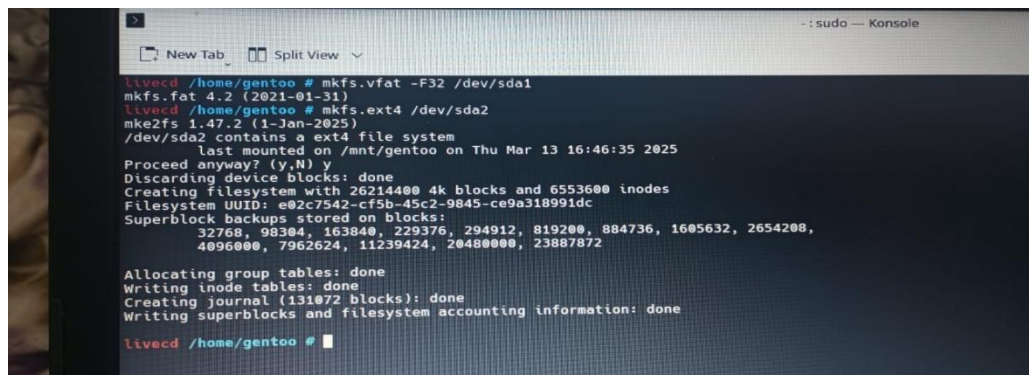
The screenshot shows a terminal window with the output of the 'fdisk -l' command for disk /dev/sda. The disk is 232.89 GiB (250059350016 bytes) and uses the gpt partitioning scheme. The output is as follows:

Device	Start	End	Sectors	Size	Type
/dev/sda1	2048	2099199	2097152	1G	EFI System
/dev/sda2	2099200	211814399	209715200	100G	Linux filesystem
Free space	211814400	488397134	276582735	131.9G	

Step 3: Formatting and Mounting Partitions

Formatted and mounted partitions through commands:

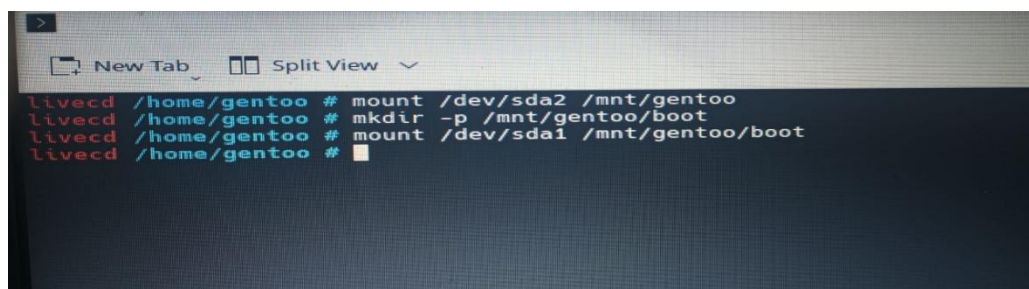
- **mkfs.vfat -F32 /dev/sda1 (EFI)**
- **mkfs.ext4 /dev/sda2 (Root)**
- **Mounted /dev/sda2 to /mnt/gentoo**
- **Created and mounted /boot under /mnt/gentoo**



```
livecd /home/gentoo # mkfs.vfat -F32 /dev/sda1
mkfs.fat 4.2 (2021-01-31)
livecd /home/gentoo # mkfs.ext4 /dev/sda2
mke2fs 1.47.2 (1-Jan-2025)
/dev/sda2 contains a ext4 file system
last mounted on /mnt/gentoo on Thu Mar 13 16:46:35 2025
Proceed anyway? (y,N) y
Discarding device blocks: done
Creating filesystem with 26214400 4k blocks and 6553600 inodes
Filesystem UUID: e02c7542-cf5b-45c2-9845-ce9a318991dc
Superblock backups stored on blocks:
32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
4096000, 7962624, 11239424, 20480000, 23887872

Allocating group tables: done
Writing inode tables: done
Creating journal (131072 blocks): done
Writing superblocks and filesystem accounting information: done

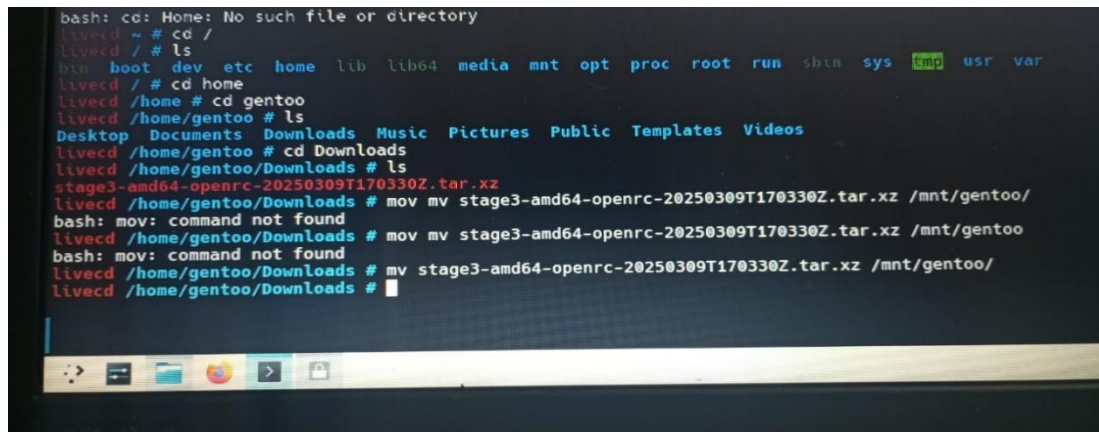
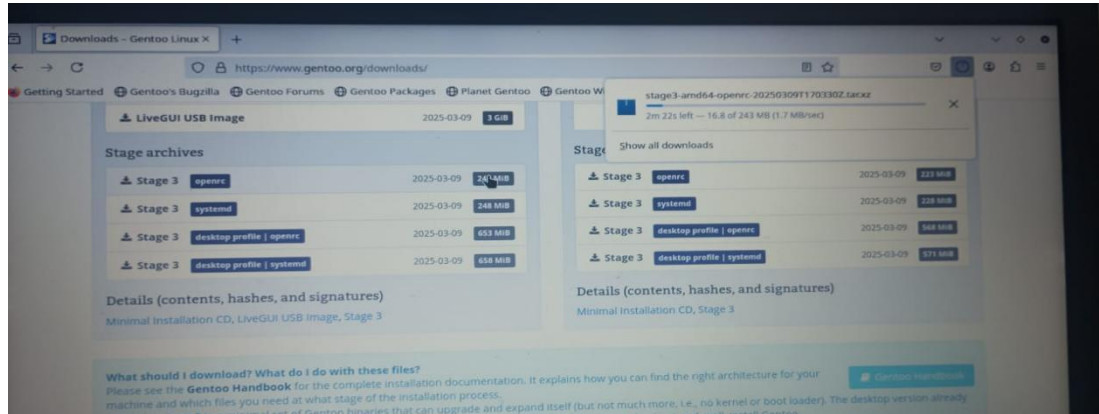
livecd /home/gentoo #
```



```
livecd /home/gentoo # mount /dev/sda2 /mnt/gentoo
livecd /home/gentoo # mkdir -p /mnt/gentoo/boot
livecd /home/gentoo # mount /dev/sda1 /mnt/gentoo/boot
livecd /home/gentoo #
```

Step 4: Downloading & Extracting Stage 3

I manually downloaded the OpenRC Stage 3 tarball from the official Gentoo website. And moved it to the required directory, and there extract it.



Step 5: Entering the Chroot Environment

Copied the system's DNS settings to ensure internet access inside the chroot.

Mounted essential virtual filesystems:

- **/proc** for process information
- **/sys** for system-related data
- **/dev** for device management

Changed root into the new Gentoo environment using the chroot command, for me to configure the system as if it were fully installed.


```
livecd /mnt/gentoo # cp --dereference /etc/resolv.conf /mnt/gentoo/etc/
livecd /mnt/gentoo # mount -t proc /proc /mnt/gentoo/proc
livecd /mnt/gentoo # mount --rbind /sys /mnt/gentoo/sys
livecd /mnt/gentoo # mount --make-rslave /mnt/gentoo/sys
livecd /mnt/gentoo # mount --rbind /dev /mnt/gentoo/dev
livecd /mnt/gentoo # mount --make-rslave /mnt/gentoo/dev
livecd /mnt/gentoo #
```

```
livecd /mnt/gentoo # chroot /mnt/gentoo /bin/bash
livecd / # source /etc/profile
livecd / # export PS1="(chroot) $PS1"
(chroot) livecd / #
```

Step 6: Syncing Portage

- Ran **emerge-webrsync** to sync the Portage tree.
- Updated the system using **emerge --sync && emerge -uDN @world**.

```
(chroot) livecd / # emerge-webrsync
* PGP verification method: gemato
* Fetching most recent snapshot ...
* Latest snapshot date: 20250312
*
* Approximate snapshot timestamp: 1741826700
* Current local timestamp: 1741894800
*
* The current local timestamp is newer than the timestamp
* of the latest snapshot. In order to force sync, use the
* --revert option or remove the timestamp file located at
* '/var/db/repos/gentoo/metadata/timestamp.x'.
(chroot) livecd / # emerge --sync && emerge -uDN @world
>>> Syncing repository 'gentoo' into '/var/db/repos/gentoo'...
* Using keys from /usr/share/openpgp-keys/gentoo-release.asc
* Refreshing keys via WKD ...
>>> Starting rsync with rsync://[2a00:1828:a00d:ffff::6]/gentoo-portage...
>>> Checking server timestamp ...
Welcome to turnstone.gentoo.org / rsync.gentoo.org

Server Address : 89.238.71.6, 2a00:1828:a00d:ffff::6
Contact Name   : mirror-admin@gentoo.org
Hardware       : 16 x Intel(R) Xeon(R) CPU           E5530 , 24152MiB RAM
Sponsor        : Manitu GmbH, St. Wendel, Germany
```

Step 7: Configuring Timezone & Locale

- Set system timezone of karachi thorough
echo "UTC" > /etc/timezone
emerge --config sys-libs/timezone-data.

- Configured locale by uncommenting **en_US.UTF-8 UTF-8** in **/etc/locale.gen**
- And then by running
locale-gen
eselect locale set en_US.utf8

```

/etc/locale.gen: list all of the locales you want to have on your system.
# See the locale.gen(5) man page for more details.
#
# The format of each line:
# <locale name> <charset>
#
# Where <locale name> starts with a name as found in /usr/share/i18n/locales/.
# It must be unique in the file as it is used as the key to locale variables.
# For non-default encodings, the <charset> is typically appended.
#
# Where <charset> is a charset located in /usr/share/i18n/charmaps/ (sans any
# suffix like ".gz").
#
# All blank lines and lines starting with # are ignored.
#
# For the default list of supported combinations, see the file:
# /usr/share/i18n/SUPPORTED
#
# Whenever glibc is emerged, the locales listed here will be automatically
# rebuilt for you. After updating this file, you can simply run 'locale-gen'
# yourself instead of re-emerging glibc.

#en_US ISO-8859-1
en_US.UTF-8 UTF-8
#ja_JP.EUC-JP EUC-JP
#ja_JP.UTF-8 UTF-8
#ja_JP EUC-JP
#en_HK ISO-8859-1
#en_PH ISO-8859-1
#de_DE ISO-8859-1
#de_DE@euro ISO-8859-15

```

Step 8: Installing and Compiling Kernel (Genkernel)

I installed the kernel sources and compiled the kernel using Genkernel with the following steps:

1. I Installed a specific kernel version:
emerge -v =sys-kernel/gentoo-sources-6.12.16
2. Accepted the necessary firmware license:
echo "sys-kernel/linux-firmware linux-fw- redistributable" | sudo tee -a /etc/portage/package.license
3. Installed Genkernel:
emerge sys-kernel/genkernel
4. Created a symbolic link for the kernel sources:
ln -sf /usr/src/linux-6.12.16-gentoo /usr/src/linux
5. Compiled the kernel using Genkernel:
genkernel all

```

--: sudo -- Konsole

New Tab Split View

(chroot) lived / # emerge -v =sys-kernel/gentoo-sources-6.12.16

* IMPORTANT: 15 news items need reading for repository 'gentoo'.
* Use eselect news read to view new items.

These are the packages that would be merged, in order:

Calculating dependencies... done!
Dependency resolution took 0.99 s (backtrack: 0/20).

[ebuild N    ] dev-libs/elfutils-0.191-r2::gentoo USE="bzip2 nls utils -debuginfod -lzma -static
,092 KiB
[ebuild N    ] virtual/libelf-3-r1:0/1::gentoo ABI_X86="(64) -32 (-x32)" 0 KiB
[ebuild N    ] app-arch/cpio-2.15::gentoo USE="nls" 1,613 KiB
[ebuild N    ] app-alternatives/cpio-0::gentoo USE="gnu -libarchive (-split-usr)" 0 KiB
[ebuild N    ] sys-kernel/gentoo-sources-6.12.16:6.12.16::gentoo USE="-build -experimental -sym

Total: 5 packages (5 new), Size of downloads: 156,324 KiB

```

```

- sys-kernel/linux-firmware-20250311::gentoo (masked by: || ( ) linux-fw-redistributable license(s)) /amd64 keyword
- sys-kernel/linux-firmware-20250211::gentoo (masked by: || ( ) linux-fw-redistributable license(s))
- sys-kernel/linux-firmware-20250109-r1::gentoo (masked by: || ( ) linux-fw-redistributable license(s))
- sys-kernel/linux-firmware-20241210-r1::gentoo (masked by: || ( ) linux-fw-redistributable license(s))

(dependency required by "sys-kernel/genkernel-4.3.16-r2::gentoo[firmware]" [ebuild])
(dependency required by "sys-kernel/genkernel" [argument])
For more information, see the MASKED PACKAGES section in the emerge
man page or refer to the Gentoo Handbook.

(chroot) lived / # ^C
(chroot) lived / # echo "sys-kernel/linux-firmware linux-fw-redistributable" | sudo tee -a /etc/portage/package.license
sys-kernel/linux-firmware linux-fw-redistributable
(chroot) lived / #

```

```

... new is up-to-date.

* IMPORTANT: 19 news items need reading for repository 'gentoo'.
* Use eselect news read to view new items.

(chroot) lived / # ln -sf /usr/src/linux-6.12.16-gentoo /usr/src/linux
(chroot) lived / # genkernel all
* Gentoo Linux Genkernel: Version 4.3.16
* Using genkernel configuration from '/etc/genkernel.conf' ...
* Running with options: all

* Working with Linux kernel 6.12.16-gentoo for x86_64
* Using kernel config file '/usr/share/genkernel/arch/x86_64/generated-config' ...
* Note: The version above is subject to change (depends on config and status of kernel sources).

* kernel: >> Initializing ...
* >> Running 'make mrproper' ...
* >> Running 'make oldconfig' ...
* >> Re-running 'make oldconfig' due to changed kernel options ...
* >> Kernel version has changed (probably due to config change) since genkernel start:
* We are now building Linux kernel 6.12.16-gentoo-x86_64 for x86_64 ...
* >> Compiling 6.12.16-gentoo-x86_64 bzImage ...
* >> Compiling 6.12.16-gentoo-x86_64 modules ...

```

```

(chroot) lived / # ln -sf /usr/src/linux-6.12.16-gentoo /usr/src/linux
(chroot) lived / # genkernel all
* Gentoo Linux Genkernel: Version 4.3.16
* Using genkernel configuration from '/etc/genkernel.conf' ...
* Running with options: all

* Working with Linux kernel 6.12.16-gentoo for x86_64
* Using kernel config file '/usr/share/genkernel/arch/x86_64/generated-config' ...
* Note: The version above is subject to change (depends on config and status of kernel sources).

* kernel: >> Initializing ...
* >> Running 'make mrproper' ...
* >> Running 'make oldconfig' ...
* >> Re-running 'make oldconfig' due to changed kernel options ...
* >> Kernel version has changed (probably due to config change) since genkernel start:
* We are now building Linux kernel 6.12.16-gentoo-x86_64 for x86_64 ...
* >> Compiling 6.12.16-gentoo-x86_64 bzImage ...
* >> Compiling 6.12.16-gentoo-x86_64 modules ...
* >> Installing 6.12.16-gentoo-x86_64 modules (and stripping) ...
* >> Generating module dependency data ...
* >> Compiling out-of-tree module(s) ...
* >> Saving config of successful build to '/etc/kernels/kernel-config-6.12.16-gentoo-x86_64'

* initramfs: >> Initializing ...
* >> Appending devices cpio data ...
* >> Appending base layout cpio data ...
* >> Appending util-linux cpio data ...
* >> Appending eudev cpio data ...
* >> Appending auxiliary cpio data ...
* >> Appending busybox cpio data ...

```


Step 9: Configuring fstab

Edited /etc/fstab to include only EFI and root partitions through **nano /etc/fstab**.

```
3387E-6DAF# /etc/fstab: static file system information.
#
# See the manpage fstab(5) for more information.
#
# NOTE: The root filesystem should have a pass number of either 0 or 1.
#       All other filesystems should have a pass number of 0 or greater than 1.
#
# NOTE: Even though we list ext4 as the type here, it will work with ext2/ext3
#       filesystems. This just tells the kernel to use the ext4 driver.
#
# NOTE: You can use full paths to devices like /dev/sda3, but it is often
#       more reliable to use filesystem labels or UUIDs. See your filesystem
#       documentation for details on setting a label. To obtain the UUID, use
#       the blkid(8) command.
#
# <fs>          <mountpoint>  <type>          <opts>          <dump> <pass>
UUID=387E-6DAF  /boot              vfat            defaults        0 2
UUID=e02c7542-cf5b-45c2-9845-ce9a318991dc /                 ext4            defaults        0 1
#LABEL=swap     none              swap            sw              0 0
/dev/cdrom       /mnt/cdrom        auto            noauto,ro       0 0
```

Step 10: Setting Up Network Configuration

- Set hostname echo "gentoo" > /etc/hostname.
- Installed and enabled dhcpcd for networking.

```
(chroot) livecd / # mount -a
(chroot) livecd / # nano /etc/fstab
(chroot) livecd / # echo "gentoo" > /etc/hostname
(chroot) livecd / # emerge --ask net-misc/dhcpcd

* IMPORTANT: 19 news items need reading for repository 'gentoo'
* Use eselect news read to view new items.

These are the packages that would be merged, in order:

Calculating dependencies... done!
Dependency resolution took 1.04 s (backtrack: 0/20).

[ebuild R ] net-misc/dhcpcd-10.1.0-r1
```

Step 11: Installing and Configuring GRUB

- Installed sys-boot/grub.
- Installed GRUB to the disk:
grub-install --target=x86_64-efi --efi-directory=/boot
grub-mkconfig -o /boot/grub/grub.cfg

```
* service drpcd added to runlevel default
(chroot) livecd / # emerge sys-boot/grub

* IMPORTANT: 19 news items need reading for repository 'gentoo'.
* Use eselect news read to view new items.

Calculating dependencies... done!
Dependency resolution took 1.31 s (backtrack: 0/20).

>>> Verifying ebuild manifests

>>> Emerging (1 of 1) sys-boot/grub-2.12-r5::gentoo
* grub-2.12.tar.xz BLAKE2B SHA512 size ;-) ...
* grub-2.12-bash-completion.patch.gz BLAKE2B SHA512 size ;-) ...
* unifont-15.0.06.pcf.gz BLAKE2B SHA512 size ;-) ...
* dejavu-sans-ttf-2.37.zip BLAKE2B SHA512 size ;-) ...
>>> Unpacking source...
>>> Unpacking grub-2.12.tar.xz to /var/tmp/portage/sys-boot/grub-2.12-r5/work
>>> Unpacking grub-2.12-bash-completion.patch.gz to /var/tmp/portage/sys-boot/grub-2.12-r5/work
>>> Unpacking unifont-15.0.06.pcf.gz to /var/tmp/portage/sys-boot/grub-2.12-r5/work
>>> Unpacking dejavu-sans-ttf-2.37.zip to /var/tmp/portage/sys-boot/grub-2.12-r5/work

(chroot) livecd / # grub-install --target=x86_64-efi --efi-directory=/boot
Installing for x86_64-efi platform.
Installation finished. No error reported.
(chroot) livecd / # 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
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
(chroot) livecd / #
```

Step 12: Creating User and Finalizing Installation

Set root password (passwd).

```
done
(chroot) livecd / # passwd

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 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 your password: "Lose4Brine$hoard".
```


Created a new user with sudo privileges.

```
Enter new password:
Weak password: not enough different characters or classes for this length.
Re-type new password:
passwd: password updated successfully
(chroot) livecd / # useradd -m -G users,wheel -s /bin/bash rafy23p0560
Creating mailbox file: No such file or directory
(chroot) livecd / # mkdir -p /var/mail
(chroot) livecd / # chmod 1777 /var/mail
(chroot) livecd / # useradd -m -G users,wheel -s /bin/bash rafy23p0560
useradd: user 'rafy23p0560' already exists
(chroot) livecd / # echo "rafy23p0560:khanzada1" | chpasswd

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 at least 3 of these 4 classes.

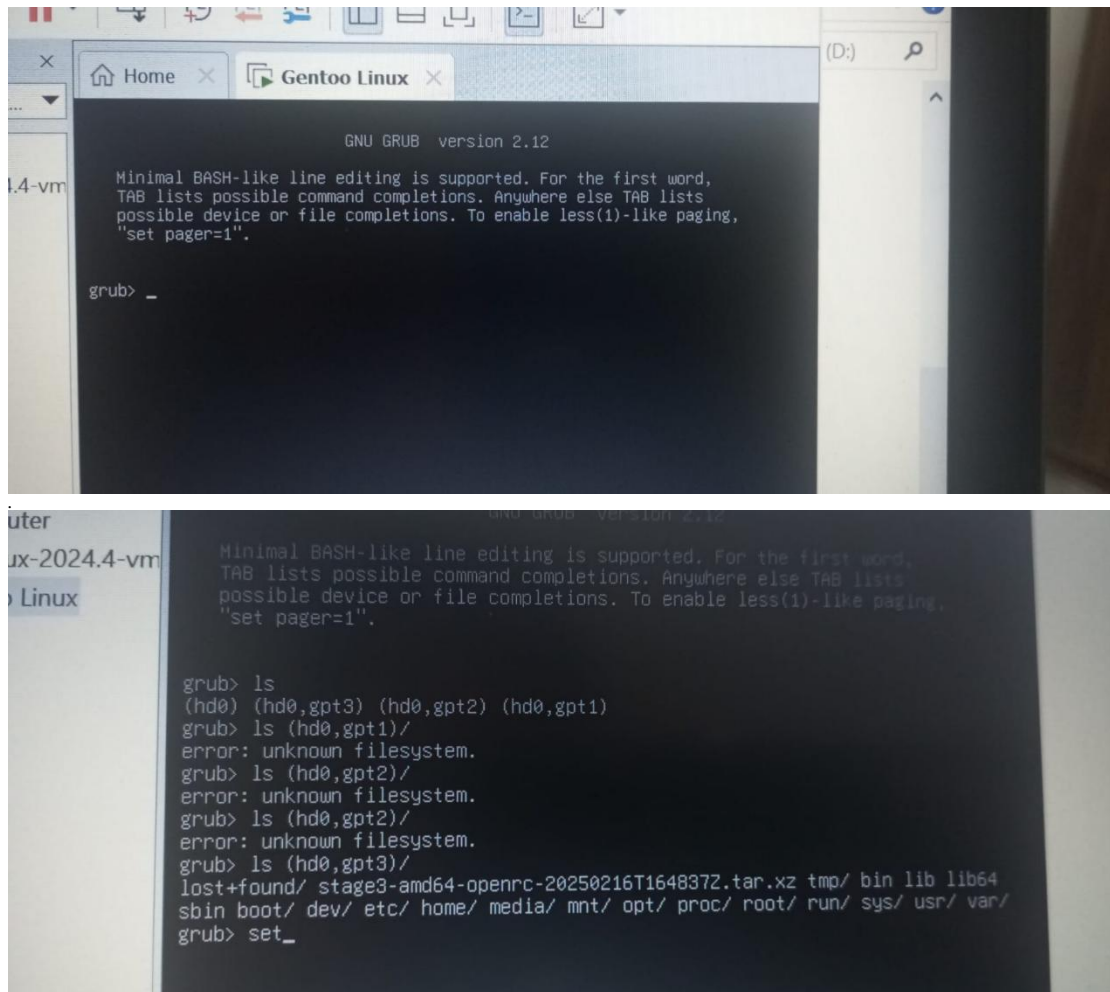
A valid passphrase should be of at least 3 words, 11 to 72 characters long, and
enough different characters.

Alternatively, if no one else can see your terminal now, you can pick this as
your password: "Help=girl9sweaty".

livecd / # ^C
livecd / # echo "rafy23p0560:alizayyas" | chpasswd --crypt-method=SHA512
livecd / #
```

Exited chroot, unmounted partitions

```
(chroot) livecd / # ^C
(chroot) livecd / # echo "rafy23p0560:alizayyas" | chpasswd --crypt-method=
(chroot) livecd / # exit
exit
livecd /mnt/gentoo # umount -l /mnt/gentoo/dev{/shm,/pts,}
livecd /mnt/gentoo # umount -R /mnt/gentoo
umount: /mnt/gentoo: target is busy.
livecd /mnt/gentoo # reboot
```

Its solution: Instead of configuring kernel manually I compile it through genkernel and ensured that GRUB was configured properly.

2. WiFi Not Connecting on Bare Metal Installation

- When I finally installed Gentoo on a friend's laptop (bare metal installation), the only real issue I faced was that WiFi was not connecting.
- The system couldn't detect or connect to available networks.

Its solution: I manually connected to WiFi using the command line with the help of ChatGPT. Once configured, everything worked fine.

5. What I Learned from This Installation

Installing Gentoo from scratch taught me a lot about Linux and system setup.

How Linux Boots – I now understand the full process, from disk partitioning to configuring GRUB and compiling the kernel.

Troubleshooting Skills – I faced issues like kernel panic and WiFi not working, but solving them helped me get better at debugging.

Working Without a GUI – Since I did everything through the terminal, I learned how powerful CLI commands can be.

Patience & Persistence – The process wasn't easy, but completing it gave me a huge sense of achievement.

Gentoo Linux: Advantages and Challenges

Advantages

Highly Customizable – I can control everything, from the kernel to installed packages.

Better Performance – Since software is compiled from source, it runs faster and is optimized for my hardware.

Great Learning Experience – Installing and using Gentoo teaches a lot about Linux internals.

Always Up to Date – Uses a rolling release model, so I will always get the latest software.

Lightweight – No unnecessary software; I will install only what I need.

Challenges

Difficult Installation – Unlike Ubuntu, Gentoo requires manual setup, which can take hours or days.

Slow Software Installation – Since everything is compiled from source, installing software takes longer.

Hard for Beginners – Managing the kernel and system updates can be very tricky.

Not for Casual Users – Needs regular maintenance, making it less suitable for everyday use.

6. Conclusion

Installing gentoo was challenging but rewarding as well. It taught me deep system-level understanding and Now, I have a fully functional, optimized Gentoo system configured exactly to my needs.