

# Getting Slackware

The official way to do this is, of course, to get it from [slackware.com](https://slackware.com).

Personally, I have [getslack](#), a bash script based on (more accurately, a trimmed-down version of) the excellent (he termed it *infamous*) [mirror-slackware-current.sh](#) by [Alien Bob](#). When going down this path, the next step would be to prepare the installation media.

## Slackware Installer ISO Image

I no longer need an ISO image (refer to USB installer below). But, I have my [slack2iso](#) script (also based on Alien Bob's script) that can help in creating one using the tree downloaded by [getslack](#).

## Slackware USB Installer

[Alien Bob](#) has provided a [script](#) to make/setup/configure a USB-based Slackware installation media. I wanted to do something simpler using the existing files in the Slackware tree that I mirrored using [getslack](#) (mentioned above). So, here is how I got that working.

1. Create a FAT32 partition
  - use `fdisk` and make sure it is bootable (bootable flag enabled)
  - use `mkdosfs` (e.g. `mkdosfs -F 32 /dev/sdb1`)
2. Use `syslinux` to provide bootloader
  - create a `/linux/boot/syslinux` folder on the USB
  - type

```
syslinux -d /linux/boot/syslinux /dev/sdb1
```

**Note:** On newer `syslinux`, use `-i` to indicate new installation

- a file `ldlinux.sys` should appear in `/linux/boot/syslinux`
3. Copy boot facilities from Slackware tree to the media
    - copy a kernel from slackware tree to `/linux/boot` (I used `huge.s`)
    - copy `initrd.img` and `message.txt` to `/linux/boot`
    - copy `isolinux.cfg` to `/linux/boot/syslinux` as `syslinux.cfg`
    - edit `syslinux.cfg` accordingly (initrd, kernel params, etc.)
  4. Copy `slackware<64>` in the Slackware tree (I used a shorter folder name like `slack` on the USB)

And... we're done! Now we have a simple Slackware USB Installer and install it on every computer we can get our hands on! 😎

[Note: GPT Disks and EFI](#)

## Note: GPT Disks and EFI

Things moving to (U)EFI and GPT... slowly leaving legacy BIOS and MBR.

Instead of MBR, we use GPT partitioning scheme:

- supports bigger disk
- supports EFI booting (easier to maintain actually :p)

Partition codes are 2-bytes instead (only 1-byte on MBR's partition table). Among the common ones:

- EF00 (EFI System Partition): this is what EFI boot look for
  - format FAT32

```
mkdosfs -F 32 -n MY1EFI /dev/sdxx
```

- 0700 (MS Basic Data): Windows Partition
  - format NTFS

```
mkntfs -f -L MY1WIN /dev/sdxx
```

- 8300 (Linux filesystem): Linux Partition
  - format EXT4

```
mkfs.ext4 -L MY1LIN /dev/sdxx
```

Once boot using EFI, efibootmgr tool can be used (available on Slackware 14.2)

- to create an entry labelled Slackware with loader file named `\efi\slackware\elilo.efi` located on first partition of first disk (`/dev/sda1`)

```
efibootmgr -c -d /dev/sda -p 1 -L "Slackware" -l  
"\efi\slackware\elilo.efi"
```

- to delete an entry xxxx (bootnum)

```
efibootmgr -b xxxx -B
```

- to re-order boot sequence

```
efibootmgr -o xxxx,yyyy,zzzz
```

From:

<http://azman.unimap.edu.my/dokuwiki/> - **Azman @UniMAP**

Permanent link:

[http://azman.unimap.edu.my/dokuwiki/doku.php?id=linux:slack\\_0get&rev=1736637719](http://azman.unimap.edu.my/dokuwiki/doku.php?id=linux:slack_0get&rev=1736637719)

Last update: **2025/01/12 07:21**

