

Arch Linux x64 Installation

Base install

First we need to download Arch Linux. The download can be made through a Torrent. Downloads are available at <https://www.archlinux.org/download/>. Once we get the ISO, we can burn it on a CD, or create a bootable USB memory using dd on Linux, or some software like Rufus on Windows.

Once the installation medium is ready, it's time to plug it into the computer. Insert the CD-ROM into the CD-ROM unit or plug the USB memory into the USB drive, and boot from it. You should be greeted with this screen.



Choose "Boot Arch Linux (x86_64)" to boot from the CD-ROM. After a short while, you should be in the prompt. The prompt looks like this

```
Arch Linux 4.18.16-arch1-1-ARCH (tty1)
archiso login: root (automatic login)
root@archiso ~ # _
```

First things first. The default keyboard layout is United States (en_us). If your keyboard is not from the states, you probably want to change it. First, list all the available layouts using “ls”. Pipe it to “more” so you can view every entry, or pipe it to grep if you are looking for an specific layout (like es):

```
# ls /usr/share/kbd/keymaps/**/*.map.gz | more
```

Once you have located your keyboard layout, load it. For instance, my keyboard is Spanish, so it uses the Spanish (es) layout:

```
# loadkeys es
```

If you are connected to the internet using an Ethernet connection, you should be connected to the internet automatically. But if you are connected using a WiFi connection, further actions might be required in order for it to work properly. Check if you are connected to the internet using ping:

```
# ping -c3 www.google.com
```

You should see something like this:

```

root@archiso ~ # ping -c3 www.google.com
PING www.google.com (172.217.17.4) 56(84) bytes of data.
64 bytes from mad07s09-in-f4.1e100.net (172.217.17.4): icmp_seq=1 ttl=53 time=4.35 ms
64 bytes from mad07s09-in-f4.1e100.net (172.217.17.4): icmp_seq=2 ttl=53 time=4.46 ms
64 bytes from mad07s09-in-f4.1e100.net (172.217.17.4): icmp_seq=3 ttl=53 time=4.18 ms

--- www.google.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 6ms
rtt min/avg/max/mdev = 4.184/4.332/4.458/0.112 ms
root@archiso ~ # _

```

In the event that this is not the case, check if your network device shows up and that it has an IP address assigned. You can do this by using the ip command

```
# ip addr
```

This shows a list of network devices and their addresses, pretty much like ipconfig on Windows.

```

root@archiso ~ # ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:be:95:99 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global noprefixroute enp0s3
        valid_lft forever preferred_lft forever
    inet6 fe80::b9cc:1d80:73cd:32a5/64 scope link
        valid_lft forever preferred_lft forever
root@archiso ~ #

```

There is a 'lo' device, which is the loopback, and the rest are the network devices (named like enpXsN, where X is the number of the interface and N is another number). If only the loopback is showing, then your network interface is not being recognized. Their IP address is the inet address (inet6 is their IPv6 counterpart). If no address is given then it's probably not picking up the configuration.

Manually setting up your network (if no DHCP is available)

If your network doesn't have any DHCP server available (for instance, it's disabled), you might need to manually set the addresses.

```
# ip address add address/prefix_len broadcast + dev interface
```

In this command, *address* refers to the address you want to give to this interface (example: 192.168.1.141). Make sure it's a valid address inside of your network. *prefix_len* refers to the length of the address, usually 24 for type C (like in the previous example). *broadcast* is the network broadcast address, usually your network address but ending in 255, but if you are segmenting your network, this might be something else. And then *interface* is the name of the interface (for instance, enp0s3).

Example:

```
# ip address add 192.168.1.141/24 192.168.1.255 + dev enp0s3
```

We also need to manually set the DNS servers we are going to use. This should be done automatically, but you can edit `/etc/resolv.conf` for this:

```
# nano resolv.conf
```

Now set the addresses of your DNS servers like this:

```
GNU nano 3.1 /etc/resolv.conf
# Generated by resolvconf
nameserver 1.1.1.1
nameserver 1.0.0.1
```

Press F3 and Enter to save, and then Ctrl+X to exit.

Do another ip address to check if everything is set up correctly. You might need to enable the interface first, to do this, use this command:

```
# ip link set interface up
```

Where `interface` is the name of your interface (for instance `enp0s3`) And then check with `ping` if the connection good.

```
# ping -c3 www.google.com
```

Setting up the DHCP

By default in home networks, every router has a DHCP server that is enabled by default. In the event that you did not alter this behavior but you still don't get any addresses, you might need to enable the DHCP first in Arch.

First, list your network interfaces with `ip address`

```
root@archiso ~ # ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
   inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
   inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
   link/ether 08:00:27:be:95:99 brd ff:ff:ff:ff:ff:ff
   inet 10.0.2.15/24 brd 10.0.2.255 scope global noprefixroute enp0s3
       valid_lft forever preferred_lft forever
   inet6 fe80::b9cc:1d80:73cd:32a5/64 scope link
       valid_lft forever preferred_lft forever
root@archiso ~ #
```

Identify your network interface name (something like `enpXsN`, where X is the number of the interface and N is another number).

Arch Linux has the dhcpd daemon by default. Check first if it is enabled and running:

```
# systemctl status dhcpd@device
```

Where *device* is the device name.

```
3 root@archiso ~ # systemctl status dhcpd@enp0s3
■ dhcpd@enp0s3.service - dhcpd on enp0s3
   Loaded: loaded (/usr/lib/systemd/system/dhcpd@.service; disabled; vendor preset: disabled)
   Active: inactive (dead) since Fri 2018-11-30 12:23:10 UTC; 25s ago
     Process: 782 ExecStop=/usr/bin/dhcpd -x enp0s3 (code=exited, status=0/SUCCESS)
     Process: 290 ExecStart=/usr/bin/dhcpd -q -w enp0s3 (code=exited, status=0/SUCCESS)
    Main PID: 488 (code=exited, status=0/SUCCESS)

Nov 30 11:29:42 archiso dhcpd[488]: enp0s3: no IPv6 Routers available
Nov 30 12:23:10 archiso systemd[1]: Stopping dhcpd on enp0s3...
Nov 30 12:23:10 archiso dhcpd[782]: sending signal TERM to pid 488
Nov 30 12:23:10 archiso dhcpd[782]: sending signal TERM to pid 488
Nov 30 12:23:10 archiso dhcpd[488]: received SIGTERM, stopping
Nov 30 12:23:10 archiso dhcpd[488]: enp0s3: removing interface
Nov 30 12:23:10 archiso dhcpd[782]: waiting for pid 488 to exit
Nov 30 12:23:10 archiso dhcpd[782]: waiting for pid 488 to exit
Nov 30 12:23:10 archiso dhcpd[488]: dhcpd exited
Nov 30 12:23:10 archiso systemd[1]: Stopped dhcpd on enp0s3.
3 root@archiso ~ # _
```

In this case, it's not running. Sometimes it might be binding to the old interface name convention, like 'eth0'. So make sure dhcpd is not binding to eth0

```
# systemctl status dhcpd@eth0
```

It should be inactive and dead

```
Nov 30 12:23:10 archiso systemd[1]: Stopped dhcpd on enp0s3.
3 root@archiso ~ # systemctl status dhcpd@eth0
■ dhcpd@eth0.service - dhcpd on eth0
   Loaded: loaded (/usr/lib/systemd/system/dhcpd@.service; disabled; vendor preset: disabled)
   Active: inactive (dead)
3 root@archiso ~ # _
```

If this is not the case, disable and stop it

```
# systemctl disable dhcpd@eth0
```

```
# systemctl stop dhcpd@eth0
```

And then start it on your device

```
# systemctl start dhcpd@device
```

And then check it's status

```
# systemctl status dhcpd@device
```

```
3 root@archiso ~ # systemctl start dhcpcd@enp0s3
root@archiso ~ # systemctl status dhcpcd@enp0s3
■ dhcpcd@enp0s3.service - dhcpcd on enp0s3
   Loaded: loaded (/usr/lib/systemd/system/dhcpcd@.service; disabled; vendor preset: disabled)
   Active: active (running) since Fri 2018-11-30 12:26:38 UTC; 7s ago
     Process: 782 ExecStop=/usr/bin/dhcpcd -x enp0s3 (code=exited, status=0/SUCCESS)
     Process: 806 ExecStart=/usr/bin/dhcpcd -q -w enp0s3 (code=exited, status=0/SUCCESS)
    Main PID: 820 (dhcpcd)
       Tasks: 1 (limit: 1738)
      Memory: 648.0K
     CGroup: /system.slice/system-dhcpcd.slice/dhcpcd@enp0s3.service
            └─820 /usr/bin/dhcpcd -q -w enp0s3

Nov 30 12:26:38 archiso systemd[1]: Starting dhcpcd on enp0s3...
Nov 30 12:26:38 archiso dhcpcd[806]: DUID 00:04:89:04:8d:77:56:fd:4f:cd:9c:ba:40:4a:5e:ec:e8:8d
Nov 30 12:26:38 archiso dhcpcd[806]: enp0s3: IAID 27:be:95:99
Nov 30 12:26:38 archiso dhcpcd[806]: enp0s3: rebinding lease of 10.0.2.15
Nov 30 12:26:38 archiso dhcpcd[806]: enp0s3: leased 10.0.2.15 for 86400 seconds
Nov 30 12:26:38 archiso dhcpcd[806]: enp0s3: adding route to 10.0.2.0/24
Nov 30 12:26:38 archiso dhcpcd[806]: enp0s3: adding default route via 10.0.2.2
Nov 30 12:26:38 archiso dhcpcd[806]: forked to background, child pid 820
Nov 30 12:26:38 archiso systemd[1]: Started dhcpcd on enp0s3.
Nov 30 12:26:39 archiso dhcpcd[820]: enp0s3: soliciting an IPv6 router
root@archiso ~ #
```

Restart your network interface, wait a few seconds, and check if you get an address

```
# ip link set interface link down
# ip link set interface link up
# ip address
```

```
root@archiso ~ # ip link set enp0s3 down
root@archiso ~ # ip link set enp0s3 up
root@archiso ~ # ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
   inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
   inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
   link/ether 08:00:27:be:95:99 brd ff:ff:ff:ff:ff:ff
   inet 10.0.2.15/24 brd 10.0.2.255 scope global noprefixroute enp0s3
       valid_lft forever preferred_lft forever
   inet6 fe80::b9cc:1d80:73cd:32a5/64 scope link
       valid_lft forever preferred_lft forever
root@archiso ~ #
```

Other instructions

For more info, refer to the Network Configuration section of the ArchLinux wiki:

https://wiki.archlinux.org/index.php/Network_configuration

System clock using NTP

Once the network is up and running, it's time to start installing the system.

Ensure the clock is accurate by using `timedatectl` to establish a connection to the NTP server:

```
# timedatectl set-ntp true
```

Disk partitioning

Now it's time for the disk partitioning. This is the most complicated part of the process, as everything else should be straight-forward.

Whether you are installing it alongside Windows or using the whole disk, the process is the same. But if you are using Windows or you have other data you want to protect, you first need to locate these first using lsblk:

```
# lsblk -f
```

```
root@archiso ~ # lsblk -f
NAME FSTYPE LABEL      UUID                                MOUNTPOINT
loop0 squashfs
sda
sr0   iso9660 ARCH_201811 2018-11-01-07-15-12-00          /run/archiso/bootmnt
root@archiso ~ #
```

Under normal circumstances, you will see each disk identified as 'sdx', where x is a letter to identify the disk or device (sda, sdb, sdc...). Partitions will appear below in a tree, identified as sdxN, where x is the letter of the device, and N the number of the partition (sda0, sda1, sda2...). A normal output would be like this:

```
NAME      FSTYPE LABEL      UUID                                MOUNTPOINT
sda
├─sda1    ntfs   WINRE_DRV  D4A45AAAA45A8EBC
├─sda2    vfat   SYSTEM_DRV 185C-DA5B
├─sda3    vfat   LRS_ESP     0E60-2E0E
├─sda4
├─sda5    ntfs   Windows8_OS 18D0632AD0630CF6
├─sda6    ntfs   LENOVO      9286FFD986FFBC33
├─sda7    ntfs   PBR_DRV     ECD06683D066543C
├─sda8
├─sda9    swap
├─sda10   ext4   bb29dda3-bdaa-4b39-86cf-4a6dc9634a1b /
sr0
```

The FSTYPE column references the format of the partition. Those related to Windows have the vfat/ntfs filesystem. You don't want to mess with neither your Windows installation, or with any of the reserved partitions used to boot Windows.

Normally you should have reserved some space when installing Windows. This space will be listed with no FSTYPE. This is the space you want to use.

In case you are using the whole disk, which is the recommended way, you don't need to worry about this. Otherwise, take note of which partitions you will be using to install it.

Now open the device of the disk you want to edit. For instance, in my case, it's sda:

```
# fdisk /dev/sda
```

```
1 root@archiso ~ # fdisk /dev/sda

Welcome to fdisk (util-linux 2.32.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0x0dff0fb4.

Command (m for help):
```

If you type F and press 'Enter', you can list all the free space in the disk.

```
Command (m for help): F

Unpartitioned space /dev/sda: 16 GiB, 17178820608 bytes, 33552384 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes

Start      End  Sectors  Size
 2048 33554431 33552384 16G

Command (m for help):
```

In this case I'm going to use all the free space in the disk, so I will create two partitions.

The first partition will have 13G and will be my root partition. The second partition will be 3G and will be my SWAP partition. Generally you want the SWAP to be double the capacity of your RAM. So if your RAM is 8G, you want your SWAP to be about 16G.

But first, if the disk is not partitioned, we need a partition table. If you are using MBR partitions (old BIOS), type 'o' and press ENTER. This will create a new MBR partition table. If you are using GPT partitions (UEFI), type 'g' instead. **DO NOT DO THIS IF YOU HAVE OTHER PARTITIONS**, this will overwrite the current partition table.

```
Command (m for help): o
Created a new DOS disklabel with disk identifier 0xf281c0f7.

Command (m for help): _
```

Now, to create the partitions.

Type 'n' and press ENTER to create a new MBR partition. Assuming you are using MBR partitions, fdisk will ask you if your partition is primary or extended. It will be a primary partition (p). Then it will ask the partition number. In this case it's the only one available, so it's 1. If you have windows installed, it might be 3.

Then it will ask the first sector, which is at the start of the free space. In the example above, we see the Start is at 2048, so that's where it starts. The end is after 13GB (the capacity of the partition). So I type +13GB.

This will create a 13GB partition.

```
Command (m for help): n
Partition type
  p   primary (0 primary, 0 extended, 4 free)
  e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-33554431, default 2048): 2048
Last sector, +sectors or +size{K,M,G,T,P} (2048-33554431, default 33554431): 13G
Value out of range.
Last sector, +sectors or +size{K,M,G,T,P} (2048-33554431, default 33554431): +13G

Created a new partition 1 of type 'Linux' and of size 13 GiB.

Command (m for help):
```

Now use 'p' to print the partition layout for this device.

```
Command (m for help): p
Disk /dev/sda: 16 GiB, 17179869184 bytes, 33554432 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x0dff0fb4

Device      Boot Start        End Sectors Size Id Type
/dev/sda1                2048 27265023 27262976 13G 83 Linux

Command (m for help): _
```

Don't worry if you screwed up, the changes haven't been written to the disk yet. You can simply quit and then try again by typing 'q' if you want to try again from scratch. You can also delete partitions by typing 'd'. This will delete the last partition you just created.

Now create the rest of the partitions. If the last one doesn't fit (out of range), try slightly lowering the size. This is due to data alignment and the headers in the fs. If you press ENTER when prompted for a value without any value, it will use the default instead.

```
Command (m for help): n
Partition type
  p   primary (1 primary, 0 extended, 3 free)
  e   extended (container for logical partitions)
Select (default p): e
Partition number (2-4, default 2): 2
First sector (27265024-33554431, default 27265024):
Last sector, +sectors or +size{K,M,G,T,P} (27265024-33554431, default 33554431): +3G
Value out of range.
Last sector, +sectors or +size{K,M,G,T,P} (27265024-33554431, default 33554431): +2.9G

Created a new partition 2 of type 'Extended' and of size 2.9 GiB.

Command (m for help): n
Partition type
  p   primary (1 primary, 1 extended, 2 free)
  l   logical (numbered from 5)
Select (default p): l

Adding logical partition 5
First sector (27267072-33302527, default 27267072):
Last sector, +sectors or +size{K,M,G,T,P} (27267072-33302527, default 33302527):

Created a new partition 5 of type 'Linux' and of size 2.9 GiB.

Command (m for help): _
```

The SWAP partition needs to be of type SWAP. You can change partition types using 't'.

```

Command (m for help): t
Partition number (1,2,5, default 5): 5
Hex code (type L to list all codes): L

 0 Empty                24 NEC DOS              81 Minix / old Lin   bf Solaris
 1 FAT12                27 Hidden NTFS Win    82 Linux swap / So   c1 DRDOS/sec (FAT-
 2 XENIX root           39 Plan 9              83 Linux              c4 DRDOS/sec (FAT-
 3 XENIX usr            3c PartitionMagic     84 OS/2 hidden or    c6 DRDOS/sec (FAT-
 4 FAT16 <32M          40 Venix 80286        85 Linux extended    c7 Syrix
 5 Extended            41 PPC PReP Boot     86 NTFS volume set   da Non-FS data
 6 FAT16               42 SFS                87 NTFS volume set   db CP/M / CTOS / .
 7 HPFS/NTFS/exFAT    4d QNX4.x              88 Linux plaintext   de Dell Utility
 8 AIX                 4e QNX4.x 2nd part    8e Linux LVM          df BootIt
 9 AIX bootable       4f QNX4.x 3rd part    93 Amoeba             e1 DOS access
 a OS/2 Boot Manag   50 OnTrack DM         94 Amoeba BBT        e3 DOS R/O
 b W95 FAT32         51 OnTrack DM6 Aux   9f BSD/OS            e4 SpeedStor
 c W95 FAT32 (LBA)   52 CP/M              a0 IBM Thinkpad hi   ea Rufus alignment
 e W95 FAT16 (LBA)   53 OnTrack DM6 Aux   a5 FreeBSD           eb BeOS fs
 f W95 Ext'd (LBA)   54 OnTrackDM6        a6 OpenBSD           ee GPT
10 OPUS              55 EZ-Drive          a7 NeXTSTEP          ef EFI (FAT-12/16/
11 Hidden FAT12      56 Golden Bow        a8 Darwin UFS         f0 Linux/PA-RISC b
12 Compaq diagnost  5c Priam Edisk        a9 NetBSD             f1 SpeedStor
14 Hidden FAT16 <3   61 SpeedStor         ab Darwin boot       f4 SpeedStor
16 Hidden FAT16     63 GNU HURD or Sys   af HFS / HFS+        f2 DOS secondary
17 Hidden HPFS/NTF   64 Novell Netware    b7 BSDI fs           fb VMware VMFS
18 AST SmartSleep    65 Novell Netware    b8 BSDI swap         fc VMware VMKCORE
1b Hidden W95 FAT3   70 DiskSecure Mult   bb Boot Wizard hid   fd Linux raid auto
1c Hidden W95 FAT3   75 PC/IX             bc Acronis FAT32 L   fe LANstep
1e Hidden W95 FAT1   80 Old Minix         be Solaris boot      ff BBT
Hex code (type L to list all codes): 82

Changed type of partition 'Linux' to 'Linux swap / Solaris'.

Command (m for help):

```

Make sure the bootable flag is enabled in your root partition, assuming you are going to install GRUB there, using the 'a' option:

```

Command (m for help): a
Partition number (1,2, default 2): 1

The bootable flag on partition 1 is enabled now.

```

Once you have everything arranged correctly (use p to print the layout and make sure everything is correct), you can simply write the changes into the disk by typing 'w'.

```

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.

root@archiso ~ #

```

And then use 'lsblk' to check the layout

```

root@archiso ~ # lsblk
NAME MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
loop0  7:0    0 474.1M  1 loop /run/archiso/sfs/airootfs
sda     8:0    0   16G  0 disk
├─sda1  8:1    0   13G  0 part
└─sda2  8:2    0    3G  0 part
sr0     11:0   1   586M  0 rom  /run/archiso/bootmnt

```

Formatting the partitions

Use mkfs to create the filesystem in the root partition. In my case this is sda1

```
# mkfs.ext4 /dev/sdxN
```

```

root@archiso ~ # mkfs.ext4 /dev/sda1
mke2fs 1.44.4 (18-Aug-2018)
Creating filesystem with 3407872 4k blocks and 851968 inodes
Filesystem UUID: 08e7f7f7-61d0-48ce-9da1-71e42bbcd5c9
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

root@archiso ~ # _

```

For the swap partition we use mkswap, and then swapon to enable it:

```
# mkswap /dev/sdxN
```

```
# swapon /dev/sdxN
```

```

1 root@archiso ~ # mkswap /dev/sda2
mkswap: /dev/sda2: warning: don't erase bootbits sectors
    (dos partition table detected). Use -f to force.
Setting up swapspace version 1, size = 3 GiB (3220172800 bytes)
no label, UUID=0ac6b470-9117-456a-8f5b-eb5969492c90
root@archiso ~ # swapon /dev/sda2
root@archiso ~ # _

```

Installing the base system

Now that we have our partitions for the system, it's time to install the base system. But first, we need to access our root filesystem. We can simply mount the root partition using mount:

```
# mount /dev/sdxN /mnt
```

And then use pacstrap to install it.

```
# pacstrap /mnt base base-devel
```

This will install both the base, and the development packages. If you only want the base, exclude the base-level package from pacstrap (I recommend installing it). This might take a while depending on your connection and your computer.

```

coreutils-8.30-1 cryptsetup-2.0.5-1 device-mapper-2.02.182-1 dhcpd-7.0.8-1
diffutils-3.6-2 e2fsprogs-1.44.4-1 fakeroot-1.23-1 file-5.35-1
filesystem-2018.8-1 findutils-4.6.0-4 flex-2.6.4-2 gawk-4.2.1-1
gcc-8.2.1+20180831-1 gcc-libs-8.2.1+20180831-1 gettext-0.19.8.1-3 glibc-2.28-5
grep-3.1-2 groff-1.22.3-8 gzip-1.9-2 inetutils-1.9.4-6 iproute2-4.19.0-1
iputils-20180629.f6aac8d-2 jfsutils-1.1.15-6 less-530-1
libtool-2.4.6+42+gb88ceb5-2 licenses-20181104-1 linux-4.19.4.arch1-1
linux-firmware-20181026.1cb4e51-1 logrotate-3.14.0-1 lvm2-2.02.182-1 m4-1.4.18-2
make-4.2.1-3 man-db-2.8.4-1 man-pages-4.16-2 mdadm-4.0-2 nano-3.2-1
netctl-1.19-1 pacman-5.1.1-1 patch-2.7.6-7 pciutils-3.6.2-1 perl-5.28.0-1
pkgconf-1.5.4-1 procps-ng-3.3.15-1 psmisc-23.2-1 reiserfsprogs-3.6.27-2
s-nail-14.9.11-1 sed-4.5-1 shadow-4.6-1 sudo-1.8.26-2 sysfsutils-2.1.0-10
systemd-239.303-1 systemd-sysucompat-239.303-1 tar-1.30-2 texinfo-6.5-2
usbutils-010-1 util-linux-2.33-2 vi-1:070224-3 which-2.21-3 xfsprogs-4.19.0-1

Total Download Size: 308.83 MiB
Total Installed Size: 1236.84 MiB

:: Proceed with installation? [Y/n]
:: Retrieving packages...
linux-api-headers-4.17.11-1-any 927.7 KiB 1205K/s 00:01 [#####] 100%
tzdata-2018g-1-x86_64 359.6 KiB 17.6M/s 00:00 [#####] 100%
iana-etc-20180913-1-any 364.9 KiB 10.8M/s 00:00 [#####] 100%
filesystem-2018.8-1-x86_64 7.5 KiB 0.00B/s 00:00 [#####] 100%
glibc-2.28-5-x86_64 9.1 MiB 2.03M/s 00:04 [#####] 100%
gcc-libs-8.2.1+20180831-1-x86_64 20.3 MiB 5.95M/s 00:03 [#####] 100%
ncurses-6.1-4-x86_64 1097.3 KiB 29.0M/s 00:00 [#####] 100%
readline-7.0.005-1-x86_64 294.5 KiB 28.8M/s 00:00 [#####] 100%
bash-4.4.023-1-x86_64 1428.3 KiB 32.4M/s 00:00 [#####] 100%
bzip2-1.0.6-8-x86_64 53.5 KiB 17.4M/s 00:00 [#####] 100%
attr-2.4.48-1-x86_64 65.0 KiB 0.00B/s 00:00 [#####] 100%
acl-2.2.53-1-x86_64 131.9 KiB 42.9M/s 00:00 [#####] 100%
gmp-6.1.2-2-x86_64 408.1 KiB 39.9M/s 00:00 [#####] 100%
libcap-2.26-1-x86_64 39.2 KiB 12.8M/s 00:00 [#####] 100%
gdbm-1.18.1-1-x86_64 160.6 KiB 39.2M/s 00:00 [#####] 100%
db-5.3.28-4-x86_64 1094.0 KiB 29.7M/s 00:00 [#####] 100%
perl-5.28.0-1-x86_64 5.5 MiB 1076K/s 00:08 [#####] 38%

```

Before doing anything else, we need to generate the fstab file. This is the file that tells the system the disk and partition layout.

```
# genfstab -U /mnt >> /mnt/etc/fstab
```

And finally, we can chroot to our root partition:

```
# arch-chroot /mnt
```

```
root@archiso ~ # arch-chroot /mnt
[root@archiso /]# _
```

Initial system configuration

We have to repeat some of the steps we already did but this time, they will be permanent on our system. The first step is to set the timezone for our computer.

Locate the timezone using ls and then create a symbolic link to it.

```
# ln -sf /usr/share/zoneinfo/region/city /etc/localtime
```

```
root@archiso ~ # arch-chroot /mnt
[root@archiso /]# ls /usr/share/zoneinfo
Africa      CET      Egypt    GMT+0    Iran      MST7MDT  Poland    UTC      posixrules
America     CST6CDT  Eire      GMT-0    Israel    Mexico    Portugal  Universal right
Antarctica  Canada   Etc       GMT0     Jamaica   NZ        ROC       W-SU    tzdata.zi
Arctic      Chile    Europe    Greenwich Japan     NZ-CHAT   ROK       WET     zone.tab
Asia        Cuba     Factory   HST      Kwajalein Nava'jo   Singapore Zulu     zone1970.tab
Atlantic    EET      GB         Hongkong Libya     PRC       Turkey    iso3166.tab
Australia   EST      GB-Eire   Iceland  MET       PST8PDT   UCT       leapseconds
Brazil      EST5EDT  GMT       Indian   MST       Pacific   US        posix
[root@archiso /]# ls /usr/share/zoneinfo/Europe
Amsterdam  Brussels  Guernsey  Lisbon    Monaco    Rome      Stockholm  Vienna
Andorra    Bucharest  Helsinki  Ljubljana Moscow     Samara    Tallinn    Vilnius
Astrakhan  Budapest   Isle_of_Man London     Nicosia   San_Marino Tirane     Volgograd
Athens     Busingen  Istanbul  Luxembourg Oslo       Sarajevo  Tiraspol   Warsaw
Belfast    Chisinau  Jersey    Madrid    Paris     Saratov   Ulyanovsk  Zagreb
Belgrade   Copenhagen Kaliningrad Malta      Podgorica Simferopol Uzhgorod  Zaporozhye
Berlin     Dublin    Kiev      Mariehamn Prague     Skopje    Vaduz     Zurich
Bratislava Gibraltar Kirov     Minsk     Riga      Sofia     Vatican
[root@archiso /]# ln -sf /usr/share/zoneinfo/Europe/Madrid /etc/localtime
[root@archiso /]#
```

And then use hwclock to create your local time:

```
# hwclock --systohc
```

Next step is to generate the locale information.

First edit /etc/locale.gen and uncomment the lines with the locale you want to use. Generally you want to uncomment en_US.UTF-8 UTF8 and then any other you might need to use. In my case the Spanish locale, es_ES.UTF-8 UTF-8

```
# nano /etc/locale.gen
```

```
#en_SC.UTF-8 UTF-8
#en_SG.UTF-8 UTF-8
#en_SG ISO-8859-1
en_US.UTF-8 UTF-8
#en_US ISO-8859-1
#en_ZA.UTF-8 UTF-8
#en_ZA ISO-8859-1
#en_ZM UTF-8
#en_ZW.UTF-8 UTF-8
#en_ZW ISO-8859-1
#eo UTF-8
#es_AR.UTF-8 UTF-8
#es_AR ISO-8859-1
#es_BO.UTF-8 UTF-8
#es_BO ISO-8859-1
#es_CL.UTF-8 UTF-8
#es_CL ISO-8859-1
#es_CO.UTF-8 UTF-8
#es_CO ISO-8859-1
#es_CR.UTF-8 UTF-8
#es_CR ISO-8859-1
#es_CU UTF-8
#es_DO.UTF-8 UTF-8
#es_DO ISO-8859-1
#es_EC.UTF-8 UTF-8
#es_EC ISO-8859-1
es_ES.UTF-8 UTF-8
#es_ES ISO-8859-1
#es_ES@euro ISO-8859-15
#es_GT.UTF-8 UTF-8
#es_GT ISO-8859-1
#es_HN.UTF-8 UTF-8
#es_HN ISO-8859-1
#es_MX.UTF-8 UTF-8
#es_MX ISO-8859-1
#es_NL.UTF-8 UTF-8
```

Save it with F3 and exit with Ctrl+X.

Use the locale-gen command and then edit the locale.conf file.

```
# locale-gen
# nano /etc/locale.conf
```

```
GNU nano 3.2 /etc/locale.conf
LANG=en_US.UTF-8_
```

Save and exit. Now edit `vconsole.conf` and set your keyboard layout there. In my case, it's a Spanish keyboard, so the layout is `es`.

```
# nano /etc/vconsole.conf
```

```
GNU nano 3.2 /etc/vconsole.conf
KEYMAP=es
```

Network configuration

First set the hostname for your computer by editing `/etc/hostname`. You can set it to almost anything you want, it's just a name for your computer over the network. I'll call it `enigmachine`.

```
# nano /etc/hostname
```

```
GNU nano 3.2 /etc/hostname
enigmachine
```

Now edit your `hosts` file to add your machine to it. This is important.

```
# nano /etc/hosts
```

```
GNU nano 3.2 /etc/hosts
# Static table lookup for hostnames.
# See hosts(5) for details.
127.0.0.1    localhost
::1         localhost
127.0.0.1    enigmachine.localdomain_
```

Now create the `initram` image that will be used by the kernel upon booting up.

```
# mkinitcpio -p linux
```



```

[root@archiso /]# mkinitcpio -p linux
==> Building image from preset: /etc/mkinitcpio.d/linux.preset: 'default'
  -> -k /boot/vmlinuz-linux -c /etc/mkinitcpio.conf -g /boot/initramfs-linux.img
==> Starting build: 4.19.4-arch1-1-ARCH
  -> Running build hook: [base]
  -> Running build hook: [udev]
  -> Running build hook: [autodetect]
  -> Running build hook: [modconf]
  -> Running build hook: [block]
  -> Running build hook: [filesystems]
  -> Running build hook: [keyboard]
  -> Running build hook: [fsck]
==> Generating module dependencies
==> Creating gzip-compressed initcpio image: /boot/initramfs-linux.img
==> Image generation successful
==> Building image from preset: /etc/mkinitcpio.d/linux.preset: 'fallback'
  -> -k /boot/vmlinuz-linux -c /etc/mkinitcpio.conf -g /boot/initramfs-linux-fallback.img -S autodetect
==> Starting build: 4.19.4-arch1-1-ARCH
  -> Running build hook: [base]
  -> Running build hook: [udev]
  -> Running build hook: [modconf]
  -> Running build hook: [block]
==> WARNING: Possibly missing firmware for module: wd719x
==> WARNING: Possibly missing firmware for module: aic94xx
  -> Running build hook: [filesystems]
  -> Running build hook: [keyboard]

```

Set a root password using the passwd command:

```
# passwd
```

You will be prompted for a password. Whatever you type won't be visible, so type slowly and make sure you don't screw up any characters.

```

[root@archiso /]# passwd
New password:
Retype new password:
passwd: password updated successfully
[root@archiso /]# _

```

Installing GRUB

First install the GRUB package. If you are installing alongside Windows, install ntfs-3g and os-prober (you can omit these if not). Press Y and ENTER when prompted if you want to proceed.

```
# pacman -S grub ntfs-3g os-prober
```

Now, if you are performing a MBR installation using a regular BIOS (or legacy BIOS mode) use the following command:

```
# grub-install --target=i386-pc /dev/sdx
```

If you are installing on UEFI mode (GPR), then do:

```
# grub-install --target=x86_64-efi --efi-directory=esp --bootloader-id=GRUB
```

```
[root@archiso /]# grub-install --target=i386-pc /dev/sda
Installing for i386-pc platform.
Installation finished. No error reported.
[root@archiso /]# _
```

And then generate the main configuration file

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

If you installed `os-prober` and `ntfs-3g`, it should also detect any Windows installation, if available.

You can now install any additional packages you want to install using `pacman`, or do it later. When you are done, exit the chroot environment, unmount the device, and reboot. Take out the installation medium from your computer and make sure you boot from your hard disk.

```
# exit
# umount /mnt
# reboot now
```

You should now reboot and boot into GRUB



And if you select “Arch Linux”, it should boot you into Arch.

Congratulations! Now you use Arch btw. You can login as root using the password you set earlier, and then create users and keep setting up the system.

```
Arch Linux 4.19.4-arch1-1-ARCH (tty1)
enigmachine login: root
Password:
[root@enigmachine ~]# _
```

Recommended actions:

Make sure your internet connection is enabled first. Repeat the steps mentioned at the beginning of this guide. If you need to enable the DHCP (you probably will need to enable it first), make sure to enable the daemon too as well as starting it.

```
# systemctl enable dhcpd@device
```

This will enable the DHCP daemon at the startup. Then start the daemon

```
# systemctl start dhcpd@device
```

Wait a few seconds (15-30), and make sure you have an address for your interface.

Create a new user with useradd and then set it's password:

```
# useradd -m enigmatico
```

```
# passwd enigmatico
```

If you want this user to have superuser rights, add him to the sudoers file like this:

```
# nano /etc/sudoers
```

```
GNU nano 3.2 /etc/sudoers
## Uncomment to enable logging of a command's output, except for
## sudoreplay and reboot. Use sudoreplay to play back logged sessions.
# Defaults log_output
# Defaults!/usr/bin/sudoreplay !log_output
# Defaults!/usr/local/bin/sudoreplay !log_output
# Defaults!REBOOT !log_output

##
## Runas alias specification
##

##
## User privilege specification
##
root ALL=(ALL) ALL

## --- EDIT HERE ---
enigmatico ALL=(ALL) ALL
## --- EDIT HERE ---

## Uncomment to allow members of group wheel to execute any command
```

Install some basic tools and programs. I will be installing the following:

- Linux headers: Headers for the current kernel. Useful to compile certain applications.
- Vim: Text editor
- Tmux: Terminal multiplexer, for multitasking.
- Lynx: A browser in your terminal.
- Irssi: IRC client for your terminal.
- Mutt: Mail client for your terminal.
- Xmms2: Audio player in the terminal.
- Alsa: Audio manager
- Git: git client, if you are going to install CDE you will need it.
- Cmake: Many projects use this, required in many cases to compile from source.

First make a full upgrade to ensure everything is up to date, including the cache of the package manager:

```
# pacman -Syuu
```

And then install everything:

```
# pacman -S linux-headers tmux lynx irssi mutt xmss2 alsa git cmake vim
```

Installing CDE

First we need the display manager. I don't think CDE works properly in wayland right now, so we will install xorg instead. Also install xinit and Compton (sinde CDE lacks of a compositor).

```
# pacman -S xorg xorg-xinit compton
```

Install the defaults for now. If you want nVidia/AMD drivers, you can do so later.

```
perl-net-smtp-ssl: git send-email TLS support
perl-authen-sasl: git send-email TLS support
perl-mediawiki-api: git mediawiki support
perl-datetime-format-iso8601: git mediawiki support
perl-lwp-protocol-https: git mediawiki https support
perl-cgi: gitweb (web interface) support
python2: various helper scripts [installed]
subversion: git svn
gnome-keyring: GNOME keyring credential helper
libsecret: libsecret credential helper [installed]
(21/25) installing shared-mime-info [#####] 100%
(22/25) installing jsoncpp [#####] 100%
Optional dependencies for jsoncpp
  jsoncpp-doc: documentation
(23/25) installing libuv [#####] 100%
(24/25) installing rhash [#####] 100%
(25/25) installing cmake [#####] 100%
Optional dependencies for cmake
  qt5-base: cmake-gui
  libxcbcommon-x11: cmake-gui
:: Running post-transaction hooks...
(1/8) Updating linux module dependencies...
(2/8) Warn about old perl modules
perl: warning: Setting locale failed.
perl: warning: Please check that your locale settings:
    LANGUAGE = (unset),
    LC_ALL = (unset),
    LANG = "en_US.UTF-8"
are supported and installed on your system.
perl: warning: Falling back to the standard locale ("C").
(3/8) Reloading system manager configuration...
(4/8) Creating system user accounts...
(5/8) Creating temporary files...
(6/8) Arming ConditionNeedsUpdate...
(7/8) Updating the info directory file...
(8/8) Updating the MIME type database...
[root@enigmachine ~]# _
```

To build CDE from source, we need a set of prerequisites first. Some of them are available from pacman:

```
# pacman -S git libxpm openmotif libxss libjpeg-turbo libjpeg6-turbo tcl m4 xorg-xfontsel rpcbind bison
xbitmaps ncurses flex
```

There are two packages that are not in the arch repository and need to be installed from source or from the AUR. These are ksh and ncompress. For this, you'll need to log in as a regular user which, if you followed this guide, you should have already done.

Now create a folder named 'sources' in your home directory and go inside:

```
$ mkdir sources && cd sources
```

Now clone the source of ksh with git and go inside it's directory

```
$ git clone https://aur.archlinux.org/ksh.git && cd ksh
```

Run makepkg to create a package, and then wait a while until it's done. This might take a while.

```
$ makepkg -si
```

After a while you will be prompted for your password. Type it, press enter, and answer Y to proceed with installation. Then wait a little bit more.

```
-> Generating .PKGINFO file...
-> Generating .BUILDINFO file...
-> Adding install file...
-> Generating .MTREE file...
-> Compressing package...
==> Leaving fakeroot environment.
==> Finished making: ksh 2014.06.25beta-1 (Fri Nov 30 17:45:33 2018)
==> Installing package ksh with pacman -U...

We trust you have received the usual lecture from the local System
administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.

[sudo] password for enigmatico:
Sorry, try again.
[sudo] password for enigmatico:
loading packages...
resolving dependencies...
looking for conflicting packages...

Packages (1) ksh-2014.06.25beta-1

Total Installed Size: 4.18 MiB

:: Proceed with installation? [Y/n] y
(1/1) checking keys in keyring [#####] 100%
(1/1) checking package integrity [#####] 100%
(1/1) loading package files [#####] 100%
(1/1) checking for file conflicts [#####] 100%
(1/1) checking available disk space [#####] 100%
:: Processing package changes...
(1/1) installing ksh [#####] 100%
mandb: can't set the locale; make sure $LC_* and $LANG are correct
```

Once it's finally done, repeat the process with ncompress. This won't take as long as the other one.

```
$ cd ..
$ git clone https://aur.archlinux.org/ncompress.git && cd ncompress
$ makepkg -si
$ cd ..
```

Once it's done, create a symlink to /usr/lib/cpp into /lib/cpp

```
# ln -s /usr/bin/cpp /lib/cpp
```

Now clone the source of cde into the sources directory.

```
$ git clone https://git.code.sf.net/p/cdesktopenv/code cdesktopenv-code && cd cdesktopenv-code/cde
```

Currently, CDE supports locales for English, German, Spanish, French, and Italian. You can either just build with support for en_US.UTF-8, or specify additional locales.

```
$ make World IMAKE_DEFINES='-DDtLocalesToBuild="en_US.UTF-8 es_ES.ISO8859-1"'
```

This will start the compilation process, which will take a while.

```
dtsrindex: Beginning Pass 1, reading records from 'CEEDOC.fzk'.
  Each dot = 20 records.
.....
dtsrindex: Rec #1000, 20% done. Est 0m 04s to end Pass 1.
.....
dtsrindex: Rec #2000, 55% done. Est 0m 00s to end Pass 1.
.....
dtsrindex: Rec #3000, 78% done. Est 0m 00s to end Pass 1.
.....
dtsrindex: Pass 1 completed in 0m 1s, read 3779 records.
  No duplicate records found, parsed 34621 words.
dtsrindex: Beginning Pass 2: batch index traversal and database update.
  Each dot = 500 words.
.....
dtsrindex: Word #25000, 72% done. Est 0m 00s to completion.
.....
dtsrindex: Pass 2 completed in 0m 1s, updated 34621 words.
dtsrindex: Exit Code = 0, Total elapsed time 0m 2s.
echo keytypes CEEDOC = Default Head Graphics Example Index Table > dtsearch.ocf
make[4]: Leaving directory '/home/enigmatico/sources/cdesktopenv-code/cde/doc/C/guides'
making all in doc/C/m-guides...
make[4]: Entering directory '/home/enigmatico/sources/cdesktopenv-code/cde/doc/C/m-guides'
make[4]: Nothing to be done for 'all'.
make[4]: Leaving directory '/home/enigmatico/sources/cdesktopenv-code/cde/doc/C/m-guides'
make[3]: Leaving directory '/home/enigmatico/sources/cdesktopenv-code/cde/doc/C'
making all in doc/en_US.UTF-8...
make[3]: Entering directory '/home/enigmatico/sources/cdesktopenv-code/cde/doc/en_US.UTF-8'
make[3]: Nothing to be done for 'all'.
make[3]: Leaving directory '/home/enigmatico/sources/cdesktopenv-code/cde/doc/en_US.UTF-8'
make[2]: Leaving directory '/home/enigmatico/sources/cdesktopenv-code/cde/doc'
make[1]: Leaving directory '/home/enigmatico/sources/cdesktopenv-code/cde'

Fri Nov 30 23:25:10 CET 2018

Full build of Release 2.3.0a of CDE complete.

[enigmatico@enigmachine cde]$_
```

Install CDE:

```
# ./admin/IntegTools/dbTools/installCDE -s ~/sources/cdesktopenv-code/cde/
```

Create a directory for the calendar service:

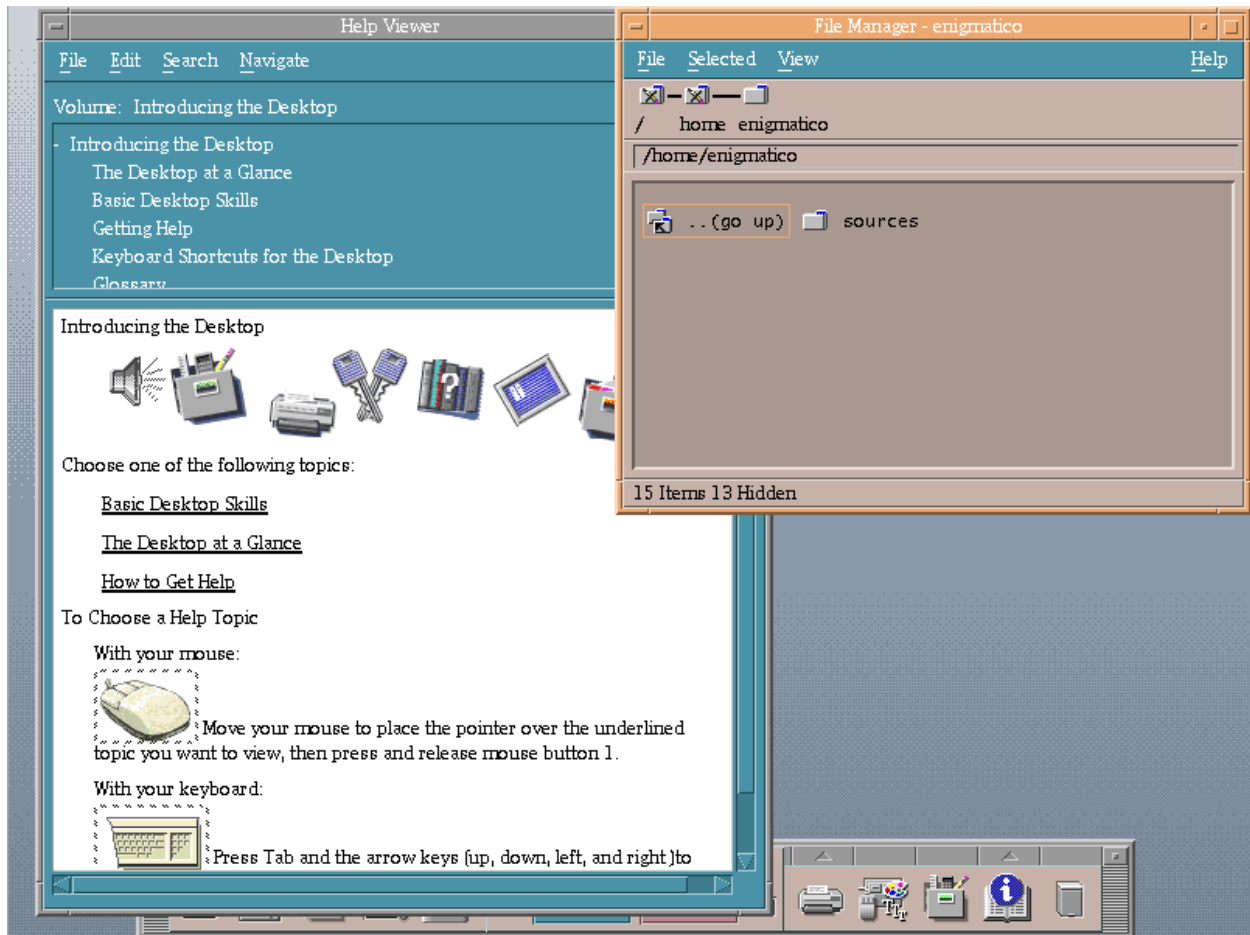
```
# mkdir -p /var/spool/calendar
```

Enable the rpcbind service and start it

```
# systemctl enable rpcbind && systemctl start rpcbind
```

Now run CDE to check if the installation was successful.

```
$ startx /usr/dt/bin/Xsession
```



Exit the session through the EXIT button in the system tray, it will bring you back to the terminal. Now let's finish configuring this by setting up the dtlogin manager and starting it on logon.

Create a startcde.sh file in your home directory like this:

```
$ nano ~/startcde.sh
```

```
GNU nano 3.2 startcde.sh
/usr/dt/bin/dtlogin -daemon
compton
```

Make it executable:

```
$ chmod +x startcde.sh
```