Xorg multiseat

Multiseat is a certain setup where N users work simultaneously on one computer. This is achieved by having N monitors, N keyboards and N mice. The advantages are quite obvious:

- Less power consumption (only one computer)
- Less hardware to purchase

Contents

The Ctrl-Alt-Fx, Alt-Fx keys mess up with virtual terminals

See also

Requirements

Keyboards and mice

Any standard PS/2 or USB keyboards will suffice. Same thing for mice.

Graphics hardware

For the best possible result you will need two graphics cards. I used an nVidia FX5500 AGP and an nVidia 6200 PCI. If you look around a bit you can certainly find new and decent PCI graphics card for a soft price.

It is possible to use only one videocard which has dual heads (like most nvidia cards will have), but this has some limitations: you have to use Xephyr on the second monitor which seems quite a messy solution from what I have read, and for optimal usage both screens need the same resolution.

If you have two pci-express slots, take advantage of them! That way you will even be able to play two games at the same time. (PCI is too slow to play comfortably)

Processors and memory

If you really are working with two users on the same computer, I would at least recommend a dual-core processor and plenty of RAM. A fast hard drive (10.000 RPM or higher) is also recommended for comfortable use.

Software

You will need Xorg with the drivers for your graphics card (according to some sources, the closed source nvidia driver works better than the open source nvidia driver for this, I have not tested this myself) and the evdev (xf86-input-evdev) driver. That's all. All this can be found in the Arch Linux core and extra repositories.

Some X knowledge

If you know how X works this will be a lot easier. Before you start, I recommend generating a clean configuration with xorgconfig that works with a single screen. Read through this xorg configuration and make yourself familiar. And as usual the manpages will provide you with most of the answers. You may reference some man pages: xorg, xserver, startx, xdm, xinit. **sudo X -configure**, **X -showopts** may give you some hint.

Definitions

For this article to be clear, I will be using the following definitions:

- screen: A screen is something Xorg can display its stuff on. A screen has a monitor and a graphics card assigned to it.
- monitor: A physical monitor like the one you are now sitting in front of.
- server layout: a definition of which screen, keyboard and mouse to use.
- seat: A workplace with a physical monitor, physical keyboard and physical mouse.

Tips and tricks

- Set up ssh on your computer, so you can ssh to the machine from another computer (such as a laptop). This is very useful because you will probably run into X not responding anymore or not giving you picture at all.
- Finding out which keyboard and mouse is which: open a terminal and use cat to find out. For example, `cat /dev/input/mouse1`. If you then move your mouse and you see all weird things happening than that is the mouse you are moving. Same goes for keyboards, which are called eventN.
- Try a basic configuration first. Do not start with the fancy stuff yet, get a very basic Xorg working first.
- Create a backup of all relevant configuration files. What do you mean you will skip this one?

About evdev

evdev is an Xorg driver which can make use of the kernel event devices, which you can find in `/dev/input`.

Attaching devices to a seat

Systemd's loginctl lets you assign devices to seats by creating udev rules.

Identify devices

Look at the default seat (seat0) status, here is an example:

```
# loginctl seat-status seat0
```

```
seat0
      Sessions: *11
       Devices:
              ├ /sys/devices/pci0000:00/0000:00:02.0/0000:03:00.0/drm/card0
              | (drm:card0)
              ├ /sys/devices/pci0000:00/0000:00:02.0/0000:03:00.0/graphics/fb0
              | (graphics:fb0) "radeondrmfb"
              ├ /sys/devices/pci0000:00/0000:00:02.0/0000:03:00.1/sound/card1
              | (sound:card1) "Generic"
              | └ /sys/devices/pci0000:00/0000:00:02.0/0000:03:00.1/sound/card1/input15
              |   (input:input15) "HD-Audio Generic HDMI/DP,pcm=3"
              ├ /sys/devices/pci0000:00/0000:00:12.0/usb3
              | (usb:usb3)
              | ├ /sys/devices/pci0000:00/0000:00:12.0/usb3/3-1/3-1:1.2/0003:046D:C52B.0006/input/input5
              | | (input:input5) "Logitech Unifying Device. Wireless PID:101b"
              | └ /sys/devices/pci0000:00/0000:00:12.0/usb3/3-1/3-1:1.2/0003:046D:C52B.0006/input/input6
              |   (input:input6) "Logitech Unifying Device. Wireless PID:200a"
              ├ /sys/devices/pci0000:00/0000:00:14.2/sound/card0
              | (sound:card0) "SB"
              | ├ /sys/devices/pci0000:00/0000:00:14.2/sound/card0/input7
              | | (input:input7) "HDA ATI SB Line"
              | └ /sys/devices/pci0000:00/0000:00:14.2/sound/card0/input9
              |   (input:input9) "HDA ATI SB Rear Mic"
              ├ /sys/devices/pci0000:00/0000:00:04.0/0000:02:00.0/drm/card1
              | (drm:card1)
              ├ /sys/devices/pci0000:00/0000:00:04.0/0000:02:00.0/graphics/fb1
              | (graphics:fb1) "radeondrmfb"
              ├ /sys/devices/pci0000:00/0000:00:04.0/0000:02:00.1/sound/card2
              | (sound:card2) "HDMI"
              | └ /sys/devices/pci0000:00/0000:00:04.0/0000:02:00.1/sound/card2/input16
              |   (input:input16) "HDA ATI HDMI HDMI/DP,pcm=3"
              ├ /sys/devices/pci0000:00/0000:00:12.1/usb4/4-1/4-1:1.0/input/input2
              | (input:input2) "CHESEN PS2 to USB Converter"
              ├ /sys/devices/pci0000:00/0000:00:12.1/usb4/4-1/4-1:1.1/input/input3
              | (input:input3) "CHESEN PS2 to USB Converter"
              └ /sys/devices/pci0000:00/0000:00:12.1/usb4/4-2/4-2:1.0/input/input4
               (input:input4) "Microsoft Microsoft 3-Button Mouse with IntelliEye(TM)"
```

If you are unsure, try comparing with `lspci` or `lsusb` to identify devices.

Assign devices

Seats are created based on graphics cards. Hence we will start by assigning a graphics card:

```
# loginctl attach seat1 /sys/devices/pci0000:00/0000:00:04.0/0000:02:00.0/drm/card1
# loginctl attach seat1 /sys/devices/pci0000:00/0000:00:04.0/0000:02:00.0/graphics/fb1
```

Then add input devices. This first example (not recommended) adds three USB devices by port and subdevice from the list above:

```
# loginctl attach seat1 /sys/devices/pci0000:00/0000:00:12.1/usb4/4-1/4-1:1.0/input/input2
# loginctl attach seat1 /sys/devices/pci0000:00/0000:00:12.1/usb4/4-1/4-1:1.1/input/input3
# loginctl attach seat1 /sys/devices/pci0000:00/0000:00:12.1/usb4/4-2/4-2:1.0/input/input4
```

This is less than ideal, because replacing the keyboard with another will only work if the new keyboard contains the same subdevices (`1.0` and `1.1`), and if the keyboard and mouse are accidentally swapped into each other's USB port, they will not be assigned to the intended seat and will end up back in the default `seat0` . It would be more flexible to add a specific USB port to the seat instead, allowing any device plugged into that USB port to be assigned to the seat (keyboard, mouse, USB sound card, etc.) Here, two USB ports are assigned to the seat, which you can see has just come from the same list above with the end part removed:

```
# loginctl attach seat1 /sys/devices/pci0000:00/0000:00:12.1/usb4/4-1
# loginctl attach seat1 /sys/devices/pci0000:00/0000:00:12.1/usb4/4-2
```

With multiple sound cards you can assign one per seat that will run out of the box with per user session pulseaudio:

```
# loginctl attach seat1 /sys/devices/pci0000:00/0000:00:04.0/0000:02:00.1/sound/card2
```

You should be able to see that the hardware has been partitioned into two seats:

```
# loginctl seat-status seat0
# loginctl seat-status seat1
```

The seat names are just string tags, so you can call a seat anything you like - `seat-1` , `seatleft` and `seat-tv` are all valid names. While you do not have to use the `seat` prefix, this is not recommended as some programs (e.g. LightDM) apply their default options only to seats matching the expression `seat*` , so any name that does not start with the word `seat` may require extra manual configuration. `seat0` is the default seat name and cannot be changed.

Any device that is not specifically tagged to a seat will end up in `seat0` . If a device is tagged to a given seat, then all child devices belong to that seat as well. This is most useful with hot pluggable interfaces, such as the above example that assigns USB ports to specific seats so that any devices plugged into those ports will automatically inherit the seat set on on the USB port itself.

`loginctl attach` creates and deletes udev rule files matching `/etc/udev/rules.d/72-seat-*.rules` , so if you wish to back up your seat configuration, these are the files to save and restore.

Each seat will not appear as ready until a device tagged `master-of-seat` appears. The default udev rules automatically apply this tag to any `drm` or `graphics` device. Thus if you do not make a one of these devices part of a seat (e.g. a visually impaired user with only keyboard and audio) then the udev rules created by `loginctl` will need to be manually edited and this tag specified before the seat becomes available for use. Look at the default udev rules for examples.

Setting up Xorg

Since you can attach devices to seats with *loginctl* there is no need to have an specific **Xorg configuration**, so if you attached devices via *loginctl* creating Xorg configuration files may be skipped. Also note that setting

`ServerFlags "AutoAdd*"` options to `false` forces Xorg to just ignore most of the *loginctl* attachments.

If you do choose to use a custom Xorg configuration (for example you have an nVidia card and want to set advanced display options) then the process is much the same as usual, with the major difference being that you define one `ServerLayout` section for each seat, and include the `MatchSeat` statement in each server layout to indicate which seat it should apply to. You can also apply `MatchSeat` directives to graphics devices and monitors themselves to avoid defining a `ServerLayout`, if you do not need any `ServerLayout` functionality. Be aware that any section missing a `MatchSeat` directive will be visible for all server layouts and seats, so make sure some other part of the configuration prevents multiple seats from trying to access the same devices. (A common mistake is allowing both seats to access the same display device, which will result in one seat working and the others failing to start.)

An example follows for this type of manual configuration. You can create parts of, or the entire, configuration set for two server layouts, each assigned with their own keyboard, mouse, video card and monitor. If you omit a part (such as the `InputDevice` section), then the Xorg defaults will apply.

xorg.conf.d tree example:

```
├── 00-keyboard.conf   #created by localectl
├── 31-monitor-seat0.conf
├── 32-monitor-seat1.conf
├── 40-serverflags.conf
├── 41-graphic-seat0.conf
├── 42-graphic-seat1.conf
├── 51-screen-seat0.conf
├── 52-screen-seat1.conf
├── 61-layout-seat0.conf
└── 62-layout-seat1.conf
```

Defining available input devices

This part of the configuration tells Xorg which input devices it has available. Input devices are keyboards and mice, but can also be, for example, touchscreens and pens.

Configure your devices as read in **Xorg#Input devices**. The "Identifier" will let you assign to a ServerLayout and "Device" defines which physical device are you configuring.

01-keyboard-seat0.conf

```
Section "InputDevice"
      Identifier     "keyboard0"
      Option         "Device"            "/dev/input/event1"
      #Option         "Device"            "/dev/input/by-path/pci-0000:00:1a.0-usb-0:1.6:1.0-event-kbd"  # or by path

      # GrabDevice no longer has any effect: https://www.spinics.net/lists/xorg/msg59881.html
      Option "GrabDevice" "on" # prevent send event to other X-servers
EndSection
```

Same for other **input devices**.

Monitors

Configure your monitors as seen on **Xorg#Monitor settings**. Pay close attention to the identifier.

31-monitor-seat0.conf

```
Section "Monitor"
   Identifier          "monitor0"
EndSection
```

ServerFlags

40-serverflags.conf

```
Section "ServerFlags"
    Option "AutoAddGPU" "off"
EndSection
```

Graphics card

Take a close look at the BusID. This option specifies which hardware card to use. You can find out the BusId's with `lspci`. However, you will soon find out this does not always match. That's because lspci displays the device address in hexadecimal form. Xorg however uses decimal form. So you will need to convert your address from hexadecimal form to decimal. Thus a device address of 0:0a:0 in lspci would become 0:10:0 in `xorg.conf`.

42-graphic-seat1.conf

```
Section "Device"
    Identifier     "graphic0"
    Driver         "nvidia"
    Option         "NoLogo"             "1"  # Remove nvidia branding at startup
    BusId          "PCI:1:0:0"
    MatchSeat      "seat-1"               # Needed when only configuring part of Xorg for non-seat0.
    Option         "Monitor-DVI-1"      "monitor1"  #assigns monitor configuration to graphic output port
EndSection
```

Another more advanced example where nVidia TwinView is used so that one seat has dual monitors, with one monitor landscape and the other portrait:

42-graphic-seat-nvidia.conf

```
Section "Device"
    Identifier "nvidia"
    Driver "nvidia"
    BusID "PCI:66:0:0"  # lspci reports this as 42:00.0, so 42 hex is 66 decimal

    # See the nVidia documentation for what these options do.  They of
    # course only apply if you have an nVidia card and you are using their
    # closed-source driver.
    Option "UseDisplayDevice" "DP-3"
    Option "MetaModes" "DP-3: 2560x1600, DP-0: 1600x1200 { Rotation=left }"
    Option "TwinViewOrientation" "LeftOf"

    # Not needed, avoid errors in logs
    Option "ConnectToAcpid" "false"

    # Do not automatically set the DPI based on screen size, use the traditional
    # value to avoid fonts becoming overly large.
    Option "UseEdidDpi" "False"
    Option "DPI" "96 x 96"

    # No MatchSeat option is needed here if it's given in the ServerLayout instead.
    #Option "MatchSeat" "seat-nvidia"
EndSection
```

Screens

Pay close attention to the "monitor" option.

51-screen-seat0.conf

```
Section "Screen"
    Identifier          "screen0"
```

```
    Device          "graphic0"
    Monitor         "monitor0"
    DefaultDepth    24
    Subsection "Display"
        Depth   24
        Modes   "1280x1024"    "1024x768"
    EndSubsection
EndSection
```

ServerLayout

Create a section for every seat (pay attention to Identifier) you have with their respective keyboards, mice and screens.

62-layout-seat1.conf

```
Section "ServerLayout"
    # The Identifier is not really important but must be unique.
    Identifier     "seat-1"

    # This controls which Screen section is used in the layout, which in turn
    # dictates which GPU and monitor will be used.  The numbers can be omitted
    # if only one Screen is used (normally the case unless you are combining
    # monitors from different video cards).
    Screen        "screen1"      0               0

    # InputDevice sections can usually be omitted
    InputDevice    "mouse1"       "CorePointer"
    InputDevice    "keyboard1"    "CoreKeyboard"

    Option        "SingleCard" "on"   # use this to simplfied isolatedevice option

    # MatchSeat means this layout will only be used when the "-seat seat-1"
    # parameter is given to /usr/bin/X, which most display managers will do
    # automatically.
    MatchSeat      "seat-1"
EndSection
```

Additional Tips:

- Make sure to delete the `~/.Xauthority` file in respective user directories before the initial reboot.
- To avoid tearing this seems to help on nearly all configurations - add this to /etc/environment:

```
CLUTTER_PAINT=disable-clipped-redraws:disable-culling
CLUTTER_VBLANK=True
```

Testing

Before we start modifying our login manager, we will first start with testing out the individual seats. If these are working, then we are good to go.

I have used twm (tiny window manager) to test out if my seats work, but there is no reason you cannot use KDE, gnome, or any other desktop environment or window manager. I have used this in my `~/.xinitrc` :

```
exec twm
```

Use the following command to test out an individual seat:

```
startx -- -layout seat0 -config xorg.conf.multiseat
```

Do this for every seat you have. If they are all working correctly and the keyboard/mouse combination matches, then

congratulations! You are almost finished! In case you are wondering why I did not you use the full path to my new configuration file, that's because X does not allow that when running as non-root. It will search for xorg.conf.multiseat relative to `/etc/X11`.

Setting up the loginmanager

KDM

Open `/usr/share/config/kdm/kdmrc` and set the following variables:

```
StaticServers=:0,:1 #In the case of two seats. If you have three this would become :0,:1,:2 and so forth.
ReserveServers=:2,:3 #You can define here as many as you want, but these should always start at the highest seat + 1.
```

Next you will need to add an [X-:n-Core] for each seat (where n = the seat)

```
[X-:0-Core]
ServerArgsLocal=-nolisten tcp -layout seat0 -seat seat0

[X-:1-Core]
ServerArgsLocal=-nolisten tcp -layout seat-1 -seat seat-1 -novtswitch
```

Add section like this for every seat you have, and do not forget to change the :0 and the -layout seat0. Note that the "-novtswitch" options should be added for all seats *except* the first one. Otherwise, you can end with rectangles of virtual terminals "showing through" on your primary screen.

For XDM

Open `/etc/X11/xdm/Xservers` and set the following variables (This sample demos two seats):

```
# NOTE: do not add -sharevts on seat0, otherwise it may reset in about 10~20 minutes automatically.
:0 local /usr/bin/X :0 vt07 -nolisten tcp  -novtswitch -layout seat0 -seat seat0
:1 local /usr/bin/X :1 vt08 -nolisten tcp -novtswitch -layout seat-1 -seat seat-1
```

Note: If seat numbers are not correctly assigned to sessions (`loginctl` will show a session without seat) by XDM, then your attached devices could have strange behavoir like lack of permissions, sound card undetected by pulseaudio... LightDM should be fine.

Also if you use the Archlinux theme edit `/etc/X11/xdm/xdm-config` for every screen:

```
DisplayManager._0.setup:      /etc/X11/xdm/arch-xdm/Xsetup
DisplayManager._0.startup:    /etc/X11/xdm/arch-xdm/Xstartup
DisplayManager._0.reset:      /etc/X11/xdm/arch-xdm/Xreset
DisplayManager._1.setup:      /etc/X11/xdm/arch-xdm/Xsetup
DisplayManager._1.startup:    /etc/X11/xdm/arch-xdm/Xstartup
DisplayManager._1.reset:      /etc/X11/xdm/arch-xdm/Xreset
```

For LightDM

LightDM has automatic multiseat detection (pulling from the seat list provided by `loginctl list-seats`). If you have a manual Xorg configuration file and want to use it, make sure you specify valid `MatchSeat` values there (see above). This is because LightDM automatically passes `-seat seat0` or similar to `/usr/bin/X`, so Xorg will ignore any sections that have a `MatchSeat` that does not match the current seat. (So if you forget to specify `MatchSeat` at all, those sections will apply to every seat which, in the case of display devices, means one or more seats likely will not start at all as they all compete for the same display.)

If you need you can configure it on `/etc/lightdm/lightdm.conf`

```
[LightDM]
run-directory=/run/lightdm

[Seat:*]
greeter-session=lightdm-gtk-greeter
greeter-hide-users=false # Bug: lightdm-gtk-greeter does not load user saved session when greeter-hide-users=true[1] (https://bugs.launchpad.net/lightdm-gtk-greeter/+bug/837002)
session-wrapper=/etc/lightdm/Xsession

[Seat:seat0]
xserver-command=/usr/bin/X :0
# If you cannot put `MatchSeat "seat0"` in the `ServerLayout` section for some reason, you can force a layout here.
xserver-layout=seat0

[Seat:seat-1]
xserver-command=/usr/bin/X :1
xserver-layout=seat-1
```

Note: Since version 1.16 Xorg server no longer handles VTs if it was started as a non-seat0. So there is no need to pass option -sharevts [2] (http://permalink.gmane.org/gmane.comp.sysutils.systemd.devel/22346).

For Auto Login multiseat (without Display Manager)

edit a script /boot/twin.sh

```
#!/bin/bash
cmd1="/bin/bash --login -c \"/usr/bin/xinit --"
cmd2="-nolisten tcp -keeptty -novtswitch -config xorg.multiseat.conf"
usr=(user1 user2)  # FIXME: assume user1, user2 is valid user id
declare -a pid
while true ; do
 for ((i=0; i<${#usr[*]}; i++)) ; do
  echo "usr[$i]=${usr[$i]} pid=${pid[$i]}"
  if [ -z "${pid[$i]}" ] || [ ! -d "/proc/${pid[$i]}" ] ; then
   # echo "pid ${pid[$i]} killed, execute again"
   cmd3="-layout seat$i vt0"$((7+i))"\""
   if [ $i -gt 0 ] ; then
    cmd3="-sharevts $cmd3"
   fi
   #echo "cmd3=$cmd3"
   /bin/su ${usr[$i]} -l -c "$cmd1 :$i $cmd2 $cmd3" &
   pid[$i]=$!
   #echo "new pid=${pid[$i]}"
  fi
 done
 sleep 5  # check process exist per 5 second
done
```

Open `/etc/inittab` and setup as follows:

```
#id:3:initdefault:
id:5:initdefault:
...
x2:5:once:/root/twin.sh > /root/twin.log 2>&1
```

Sound

One sound card for each seat

As soon as you **attach** the sound card to the seat, Pulseaudio will detect and use it.

Multiple users on single sound card: PulseAudio

If two users want to use the sound card simultaneously, it is necessary to use a sound server, **PulseAudio** being most prevalent. Usually, the PulseAudio server runs only for active user and does not allow for multiple user instances. Solution to this problem is using the system-wide PulseAudio server. Although this approach is discouraged by its authors, it is probably most applicable setup.

Configuring for multiple users, without system-mode daemon

This results in all the mixing transferred to a single server.

Copy the default configuration to the main user's home directory:

```
$ cp /etc/pulse/default.pa ~/.pulse/
```

or:

```
$ cp /etc/pulse/default.pa ~/.config/pulse/
```

Edit the file, adding at the end:

```
load-module module-native-protocol-tcp auth-ip-acl=127.0.0.1
```

Restart pulseaudio, if already running:

```
$ pulseaudio -k
$ pulseaudio --daemonize=no
```

Repeat this procedure for each secondary user, as the user:

```
$ echo "default-server = 127.0.0.1" > ~/.pulse/client.conf
```

Configuring for system-wide PulseAudio

- Create user pulse and put it into group audio (PulseAudio drops root privileges and changes to user pulse. Group membership allows for device access.)
- Create group pulse-access and put users, who will play sound locally into it (Group membership is used for access control for local access to PA daemon.)
- In /etc/pulse/default.pa state explicitly the access rights

```
load-module module-native-protocol-unix auth-group=pulse-access
auth-group-enable=1
```

Create /etc/dbus-1/system.d/pulseaudio.conf with this content **/etc/dbus-1/system.d/pulseaudio.conf (https://bbs.archlinux.org/viewtopic.php?pid=563481#p563481)**:

```
<?xml version="1.0" encoding="UTF-8"?>
<busconfig>
    <policy user="pulse">
        <allow own="org.pulseaudio.Server"/>
        <allow send_destination="org.pulseaudio.Server"/>
        <allow receive_sender="org.pulseaudio.Server"/>
    </policy>
</busconfig>
```

Start PA as system-wide, under root:

```
# pulseaudio --system
```

In `/var/run/pulse` should appear files for communication with daemon, namely pid and native.

User access

You can check communication with system daemon as non-root by e.g. `pactl -s "unix:/var/run/pulse/native" list`.

It is possible to enable automatic network connection to local daemon in `/etc/pulse/client.conf`:

```
auto-connect-localhost = yes
```

The users should be able to connect to PA server. All the cons for system-wide daemon become essentially pros, e.g. ability to control volume of other users streams in pavucontrol.

Troubleshooting

It is possible to enable the http interface to PA for debugging in /etc/pulse/default.pa `load-module module-http-protocol-tcp` and then connect to it at **http://localhost:4714/**

Multiple users on single sound card: ALSA

Setup with PulseAudio removed, shared audio through an alsamixer equalizer and software mixing. The relevant hardware used is an ATI Radeon HD 5850 and an Intel Sandy Bridge (onboard) HD 3000. Configuration steps may vary.

Specific hardware could be addressed in `/etc/asound.conf` to allocate audio to both users simultaneously if required. Another option enables two different users to both access audio with their own equalizer (described further down). The sound card will be shared equally among the seats using ALSA with PulseAudio removed - libpulse itself should be kept for various software dependencies however.

Next, remove respective `~/.asoundrc` files (as well as related PulseAudio config files if you removed that) and follow this template with `/etc/asound.conf` for sound:

```
defaults.pcm.rate_converter "samplerate_best"

ctl.equal {
 type equal;
}

pcm.plugequal {
  type equal;
  # Modify the line below if you do not
  # want to use sound card 0.
  #slave.pcm "plughw:0,0";
  #by default we want to play from more sources at time:
  slave.pcm "plug:dmix";
}
#pcm.equal {
  # If you do not want the equalizer to be your
  # default soundcard comment the following
  # line and uncomment the above line. (You can
  # choose it as the output device by addressing
  # it with specific apps,eg mpg123 -a equal 06.Back_In_Black.mp3)
pcm.!default {
  type plug;
  slave.pcm plugequal;
}
```

Only the last user to login will have audio control and access but if you wish to share audio equally among different users, add each user to the audio group:

Example:

```
usermod -a -G ftp joeblow
usermod -a -G ftp jillschmill
```

Then put this in each user's respective ~/.asoundrc rather than using /etc/asound.conf (this option also contains various tweaks to improve audio quality):

```
defaults.pcm.rate_converter "samplerate_best"

ctl.equal {
 type equal;
}

pcm.ossmix {
   type dmix
   ipc_key 1024 # must be unique!
   ipc_key_add_uid false   # let multiple users share
   ipc_perm 0666          # IPC permissions for multi-user sharing (octal, default 0600)
   slave {
      pcm "hw:0,0"     # you cannot use a "plug" device here, darn.
      period_time 0
      period_size 1024 # must be power of 2.
      buffer_size 8192  # ditto.
      rate 44100
     #format "S32_LE"
     #periods 128 # dito.
     #rate 8000 # with rate 8000 you *will* hear,
     # if ossmix is used :)
   }
   # bindings are cool. This says, that only the first
   # two channels are to be used by dmix, which is
   # enough for (most) oss apps and also lets
   # multichannel chips work much faster:
   bindings {
      0 0 # from 0 => to 0
      1 1 # from 1 => to 1
   }
}

pcm.plugequal {
  type equal;
  # Modify the line below if you do not
  # want to use sound card 0.
  #slave.pcm "plughw:0,0";
  #by default we want to play from more sources at time:
  slave.pcm "plug:ossmix";
}

#pcm.equal {
  # If you do not want the equalizer to be your
  # default soundcard comment the following
  # line and uncomment the above line. (You can
  # choose it as the output device by addressing
  # it with specific apps,eg mpg123 -a equal 06.Back_In_Black.mp3)
pcm.!default {
  type plug;
  slave.pcm plugequal;
}
```

Accessing the equalizer can be done with:

```
alsaequal -D equal
```

Each user has an equalizer they can configure separately.

Make sure to turn down and mute the audio channels that you do not use, turn off auto-mute microphone, and make sure no channel has a gain higher than 0 to avoid ALSA audio bugs. This can be done via alsamixer.

Troubleshooting

Xorg will not start

Look at the log files in `/var/log/Xorg.*.log`. There will be one for each seat. Reading the log file will show you which devices the seat is trying to use, and you may see something like both seats are trying to use the same display device, because you have not correctly assigned it to a particular seat.

If you are using a display manager, find out where its log files are so you can check that it is starting the Xorg server properly, and passing the required `-seat` parameter.

My Windows key does not work anymore

Put this in **a startup file**:

```
xmodmap -e "add Mod4 = Super_L Super_R"
```

Unreliable behaviour (black picture without cursor)

If everything seems to be set up correctly, but for some reason you always get a black picture without a cursor, try setting the first initialized card in the BIOS to be the PCI card one.

Little black boxes/dots on the desktop

This is actually portions of the virtual terminals being painted on top of X. It seems to be caused by the Linux kernel framebuffer. This can be fixed by disabling the framebuffer, or by removing the "-sharevts" option from the primary seat's X args.

Multimedia keys not working

If your keyboard(s) has extra "multimedia" keys, you may find that they stopped working in your multiseat setup. This is because such keyboards are often represented as more then one "event" device. As you did above, cat each /dev/input/event* device, this time pressing multimedia keys. Once you have found the right event device, add a separate keyboard InputDevice section for it, then add that InputDevice section to the corresponding ServerLayout section with the "SendCoreEvents" option, which indicates that input from this device should be handled, despite not being the core keyboard. In the end you should have sections something like the following:

```
Section "InputDevice"
   Identifier     "Keyboard0"
   Driver         "evdev"
   Option         "Device" "/dev/input/event6"
   Option         "XkbModel" "evdev"
EndSection

Section "InputDevice"
   Identifier     "Keyboard0Multimedia"
   Driver         "evdev"
   Option         "Device" "/dev/input/event7"
   Option         "XkbModel" "evdev"
EndSection

Section "ServerLayout"
   Identifier     "Layout0"
   Screen      0  "Screen0" 0 0
   InputDevice    "Keyboard0" "CoreKeyboard"
   InputDevice    "Keyboard0Multimedia" "SendCoreEvents"
   InputDevice    "Mouse0" "CorePointer"
```

```
    Option      "AutoAddDevices" "no"
EndSection
```

The Ctrl-Alt-Fx, Alt-Fx keys mess up with virtual terminals

(Oct 2010) I follows this guide and everything works, except for Atl-F1, Atl-F2,... mess things up. Then I follow this guide **https://help.ubuntu.com/community/MultiseatX** (read the part for Ubuntu 10.04):

```
# cd /usr/bin
# ln -s X X0
# ln -s X X1
```

Then fix in the **/usr/share/config/kdm/kdmrc** as follow

```
[General]
ConsoleTTYs=tty1,tty2,tty3,tty4,tty5,tty6
ServerVTs=7,8
StaticServers=:0,:1
ReserveServers=:2,:3
...

[X-:0-Core]
ServerVT=8
ServerCmd=/usr/bin/X1
ServerArgsLocal=-nolisten tcp -sharevts -novtswitch -keeptty -layout Seat1 -isolateDevice PCI:1:0:0

[X-:1-Core]
ServerVT=7
ServerCmd=/usr/bin/X0
ServerArgsLocal=-nolisten tcp -novtswitch -keeptty -layout Seat0 -isolateDevice PCI:0:2:0
...
```

It works for my computer: one on-board Intel card (xf86-video-intel driver), and one Nvidia card (xf86-video-nouveau driver). You can check if the parameters are passed correctly by:

```
$ pgrep -af PCI
root    16993  1.6  1.3  32900 26772 ?      S    08:09   0:19 /usr/bin/X0 :1 vt7 -nolisten tcp -novtswitch -keeptty -layout Seat0 -isolateDevice PCI:0:2:0 -auth /var/
run/xauth/A:1-ES6CCb
root    17124  5.9  0.5  18996 11980 ?      S    08:09   1:09 /usr/bin/X1 :0 vt8 -nolisten tcp -sharevts -novtswitch -keeptty -layout Seat1 -isolateDevice PCI:1:0:0 -
auth /var/run/xauth/A:0-Wgiyza
```

The **ServerVT=7**, **ServerVT=8** would be pass to as **vt7**, **vt8**

See also

- **Multi-pointer X** explains how to setup two separate pair of devices on the same session.
- **The original Arch Forums thread (https://bbs.archlinux.org/viewtopic.php?id=105450)**.
- **Multiseat using loginctl (https://code.lexarcana.com/posts/simple-multiseat-setup-on-fedora-17.html)**.
- **Using udev to configure multi-seat automatically (https://code.lexarcana.com/posts/using-udev-to-configure-fedora-multi-seat-automatically.html)**.