gentoo linux (/) Wiki

PipeWire

From Gentoo Wiki
Jump to:navigation Jump to:search

PipeWire is a low-latency, graph-based, processing engine and server, for interfacing with audio and video devices. It can be used to support use-cases currently handled by ALSA (/wiki/ALSA), PulseAudio (/wiki/PulseAudio), and/or JACK (/wiki/JACK), and aims to improve handling of audio and video under Linux.

Note
Wayland systems use PipeWire to provide desktop portal functionality such as screen sharing and remote desktop.

Some key features of PipeWire include:

- Minimal latency capture/playback of audio *and* video.
- Real-time multimedia processing.
- Multi-process architecture allowing multimedia content sharing between applications.
- Seamless support for PulseAudio, JACK, ALSA, and GStreamer (/index.php?title=GStreamer&action=edit&redlink=1).
- Applications sandboxing support with Flatpak (/wiki/Flatpak), with a security model that facilitates interacting containerized applications.

Resources

- Home (https://pipewire.org/)
- Official documentation (https://gitlab.freedesktop.org/pipewire/pipewire/-/wikis/home)
- Package information (https://packages.gentoo.org/packages/media-video/pipewire)
- Wikipedia (https://en.wikipedia.org/wiki/PipeWire)
- Open Hub (https://www.openhub.net/p/pipewire)
- GitLab (https://gitlab.freedesktop.org/pipewire/pipewire.git)
- `pipewire (irc://irc.oftc.net/pipewire)(on irc://irc.oftc.net (irc://irc.oftc.net))`(registration required)

PipeWire currently ships a PipeWire daemon, an example session manager, tools to introspect and use the PipeWire Daemon, a library to develop PipeWire applications and plugins, and the SPA (Simple Plugin API) used by both the PipeWire daemon and the PipeWire library.

Contents

Installation

Kernel

Needed `media-video/wireplumber` (https://packages.gentoo.org/packages/media-video/wireplumber) options:

KERNEL

```
Device Drivers  --->
  <*> Sound card support     --->
    <*> Advanced Linux Sound Architecture     --->
      -*-  Sound Proc FS Support
      [*]    Verbose procfs contents
```

USE flags

To use PipeWire as a sound server, specify the *sound-server* USE flag (/wiki/USE_flag) on `media-video/pipewire` (https://packages.gentoo.org/packages/media-video/pipewire)[1].

Then, if using PipeWire as a replacement for the PulseAudio sound server:

1. Ensure `media-sound/pulseaudio-daemon` (https://packages.gentoo.org/packages/media-sound/pulseaudio-daemon) is not installed. This is necessary to avoid issues resulting from running more than one sound server.
2. Ensure `media-libs/libpulse` (https://packages.gentoo.org/packages/media-libs/libpulse) is installed, which will allow PipeWire to emulate a PulseAudio sound server. Not many applications currently support PipeWire's native API.
3. Ensure the *pulseaudio* USE flag is still set globally.

PipeWire requires the presence of a D-Bus (/wiki/D-Bus) session bus and an XDG-compliant environment. Both requirements should be met by a desktop profile (/wiki/Profile_(Portage)); on such systems, starting PipeWire is as simple as running the **`pipewire`** binary. On other profiles, if using OpenRC, permissions requirements might also need the *elogind* USE flag on the `media-video/wireplumber` (https://packages.gentoo.org/packages/media-video/wireplumber) package, together with elogind (/wiki/Elogind) itself.

To enable direct screencasting support on applications offering it, specify the *screencast* USE flag on the relevant packages. Otherwise, screencasting support may also be provided through the PulseAudio or JACK compatibility layers.

USE flags for media-video/pipewire (https://packages.gentoo.org/packages/media-video/pi...	
+man (https://packages.gentoo.org/useflags/+man)	Build and install man pages
X (https://packages.gentoo.org/useflags/X)	Enable audible bell for X11
bluetooth (https://packages.gentoo.org/useflags/bluetooth)	Enable Bluetooth Support
dbus (https://packages.gentoo.org/useflags/dbus)	Enable dbus support for anything that needs it (gpsd, gnomemeeting, etc)
doc (https://packages.gentoo.org/useflags/doc)	Add extra documentation (API, Javadoc, etc). It is recommended to enable per package instead of globally

More information about USE flags (/wiki/Handbook:AMD64/Working/USE) Data provided by the Gentoo Package Database (https://packages.gentoo.org) · Last update: 2025-11-29 16:05

Emerge

Once the USE flags have been specified, rebuild the affected packages:

```
root # emerge --ask --verbose --changed-use --update --deep @world
```

Alternatively, PipeWire may be emerged independently, though the previous method is usually what is required:

```
root # emerge --ask media-video/pipewire
```

Typically, the WirePlumber session manager should also be installed. WirePlumber is a replacement for **pipewire-media-session**; refer to this article (https://www.collabora.com/news-and-blog/blog/2020/05/07/wireplumber-the-pipewire-session-manager/) for an introduction and the details. The `media-video/wireplumber` (https://packages.gentoo.org/packages/media-video/wireplumber) package provides the **wireplumber.service** systemd service and the **`wireplumber`** binary used by **gentoo-pipewire-launcher**.

```
root # emerge --ask media-video/wireplumber
```

Configuration

See also
Detailed, non Gentoo-specific, configuration documentation can be found at the project's official wiki (https://gitlab.freedesktop.org/pipewire/pipewire/-/wikis/Config-PipeWire).

Tip
Typically, things work reasonably well out of the box, and PipeWire's system-level configuration is usually best left unmodified.

Audio Groups

PipeWire's default configuration tries to use *realtime* scheduling to increase audio thread priorities. It's recommended that users are in the **pipewire** group. If the user doesn't have the necessary permissions for this, the configuration will try to use RTKit instead, so the package may need to be installed. This behavior is defined under the `context.modules` portion of PipeWire's configuration.

```
root # usermod -aG pipewire larry
root # emerge --ask sys-auth/rtkit
```

For the best experience with fast user switching, it is recommended that you remove your user from the **audio** group unless you rely on the **audio** group for device access control or ACLs:

```
root # usermod -rG audio larry
```

PipeWire also recognizes multiple environment variables that allow these settings to be changed, per-user, or for individual commands.

No additional configuration is required to use PipeWire as video server; this functionality is enabled by default.

File locations

The default configuration should be fine for most users, but if configuration is required, one can find PipeWire's configuration files at:

1. **`~/.config/pipewire/pipewire.conf`** - User PipeWire configuration
2. **`/etc/pipewire/pipewire.conf`** - System PipeWire configuration
3. **`/usr/share/pipewire/pipewire.conf`** - Sample PipeWire configuration

> **Note**
> The PipeWire configuration files do not exist by default, they must be created by copying the sample configuration.

If customization is required, run the following:

```
root # cp /usr/share/pipewire/pipewire.conf /etc/pipewire/pipewire.conf
user $ cp /usr/share/pipewire/pipewire.conf ~/.config/pipewire/pipewire.conf
```

If not, just leave it alone.

Configuration fragments

Since *0.3.45*, fragments of the config file can be copied to a file (with a .conf extension) in the following directories[2]:

1. **`/usr/share/pipewire/pipewire.conf.d/`**
2. **`/etc/pipewire/pipewire.conf.d/`**
3. **`~/.config/pipewire/pipewire.conf.d/`**

context.properties

The following configuration options are applied under `context.properties` in PipeWire's configuration

Sample rates

PipeWire uses a global sample rate in the audio processing pipeline. All signals are converted to this sample rate and then converted to the sample rate of the device.

FILE `pipewire.conf` **Change the default sample rate to 192000hz**

```
context.properties = {
    default.clock.rate = 192000
    default.clock.allowed-rates = [ 192000 48000 44100 ]  # Up to 16 can be specified
}
```

> **Tip**
> Setting `default.clock.allowed-rates` to contain rates the output device supports eliminates the need to resample.[3] This not enabled by default because of kernel driver bugs (https://gitlab.freedesktop.org/pipewire/pipewire/-/issues/1547) in kernels before 5.16.

Sound Server Configuration

> **Tip**
> SDDM users should make sure to start it up via the proper service (/wiki/SDDM#Service) or XDG_RUNTIME_DIR may not be set up correctly.

systemd

PipeWire provides socket and service files when built with the `systemd` USE flag.

If the PulseAudio user service is enabled, disable it; this is safe to do even if the user service was not in use.

```
user $ systemctl --user disable --now pulseaudio.socket pulseaudio.service
```

While PipeWire does not appear to utilize the **~/.config/pulse/** directory beyond the cookie file, it may be a good idea to rename or delete it.

Enable the `wireplumber` service and the `pipewire-pulse` socket; enabling the `pipewire-pulse` socket will cause the `pipewire-pulse` service to be started if required. That service will in turn start the `pipewire` service.

```
user $ systemctl --user enable --now pipewire-pulse.socket wireplumber.service
```

Socket activation means the `pipewire` service will only be started when required, which is usually sufficient. However, the `pipewire` service can be always started when the user logs in by enabling pipewire.service:

```
user $ systemctl --user enable --now pipewire.service
```

In these cases, the `--now` flag is optional, but probably safe to use, as starting PipeWire with default configuration merely allows using new interfaces and doesn't change the existing ones, i.e. non-PipeWire clients continue using the same libraries and services they were using previously.

OpenRC

There is no standardized non-systemd way to start PipeWire services - **pipewire**, **pipewire-pulse** and **wireplumber** - when starting a graphical shell, and users need to choose the correct approach based on how their graphical environment is started.

OpenRC 0.60 introduced support for user services, and eventually PipeWire services will be provided via that functionality; however, as of 2025-11-22, **gentoo-pipewire-launcher** should continue to be used.

> **Important**
> PipeWire must be started before anything that might try to connect to any sound input or output, such as a volume monitoring applet.

Prerequisites

- *XDG_RUNTIME_DIR*

This variable is usually set automatically by systemd-logind or elogind (/wiki/Elogind). seatd (/wiki/Seatd) on the other hand does need some manual steps (https://man.sr.ht/~kennylevinsen/seatd/). Users **only need to set it manually** if not using a session/seat manager. Those systems like the ones using seatd or those that don't use any of these seat management daemons will need to set *XDG_RUNTIME_DIR* manually, e.g. in **~/.bash_profile**, **~/.zprofile**, etc.

FILE

```
# Ensure XDG_RUNTIME_DIR is set
if test -z "$XDG_RUNTIME_DIR"; then
    export XDG_RUNTIME_DIR=$(mktemp -d /tmp/$(id -u)-runtime-dir.XXX)
fi
```

- *DBUS_SESSION_BUS_ADDRESS*

This variable is usually set automatically by desktop environments such as GNOME or KDE. Note that a D-Bus session bus is *not* what is typically being provided by the system D-Bus service, which is instead providing a *system* bus for things like hardware events, etc.

When running a window manager, such as i3, bspwm or Sway, the WM needs to be started with **dbus-launch**, which will set *DBUS_SESSION_BUS_ADDRESS* for the session. For example, if starting X via **startx**, the final line of **~/.xinitrc** could be:

FILE **~/.xinitrc**

```
exec dbus-launch --exit-with-session bspwm
```

Note that use of **exec** in this context means that *no subsequent commands in that file will be executed*, as the shell process interpreting the file will be replaced by the **dbus-launch** process. Any commands needing to be run from within the new session should be added to the WM's configuration/setup file, e.g. **~/.config/bspwm/bspwmrc**.

More generally, **dbus-launch** can be run directly from the command line:

```
user $ dbus-launch --exit-with-session sway
```

gentoo-pipewire-launcher

gentoo-pipewire-launcher is a convenience script for systems not running systemd, e.g. OpenRC systems. It will only be installed if the *systemd* USE flag is not enabled.

> **Note**
> **gentoo-pipewire-launcher** is a temporary solution for OpenRC-based systems. Eventually, once OpenRC user services (introduced in OpenRC 0.60) are well-established, **gentoo-pipewire-launcher** might be split into a separate package or removed altogether.

As documented in the gentoo-pipewire-launcher(1) man page, the **gentoo-pipewire-launcher** script starts:

- a PipeWire server;
- a WirePlumber session manager, required to make use of PipeWire servers;
- a **pipewire-pulse** PipeWire server, required for PulseAudio compatibility.

Before doing so, the script terminates any existing PipeWire or WirePlumber instances. For further details, refer to the comments in the script.

gentoo-pipewire-launcher should be started in an environment which has the *DBUS_SESSION_BUS_ADDRESS* environment variable set appropriately, i.e. within the context of the program started by **dbus-launch** or **dbus-run-session**.

gentoo-pipewire-launcher supports logging, via **${XDG_CONFIG_HOME}/gentoo-pipewire-launcher.conf**. The variables *GENTOO_PIPEWIRE_LOG*, *GENTOO_PIPEWIRE_PULSE_LOG*, and *GENTOO_WIREPLUMBER_LOG* can be used to specify the absolute path of a file to which logs should be written. If these variables are not set, log output will go to **/dev/null**.

gentoo-pipewire-launcher sources the **gentoo-pipewire-launcher.conf** file, such that the conf file can be used to add variables (e.g. *PIPEWIRE_DEBUG=4*) to the environment of the PipeWire and WirePlumber processes it starts.

GUI environments (Desktop Environments and Window Managers)

Gentoo's PipeWire package installs the **/etc/xdg/autostart/pipewire.desktop** autostart file. However, not all GUI environments make use of autostart files: Plasma, GNOME, XFCE and Cinnamon do, but various window managers (such as Fluxbox) do not. Environments which make use of autostart files *must not* start PipeWire via **~/.xprofile**; environments not making use of autostart files must start PipeWire by calling **gentoo-pipewire-launcher** from either the environment's startup file (see below), or **~/.xprofile** (creating that file if necessary):

FILE **~/.xprofile**

```
gentoo-pipewire-launcher &
```

Wayland / Sway / Window Managers

The XDG autostart approach should work for Wayland (/wiki/Wayland) as well as X, but it is not clear whether **.xprofile** is necessarily sourced when starting a Wayland session.

> **Note**
> `gentoo-pipewire-launcher` has a `restart` option which will shutdown any running instances of PipeWire and start a new one; this means that any app using sound will stutter. It's okay to use the `restart` option here as long as we run it with Sway's `exec` command and not the `exec_always` command. `exec` prevents sound stuttering when reloading Sway. `restart` prevents PipeWire failing to start after Sway exits and runs again.

For Sway (/wiki/Sway), edit **~/.config/sway/config** to add:

FILE **~/.config/sway/config**

```
exec gentoo-pipewire-launcher restart &
```

For dwm (/wiki/Dwm), edit **~/.dwm/dwmrc** to add:

FILE **~/.dwm/dwmrc**

```
gentoo-pipewire-launcher &
```

For wayfire (/wiki/Wayfire), edit **~/.config/wayfire.ini** to add:

FILE **~/.config/wayfire.ini**

```
[autostart]
pipewire = gentoo-pipewire-launcher
```

More generally, when using a window manager (/wiki/Window_manager), a call to **gentoo-pipewire-launcher** needs to be added to the WM's configuration/setup file, e.g.:

FILE **~/.config/i3/config**

```
exec gentoo-pipewire-launcher &
```

Restarting PipeWire and WirePlumber

To restart PipeWire and WirePlumber under OpenRC, e.g. to pick up configuration changes, run **gentoo-pipewire-launcher** with the **restart** argument to have it first shut down the existing instances from within the relevant D-Bus session:

```
user $ nohup gentoo-pipewire-launcher restart &
```

Using **nohup** allows the terminal to be closed without terminating **gentoo-pipewire-launcher**. Where output is directed might depend on the implementation of **nohup**; depending on the shell, **nohup** might be a builtin or an external command (e.g. `nohup(1)` (https://man.archlinux.org/man/nohup.1.en) (/wiki/Special:MyLanguage/man_page)), so check the shell's documentation.

Note that due to this wireplumber bug (https://gitlab.freedesktop.org/pipewire/wireplumber/-/issues/505), **nohup** may not work, so you may need to use an alternative. In Bash, for instance, you can omit **nohup**, and then run **disown** right afterwards:

```
user $ gentoo-pipewire-launcher restart &
user $ disown
```

User Services

> **Warning**
> This method is still in development. Unless you are an expert or want to experiment, please use the **gentoo-pipewire-launcher** method for OpenRC systems for now. See `bug #964059 (https://bugs.gentoo.org/show_bug.cgi?id=964059)` for a discussion of pending work.

OpenRC has built-in and enabled by default support for user services since version 0.60 OpenRC#User_services (/wiki/OpenRC#User_services). Similarly to systemd, they can be used to automatically launch and stop PipeWire on login and logout.

To enable the PipeWire services run:

```
user $ rc-update add -U pipewire default
user $ rc-update add -U pipewire-pulse default
user $ rc-update add -U wireplumber default
```

To start the services without enabling them:

```
user $ rc-service --user pipewire start
user $ rc-service --user pipewire-pulse start
user $ rc-service --user wireplumber start
```

Verifying PulseAudio server emulation

```
user $ LANG=C pactl info | grep "Server Name"
```

```
Server Name: PulseAudio (on PipeWire 0.3.39)
```

Adding multi user support

The default configuration only allows the current user (who started **pipewire-pulse**) to play audio.

For multi-user support, the TCP interface is needed:

```
root # cp -r /usr/share/pipewire/ /etc/
root # $EDITOR /etc/pipewire/pipewire-pulse.conf
```

Locate `server.address = [` section and uncomment one of the available TCP options.

Usage

Controlling the volume

Pipewire volume can be controlled using Pulseaudio tools like **`pactl`** (`media-libs/libpulse` (https://packages.gentoo.org/packages/media-libs/libpulse)). It can also be controlled without depending on Pulseaudio:

Using **`wpctl`** (`media-video/wireplumber` (https://packages.gentoo.org/packages/media-video/wireplumber)). Example: to increase volume by 2%:

```
user $ wpctl set-volume @DEFAULT_AUDIO_SINK@ 2%+
```

Or using **`pw-cli`** (`media-video/pipewire` (https://packages.gentoo.org/packages/media-video/pipewire)):[4]

1. Find the stream node id:

```
user $ pw-cli ls Node
```

2. Change the needed properties (To introspect the properties of a node, run **`pw-cli e <node-id> Props`**). Example: to unmute and set volume to 0.3:

```
user $ pw-cli s <node-id> Props '{ mute: false, channelVolumes: [ 0.3, 0.3 ] }'
```

Checking the sample rate

`pw-metadata` can be used to check the current sample rate, along with other configuration:

```
user $ pw-metadata -n settings
```

```
Found "settings" metadata 31
update: id:0 key:'log.level' value:'2' type:''
update: id:0 key:'clock.rate' value:'192000' type:''
update: id:0 key:'clock.allowed-rates' value:'[ 192000, 48000, 44100 ]' type:''
update: id:0 key:'clock.quantum' value:'1024' type:''
update: id:0 key:'clock.min-quantum' value:'32' type:''
update: id:0 key:'clock.max-quantum' value:'2048' type:''
update: id:0 key:'clock.force-quantum' value:'0' type:''
update: id:0 key:'clock.force-rate' value:'0' type:''
```

PulseAudio Server

If Pipewire is being used as a PulseAudio backend, the sample rate and bit depth, or `Default Sample Specification` , can be checked with:

```
user $ pactl info
```

```
Server String: /run/user/1000/pulse/native
Library Protocol Version: 35
Server Protocol Version: 35
Is Local: yes
Client Index: 213
Tile Size: 65472
User Name: larry
Host Name: gentoo
Server Name: PulseAudio (on PipeWire 0.3.71)
Server Version: 15.0.0
Default Sample Specification: float32le 2ch 192000Hz
Default Channel Map: front-left,front-right
Default Sink: alsa_output.usb-Generic_USB_Audio-00.pro-output-2
Default Source: alsa_input.usb-Focusrite_Scarlett_Solo_USB-00.pro-input-0
```

Setting the Sample Rate at Runtime

`pw-metadata -n settings 0 clock.rate` can be used to adjust the clock rate at runtime:

```
user $ pw-metadata -n settings 0 clock.rate 384000
```

```
Found "settings" metadata 31
set property: id:0 key:clock.rate value:384000 type:(null)
```

Tip
Results can be verified with **`pw-metadata -n settings`**, and this procedure can be used for other config values.

GUI patchbays

Any of the patchbays are great to use:

`media-sound/helvum` (https://packages.gentoo.org/packages/media-sound/helvum) is a GTK-based pipewire patchbay. Simple and easy to use.

`media-sound/qpwgraph` (https://packages.gentoo.org/packages/media-sound/qpwgraph) is a QT-based pipewire patchbay. Can save layout. Monitor-streams are not joint with original objects, since sometimes not intuitive. Yet simple and easy to use.

coppwr (https://github.com/dimtpap/coppwr) advertise itself as low-level pipewire patchbay. Seems to be very interesting to use. If you used helvum or qpwgraph, this can be relatively easy to use. Currently there's only flatpak package for gentoo. To install, get Flatpak (/wiki/Flatpak#Installation) installed, then:

```
user $ flatpak install io.github.dimtpap.coppwr
```

Tip
If you are using KDE's volume applet and if you get a lot of useless objects like `Plasma PA*` which do not disappear after closing the menu at any of the patchbays, then just open and close a few times the menu for volume. Usually it removes all of such at an attempt.

Streaming audio through RTP at network using pipewire native module

Check next documentation (https://gitlab.freedesktop.org/pipewire/pipewire/-/wikis/Guide-Network-RTP) for that. Start the listener, then start broadcaster. At config for listener change 0.0.0.0 to source ip-address, for broadcaster config change to destination ip-address too.

Then use any patchbay to sink prefered audio stream to a new sink called **`localhost.localdomain`** with your names. Do the same for listener: use previously mentioned tools for streaming to prefered device.

If you got error code 69 at restarting pipewire, it seems config is bad. If error code is 70, it seems that it tried to create RTP connection and failed. Check whether ip-addresses and ports at configs are right.

Replacing JACK

To have JACK clients routed through PipeWire, currently the only method supported by Gentoo is to run them via pw-jack which uses *LD_PRELOAD* to redirect clients from JACK's original libraries to PipeWire's alternatives:

```
user $ pw-jack qsynth
```

Important
Existing JACK users are likely to have realtime capability set up but new users are advised to raise the *RLIMIT_MEMLOCK* value from Gentoo's default of 64 kilobytes to 256 kilobytes on all PipeWire users that want to use its JACK emulation. For instructions on how to achieve that please see the subsection on that below (also listed in the table of contents for this page). Failure to do this will likely cause at least occasional buffer underuns (xruns) as a single page fault is likely to spend half to the entire length of a buffer just in kernel time to resolve.

It should be noted that either due to PipeWire incompleteness or Gentoo configuration shortcomings, not every client will work. Some may even ungracefully exit due to missing symbols. It will likely also require re-configuration of JACK clients, because they will attempt to use their old configuration files, if such exist.

Alternatively it should be possible to have PipeWire connect to a real jackd and act as a gateway for non-JACK applications but, unless there already is a working JACK setup, this is not recommended for the overall worse user experience with JACK.

Stuttering recordings in applications like `media-sound/ardour` (https://packages.gentoo.org/packages/media-sound/ardour) might be resolved using a

WirePlumber script:

FILE **~/.config/wireplumber/main.lua.d/latency.lua**

```
table.insert(alsa_monitor.rules, {
  matches = {
    {
      -- replace device as described below
      { "node.name", "equals", "device" },
    },
  },
  apply_properties = {
    -- If 64 doesn't work, try bigger values that are a power of 2 (128, 256, 512, 1024, 2048,
etc.)
    ["api.alsa.headroom"] = 64,
  },
})
```

Executing:

user $ `wpctl status`
outputs the list of devices currently available on the system. The audio device should appear under Audio -> Sinks (unsure which one to pick? Use the one with the star (*) at the beginning. It represents the currently used device). Remember this number and execute:

user $ `wpctl inspect <number> | grep node.name`
This should output one line, for example `* node.name = "alsa_output.usb-Lenovo_ThinkPad.analog-stereo"`. Replace `device` in the script with the device from the output (in this example it would be `alsa_output.usb-Lenovo_ThinkPad.analog-stereo`). The problem should be resolved after restarting PipeWire. If the issue is not resolved, more information can be found here (https://gitlab.freedesktop.org/pipewire/pipewire/-/issues/2257) or here (https://forum.manjaro.org/t/howto-troubleshoot-crackling-in-pipewire/82442).

Troubleshooting

Screensharing doesn't work with Chrome

Change "WebRTC PipeWire support" to "Enabled" from chrome://flags.

Increasing RLIMIT_MEMLOCK for JACK clients (and PulseAudio clients with OpenRC)

Unlike the case of PulseAudio clients, for which the user service does memory locking of buffers, JACK clients do it themselves. In the likely event that the clients report being unable to lock memory, in addition to the hard limit (max permitted value) described in previous sub-section on PulseAudio RLIMIT_MEMLOCK, the soft limit (the default value) must also be raised:

FILE **/etc/security/limits.d/50-custom.conf**

```
# This both raises the max and sets the default lockable memory limit of every process running
under a non-system account (except for nobody) from default 64 to 256 kilobytes (in increments
of ''page size'')
1000:65533      -    memlock 256
```

Bluetooth

Important
PipeWire should already be enabling the bluez5 module when built with the `pulseaudio` *USE* flag. Please check if it's being set system-wide as PulseAudio API for clients is not going anywhere in a hurry.

By default, Bluetooth (/wiki/Bluetooth) support is disabled to avoid clashes with PulseAudio's Bluetooth stack since currently the main use case is as an addition to, not a replacement for PA. Uncomment PipeWire's bluez5 module for Bluetooth devices to be listed:

FILE **/etc/pipewire/media-session.d/media-session.conf**

```
...
modules = {
    ...
    default = [
        ...
        bluez5          # bluetooth support
        ...
    ]
    ...
```

Then run:

```
user $ systemctl --user restart pipewire-pulse.service
```

Restarting PipeWire after crash (with OpenRC)

When PipeWire crashes, all sound devices disappear. Three processes need to be created:

1. **pipewire**
2. **wireplumber**
3. **pipewire -c pipewire-pulse.conf**

Do not run these directly. Instead, start them with:

```
user $ /usr/bin/gentoo-pipewire-launcher &
```

from the environment with a running D-Bus session bus (e.g. the desktop). An environment with a D-Bus session bus will have the *DBUS_SESSION_BUS_ADDRESS* variable set; if it is not set, refer to the OpenRC (/wiki/PipeWire#OpenRC) section for how to ensure a D-Bus session bus is running for the environment.

Crackling and stuttering

Crackling and stuttering might be reduced or eliminated by setting *default.clock.min-quantum* appropriately in **pipewire.conf**[5]:

FILE **/etc/pipewire/pipewire.conf**

```
default.clock.min-quantum = 2048
```

Stuttering on a VM

Stuttering on a VM may be caused by insufficient headroom for alsa in wireplumber. By default on a vm the headroom is increased to 2048 in **/usr/share/wireplumber/wireplumber.conf.d/alsa-vm.conf**

It might be eliminated by raising it further to 8096. [6]

FILE **/etc/wireplumber/wireplumber.conf.d/30-alsa.conf**

```
monitor.alsa.rules = [
  {
    matches = [
      # This matches the value of the 'node.name' property of the node.
      {
        node.name = "~alsa_output.*"
      }
    ]
    actions = {
      # Apply all the desired node specific settings here.
      update-props = {
        api.alsa.period-size   = 1024
        api.alsa.headroom      = 8192
      }
    }
  }
]
```

Disabling or delaying audio sink suspension

By default, audio devices are put into standby after 5 seconds of no audio. This can be annoying, because some devices take a few seconds to start up again. With `media-video/wireplumber` (https://packages.gentoo.org/packages/media-video/wireplumber), the delay can be configured or the functionality can be disabled completely.

To make it simple, this user config applies to all audio sources and audio sinks. `session.suspend-timeout-seconds = 0` disables the suspend functionality, alternatively the delay can also be increased.

FILE **`~/.config/wireplumber/wireplumber.conf.d/51-disable-suspension.conf`**

```
monitor.alsa.rules = [
  {
    matches = [
      {
        # Matches all sources
        node.name = "~alsa_input.*"
      },
      {
        # Matches all sinks
        node.name = "~alsa_output.*"
      }
    ]
    actions = {
      update-props = {
        session.suspend-timeout-seconds = 0
      }
    }
  }
]
# bluetooth devices
monitor.bluez.rules = [
  {
    matches = [
      {
        # Matches all sources
        node.name = "~bluez_input.*"
      },
      {
        # Matches all sinks
        node.name = "~bluez_output.*"
      }
    ]
    actions = {
      update-props = {
        session.suspend-timeout-seconds = 0
      }
    }
  }
]
```

When using `systemd`, restart `pipewire.service` and `pipewire-pulse.service` afterwards for the setting to take effect or alternatively reboot.

```
user $ systemctl restart --user pipewire.service pipewire-pulse.service
```

Dummy Output with PipeWire running

If the `pipewire` and `wireplumber` services are running and PipeWire is not detecting your audio input/output devices, only *Dummy Output* will show up as an output device. This may be because ACL is missing. Make sure the `acl (https://packages.gentoo.org/useflags/acl)` (`/wiki/USE_flag`) USE flag is not disabled in **`/etc/portage/make.conf`** and rerun emerge if a change was made.

See also

- PulseAudio (/wiki/PulseAudio) — a multi-platform, open source, *sound server* that provides a number of features on top of the low-level audio interface ALSA (/wiki/ALSA)
- ALSA (/wiki/ALSA) — describes the setup of a sound card with **ALSA** (**A**dvanced **L**inux **S**ound **A**rchitecture).
- Technical notes on the packaging of PipeWire (/wiki/User:Sam/PipeWire_changes)

External resources

- Pipewire Guide (https://github.com/mikeroyal/PipeWire-Guide/blob/main/README.md)
- PipeWire FAQ (https://gitlab.freedesktop.org/pipewire/pipewire/-/wikis/FAQ)
- Introduction to Pipewire (https://fedoramagazine.org/introduction-to-pipewire/) (2025-02-07)
- An introduction to PipeWire (https://bootlin.com/blog/an-introduction-to-pipewire/) (2022-06-20)
- PipeWire under the hood (https://venam.nixers.net/blog/unix/2021/06/23/pipewire-under-the-hood.html) (2021-06-23) - blog post explaining PipeWire from a unique perspective
- PipeWire: The Linux audio/video bus (https://lwn.net/Articles/847412/), on Linux Weekly News (2021-03-02)

References

1. https://www.gentoo.org/support/news-items/2022-07-29-pipewire-sound-server.html (https://www.gentoo.org/support/news-items/2022-07-29-pipewire-sound-server.html)
2. https://gitlab.freedesktop.org/pipewire/pipewire/-/wikis/Config-PipeWire#configuration-file-pipewireconf (https://gitlab.freedesktop.org/pipewire/pipewire/-/wikis/Config-PipeWire#configuration-file-pipewireconf)
3. PipeWire's Issue 1523 (https://gitlab.freedesktop.org/pipewire/pipewire/-/issues/1523)
4. https://gitlab.freedesktop.org/pipewire/pipewire/-/wikis/Migrate-PulseAudio#sink-inputsource-output-volumemute (https://gitlab.freedesktop.org/pipewire/pipewire/-/wikis/Migrate-PulseAudio#sink-inputsource-output-volumemute)
5. Post on Gentoo forums (https://forums.gentoo.org/viewtopic-p-8819425.html#8819425). Retrieved on 2024-03-13.
6. Pipewire troubleshooting instructions (https://gitlab.freedesktop.org/pipewire/pipewire/-/wikis/Troubleshooting#stuttering-audio-in-virtual-machine). Retrieved on 2024-09-02

Retrieved from "https://wiki.gentoo.org/index.php?title=PipeWire&oldid=1420907 (https://wiki.gentoo.org/index.php?title=PipeWire&oldid=1420907)"

- This page was last edited on 21 November 2025, at 22:29.
- Privacy policy (/wiki/Gentoo_Wiki:Privacy_policy)
- About Gentoo Wiki (/wiki/Gentoo_Wiki:About)
- Disclaimers (/wiki/Gentoo_Wiki:General_disclaimer)

© 2001–2025 Gentoo Authors
Gentoo is a trademark of the Gentoo Foundation, Inc. and of Förderverein Gentoo e.V. The contents of this document, unless otherwise expressly stated, are licensed under the CC-BY-SA-4.0 (https://creativecommons.org/licenses/by-sa/4.0/) license. The Gentoo Name and Logo Usage Guidelines (https://www.gentoo.org/inside-gentoo/foundation/name-logo-guidelines.html) apply.