Arch Linux as a lightweight audio platform

David Runge

Linux Audio Conference 2015

10.04.2015

What, Where, Why?

- “Arch Linux exposes the user to the system without hiding any details.” [man 7 archlinux]

What, Where, Why?

- “Arch Linux exposes the user to the system without hiding any details.” [man 7 archlinux]
- https://www.archlinux.org

What, Where, Why?

- “Arch Linux exposes the user to the system without hiding any details.” [man 7 archlinux]
- https://www.archlinux.org
- 36 developers, 40 trusted users, 8 support staff

What, Where, Why?

- “Arch Linux exposes the user to the system without hiding any details.” [man 7 archlinux]
- https://www.archlinux.org
- 36 developers, 40 trusted users, 8 support staff
- ABS & AUR

What, Where, Why?

- “Arch Linux exposes the user to the system without hiding any details.” [man 7 archlinux]
- https://www.archlinux.org
- 36 developers, 40 trusted users, 8 support staff
- ABS & AUR
- pacman/ aura

What, Where, Why?

- “Arch Linux exposes the user to the system without hiding any details.” [man 7 archlinux]
- https://www.archlinux.org
- 36 developers, 40 trusted users, 8 support staff
- ABS & AUR
- pacman/ aura
- Archiso

What, Where, Why?

- “Arch Linux exposes the user to the system without hiding any details.” [man 7 archlinux]
- https://www.archlinux.org
- 36 developers, 40 trusted users, 8 support staff
- ABS & AUR
- pacman/ aura
- Archiso
- rolling release, systemd

What, Where, Why?

- “Arch Linux exposes the user to the system without hiding any details.” [man 7 archlinux]
- https://www.archlinux.org
- 36 developers, 40 trusted users, 8 support staff
- ABS & AUR
- pacman/ aura
- Archiso
- rolling release, systemd
- lightweight, high customizability, easy build system

Preparations

Note: *The following assumes you also configure, what you install!*

- Choose supported hardware!!!

Preparations

Note: *The following assumes you also configure, what you install!*

- Choose supported hardware!!!
- Install Archlinux

Preparations

Note: *The following assumes you also configure, what you install!*

- Choose supported hardware!!!
- Install Archlinux
- Install cpupower

Preparations

Note: *The following assumes you also configure, what you install!*

- Choose supported hardware!!!
- Install Archlinux
- Install cpupower
- Install linux-rt

Preparations

Note: *The following assumes you also configure, what you install!*

- Choose supported hardware!!!
- Install Archlinux
- Install cpupower
- Install linux-rt
- Install tuna & rt-tests

Preparations

Note: *The following assumes you also configure, what you install!*

- Choose supported hardware!!!
- Install Archlinux
- Install cpupower
- Install linux-rt
- Install tuna & rt-tests
- Install rts & uenv

Preparations

Note: *The following assumes you also configure, what you install!*

- Choose supported hardware!!!
- Install Archlinux
- Install cpupower
- Install linux-rt
- Install tuna & rt-tests
- Install rts & uenv
- Install jack2

Preparations

Note: *The following assumes you also configure, what you install!*

- Choose supported hardware!!!
- Install Archlinux
- Install cpupower
- Install linux-rt
- Install tuna & rt-tests
- Install rts & uenv
- Install jack2
- Install awesome

Preparations

Note: *The following assumes you also configure, what you install!*

- Choose supported hardware!!!
- Install Archlinux
- Install cpupower
- Install linux-rt
- Install tuna & rt-tests
- Install rts & uenv
- Install jack2
- Install awesome
- Boot into realtime kernel

Checks

-
  ```
  # Add your user to the audio group
  gpasswd -a <username> audio
  # Fully log out and log back in again
  ```

Checks

-
  ```
  # Add your user to the audio group
  gpasswd -a <username> audio
  # Fully log out and log back in again
  ```

-
  ```
  # Have a look at your hardware interrupts
  cat /proc/interrupts

  # Find the devices (and sometimes their parent devices),
  # that you will use/ need for your audio setup
  # You can also use your audio card's IRQ for the next test
  ```

Checks

-
  ```
  # Add your user to the audio group
  gpasswd -a <username> audio
  # Fully log out and log back in again
  ```

-
  ```
  # Have a look at your hardware interrupts
  cat /proc/interrupts

  # Find the devices (and sometimes their parent devices),
  # that you will use/ need for your audio setup
  # You can also use your audio card's IRQ for the next test
  ```

-
  ```
  # clone the realtimeconfigquickscan and run it
  git clone https://github.com/raboof/realtimeconfigquickscan
  cd realtimeconfigquickscan
  ./realtimeconfigquickscan
  # Try and meet all criteria
  # (except maybe the one for 'fs.inotify.max_user_watches')
  ```

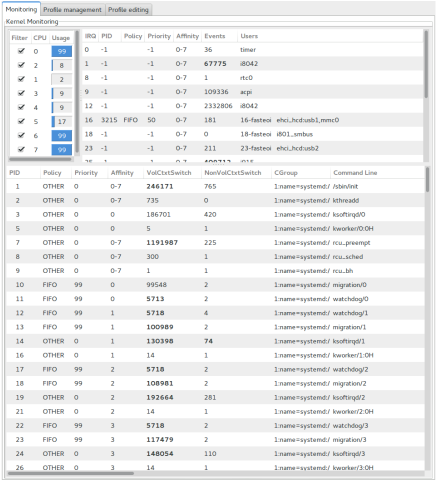
Systemd Conditional “Hook”

Listing 1: Add systemd.setenv=REALTIME=true to your kernel command line

```
# syslinux: /boot/syslinux/syslinux.cfg
[..]
LINUX ../vmlinuz-linux-rt
APPEND root=/dev/<your-root-fs> systemd.setenv=REALTIME=true rw
[..]

# grub: /boot/grub/grub.cfg (or set stuff using /etc/default/grub)
[..]
linux /boot/vmlinuz-linux-rt root=/dev/<your-root-fs> systemd.setenv=REALTIME=true rw
[..]
```

IRQs, cgroups & CPU settings



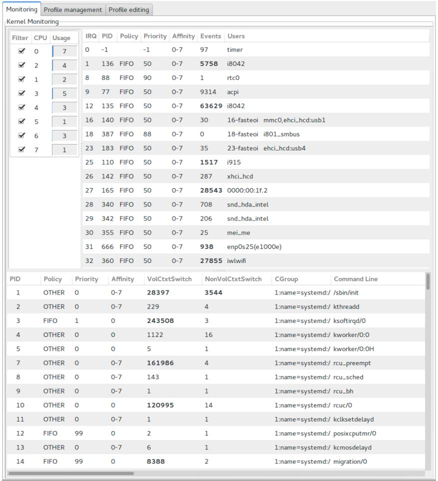
Filter	CPU	Usage
☑	0	99
☑	2	8
☑	1	2
☑	3	9
☑	4	9
☑	5	17
☑	6	99
☑	7	99

IRQ	PID	Policy	Priority	Affinity	Events	Users
0	-1		-1	0-7	36	timer
1	-1		-1	0-7	**67775**	i8042
8	-1		-1	0-7	1	rtc0
9	-1		-1	0-7	109336	acpi
12	-1		-1	0-7	2332806	i8042
16	3215	FIFO	50	0-7	181	16-fasteoi ehci_hcd:usb1,mmc0
18	-1		-1	0-7	0	18-fasteoi i801_smbus
23	-1		-1	0-7	211	23-fasteoi ehci_hcd:usb2
25	-1		-1	0-7	**408712**	i915

PID	Policy	Priority	Affinity	VolCtxtSwitch	NonVolCtxtSwitch	CGroup	Command Line
1	OTHER	0	0-7	**246171**	765	1:name=systemd:/	/sbin/init
2	OTHER	0	0-7	735	0	1:name=systemd:/	kthreadd
3	OTHER	0	0	186701	420	1:name=systemd:/	ksoftirqd/0
5	OTHER	0	0	5	1	1:name=systemd:/	kworker/0:0H
7	OTHER	0	0-7	**1191987**	225	1:name=systemd:/	rcu_preempt
8	OTHER	0	0-7	300	1	1:name=systemd:/	rcu_sched
9	OTHER	0	0-7	1	1	1:name=systemd:/	rcu_bh
10	FIFO	99	0	99548	2	1:name=systemd:/	migration/0
11	FIFO	99	0	**5713**	2	1:name=systemd:/	watchdog/0
12	FIFO	99	1	**5718**	4	1:name=systemd:/	watchdog/1
13	FIFO	99	1	**100989**	2	1:name=systemd:/	migration/1
14	OTHER	0	1	**130398**	**74**	1:name=systemd:/	ksoftirqd/1
16	OTHER	0	1	14	1	1:name=systemd:/	kworker/1:0H
17	FIFO	99	2	**5718**	2	1:name=systemd:/	watchdog/2
18	FIFO	99	2	**108981**	2	1:name=systemd:/	migration/2
19	OTHER	0	2	**192664**	281	1:name=systemd:/	ksoftirqd/2
21	OTHER	0	2	14	1	1:name=systemd:/	kworker/2:0H
22	FIFO	99	3	**5718**	2	1:name=systemd:/	watchdog/3
23	FIFO	99	3	**117479**	2	1:name=systemd:/	migration/3
24	OTHER	0	3	**148054**	110	1:name=systemd:/	ksoftirqd/3
26	OTHER	0	3	14	1	1:name=systemd:/	kworker/3:0H

Figure: Sample tuna display of IRQs and kthreads on linux

IRQs, cgroups & CPU settings



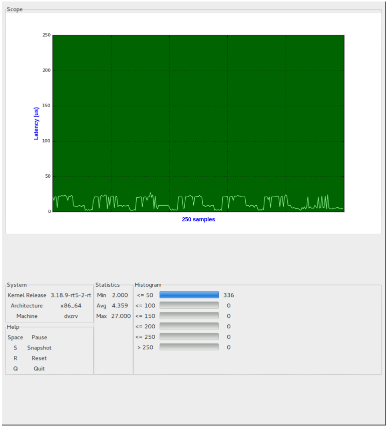
Filter	CPU	Usage
☑	0	7
☑	2	4
☑	1	2
☑	3	5
☑	4	3
☑	5	1
☑	6	3
☑	7	1

IRQ	PID	Policy	Priority	Affinity	Events	Users
0	-1		-1	0-7	97	timer
1	136	FIFO	50	0-7	**5758**	i8042
8	88	FIFO	90	0-7	1	rtc0
9	77	FIFO	50	0-7	9314	acpi
12	135	FIFO	50	0-7	**63629**	i8042
16	140	FIFO	50	0-7	30	16-fasteoi mmc0,ehci_hcd:usb1
18	387	FIFO	88	0-7	0	18-fasteoi i801_smbus
23	183	FIFO	50	0-7	35	23-fasteoi ehci_hcd:usb4
25	110	FIFO	50	0-7	**1517**	i915
26	142	FIFO	50	0-7	287	xhci_hcd
27	165	FIFO	50	0-7	**28543**	0000:00:1f.2
28	340	FIFO	50	0-7	708	snd_hda_intel
29	342	FIFO	50	0-7	206	snd_hda_intel
30	355	FIFO	50	0-7	25	mei_me
31	666	FIFO	50	0-7	**938**	enp0s25(e1000e)
32	360	FIFO	50	0-7	**27855**	iwlwifi

PID	Policy	Priority	Affinity	VolCtxtSwitch	NonVolCtxtSwitch	CGroup	Command Line
1	OTHER	0	0-7	**28397**	**3544**	1:name=systemd:/	/sbin/init
2	OTHER	0	0-7	229	4	1:name=systemd:/	kthreadd
3	FIFO	1	0	**243508**	3	1:name=systemd:/	ksoftirqd/0
4	OTHER	0	0	1122	16	1:name=systemd:/	kworker/0:0
5	OTHER	0	0	5	1	1:name=systemd:/	kworker/0:0H
7	OTHER	0	0-7	**161986**	4	1:name=systemd:/	rcu_preempt
8	OTHER	0	0-7	143	1	1:name=systemd:/	rcu_sched
9	OTHER	0	0-7	1	1	1:name=systemd:/	rcu_bh
10	OTHER	0	0	**120995**	14	1:name=systemd:/	rcuc/0
11	OTHER	0	0-7	1	1	1:name=systemd:/	kclksetdelayd
12	FIFO	99	0	2	1	1:name=systemd:/	posixcputmr/0
13	OTHER	0	0-7	6	1	1:name=systemd:/	kcmosdelayd
14	FIFO	99	0	**8388**	2	1:name=systemd:/	migration/0

Figure: Sample tuna display of IRQs and kthreads on linux-rt

Cyclictest and oscilloscope



System

Kernel Release	3.18.9-rt5-2-rt
Architecture	x86_64
Machine	dvzrv

Help

Space	Pause
S	Snapshot
R	Reset
Q	Quit

Statistics

Min	2.000
Avg	4.359
Max	27.000

Histogram

<= 50	336
<= 100	0
<= 150	0
<= 200	0
<= 250	0
> 250	0

Figure: Example Oscilloscope output (being fed by cyclictest)

Listing 2: Cyclictest feeding oscilloscope

```
# Start cyclictest (package: rt-tests) and feed it to oscilloscope (package: tuna)
cyclictest --smp -n -p99 -m -v | oscilloscope >/dev/null
```

About

- *systemd is a system and service manager for Linux operating systems. When run as first process on boot (as PID 1), it acts as init system that brings up and maintains userspace services.* (**man 1 init**)

About

- *systemd is a system and service manager for Linux operating systems. When run as first process on boot (as PID 1), it acts as init system that brings up and maintains userspace services.* (**man 1 init**)
- **systemctl** <*start*/*stop*/*enable*/*disable*> *.{service,target,timer}

About

- *systemd is a system and service manager for Linux operating systems. When run as first process on boot (as PID 1), it acts as init system that brings up and maintains userspace services.* (**man 1 init**)
- **systemctl** <*start*/*stop*/*enable*/*disable*> *.{service,target,timer}
- /run/systemd/system/*,
  /usr/lib/systemd/{system,user}/*,
  /etc/systemd/{system,user}/*

About

- *systemd is a system and service manager for Linux operating systems. When run as first process on boot (as PID 1), it acts as init system that brings up and maintains userspace services.* (**man 1 init**)
- **systemctl** <*start*/*stop*/*enable*/*disable*> *.{service,target,timer}
- /run/systemd/system/*,
  /usr/lib/systemd/{system,user}/*,
  /etc/systemd/{system,user}/*
- /etc/systemd/journald.conf,
  /etc/systemd/logind.conf,
  /etc/systemd/system.conf,
  /etc/systemd/user.conf

Realtime

- ~~My Service Can’t Get Realtime!~~

Realtime

- ~~My Service Can't Get Realtime!~~
- Let's ignore dbus as long as it is not kdbus

Realtime

- ~~My Service Can’t Get Realtime!~~
- Let’s ignore dbus as long as it is not kdbus
- *“[. . .] When a process is in a cgroup in the “cpu” controller, and no RT budget is set for that cgroup, then RT is not available to it. This is very unfortunate. I’d love to assign an RT budget by default from systmed, but this isn’t really doable, since there’s no sane RT budget one could assign a cgroup given the current semantics of it (which require that all RT budgets of cgroups within another cgroup must sum up to less than 1/1. . .).*

  THis is something that needs to be cleaned up in the kernel, and then we can expose this nicer in systemd.

  For now, my recommendation would be to disable the RT cgroup stuff in the kernel, and thus forego the whole problem. [. . .]“ ~**Lennart Poettering**

Realtime

- ~~My Service Can't Get Realtime!~~

- Let's ignore dbus as long as it is not kdbus
- *"[. . .] When a process is in a cgroup in the "cpu" controller, and no RT budget is set for that cgroup, then RT is not available to it. This is very unfortunate. I'd love to assign an RT budget by default from systmed, but this isn't really doable, since there's no sane RT budget one could assign a cgroup given the current semantics of it (which require that all RT budgets of cgroups within another cgroup must sum up to less than 1/1. . .).*

  THis is something that needs to be cleaned up in the kernel, and then we can expose this nicer in systemd.

  For now, my recommendation would be to disable the RT cgroup stuff in the kernel, and thus forego the whole problem. [. . .]" ∼**Lennart Poettering**

- *"[. . .] anyone who wants to control cpu cgroups will be required to also become responsible for distributing RT scheduling."* ∼**Tejun Heo**

Realtime

- ~~My Service Can’t Get Realtime!~~
- Let’s ignore dbus as long as it is not kdbus
- *“[. . .] When a process is in a cgroup in the “cpu” controller, and no RT budget is set for that cgroup, then RT is not available to it. This is very unfortunate. I’d love to assign an RT budget by default from systmed, but this isn’t really doable, since there’s no sane RT budget one could assign a cgroup given the current semantics of it (which require that all RT budgets of cgroups within another cgroup must sum up to less than 1/1. . .).*

  THis is something that needs to be cleaned up in the kernel, and then we can expose this nicer in systemd.

  For now, my recommendation would be to disable the RT cgroup stuff in the kernel, and thus forego the whole problem. [. . .]“ ~**Lennart Poettering**
- *“[. . .] anyone who wants to control cpu cgroups will be required to also become responsible for distributing RT scheduling.”* ~**Tejun Heo**
- **ControlGroup**, **ControlGroupAttribute** directives removed in systemd > 205

man 5 systemd.exec

[. . .]

IOSchedulingClass = Sets the IO scheduling class for executed processes. Takes an integer between 0 and 3 or one of the strings none, realtime, best-effort or idle. See ioprio_set(2) for details.

IOSchedulingPriority = Sets the IO scheduling priority for executed processes. Takes an integer between 0 (highest priority) and 7 (lowest priority). The available priorities depend on the selected IO scheduling class (see above). See ioprio_set(2) for details.

CPUSchedulingPolicy = Sets the CPU scheduling policy for executed processes. Takes one of other, batch, idle, fifo or rr. See sched_setscheduler(2) for details.

CPUSchedulingPriority = Sets the CPU scheduling priority for executed processes. The available priority range depends on the selected CPU scheduling policy (see above). For real-time scheduling policies an integer between 1 (lowest priority) and 99 (highest priority) can be used. See sched_setscheduler(2) for details.

CPUSchedulingResetOnFork = Takes a boolean argument. If true, elevated CPU scheduling priorities and policies will be reset when the executed processes fork, and can hence not leak into child processes. See sched$_setscheduler(2)fordetails.Defaultstofalse$.

CPUAffinity = Controls the CPU affinity of the executed processes. Takes a space-separated list of CPU indices. This option may be specified more than once in which case the specified CPU affinity masks are merged. If the empty string is assigned, the mask is reset, all assignments prior to this will have no effect. See sched_setaffinity(2) for details.

[. . .]

LimitCPU=, **LimitFSIZE**=, **LimitDATA**=, **LimitSTACK**=, **LimitCORE**=, **LimitRSS**=, **LimitNOFILE**=, **LimitAS**=, **LimitNPROC**=, **LimitMEMLOCK**=, **LimitLOCKS**=, **LimitSIGPENDING**=, **LimitMSGQUEUE**=, **LimitNICE**=, **LimitRTPRIO**=, **LimitRTTIME**= These settings set both soft and hard limits of various resources for executed processes. See setrlimit(2) for details. Use the string infinity to configure no limit on a specific resource.

[. . .]

Custom JACK systemd –user service

Listing 3: /etc/conf.d/fw1

```
# Sample configuration file for a JACK systemd --user service, using a firewire device
NAME="default"
DRIVER="firewire"
DEVICE="/dev/fw1"
NOMLOCK=""
REALTIME="-R"
PORTMAX=512
UNLOCK="-u"
VERBOSE="-v"
DRIVER_SETTINGS="\
  -n 3 \
  -p 256\
  -r 48000"
```

Listing 4: /usr/lib/systemd/user/jack@.service

```
[Unit]
Description=JACK Audio
After=sound.target local-fs.target

[Service]
EnvironmentFile=-/etc/conf.d/%i
EnvironmentFile=-%h/.config/jack/%i
ExecStart=/usr/bin/jackd -n $NAME $REALTIME -p $PORTMAX -d $DRIVER -d $DEVICE $DRIVER_SETTINGS
CPUSchedulingPolicy=rr
CPUSchedulingPriority=70
LimitRTPRIO=71
LimitRTTIME=-1

[Install]
WantedBy=default.target
```

Conditional cpupower (cpupower-rt)

Listing 5: /usr/lib/systemd/system/cpupower-rt.service (package: uenv)

```
[Unit]
Description=Apply cpupower configuration
ConditionKernelCommandLine=systemd.setenv=REALTIME=true
After=cpupower.service

[Service]
Type=oneshot
ExecStart=/usr/lib/systemd/scripts/cpupower-rt
RemainAfterExit=yes

[Install]
WantedBy=multi-user.target
```

cpupower-rt

Listing 6: /etc/default/cpupower-rt

```
# Define CPUs governor
# valid governors: ondemand, performance, powersave, conservative, userspace.
governor=’performance’

# Limit frequency range
# Valid suffixes: Hz, kHz (default), MHz, GHz, THz
#min_freq="2.25GHz"
#max_freq="3.4GHz"

# Specific frequency to be set.
# Requires userspace governor to be available.
# Do not set governor field if you use this one.
#freq=

# Utilizes cores in one processor package/socket first before processes are
# scheduled to other processor packages/sockets.
# See man (1) CPUPOWER-SET for additional details.
#mc_scheduler=

# Utilizes thread siblings of one processor core first before processes are
# scheduled to other cores. See man (1) CPUPOWER-SET for additional details.
#smp_scheduler=

#  Sets a register on supported Intel processore which allows software to convey
# its policy for the relative importance of performance versus energy savings to
# the  processor. See man (1) CPUPOWER-SET for additional details.
perf_bias=0

# vim:set ts=2 sw=2 ft=sh et:
```

Conditional Compositing

Listing 7: /usr/lib/systemd/user/compton.service

```
[Unit]
Description=Compton X Compositor
After=display-manager.service local-fs.target
ConditionFileIsExecutable=/usr/bin/compton
ConditionKernelCommandLine=!systemd.setenv=REALTIME=true

[Service]
Type=forking
ExecStart=/usr/bin/compton --config %h/.config/compton.conf -b
Restart=always

[Install]
WantedBy=default.target
```

Tuna scripting

With the help of **tuna** and **rts** you can create a customized IRQ and kthread scheduling setup!

Listing 8: /etc/rts/examples/w540-expresscard-firewire

```
# IRQ scheduling

# set priority for hardware clock rtc0
tuna -q 8 -p 90
# set priority for device with irq 18 (firewire_ohci, i801_smbus)
tuna -q 18 -p 85

# kthread scheduling
tuna -t ’irq/*rtc0*’ -p 90
tuna -t ’irq/*i801_smb*’ -p 88
tuna -t ’irq/*firewire*’ -p 86
```

Listing 9: /usr/lib/systemd/system/rts@.service

```
[Unit]
Description=Apply Realtime-Kernel specific settings
After=multi-user.target sound.target
ConditionKernelCommandLine=systemd.setenv=REALTIME=true
ConditionFileIsExecutable=/usr/bin/tuna

[Service]
Type=oneshot
ExecStart=/usr/lib/systemd/scripts/rts %i
RemainAfterExit=true

[Install]
WantedBy=multi-user.target
```

Info

- *“awesome is a highly configurable, next generation framework window manager for X. It is very fast, extensible and licensed under the GNU GPLv2 license.”*

Info

- *“awesome is a highly configurable, next generation framework window manager for X. It is very fast, extensible and licensed under the GNU GPLv2 license.”*
- Using LUA, versatile and complex setups are possible

Info

- *“awesome is a highly configurable, next generation framework window manager for X. It is very fast, extensible and licensed under the GNU GPLv2 license.”*
- Using LUA, versatile and complex setups are possible
- Being keyboard-based this might or might not be what you want

Info

- *"awesome is a highly configurable, next generation framework window manager for X. It is very fast, extensible and licensed under the GNU GPLv2 license."*
- Using LUA, versatile and complex setups are possible
- Being keyboard-based this might or might not be what you want
- Many themes available

Info

- *“awesome is a highly configurable, next generation framework window manager for X. It is very fast, extensible and licensed under the GNU GPLv2 license.”*
- Using LUA, versatile and complex setups are possible
- Being keyboard-based this might or might not be what you want
- Many themes available
- Auto-tiling!

Info

- *"awesome is a highly configurable, next generation framework window manager for X. It is very fast, extensible and licensed under the GNU GPLv2 license."*
- Using LUA, versatile and complex setups are possible
- Being keyboard-based this might or might not be what you want
- Many themes available
- Auto-tiling!
- Integrates well with Desktop/Session Managers

Info

- *"awesome is a highly configurable, next generation framework window manager for X. It is very fast, extensible and licensed under the GNU GPLv2 license."*
- Using LUA, versatile and complex setups are possible
- Being keyboard-based this might or might not be what you want
- Many themes available
- Auto-tiling!
- Integrates well with Desktop/Session Managers
- Extensible via widgets

Info

- *“awesome is a highly configurable, next generation framework window manager for X. It is very fast, extensible and licensed under the GNU GPLv2 license.”*
- Using LUA, versatile and complex setups are possible
- Being keyboard-based this might or might not be what you want
- Many themes available
- Auto-tiling!
- Integrates well with Desktop/Session Managers
- Extensible via widgets
- Well documented, vast wiki and a helpful community

Info

- *“awesome is a highly configurable, next generation framework window manager for X. It is very fast, extensible and licensed under the GNU GPLv2 license.”*
- Using LUA, versatile and complex setups are possible
- Being keyboard-based this might or might not be what you want
- Many themes available
- Auto-tiling!
- Integrates well with Desktop/Session Managers
- Extensible via widgets
- Well documented, vast wiki and a helpful community
- Suited for work with lightweight and realtime environments

Contact

Questions, suggestions, blame?

Mail:	dave@sleepmap.de
XMPP:	dvzrv@sleepmap.de
IRC:	dvzrv@{efnet,freenode,oftc}

Arch Linux as a lightweight audio platform

David Runge

Archlinux

Installation

Config

Tests

Systemd

Realtime

Conditionals

IRQ & kthread scheduling

Awesome

References

Julien Danjou.
Awesome window manager, 2015.
URL http://awesome.naquadah.org/.

FreeDesktop Foundation.
Systemd Index, 2015.
URL http://www.freedesktop.org/software/systemd/man/index.html.

Judd Vinet & Aaron Griffin.
Arch Linux, 2015.
URL https://www.archlinux.org.

JACK Audio Connection Kit.
JACK Audio Connection Kit, 2015.
URL http://jackaudio.org/.

Libre Music Production.
Libre Music Production, 2015.
URL http://libremusicproduction.com/.

David Runge.
rts, 2015.
URL http://sleepmap.de/projects/rts/.

David Runge.
uenv, 2015.
URL http://sleepmap.de/projects/uenv/.

Lana Brindley & Alison Young.
Tuna User Guide, 2015.
URL https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_MRG/1.3/html-single/Tuna_User_Guide/index.html.