tldr pages

Simplified and community driven man pages

adduser

User addition utility.

- Create a new user with a default home directory and prompt the user to set a password:

  ```
  adduser {{username}}
  ```

- Create a new user without a home directory:

  ```
  adduser --no-create-home {{username}}
  ```

- Create a new user with a home directory at the specified path:

  ```
  adduser --home {{path/to/home}} {{username}}
  ```

- Create a new user with the specified shell set as the login shell:

  ```
  adduser --shell {{path/to/shell}} {{username}}
  ```

- Create a new user belonging to the specified group:

  ```
  adduser --ingroup {{group}} {{username}}
  ```

alpine

An email client and Usenet newsgroup program with a pico/nano-inspired interface.

Supports most modern email services through IMAP.

- Open alpine normally:

  ```
  alpine
  ```

- Open alpine directly to the message composition screen to send an email to a given email address:

  ```
  alpine {{email@example.net}}
  ```

- Quit alpine:

  ```
  'q' then 'y'
  ```

apt-cache

Debian and Ubuntu package query tool.

- Search for a package in your current sources:

  ```
  apt-cache search {{query}}
  ```

- Show information about a package:

  ```
  apt-cache show {{package}}
  ```

- Show whether a package is installed and up to date:

  ```
  apt-cache policy {{package}}
  ```

- Show dependencies for a package:

  ```
  apt-cache depends {{package}}
  ```

- Show packages that depend on a particular package:

  ```
  apt-cache rdepends {{package}}
  ```

apt-get

Debian and Ubuntu package management utility.

- Update the list of available packages and versions (it's recommended to run this before other `apt-get` commands):

  ```
  apt-get update
  ```

- Install a package, or update it to the latest available version:

  ```
  apt-get install {{package}}
  ```

- Remove a package:

  ```
  apt-get remove {{package}}
  ```

- Upgrade all installed packages to their newest available versions:

  ```
  apt-get upgrade
  ```

- Remove all packages that are no longer needed:

  ```
  apt-get autoremove
  ```

- Upgrade installed packages (like `upgrade`), but remove obsolete packages and install additional packages to meet new dependencies:

  ```
  apt-get dist-upgrade
  ```

apt-key

Key management utility for the APT Package Manager on Debian and Ubuntu.

- List trusted keys:

  ```
  apt-key list
  ```

- Add a key to the trusted keystore:

  ```
  apt-key add {{public_key_file.asc}}
  ```

- Delete a key from the trusted keystore:

  ```
  apt-key del {{key_id}}
  ```

- Add a remote key to the trusted keystore:

  ```
  wget -qO - {{https://host.tld/filename.key}} | apt-key add -
  ```

- Add a key from keyserver with only key id:

  ```
  apt-key adv --keyserver {{pgp.mit.edu}} --recv {{KEYID}}
  ```

apt

Package management utility for Debian based distributions.

- Update the list of available packages and versions (it's recommended to run this before other `apt` commands):

  ```
  sudo apt update
  ```

- Search for a given package:

  ```
  apt search {{package}}
  ```

- Show information for a package:

  ```
  apt show {{package}}
  ```

- Install a package, or update it to the latest available version:

  ```
  sudo apt install {{package}}
  ```

- Remove a package (using `purge` instead also removes its configuration files):

  ```
  sudo apt remove {{package}}
  ```

- Upgrade all installed packages to their newest available versions:

  ```
  sudo apt upgrade
  ```

aptitude

Debian and Ubuntu package management utility.

- Synchronize list of packages and versions available. This should be run first, before running subsequent aptitude commands:

  ```
  aptitude update
  ```

- Install a new package and its dependencies:

  ```
  aptitude install {{package}}
  ```

- Search for a package:

  ```
  aptitude search {{package}}
  ```

- Remove a package and all packages depending on it:

  ```
  aptitude remove {{package}}
  ```

- Upgrade installed packages to newest available versions:

  ```
  aptitude upgrade
  ```

- Upgrade installed packages (like `aptitude upgrade`) including removing obsolete packages and installing additional packages to meet new package dependencies:

  ```
  aptitude full-upgrade
  ```

archey

Simple tool for stylishly displaying system information.

- Show system information:

  ```
  archey
  ```

arp-scan

Send ARP packets to hosts (specified as IP addresses or hostnames) to scan the local network.

- Scan the current local network:

  ```
  arp-scan --localnet
  ```

- Scan an IP network with a custom bitmask:

  ```
  arp-scan {{192.168.1.1}}/{{24}}
  ```

- Scan an IP network within a custom range:

  ```
  arp-scan {{127.0.0.0}}-{{127.0.0.31}}
  ```

- Scan an IP network with a custom net mask:

  ```
  arp-scan {{10.0.0.0}}:{{255.255.255.0}}
  ```

at

Executes commands at a specified time.

- Open an `at` prompt to create a new set of scheduled commands, press `Ctrl + D` to save and exit:

  ```
  at {{hh:mm:ss}}
  ```

- Execute the commands and email the result using a local mailing program such as sendmail:

  ```
  at {{hh:mm:ss}} -m
  ```

- Execute a script at the given time:

  ```
  at {{hh:mm:ss}} -f {{path/to/file}}
  ```

beep

A utility to beep the PC speaker.

- Play a beep:

  ```
  beep
  ```

- Play a beep that repeats:

  ```
  beep -r {{repetitions}}
  ```

- Play a beep at a specified frequency (Hz) and duration (milliseconds):

  ```
  beep -f {{frequency}} -l {{duration}}
  ```

- Play each new frequency and duration as a distinct beep:

  ```
  beep -f {{frequency}} -l {{duration}} -n -f {{frequency} -l
  {{duration}}
  ```

- Play the C major scale:

  ```
  beep -f 262 -n -f 294 -n -f 330 -n -f 349 -n -f 392 -n -f 440
  -n -f 494 -n -f 523
  ```

bmon

Monitor bandwidth and capture network related statistics.

- Display the list of all the interfaces:

  ```
  bmon -a
  ```

- Display data transfer rates in bits per second:

  ```
  bmon -b
  ```

- Set policy to define which network interface(s) is/are displayed:

  ```
  bmon -p {{interface_1,interface_2,interface_3}}
  ```

- Set interval (in seconds) in which rate per counter is calculated:

  ```
  bmon -R {{2.0}}
  ```

brew

The Homebrew package manager for Linux.

- Search for available formulas:

  ```
  brew search {{text}}
  ```

- Install the latest stable version of a formula (use `--devel` for development versions):

  ```
  brew install {{formula}}
  ```

- List all installed formulae:

  ```
  brew list
  ```

- Update an installed formula (if no formula name is given, all installed formulae are updated):

  ```
  brew upgrade {{formula}}
  ```

- Fetch the newest version of Linuxbrew and all formulae from GitHub:

  ```
  brew update
  ```

- Show formulae that have a more recent version available:

  ```
  brew outdated
  ```

- Display information about a formula (version, installation path, dependencies, etc.):

  ```
  brew info {{formula}}
  ```

- Check your Linuxbrew installation for potential problems:

  ```
  brew doctor
  ```

bzip2

A block-sorting file compressor.

- Compress file:

  ```
  bzip2 {{path/to/file_to_compress}}
  ```

- Decompress file:

  ```
  bzip2 -d {{path/to/compressed_file.bz2}}
  ```

- Decompress to console:

  ```
  bzip2 -dc {{path/to/compressed_file.bz2}}
  ```

cal

Prints calendar information, with the current day highlighted.

- Display a calendar for the current month:

  ```
  cal
  ```

- Display previous, current and next month:

  ```
  cal --three
  ```

- Use monday as the first day of the week:

  ```
  cal --monday
  ```

- Display a calendar for a specific year (4 digits):

  ```
  cal {{year}}
  ```

- Display a calendar for a specific month and year:

  ```
  cal {{month}} {{year}}
  ```

calc

An interactive arbitrary-precision calculator on the terminal.

- Start calc in interactive mode:

  ```
  calc
  ```

- Perform a calculation in non-interactive mode:

  ```
  calc -p '{{85 * (36 / 4)}}'
  ```

chage

Change user account and password expiry information.

- List password information for the user:

  ```
  chage -l {{user_name}}
  ```

- Enable password expiration in 10 days:

  ```
  sudo chage -M {{10}} {{user_name}}
  ```

- Disable password expiration:

  ```
  sudo chage -M -1 {{user_name}}
  ```

- Set account expiration date:

  ```
  sudo chage -E {{YYYY-MM-DD}}
  ```

- Force user to change password on next log in:

  ```
  sudo chage -d 0
  ```

chattr

Change attributes of files or folders.

- Make a file or folder immutable to changes and deletion, even by superuser:

  ```
  chattr +i {{path}}
  ```

- Make a file or folder mutable:

  ```
  chattr -i {{path}}
  ```

- Recursively make an entire folder and contents immutable:

  ```
  chattr -R +i {{folder}}
  ```

chroot

Run command or interactive shell with special root directory.

- Run command as new root directory:

  ```
  chroot {{/path/to/new/root}} {{command}}
  ```

- Specify user and group (ID or name) to use:

  ```
  chroot --userspec={{user:group}}
  ```

clear

Clears the screen of the terminal.

- Clear the screen (equivalent to typing Control-L when using the bash shell):

  ```
  clear
  ```

cmus

Commandline Music Player.

Use arrow keys to navigate, `<enter/return>` to select, and numbers 1-8 switch between different views.

- Open cmus from specified directory:

  ```
  cmus {{path/to/directory}}
  ```

- Add file/directory to library:

  ```
  :add {{path/to/file_or_directory}}
  ```

- Pause/unpause current song:

  ```
  c
  ```

- Toggle shuffle mode on/off:

  ```
  s
  ```

- Quit cmus:

  ```
  q
  ```

compose

An alias to a `run-mailcap`'s action compose.

Originally `run-mailcap` is used to mime-type/file.

- Compose action can be used to compose any existing file or new on default mailcap edit tool:

  ```
  compose {{filename}}
  ```

- With `run-mailcap`:

  ```
  run-mailcap --action=compose {{filename}}
  ```

cpufreq-aperf

Calculate the average CPU frequency over a time period.

Requires root privileges.

- Start calculating, defaulting to all CPU cores and 1 second refresh interval:

  ```
  sudo cpufreq-aperf
  ```

- Start calculating for CPU 1 only:

  ```
  sudo cpufreq-aperf -c {{1}}
  ```

- Start calculating with a 3 seconds refresh interval for all CPU cores:

  ```
  sudo cpufreq-aperf -i {{3}}
  ```

- Calculate only once:

  ```
  sudo cpufreq-aperf -o
  ```

cpufreq-info

A tool to show CPU frequency information.

- Show CPU frequency information for all CPUs:

  ```
  cpufreq-info
  ```

- Show CPU frequency information for the specified CPU:

  ```
  cpufreq-info -c {{cpu_number}}
  ```

- Show the allowed minimum and maximum CPU frequency:

  ```
  cpufreq-info -l
  ```

- Show the current minimum and maximum CPU frequency and policy in table format:

  ```
  cpufreq-info -o
  ```

- Show available CPU frequency policies:

  ```
  cpufreq-info -g
  ```

- Show current CPU work frequency in a human-readable format, according to the cpufreq kernel module:

  ```
  cpufreq-info -f -m
  ```

- Show current CPU work frequency in a human-readable format, by reading it from hardware (only available to root):

  ```
  sudo cpufreq-info -w -m
  ```

cpufreq-set

A tool to modify CPU frequency settings.

The frequency value should range between the output of command `cpufreq-info -l`.

- Set the CPU frequency policy of CPU 1 to "userspace":

  ```
  sudo cpufreq-set -c {{1}} -g {{userspace}}
  ```

- Set the current minimum CPU frequency of CPU 1:

  ```
  sudo cpufreq-set -c {{1}} --min {{min_frequency}}
  ```

- Set the current maximum CPU frequency of CPU 1:

  ```
  sudo cpufreq-set -c {{1}} --max {{max_frequency}}
  ```

- Set the current work frequency of CPU 1:

  ```
  sudo cpufreq-set -c {{1}} -f {{work_frequency}}
  ```

cpuid

Display detailed information about all CPUs.

- Display information for all CPUs:

  ```
  cpuid
  ```

- Display information only for the current CPU:

  ```
  cpuid -1
  ```

- Display raw hex information with no decoding:

  ```
  cpuid -r
  ```

cryptsetup

Manage plain dm-crypt and LUKS (Linux Unified Key Setup) encrypted volumes.

- Initialize a LUKS volume (overwrites all data on the partition):

  ```
  cryptsetup luksFormat {{/dev/sda1}}
  ```

- Open a LUKS volume and create a decrypted mapping at /dev/mapper/{{target}}:

  ```
  cryptsetup luksOpen {{/dev/sda1}} {{target}}
  ```

- Remove an existing mapping:

  ```
  cryptsetup luksClose {{target}}
  ```

dash

Debian Almquist Shell.

Modern POSIX-compliant implementation of `sh` (isn't Bash compatible).

- Start interactive shell:

  ```
  dash
  ```

- Execute a command:

  ```
  dash -c "{{command}}"
  ```

- Run commands from a file:

  ```
  dash {{file.sh}}
  ```

- Run commands from a file, logging all commands executed to the terminal:

  ```
  dash -x {{file.sh}}
  ```

date

Set or display the system date.

- Display the current date using the default locale's format:

  ```
  date +"%c"
  ```

- Display the current date in UTC and ISO 8601 format:

  ```
  date -u +"%Y-%m-%dT%H:%M:%SZ"
  ```

- Display the current date as a Unix timestamp (seconds since the Unix epoch):

  ```
  date +%s
  ```

- Display a specific date (represented as a Unix timestamp) using the default format:

  ```
  date -d @1473305798
  ```

dd

Convert and copy a file.

- Make a bootable usb drive from an isohybrid file (such like archlinux-xxx.iso) and show the progress:

```
dd if={{file.iso}} of=/dev/{{usb_drive}} status=progress
```

- Clone a drive to another drive with 4MB block, ignore error and show progress:

```
dd if=/dev/{{source_drive}} of=/dev/{{dest_drive}} bs=4M
conv=noerror status=progress
```

- Generate a file of 100 random bytes by using kernel random driver:

```
dd if=/dev/urandom of={{random_file}} bs=100 count=1
```

- Benchmark the write performance of a disk:

```
dd if=/dev/zero of={{file_1GB}} bs=1024 count=1000000
```

- Check progress of an ongoing dd operation (Run this command from another shell):

```
kill -USR1 $(pgrep ^dd)
```

dmesg

Write the kernel messages to standard output.

- Show kernel messages:

  ```
  dmesg
  ```

- Show kernel error messages:

  ```
  dmesg --level err
  ```

- Show kernel messages and keep reading new ones, similar to `tail -f` (available in kernels 3.5.0 and newer):

  ```
  dmesg -w
  ```

- Show how much physical memory is available on this system:

  ```
  dmesg | grep -i memory
  ```

- Show kernel messages 1 page at a time:

  ```
  dmesg | less
  ```

dmidecode

Display the DMI (alternatively known as SMBIOS) table contents in a human-readable format.

Requires root privileges.

- Show all DMI table contents:

  ```
  sudo dmidecode
  ```

- Show the BIOS version:

  ```
  sudo dmidecode -s bios-version
  ```

- Show the system's serial number:

  ```
  sudo dmidecode -s system-serial-number
  ```

- Show all the BIOS information:

  ```
  sudo dmidecode -t bios
  ```

- Show all the system information:

  ```
  sudo dmidecode -t system
  ```

dnf

Package management utility for RHEL, Fedora, and CentOS (replaces yum).

- Synchronize list of packages and versions available. This should be run first, before running subsequent dnf commands:

  ```
  dnf update
  ```

- Install a new package:

  ```
  dnf install {{package}}
  ```

- Install a new package and assume yes to all questions:

  ```
  dnf -y install {{package}}
  ```

- Remove a package:

  ```
  dnf remove {{package}}
  ```

- Upgrade installed packages to newest available versions:

  ```
  dnf upgrade
  ```

dpkg-query

A tool that shows information about installed packages.

- List all installed packages:

  ```
  dpkg-query -l
  ```

- List installed packages matching a pattern:

  ```
  dpkg-query -l '{{pattern}}'
  ```

- List all files installed by a package:

  ```
  dpkg-query -L {{package_name}}
  ```

- Show information about a package:

  ```
  dpkg-query -s {{package_name}}
  ```

dpkg

Debian package manager.

- Install a package:

  ```
  dpkg -i {{/path/to/file}}
  ```

- Remove a package:

  ```
  dpkg -r {{package_name}}
  ```

- List installed packages:

  ```
  dpkg -l {{pattern}}
  ```

- List package contents:

  ```
  dpkg -L {{package_name}}
  ```

- Find out which package owns a file:

  ```
  dpkg -S {{file_name}}
  ```

du

Disk usage: estimate and summarize file and folder space usage.

- List the sizes of a folder and any subfolders, in the given unit (B/KB/MB):

```
du -{{b|k|m}} {{path/to/folder}}
```

- List the sizes of a folder and any subfolders, in human-readable form (i.e. auto-selecting the appropriate unit for each size):

```
du -h {{path/to/folder}}
```

- Show the size of a single folder, in human readable units:

```
du -sh {{path/to/folder}}
```

- List the human-readable sizes of a folder and of all the files and folders within it:

```
du -ah {{path/to/folder}}
```

- List the human-readable sizes of a folder and any subfolders, up to N levels deep:

```
du -h --max-depth=N {{path/to/folder}}
```

- List the human-readable size of all .jpg files in subfolders of the current folder, and show a cumulative total at the end:

```
du -ch */*.jpg
```

edit

An alias to a `run-mailcap`'s action edit.

Originally `run-mailcap` is used to process/edit mime-type/file.

- Edit action can be used to view any file on default mailcap explorer:

  ```
  edit {{filename}}
  ```

- With `run-mailcap`:

  ```
  run-mailcap --action=edit {{filename}}
  ```

edquota

Edit quotas for a user or group. By default it operates on all file systems with quotas.

Quota information is stored permanently in the `quota.user` and `quota.group` files in the root of the filesystem.

- Edit quota of the current user:

  ```
  edquota --user $(whoami)
  ```

- Edit quota of a specific user:

  ```
  sudo edquota --user {{username}}
  ```

- Edit quota for a group:

  ```
  sudo edquota --group {{group}}
  ```

- Restrict operations to a given filesystem (by default edquota operates on all filesystems with quotas):

  ```
  sudo edquota --file-system {{filesystem}}
  ```

- Edit the default grace period:

  ```
  sudo edquota -t
  ```

- Duplicate a quota to other users:

  ```
  sudo edquota -p {{reference_user}} {{destination_user1}}
  {{destination_user2}}
  ```

emerge

Gentoo Linux package manager utility.

- Synchronize all packages:

  ```
  emerge --sync
  ```

- Update all packages, including dependencies:

  ```
  emerge -uDNav @world
  ```

- Resume a failed updated, skipping the failing package:

  ```
  emerge --resume --skipfirst
  ```

- Install a new package, with confirmation:

  ```
  emerge -av {{package_name}}
  ```

- Remove a package, with confirmation:

  ```
  emerge -Cav {{package_name}}
  ```

- Remove orphaned packages (that were installed only as dependencies):

  ```
  emerge -avc
  ```

- Search the package database for a keyword:

  ```
  emerge -S {{keyword}}
  ```

equery

View information about Portage packages.

- List all installed packages:

  ```
  equery list '*'
  ```

- Search for installed packages in the Portage tree and in overlays:

  ```
  equery list -po {{package_name}}
  ```

- List all packages that depend on a given package:

  ```
  equery depends {{package_name}}
  ```

- List all packages that a given package depends on:

  ```
  equery depgraph {{package_name}}
  ```

- List all files installed by a package:

  ```
  equery files --tree {{package_name}}
  ```

eval

Execute arguments as a single command in the current shell and return its result.

- Call `echo` with the "foo" argument:

  ```
  eval "{{echo foo}}"
  ```

- Set a variable in the current shell:

  ```
  eval "{{foo=bar}}"
  ```

expand

Convert tabs to spaces.

- Convert tabs in each file to spaces, writing to standard output:

  ```
  expand {{file}}
  ```

- Convert tabs to spaces, reading from standard input:

  ```
  expand
  ```

- Do not convert tabs after non blanks:

  ```
  expand -i {{file}}
  ```

- Have tabs a certain number of characters apart, not 8:

  ```
  expand -t={{number}} {{file}}
  ```

- Use comma separated list of explicit tab positions:

  ```
  expand -t={{list}}
  ```

export

Command to mark shell variables in the current environment to be exported with any newly forked child processes.

- Set a new environment variable:

  ```
  export {{VARIABLE}}={{value}}
  ```

- Remove an environment variable:

  ```
  export -n {{VARIABLE}}
  ```

- Append something to the PATH variable:

  ```
  export PATH=$PATH:{{path/to/append}}
  ```

expr

Evaluate expressions and manipulate strings.

- Get string length:

  ```
  expr length {{string}}
  ```

- Evaluate logical or math expression with an operator ('+', '-', '*', '&', '|', etc.). Special symbols should be escaped:

  ```
  expr {{first_argument}} {{operator}} {{second_argument}}
  ```

- Get position of the first character in 'string' that matches 'substring':

  ```
  echo $(expr index {{string}} {{substring}})
  ```

- Extract part of the string:

  ```
  echo $(expr substr {{string}} {{position_to_start}}
  {{number_of_characters}}
  ```

- Extract part of the string which matches a regular expression:

  ```
  echo $(expr {{string}} : '\({{regular_expression}}\)')
  ```

eyeD3

Read and manipulate metadata of MP3 files.

- View information about an MP3 file:

  ```
  eyeD3 {{filename.mp3}}
  ```

- Set the title of an MP3 file:

  ```
  eyeD3 --title {{"A Title"}} {{filename.mp3}}
  ```

- Set the album of all the MP3 files in a directory:

  ```
  eyeD3 --album {{"Album Name"}} {{*.mp3}}
  ```

- Set the front cover art for an MP3 file:

  ```
  eyeD3 --add-image {{front_cover.jpeg}}:FRONT_COVER:
  {{filename.mp3}}
  ```

fatlabel

Sets or gets the label of a FAT32 partition.

- Get the label of a FAT32 partition:

  ```
  fatlabel {{/dev/sda1}}
  ```

- Set the label of a FAT32 partition:

  ```
  fatlabel {{/dev/sdc3}} "{{new_label}}"
  ```

fc-list

List available fonts installed on the system.

- Return a list of installed fonts with given name:

  ```
  fc-list | grep '{{DejaVu Serif}}'
  ```

fc-match

Match available fonts.

- Return a sorted list of best matching fonts:

  ```
  fc-match -s '{{DejaVu Serif}}'
  ```

fc-pattern

Shows information about a font matching a pattern.

- Display default information about a font:

  ```
  fc-pattern -d '{{DejaVu Serif}}'
  ```

feh

Lightweight image viewing utility.

- View images locally or using a URL:

  ```
  feh {{path/to/images}}
  ```

- View images recursively:

  ```
  feh --recursive {{path/to/images}}
  ```

- View images without window borders:

  ```
  feh --borderless {{path/to/images}}
  ```

- Exit after the last image:

  ```
  feh --cycle-once {{path/to/images}}
  ```

- Set the slideshow cycle delay:

  ```
  feh --slideshow-delay {{seconds}} {{path/to/images}}
  ```

- Set your wallpaper (centered, filled, maximized, scaled or tiled):

  ```
  feh --bg-{{center|fill|max|scale|tile}} {{path/to/image}}
  ```

figlet

Generate ASCII banners from user input.

- Generate by directly inputting text:

  ```
  figlet {{input_text}}
  ```

- Use a custom font file:

  ```
  figlet {{input_text}} -f {{font_file_name}}
  ```

- Pipe command output through figlet:

  ```
  {{command}} | figlet
  ```

- Show available figlet fonts:

  ```
  showfigfonts {{optional_string_to_display}}
  ```

file

Determine file type.

- Give a description of the type of the specified file. Works fine for files with no file extension:

  ```
  file {{filename}}
  ```

- Look inside a zipped file and determine the file type(s) inside:

  ```
  file -z {{foo.zip}}
  ```

- Allow file to work with special or device files:

  ```
  file -s {{filename}}
  ```

- Don't stop at first file type match; keep going until the end of the file:

  ```
  file -k {{filename}}
  ```

- Determine the mime encoding type of a file:

  ```
  file -i {{filename}}
  ```

findmnt

Find your filesystem.

- List all mounted filesystems:

  ```
  findmnt
  ```

- Search for a device:

  ```
  findmnt {{/dev/sdb1}}
  ```

- Search for a mountpoint:

  ```
  findmnt {{/}}
  ```

- Find filesystems in specific type:

  ```
  findmnt -t {{ext4}}
  ```

- Find filesystems with specific label:

  ```
  findmnt LABEL={{BigStorage}}
  ```

firewall-cmd

The firewalld command line client.

- View the available firewall zones:

  ```
  firewall-cmd --get-active-zones
  ```

- View the rules which are currently applied:

  ```
  firewall-cmd --list-all
  ```

- Permanently open the port for a service in the specified zone (like port `443` when in the `public` zone):

  ```
  firewall-cmd --permanent --zone={{public}} --add-service=
  {{https}}
  ```

- Permanently close the port for a service in the specified zone (like port `80` when in the `public` zone):

  ```
  firewall-cmd --permanent --zone={{public}} --remove-service=
  {{http}}
  ```

- Reload firewalld to force rule changes to take effect:

  ```
  firewall-cmd --reload
  ```

foreman

Manage Procfile-based applications.

- Start an application with the Procfile in the current directory:

  ```
  foreman start
  ```

- Start an application with a specified Procfile:

  ```
  foreman start -f {{Procfile}}
  ```

- Start a specific application:

  ```
  foreman start {{process}}
  ```

- Validate Procfile format:

  ```
  foreman check
  ```

- Run one-off commands with the process's environment:

  ```
  foreman run {{command}}
  ```

- Start all processes except the one named "worker":

  ```
  foreman start -m all=1,{{worker}}=0
  ```

free

Display amount of free and used memory in the system.

- Display system memory:

  ```
  free
  ```

- Display memory in Bytes/KB/MB/GB:

  ```
  free -{{b|k|m|g}}
  ```

- Display memory in human readable units:

  ```
  free -h
  ```

- Refresh the output every 2 seconds:

  ```
  free -s {{2}}
  ```

fsck

Check the integrity of a filesystem or repair it. The filesystem should be unmounted at the time the command is run.

- Check filesystem /dev/sda, reporting any damaged blocks:

  ```
  fsck {{/dev/sda}}
  ```

- Check filesystem /dev/sda, reporting any damaged blocks and interactively letting the user choose to repair each one:

  ```
  fsck -r {{/dev/sda}}
  ```

- Check filesystem /dev/sda, reporting any damaged blocks and automatically repairing them:

  ```
  fsck -a {{/dev/sda}}
  ```

fuser

Display process IDs currently using files or sockets.

Require admin privileges.

- Identify process using a TCP socket:

  ```
  fuser -n tcp {{port}}
  ```

genkernel

Gentoo Linux utility to compile and install kernels.

- Automatically compile and install a generic kernel:

  ```
  sudo genkernel all
  ```

- Build and install the bzImage|initramfs|kernel|ramdisk only:

  ```
  sudo genkernel {{bzImage|initramfs|kernel|ramdisk}}
  ```

- Apply changes to the kernel configuration before compiling and installing:

  ```
  sudo genkernel --menuconfig all
  ```

- Generate a kernel with a custom name:

  ```
  sudo genkernel --kernname={{custom_name}} all
  ```

- Use a kernel source outside of the default directory /usr/src/linux:

  ```
  sudo genkernel --kerneldir={{path/to/directory}} all
  ```

getent

Get entries from Name Service Switch libraries.

- Get list of all groups:

  ```
  getent group
  ```

- See the members of a group:

  ```
  getenet group {{group_name}}
  ```

- Get list of all services:

  ```
  getent services
  ```

- Find a username by UID:

  ```
  getent passwd 1000
  ```

- Perform a reverse DNS lookup:

  ```
  getent hosts {{host}}
  ```

getfacl

Get file access control lists.

- Display the file access control list:

  ```
  getfacl {{path/to/file_or_folder}}
  ```

- Display the file access control list with numeric user and group IDs:

  ```
  getfacl -n {{path/to/file_or_folder}}
  ```

- Display the file access control list with tabular output format:

  ```
  getfacl -t {{path/to/file_or_folder}}
  ```

gitk

A graphical git repository browser.

- Show the repository browser for the current git repository:

  ```
  gitk
  ```

- Show repository browser for a specific file or folder:

  ```
  gitk {{path/to/file_or_folder}}
  ```

- Show commits made since 1 week ago:

  ```
  gitk --since={{"1 week ago"}}
  ```

- Show commits older than 1/1/2016:

  ```
  gitk --until={{"1/1/2015"}}
  ```

- Show at most 100 changes in all branches:

  ```
  gitk --max-count={{100}} --all
  ```

groupadd

Add user groups to the system.

- Create a new Linux group:

  ```
  groupadd {{group_name}}
  ```

- Create new group with a specific groupid:

  ```
  groupadd {{group_name}} -g {{group_id}}
  ```

groupdel

Delete existing user groups from the system.

- Delete an existing group:

  ```
  groupdel {{group_name}}
  ```

groupmod

Modify existing user groups in the system.

- Change the group name:

  ```
  groupmod -n {{new_group_name}} {{old_group_name}}
  ```

- Change the group id:

  ```
  groupmod -g {{new_group_id}} {{old_group_name}}
  ```

halt

Power off or reboot the machine.

- Power the machine off:

  ```
  halt
  ```

- Reboot the machine:

  ```
  halt --reboot
  ```

hardinfo

Show hardware information in GUI window.

- Start hardinfo:

  ```
  hardinfo
  ```

- Print report to standard output:

  ```
  hardinfo -r
  ```

- Save report to HTML file:

  ```
  hardinfo -r -f html > hardinfo.html
  ```

head

Output the first part of files.

- Output the first few lines of a file:

  ```
  head -n {{count_of_lines}} {{filename}}
  ```

- Output the first few bytes of a file:

  ```
  head -c {{size_in_bytes}} {{filename}}
  ```

- Output everything but the last few lines of a file:

  ```
  head -n -{{count_of_lines}} {{filename}}
  ```

- Output everything but the last few bytes of a file:

  ```
  head -c -{{size_in_bytes}} {{filename}}
  ```

hostname

Show or set the system's host name.

- Show current host name:

  ```
  hostname
  ```

- Show the network address of the host name:

  ```
  hostname -i
  ```

- Show all network addresses of the host:

  ```
  hostname -I
  ```

- Show the FQDN (Fully Qualified Domain Name):

  ```
  hostname --fqdn
  ```

- Set current host name:

  ```
  hostname {{new_hostname}}
  ```

hostnamectl

Get or set the hostname of the computer.

- Get the hostname of the computer:

  ```
  hostnamectl
  ```

- Set the hostname of the computer:

  ```
  sudo hostnamectl set-hostname "{{some_hostname}}"
  ```

htop

Display dynamic real-time information about running processes. An enhanced version of `top`.

- Start htop:

  ```
  htop
  ```

- Start htop displaying only processes owned by given user:

  ```
  htop -u {{user_name}}
  ```

- Get help about interactive commands:

  ```
  ?
  ```

http-prompt

An interactive command-line HTTP client featuring autocomplete and syntax highlighting.

- Launch a session targeting the default url of http://localhost:8000 or the previous session:

  ```
  http-prompt
  ```

- Launch a session with a given url:

  ```
  http-prompt {{http://example.com}}
  ```

- Launch a session with some initial options:

  ```
  http-prompt {{localhost:8000/api}} --auth
  {{username:password}}
  ```

httpie

A user friendly command line HTTP tool.

- Send a GET request (default method with no request data):

  ```
  http {{https://example.com}}
  ```

- Send a POST request (default method with request data):

  ```
  http {{https://example.com}} {{hello=World}}
  ```

- Send a POST request with redirected input:

  ```
  http {{https://example.com}} < {{file.json}}
  ```

- Send a PUT request with a given json body:

  ```
  http PUT {{https://example.com/todos/7}} {{hello=world}}
  ```

- Send a DELETE request with a given request header:

  ```
  http DELETE {{https://example.com/todos/7}} {{API-Key:foo}}
  ```

- Show the whole HTTP exchange (both request and response):

  ```
  http -v {{https://example.com}}
  ```

- Download a file:

  ```
  http --download {{https://example.com}}
  ```

hwclock

Used for reading or changing the hardware clock. Usually requires root.

- Display the current time as reported by the hardware clock:

  ```
  hwclock
  ```

- Write the current software clock time to the hardware clock (sometimes used during system setup):

  ```
  hwclock --systohc
  ```

- Write the current hardware clock time to the software clock:

  ```
  hwclock --hctosys
  ```

i7z

An Intel CPU (only i3, i5 and i7) realtime reporting tool.

- Start i7z (needs to be run in super user mode):

  ```
  sudo i7z
  ```

ifdown

Disable network interfaces.

- Disable interface eth0:

  ```
  ifdown {{eth0}}
  ```

- Disable all interfaces which are enabled:

  ```
  ifdown -a
  ```

ifup

Tool used to enable network interfaces.

- Enable interface eth0:

  ```
  ifup {{eth0}}
  ```

- Enable all the interfaces defined with "auto" in /etc/network/interfaces:

  ```
  ifup -a
  ```

iostat

Report statistics for devices and partitions.

- Display a report of CPU and disk statistics since system startup:

  ```
  iostat
  ```

- Display a report of CPU and disk statistics with units converted to megabytes:

  ```
  iostat -m
  ```

- Display CPU statistics:

  ```
  iostat -c
  ```

- Display disk statistics with disk names (including LVM):

  ```
  iostat -N
  ```

- Display extended disk statistics with disk names for device "sda":

  ```
  iostat -xN {{sda}}
  ```

- Display incremental reports of CPU and disk statistics every 2 seconds:

  ```
  iostat {{2}}
  ```

ip

Show / manipulate routing, devices, policy routing and tunnels.

- List interfaces with detailed info:

  ```
  ip a
  ```

- Display the routing table:

  ```
  ip r
  ```

- Make an interface up/down:

  ```
  ip link set {{interface}} up/down
  ```

- Add/Delete an ip address to an interface:

  ```
  ip addr add/del {{ip}}/{{mask}} dev {{interface}}
  ```

- Add a default route:

  ```
  ip route add default via {{ip}} dev {{interface}}
  ```

iptables

Program that allows configuration of tables, chains and rules provided by the Linux kernel firewall.

- See chains and rules for specific table:

```
sudo iptables -t {{table_name}} -vnL
```

- Set chain policy rule:

```
sudo iptables -P {{chain}} {{rule}}
```

- Append rule to chain policy for IP:

```
sudo iptables -A {{chain}} -s {{ip}} -j {{rule}}
```

- Append rule to chain policy for IP considering protocol and port:

```
sudo iptables -A {{chain}} -s {{ip}} -p {{protocol}} --dport
{{port}} -j {{rule}}
```

- Delete chain rule:

```
sudo iptables -D {{chain}} {{rule_line_number}}
```

- Save iptables configuration:

```
sudo iptables-save > {{path/to/iptables_file}}
```

isoinfo

Utility programs for dumping and verifying ISO disk images.

- List all the files included in an ISO image:

  ```
  isoinfo -f -i {{path/to/image.iso}}
  ```

- E[x]tract a specific file from an ISO image and send it out stdout:

  ```
  isoinfo -i {{path/to/image.iso}} -x {{/PATH/TO/FILE/INSIDE/
  ISO.EXT}}
  ```

- Show header information for an ISO disk image:

  ```
  isoinfo -d -i {{path/to/image.iso}}
  ```

jobs

BASH builtin for viewing information about processes spawned by the current shell.

- View jobs spawned by the current shell:

  ```
  jobs
  ```

- List jobs and their process ids:

  ```
  jobs -l
  ```

- Display information about jobs with changed status:

  ```
  jobs -n
  ```

- Display process id of process group leader:

  ```
  jobs -p
  ```

- Display running processes:

  ```
  jobs -r
  ```

- Display stopped processes:

  ```
  jobs -s
  ```

journalctl

Query the systemd journal.

- Show all messages from this boot:

  ```
  journalctl -b
  ```

- Show all messages from last boot:

  ```
  journalctl -b -1
  ```

- Follow new messages (like `tail -f` for traditional syslog):

  ```
  journalctl -f
  ```

- Show all messages by a specific unit:

  ```
  journalctl -u {{unit}}
  ```

- Show all messages by a specific process:

  ```
  journalctl _PID={{pid}}
  ```

- Show all messages by a specific executable:

  ```
  journalctl {{/path/to/executable}}
  ```

kexec

Directly reboot into a new kernel.

- Load a new kernel:

```
kexec -l {{path/to/kernel}} --initrd={{path/to/initrd}} --
command-line={{arguments}}
```

- Load a new kernel with current boot parameters:

```
kexec -l {{path/to/kernel}} --initrd={{path/to/initrd}} --
reuse-cmdline
```

- Execute a currently loaded kernel:

```
kexec -e
```

- Unload current kexec target kernel:

```
kexec -u
```

lastlog

Show the most recent login of all users or of a given user.

- Display the most recent login of all users:

  ```
  lastlog
  ```

- Display lastlog record of the specified user:

  ```
  lastlog -u {{username}}
  ```

- Display records before than 7 days:

  ```
  lastlog -b {{7}}
  ```

- Display records more recent than 3 days:

  ```
  lastlog -t {{3}}
  ```

ldconfig

Configure symlinks and cache for shared library dependencies.

- Update symlinks and rebuild the cache (usually run when a new library is installed):

```
sudo ldconfig
```

- Update the symlinks for a given directory:

```
sudo ldconfig -n {{path/to/directory}}
```

- Print the libraries in the cache and check whether a given library is present:

```
ldconfig -p | grep {{library_name}}
```

ldd

Display shared library dependencies.

- Display shared library dependencies of a binary:

  ```
  ldd {{path/to/binary}}
  ```

- Display unused direct dependencies:

  ```
  ldd -u {{path/to/binary}}
  ```

light

CLI to control the backlight of your screen.

- Get the current backlight value in percent:

  ```
  light
  ```

- Set the backlight value to 50 percent:

  ```
  light -S {{50}}
  ```

- Reduce 20 percent from the current backlight value:

  ```
  light -U {{20}}
  ```

- Add 20 percent to the current backlight value:

  ```
  light -A {{20}}
  ```

lldb

The LLVM Low-Level Debugger.

- Debug an executable:

  ```
  lldb {{executable}}
  ```

- Attach `lldb` to a running process with a given PID:

  ```
  lldb -p {{pid}}
  ```

- Wait for a new process to launch with a given name, and attach to it:

  ```
  lldb -w -n {{process_name}}
  ```

locate

Find filenames quickly.

- Look for pattern in the database. Note: the database is recomputed periodically (usually weekly or daily):

  ```
  locate {{pattern}}
  ```

- Look for a file by its exact filename (a pattern containing no globbing characters is interpreted as `*pattern*`):

  ```
  locate */{{filename}}
  ```

- Recompute the database. You need to do it if you want to find recently added files:

  ```
  sudo updatedb
  ```

logger

Add messages to syslog (/var/log/syslog).

- Log a message to syslog:

  ```
  logger {{message}}
  ```

- Take input from stdin and log to syslog:

  ```
  echo {{log_entry}} | logger
  ```

- Send the output to a remote syslog server running at a given port. Default port is 514:

  ```
  echo {{log_entry}} | logger --server {{hostname}} --port {{port}}
  ```

- Use a specific tag for every line logged. Default is the name of logged in user:

  ```
  echo {{log_entry}} | logger --tag {{tag}}
  ```

- Log messages with a given priority. Default is `user.notice`. See `man logger` for all priority options:

  ```
  echo {{log_entry}} | logger --priority {{user.warning}}
  ```

logwatch

Summarizes many different logs for common services (e.g., apache, pam_unix, sshd, etc.) in a single report.

- Analyze logs for a range of dates at certain level of detail:

  ```
  logwatch --range {{yesterday|today|all|help}} --detail {{low|
  medium|others}}'
  ```

- Restrict report to only include information for a selected service:

  ```
  logwatch --range {{all}} --service {{apache|pam_unix|etc}}
  ```

losetup

Set up and control loop devices.

- List loop devices with detailed info:

  ```
  losetup -a
  ```

- Attach a file to a given loop device:

  ```
  sudo losetup /dev/{{loop}} /{{path/to/file}}
  ```

- Detach all loop devices:

  ```
  sudo losetup -D
  ```

- Detach a given loop device:

  ```
  sudo losetup -d /dev/{{loop}}
  ```

lsattr

List file attributes on a Linux file system.

- Display the attributes of the files in the current directory:

  ```
  lsattr
  ```

- List the attributes of files in a particular path:

  ```
  lsattr {{path}}
  ```

- List file attributes recursively in the current and subsequent directories:

  ```
  lsattr -R
  ```

- Show attributes of all the files in the current directory, including hidden ones:

  ```
  lsattr -a
  ```

- Display attributes of directories in the current directory:

  ```
  lsattr -d
  ```

lsb_release

Provides certain LSB (Linux Standard Base) and distribution-specific information.

- Print all available information:

  ```
  lsb_release -a
  ```

- Print a description (usually the full name) of the operating system:

  ```
  lsb_release -d
  ```

- Print only the operating system name (ID), suppressing the field name:

  ```
  lsb_release -i -s
  ```

- Print the release number and codename of the distribution, suppressing the field names:

  ```
  lsb_release -rcs
  ```

lsblk

Lists information about devices.

- List all storage devices in a tree-like format:

  ```
  lsblk
  ```

- Also list empty devices:

  ```
  lsblk -a
  ```

- Print the SIZE column in bytes rather than in a human-readable format:

  ```
  lsblk -b
  ```

- Output info about filesystems:

  ```
  lsblk -f
  ```

- Use ASCII characters for tree formatting:

  ```
  lsblk -i
  ```

- Output info about block-device topology:

  ```
  lsblk -t
  ```

lscpu

Displays information about the CPU architecture.

- Display information about all CPUs:

  ```
  lscpu
  ```

- Display information in a table:

  ```
  lscpu --extended
  ```

- Display only information about offline CPUs in a table:

  ```
  lscpu --extended --offline
  ```

lshw

List detailed information about hardware configurations as root user.

- Launch the GUI:

  ```
  sudo lshw -X
  ```

- List all hardwares in tabular format:

  ```
  sudo lshw -short
  ```

- List all disks and storage controllers in tabular format:

  ```
  sudo lshw -class disk -class storage -short
  ```

- Save all network interfaces to an HTML file:

  ```
  sudo lshw -class network -html > {{interfaces.html}}
  ```

lsmod

Shows the status of linux kernel modules.

See also `modprobe`, which loads kernel modules.

- List all currently loaded kernel modules:

  ```
  lsmod
  ```

lspci

List all PCI devices.

- Show a brief list of devices:

  ```
  lspci
  ```

- Display additional info:

  ```
  lspci -v
  ```

- Display drivers and modules handling each device:

  ```
  lspci -k
  ```

- Show a specific device:

  ```
  lspci -s {{00:18.3}}
  ```

- Dump info in a readable form:

  ```
  lspci -vm
  ```

lsscsi

List SCSI devices (or hosts) and their attributes.

- List all SCSI devices:

  ```
  lsscsi
  ```

- List all SCSI devices with detailed attributes:

  ```
  lsscsi -L
  ```

- List all SCSI devices with human readable disk capacity:

  ```
  lsscsi -s
  ```

ltrace

Display dynamic library calls of a process.

- Print (trace) library calls of a program binary:

  ```
  ltrace ./{{program}}
  ```

- Count library calls. Print a handy summary at the bottom:

  ```
  ltrace -c {{/path/to/program}}
  ```

- Trace calls to malloc and free, omit those done by libc:

  ```
  ltrace -e malloc+free-@libc.so* {{/path/to/program}}
  ```

- Write to file instead of terminal:

  ```
  ltrace -o {{file}} {{/path/to/program}}
  ```

lvcreate

Creates a logical volume in an existing volume group.

A volume group is a collection of logical and physical volumes.

- Create a logical volume of 10 gigabytes in the volume group vg1:

  ```
  lvcreate -L {{10G}} {{vg1}}
  ```

- Create a 1500 megabyte linear logical volume named mylv in the volume group vg1:

  ```
  lvcreate -L {{1500}} -n {{mylv}} {{vg1}}
  ```

- Create a logical volume called mylv that uses 60% of the total space in volume group vg1:

  ```
  lvcreate -l {{60%VG}} -n {{mylv}} {{vg1}}
  ```

- Create a logical volume called mylv that uses all of the unallocated space in the volume group vg1:

  ```
  lvcreate -l {{100%FREE}} -n {{mylv}} {{vg1}}
  ```

lxc

Manage Linux containers using the lxd REST API.

Any container names or patterns can be prefixed with the name of a remote server.

- List local containers matching a string. Omit the string to list all local containers:

  ```
  lxc list {{match_string}}
  ```

- List images matching a string. Omit the string to list all images:

  ```
  lxc image list [{{remote}}:]{{match_string}}
  ```

- Create a new container from an image:

  ```
  lxc launch [{{remote}}:]{{image}} {{container}}
  ```

- Start a container:

  ```
  lxc start [{{remote}}:]{{container}}
  ```

- Stop a container:

  ```
  lxc stop [{{remote}}:]{{container}}
  ```

- Show detailed info about a container:

  ```
  lxc info [{{remote}}:]{{container}}
  ```

- Take a snapshot of a container:

  ```
  lxc snapshot [{{remote}}:]{{container}} {{snapshot}}
  ```

md5sum

Calculate MD5 cryptographic checksums.

- Calculate the MD5 checksum for a file:

  ```
  md5sum {{filename1}}
  ```

- Calculate MD5 checksums for multiple files:

  ```
  md5sum {{filename1}} {{filename2}}
  ```

- Read a file of MD5SUMs and verify all files have matching checksums:

  ```
  md5sum -c {{filename.md5}}
  ```

mdadm

RAID management utility.

- Create array:

  ```
  mdadm --create {{/path/to/raid_device_file}} --level
  {{raid_level}} --raid-devices {{number_of_disks}} {{/path/to/
  disk_device_file}}
  ```

- Stop array:

  ```
  mdadm -S {{/path/to/raid_device_file}}
  ```

- Mark disk as failed:

  ```
  mdadm {{/path/to/raid_device_file}} -f {{/path/to/
  disk_device_file}}
  ```

- Remove disk:

  ```
  mdadm {{/path/to/raid_device_file}} -r {{/path/to/
  disk_device_file}}
  ```

- Add disk to array:

  ```
  mdadm {{/path/to/raid_device_file}} -a {{/path/to/
  disk_device_file}}
  ```

- Show RAID info:

  ```
  mdadm -D {{/path/to/raid_device_file}}
  ```

microcom

A minimalistic terminal program, used to access remote devices via a serial, CAN or telnet connection from the console.

- Open a serial port using the specified baud rate:

  ```
  microcom --port {{path/to/serial_port}} --speed {{baud_rate}}
  ```

- Establish a telnet connection to the specified host:

  ```
  microcom --telnet {{hostname}}:{{port}}
  ```

mke2fs

Creates a Linux filesystem inside a partition.

- Create an ext2 filesystem in partition 1 of device b (`sdb1`):

  ```
  mkfs.ext2 {{/dev/sdb1}}
  ```

- Create an ext3 filesystem in partition 1 of device b (`sdb1`):

  ```
  mkfs.ext3 {{/dev/sdb1}}
  ```

- Create an ext3 filesystem in partition 1 of device b (`sdb1`):

  ```
  mkfs.ext3 {{/dev/sdb1}}
  ```

mkfs.cramfs

Creates a ROM filesystem inside a partition.

- Create a ROM filesystem inside partition 1 on device b (`sdb1`):

  ```
  mkfs.cramfs {{/dev/sdb1}}
  ```

- Create a ROM filesystem with a volume-name:

  ```
  mkfs.cramfs -n {{volume_name}} {{/dev/sdb1}}
  ```

mkfs.exfat

Creates an exfat filesystem inside a partition.

- Create an exfat filesystem inside partition 1 on device b (`sdb1`):

  ```
  mkfs.exfat {{/dev/sdb1}}
  ```

- Create filesystem with a volume-name:

  ```
  mkfs.exfat -n {{volume_name}} {{/dev/sdb1}}
  ```

- Create filesystem with a volume-id:

  ```
  mkfs.exfat -i {{volume_id}} {{/dev/sdb1}}
  ```

mkfs.fat

Creates an MS-DOS filesystem inside a partition.

- Create a fat filesystem inside partition 1 on device b (`sdb1`):

  ```
  mkfs.fat {{/dev/sdb1}}
  ```

- Create filesystem with a volume-name:

  ```
  mkfs.fat -n {{volume_name}} {{/dev/sdb1}}
  ```

- Create filesystem with a volume-id:

  ```
  mkfs.fat -i {{volume_id}} {{/dev/sdb1}}
  ```

- Use 5 instead of 2 file allocation tables:

  ```
  mkfs.fat -f 5 {{/dev/sdb1}}
  ```

mkfs.minix

Creates a Minix filesystem inside a partition.

- Create a Minix filesystem inside partition 1 on device b (`sdb1`):

  ```
  mkfs.minix {{/dev/sdb1}}
  ```

mkfs.ntfs

Creates a NTFS filesystem inside a partition.

- Create a NTFS filesystem inside partition 1 on device b (`sdb1`):

  ```
  mkfs.ntfs {{/dev/sdb1}}
  ```

- Create filesystem with a volume-label:

  ```
  mkfs.ntfs -L {{volume_label}} {{/dev/sdb1}}
  ```

- Create filesystem with specific UUID:

  ```
  mkfs.ntfs -U {{UUID}} {{/dev/sdb1}}
  ```

mkfs.vfat

Creates an MS-DOS filesystem inside a partition.

- Create a.vfat filesystem inside partition 1 on device b (`sdb1`):

  ```
  mkfs.vfat {{/dev/sdb1}}
  ```

- Create filesystem with a volume-name:

  ```
  mkfs.vfat -n {{volume_name}} {{/dev/sdb1}}
  ```

- Create filesystem with a volume-id:

  ```
  mkfs.vfat -i {{volume_id}} {{/dev/sdb1}}
  ```

- Use 5 instead of 2 file allocation tables:

  ```
  mkfs.vfat -f 5 {{/dev/sdb1}}
  ```

mkisofs

Create ISO files from folders.

Also aliased as `genisoimage`.

- Create an ISO from a folder:

  ```
  mkisofs -o {{filename.iso}} {{path/to/source_folder}}
  ```

- Set the disc label when creating an ISO:

  ```
  mkisofs -o {{filename.iso}} -V {{"label_name"}} {{path/to/
  source_folder}}
  ```

modinfo

Extract information about a Linux kernel module.

- List all attributes of a kernel module:

```
modinfo {{kernel_module}}
```

- List the specified attribute only:

```
modinfo -F {{author|description|license|parm|filename}}
{{kernel_module}}
```

mpstat

Report CPU statistics.

- Display CPU statistics every 2 seconds:

  ```
  mpstat {{2}}
  ```

- Display 5 reports, one by one, at 2 second intervals:

  ```
  mpstat {{2}} {{5}}
  ```

- Display 5 reports, one by one, from a given processor, at 2 second intervals:

  ```
  mpstat -P {{0}} {{2}} {{5}}
  ```

mycli

A CLI for MySQL, MariaDB, and Percona with auto-completion and syntax highlighting.

- Connect to a database with the currently logged in user:

  ```
  mycli {{database_name}}
  ```

- Connect to a database with the specified user:

  ```
  mycli -u {{user}} {{database_name}}
  ```

- Connect to a database on the specified host with the specified user:

  ```
  mycli -u {{user}} -h {{host}} {{database_name}}
  ```

n

Tool to manage multiple node versions.

- Install a given version of node. If the version is already installed, it will be activated:

  ```
  n {{version}}
  ```

- Display installed versions and interactively activate one of them:

  ```
  n
  ```

- Remove a version:

  ```
  n rm {{version}}
  ```

- Execute a file with a given version:

  ```
  n use {{version}} {{file.js}}
  ```

- Output binary path for a version:

  ```
  n bin {{version}}
  ```

ncat

Use the normal `cat` functionality over networks.

- Listen for input on the specified port and write it to the specified file:

  ```
  ncat -l {{port}} > {{/path/to/file}}
  ```

- Write output of specified file to the specified host on the specified port:

  ```
  ncat {{address}} {{port}} < {{/path/to/file}}
  ```

nethogs

Monitor bandwidth usage per process.

- Start nethogs as root (default device is eth0):

  ```
  sudo nethogs
  ```

- Monitor bandwidth on specific device:

  ```
  sudo nethogs {{device}}
  ```

- Monitor bandwidth on multiple devices:

  ```
  sudo nethogs {{device1}} {{device2}}
  ```

- Specify refresh rate:

  ```
  sudo nethogs -t {{seconds}}
  ```

netstat

Displays various networks related information such as open connections, open socket ports etc.

- List all ports:

  ```
  netstat -a
  ```

- List all listening ports:

  ```
  netstat -l
  ```

- List listening TCP ports:

  ```
  netstat -t
  ```

- Display PID and program names:

  ```
  netstat -p
  ```

- List information continuously:

  ```
  netstat -c
  ```

- List routes and do not resolve IP to hostname:

  ```
  netstat -rn
  ```

- List listening TCP and UDP ports (+ user and process if you're root):

  ```
  netstat -lepunt
  ```

nl

A utility for numbering lines, either from a file, or from standard input.

- Number non-blank lines in a file:

  ```
  nl {{file}}
  ```

- Read from standard output:

  ```
  cat {{file}} | nl {{options}} -
  ```

- Number only the lines with printable text:

  ```
  nl -t {{file}}
  ```

- Number all lines including blank lines:

  ```
  nl -b a {{file}}
  ```

- Number only the body lines that match a basic regular expression (BRE) pattern:

  ```
  nl -b p'FooBar[0-9]' {{file}}
  ```

nm

List symbol names in object files.

- List global (extern) functions in a file (prefixed with T):

  ```
  nm -g {{file.o}}
  ```

- Demangle C++ symbols (make them readable):

  ```
  nm --demangle {{file.o}}
  ```

- List only undefined symbols in a file:

  ```
  nm -u {{file.o}}
  ```

- List all symbols, even debugging symbols:

  ```
  nm -a {{file.o}}
  ```

nmcli

A command line tool for controlling NetworkManager.

- List all NetworkManager connections (shows name, uuid, type and device):

  ```
  nmcli connection
  ```

- Activate a connection by specifying an uuid:

  ```
  nmcli connection up uuid {{uuid}}
  ```

- Deactivate a connection:

  ```
  nmcli connection down uuid {{uuid}}
  ```

- Print statuses of network interfaces:

  ```
  nmcli device status
  ```

nmon

A system administrator, tuner, and benchmark tool.

- Start nmon:

  ```
  nmon
  ```

- Save records to file ("-s 300 -c 288" by default):

  ```
  nmon -f
  ```

- Save records to file with a total of 240 measurements, by taking 30 seconds between each measurement:

  ```
  nmon -f -s {{30}} -c {{240}}
  ```

nmtui

Text user interface for controlling NetworkManager.

Use arrow keys to navigate, enter to select an option.

- Open the user interface:

  ```
  nmtui
  ```

- Show a list of available connections, with the option to activate or deactivate them:

  ```
  nmtui connect
  ```

- Connect to a given network:

  ```
  nmtui connect {{name|uuid|device|SSID}}
  ```

- Edit/Add/Delete a given network:

  ```
  nmtui edit {{name|id}}
  ```

- Set the system hostname:

  ```
  nmtui hostname
  ```

notify-send

Uses the current desktop environment's notification system to create a notification.

- Show a notification with the title "Test" and the content "This is a test":

  ```
  notify-send {{"Test"}} {{"This is a test"}}
  ```

- Show a notification with a custom icon:

  ```
  notify-send -i {{icon.png}} {{"Test"}} {{"This is a test"}}
  ```

- Show a notification for 5 seconds:

  ```
  notify-send -t 5000 {{"Test"}} {{"This is a test"}}
  ```

ntfsfix

Fix common problems on an NTFS partition.

- Fix a given NTFS partition:

  ```
  sudo ntfsfix {{/dev/sdb2}}
  ```

objdump

View information about object files.

- Display the file header information:

  ```
  objdump -f {{binary}}
  ```

- Display the dis-assembled output of executable sections:

  ```
  objdump -d {{binary}}
  ```

- Display a complete binary hex dump of all sections:

  ```
  objdump -s {{binary}}
  ```

pacaur

A utility for Arch Linux to build and install packages from the Arch User Repository.

- Synchronize and update all packages (includes AUR):

  ```
  pacaur -Syu
  ```

- Synchronize and update only AUR packages:

  ```
  pacaur -Syua
  ```

- Install a new package (includes AUR):

  ```
  pacaur -S {{package_name}}
  ```

- Remove a package and its dependencies (includes AUR packages):

  ```
  pacaur -Rs {{package_name}}
  ```

- Search the package database for a keyword (includes AUR):

  ```
  pacaur -Ss {{keyword}}
  ```

- List all currently installed packages (includes AUR packages):

  ```
  pacaur -Qs
  ```

pacman

Arch Linux package manager utility.

- Synchronize and update all packages:

  ```
  pacman -Syu
  ```

- Install a new package:

  ```
  pacman -S {{package_name}}
  ```

- Remove a package and its dependencies:

  ```
  pacman -Rs {{package_name}}
  ```

- Search the package database for a regular expression or keyword:

  ```
  pacman -Ss "{{search_pattern}}"
  ```

- List installed packages and versions:

  ```
  pacman -Q
  ```

- List only the explicitly installed packages and versions:

  ```
  pacman -Qe
  ```

- Find which package owns a certain file:

  ```
  pacman -Qo {{filename}}
  ```

- Empty package cache to free up space:

  ```
  pacman -Scc
  ```

pasuspender

Temporarily suspends `pulseaudio` while another command is running to allow access to alsa.

- Suspend pulseaudio while running `jackd`:

  ```
  pasuspender -- {{jackd -d alsa --device hw:0}}
  ```

pdfgrep

Search text in PDF files.

- Find lines that match pattern in a PDF:

```
pdfgrep {{pattern}} {{file.pdf}}
```

- Include file name and page number for each matched line:

```
pdfgrep --with-filename --page-number {{pattern}}
{{file.pdf}}
```

- Do a case insensitive search for lines that begin with "foo" and return the first 3 matches:

```
pdfgrep --max-count {{3}} --ignore-case {{'^foo'}}
{{file.pdf}}
```

- Find pattern in files with a .pdf extension in the current directory recursively:

```
pdfgrep --recursive {{pattern}}
```

- Find pattern on files that match a specific glob in the current directory recursively:

```
pdfgrep --recursive --include {{'*book.pdf'}} {{pattern}}
```

perf

Framework for linux performance counter measurements.

- Display basic performance counter stats for a command:

  `perf stat {{gcc hello.c}}`

- Display system-wide real time performance counter profile:

  `sudo perf top`

- Run a command and record its profile into "perf.data":

  `sudo perf record {{command}}`

- Read "perf.data" (created by `perf record`) and display the profile:

  `sudo perf report`

pkgadd

Add a package to a CRUX system.

- Install a local software package:

  ```
  pkgadd {{package_name}}
  ```

- Update an already installed package from a local package:

  ```
  pkgadd -u {{package_name}}
  ```

pkginfo

Query the package database on a CRUX system.

- List installed packages and their versions:

  ```
  pkginfo -i
  ```

- List files owned by a package:

  ```
  pkginfo -l {{package_name}}
  ```

- List the owner(s) of files matching a pattern:

  ```
  pkginfo -o {{pattern}}
  ```

- Print the footprint of a file:

  ```
  pkginfo -f {{file}}
  ```

pkgmk

Make a binary package for use with pkgadd on CRUX.

- Make and download a package:

  ```
  pkgmk -d
  ```

- Install the package after making it:

  ```
  pkgmk -d -i
  ```

- Upgrade the package after making it:

  ```
  pkgmk -d -u
  ```

- Ignore the footprint when making a package:

  ```
  pkgmk -d -if
  ```

- Ignore the MD5 sum when making a package:

  ```
  pkgmk -d -im
  ```

- Update the package's footprint:

  ```
  pkgmk -uf
  ```

pkgrm

Remove a package from a CRUX system.

- Remove an installed package:

  ```
  pkgrm {{package_name}}
  ```

popd

Remove a directory placed on the directory stack by the `pushd` command.

- Remove the top directory from the stack and cd to it:

  ```
  popd
  ```

- Remove the Nth directory (starting from zero to the left from the list printed with `dirs`):

  ```
  popd +N
  ```

- Remove the Nth directory (starting from zero to the right from the list printed with `dirs`):

  ```
  popd -N
  ```

ports

Update/list the ports tree on a CRUX system.

- Update the ports tree:

  ```
  ports -u
  ```

- List the ports in the current tree:

  ```
  ports -l
  ```

- Check the differences between installed packages and the ports tree:

  ```
  ports -d
  ```

print

An alias to a `run-mailcap`'s action print.

Originally `run-mailcap` is used to process mime-type/file.

- Print action can be used to print any file on default run-mailcap tool:

  ```
  print {{filename}}
  ```

- With `run-mailcap`:

  ```
  run-mailcap --action=print {{filename}}
  ```

prt-get

The advanced CRUX package manager.

- Install a package:

  ```
  prt-get install {{package_name}}
  ```

- Install a package with dependency handling:

  ```
  prt-get depinst {{package_name}}
  ```

- Update a package manually:

  ```
  prt-get upgrade {{package_name}}
  ```

- Remove a package:

  ```
  prt-get remove {{package_name}}
  ```

- Upgrade the system from the local ports tree:

  ```
  prt-get sysup
  ```

- Search the ports tree:

  ```
  prt-get search {{package_name}}
  ```

- Search for a file in a package:

  ```
  prt-get fsearch {{file}}
  ```

pstree

A convenient tool to show running processes as a tree.

- Display a tree of processes:

  ```
  pstree
  ```

- Display a tree of processes with PIDs:

  ```
  pstree -p
  ```

- Display all process trees rooted at processes owned by specified user:

  ```
  pstree {{user}}
  ```

pulseaudio

The pulseaudio sound system daemon and manager.

- Check if pulseaudio is running (a non-zero exit code means it is not running):

```
pulseaudio --check
```

- Start the pulseaudio daemon in the background:

```
pulseaudio --start
```

- Kill the running pulseaudio daemon:

```
pulseaudio --kill
```

- List available modules:

```
pulseaudio --dump-modules
```

- Load a module into the currently running daemon with the specified arguments:

```
pulseaudio --load="{{module_name}} {{arguments}}"
```

pushd

Place a directory on a stack so it can be accessed later.

See also `popd` to switch back to original directory.

- Switch to directory and push it on the stack:

  ```
  pushd < {{directory}}
  ```

- Switch first and second directories on the stack:

  ```
  pushd
  ```

- Rotate stack by making the 5th element the top of the stack:

  ```
  pushd +4
  ```

pvcreate

Initialize a physical volume (disk or partition) for use by the Logical Volume Manager (LVM).

- Initialize the `/dev/sda1` volume for use by LVM:

  ```
  pvcreate {{/dev/sda1}}
  ```

- Force the creation without any confirmation prompts:

  ```
  pvcreate --force {{/dev/sda1}}
  ```

pwgen

Generate pronounceable passwords.

- Generate random password with s[y]mbols:

  ```
  pwgen -y {{length}}
  ```

- Generate secure, hard-to-memorize passwords:

  ```
  pwgen -s {{length}}
  ```

- Generate password with at least one capital letter in them:

  ```
  pwgen -c {{length}}
  ```

qsub

Submits a script to the queue management system TORQUE.

- Submit a script with default settings (depends on TORQUE settings):

  ```
  qsub {{script.sh}}
  ```

- Submit a script with a specified wallclock runtime limit of 1 hour, 2 minutes and 3 seconds:

  ```
  qsub -l walltime={{1}}:{{2}}:{{3}} {{script.sh}}
  ```

- Submit a script that is executed on 2 nodes using 4 cores per node:

  ```
  qsub -l nodes={{2}}:ppn={{4}} {{script.sh}}
  ```

- Submit a script to a specific queue. Note that different queues can have different maximum and minimum runtime limits:

  ```
  qsub -q {{queue_name}} {{script.sh}}
  ```

quotacheck

Scan a filesystem for disk usage; create, check and repair quota files.

It is best to run quota check with quotas turned off to prevent damage or loss to quota files.

- Check quotas on all mounted non-NFS filesystems:

  ```
  sudo quotacheck --all
  ```

- Force check even if quotas are enabled (this can cause damage or loss to quota files):

  ```
  sudo quotacheck --force {{mountpoint}}
  ```

- Check quotas on a given filesystem in debug mode:

  ```
  sudo quotacheck --debug {{mountpoint}}
  ```

- Check quotas on a given filesystem, displaying the progress:

  ```
  sudo quotacheck --verbose {{mountpoint}}
  ```

- Check user quotas:

  ```
  sudo quotacheck --user {{user}} {{mountpoint}}
  ```

- Check group quotas:

  ```
  sudo quotacheck --group {{group}} {{mountpoint}}
  ```

rdesktop

Remote Desktop Protocol client.

It can be used to connect the remote computer using the RDP protocol.

- Connect to a remote computer (default port is 3389):

```
rdesktop -u {{username}} -p {{password}} {{host:port}}
```

- Simple Examples:

```
rdesktop -u Administrator -p passwd123 192.168.1.111:3389
```

- Connect to a remote computer with full screen (press `Ctrl + Alt + Enter` to exist):

```
rdesktop -u {{username}} -p {{password}} -f {{host:port}}
```

- Use the customed resolution (use the letter 'x' between the number):

```
rdesktop -u {{username}} -p {{password}} -g 1366x768
{{host:port}}
```

- Connect to a remote computer using domain user:

```
rdesktop -u {{username}} -p {{password}} -d {{domainname}}
{{host:port}}
```

- Use the 16 bit color (speed up):

```
rdesktop -u {{username}} -p {{password}} -a 16 {{host:port}}
```

reboot

Reboot the system.

- Reboot immediately:

  ```
  reboot
  ```

- Reboot immediately without gracefully shutdown:

  ```
  reboot -f
  ```

repquota

Display a summary of existing file quotas for a filesystem.

- Report stats for all quotas in use:

  ```
  sudo repquota -all
  ```

- Report quota stats for all users, even those who aren't using any of their quota:

  ```
  sudo repquota -v {{filesystem}}
  ```

- Report on quotas for users only:

  ```
  repquota --user {{filesystem}}
  ```

- Report on quotas for groups only:

  ```
  sudo repquota --group {{filesystem}}
  ```

- Report on used quota and limits in a human-readable format:

  ```
  sudo repquota --human-readable {{filesystem}}
  ```

- Report on all quotas for users and groups in a human-readable format:

  ```
  sudo repquota -augs
  ```

rofi

An application launcher and window switcher.

- Show the list of apps:

  ```
  rofi -show drun
  ```

- Show the list of all commands:

  ```
  rofi -show run
  ```

- Switch between windows:

  ```
  rofi -show window
  ```

rpm

RPM Package Manager.

- Show version of httpd package:

  ```
  rpm -q {{httpd}}
  ```

- List versions of all matching packages:

  ```
  rpm -qa '{{mariadb*}}'
  ```

- Identify owner of a file and show version of the package:

  ```
  rpm -qf {{/etc/postfix/main.cf}}
  ```

- List package-owned files:

  ```
  rpm -ql {{kernel}}
  ```

- Show scriptlets from an RPM file:

  ```
  rpm -qp --scripts {{some.rpm}}
  ```

- Show changed, missing and/or incorrectly installed files of matching packages:

  ```
  rpm -Va '{{php-*}}'
  ```

rspamc

Command line client for rspamd servers.

- Train the bayesian filter to recognise an email as spam:

  ```
  rspamc learn_spam {{path/to/email_file}}
  ```

- Train the bayesian filter to recognise an email as ham:

  ```
  rspamc learn_ham {{path/to/email_file}}
  ```

- Generate a manual report on an email:

  ```
  rspamc symbols {{path/to/email_file}}
  ```

- Show server statistics:

  ```
  rspamc stat
  ```

rtorrent

Download torrents over the command line.

- Add a torrent file or magnet to be downloaded:

  `rtorrent {{torrent_or_magnet}}`

- Start the download:

  `<Ctrl>S`

- View details about downloading torrent:

  `->`

- Close rtorrent safely:

  `<Ctrl>Q`

run-mailcap

Run MailCap Programs.

Run mailcap view, see, edit, compose, print - execute programs via entries in the mailcap file (or any of its aliases) will use the given action to process each mime-type/file.

- Individual actions/programs on run-mailcap can be invoked with action flag:

  ```
  run-mailcap --action=ACTION [--option[=value]]
  ```

- In simple language:

  ```
  run-mailcap --action=ACTION {{filename}}
  ```

- Turn on extra information:

  ```
  run-mailcap --action=ACTION --debug {{filename}}
  ```

- Ignore any "copiousoutput" directive and forward output to STD-OUT:

  ```
  run-mailcap --action=ACTION --nopager {{filename}}
  ```

- Display the found command without actually executing it:

  ```
  run-mailcap --action=ACTION --norun {{filename}}
  ```

runit

3-stage init system.

- Start runit's 3-stage init scheme:

  ```
  runit
  ```

- Shut down runit:

  ```
  kill --CONT {{runit_pid}}
  ```

runsv

Start and manage a runit service.

- Start a runit service as the current user:

  ```
  runsv {{path/to/service}}
  ```

- Start a runit service as root:

  ```
  sudo runsv {{path/to/service}}
  ```

runsvchdir

Change the directory `runsvdir` uses by default.

- Switch `runsvdir` directories:

  ```
  sudo runsvchdir {{/path/to/directory}}
  ```

runsvdir

Run an entire directory of services.

- Start and manage all services in a directory as the current user:

  ```
  runsvdir {{path/to/services}}
  ```

- Start and manage all services in a directory as root:

  ```
  sudo runsvdir {{path/to/services}}
  ```

- Start services in separate sessions:

  ```
  runsvdir -P {{path/to/services}}
  ```

sar

Monitor performance of various Linux subsystems.

- Report I/O and transfer rate issued to physical devices, one per second (press CTRL +C to quit):

```
sar -b {{1}}
```

- Report a total of 10 network device statistics, one per 2 seconds:

```
sar -n DEV {{2}} {{10}}
```

- Report CPU utilization, one per 2 seconds:

```
sar -u ALL {{2}}
```

- Report a total of 20 memory utilization statistics, one per second:

```
sar -r ALL {{1}} {{20}}
```

- Report the run queue length and load averages, one per second:

```
sar -q {{1}}
```

- Report paging statistics, one per 5 seconds:

```
sar -B {{5}}
```

see

Alias to `run-mailcap`'s view.

An alias to a `run-mailcap`'s action print.

- See action can be used to view any file (usually image) on default mailcap explorer:

  ```
  see {{filename}}
  ```

- Using with `run-mailcap`:

  ```
  run-mailcap --action=view {{filename}}
  ```

sensible-browser

Open the default browser.

- Open a new window of the default browser:

  ```
  sensible-browser
  ```

- Open a url in the default browser:

  ```
  sensible-browser {{url}}
  ```

service

Manage services by running init scripts.

The full script path should be omitted (/etc/init.d/ is assumed).

- Start/Stop/Restart/Reload service (start/stop should always be available):

  ```
  service {{init_script}} {{start|stop|restart|reload}}
  ```

- Do a full restart (runs script twice with start and stop):

  ```
  service {{init_script}} --full-restart
  ```

- Show the current status of a service:

  ```
  service {{init_script}} status
  ```

- List the status of all services:

  ```
  service --status-all
  ```

setfacl

Set file access control lists (ACL).

- Modify ACL of a file for user with read and write access:

  ```
  setfacl -m u:{{username}}:rw {{file}}
  ```

- Modify default ACL of a file for all users:

  ```
  setfacl -d -m u::rw {{file}}
  ```

- Remove ACL of a file for an user:

  ```
  setfacl -x u:{{username}} {{file}}
  ```

- Remove all ACL entries of a file:

  ```
  setfacl -b {{file}}
  ```

sha1sum

Calculate SHA1 cryptographic checksums.

- Calculate the SHA1 checksum for a file:

  ```
  sha1sum {{filename1}}
  ```

- Calculate SHA1 checksums for multiple files:

  ```
  sha1sum {{filename1}} {{filename2}}
  ```

- Read a file of SHA1 sums and verify all files have matching checksums:

  ```
  sha1sum -c {{filename.sha1}}
  ```

sha224sum

Calculate SHA224 cryptographic checksums.

- Calculate the SHA224 checksum for a file:

  ```
  sha224sum {{filename1}}
  ```

- Calculate SHA224 checksums for multiple files:

  ```
  sha224sum {{filename1}} {{filename2}}
  ```

- Read a file of SHA224 sums and verify all files have matching checksums:

  ```
  sha224sum -c {{filename.sha224}}
  ```

sha256sum

Calculate SHA256 cryptographic checksums.

- Calculate the SHA256 checksum for a file:

  ```
  sha256sum {{filename1}}
  ```

- Calculate SHA256 checksums for multiple files:

  ```
  sha256sum {{filename1}} {{filename2}}
  ```

- Read a file of SHA256 sums and verify all files have matching checksums:

  ```
  sha256sum -c {{filename.sha256}}
  ```

sha384sum

Calculate SHA384 cryptographic checksums.

- Calculate the SHA384 checksum for a file:

  ```
  sha384sum {{filename1}}
  ```

- Calculate SHA384 checksums for multiple files:

  ```
  sha384sum {{filename1}} {{filename2}}
  ```

- Read a file of SHA384 sums and verify all files have matching checksums:

  ```
  sha384sum -c {{filename.sha384}}
  ```

sha512sum

Calculate SHA512 cryptographic checksums.

- Calculate the SHA512 checksum for a file:

  ```
  sha512sum {{filename1}}
  ```

- Calculate SHA512 checksums for multiple files:

  ```
  sha512sum {{filename1}} {{filename2}}
  ```

- Read a file of SHA512 sums and verify all files have matching checksums:

  ```
  sha512sum -c {{filename.sha512}}
  ```

shasum

Calculate or check cryptographic SHA checksums.

- Calculate the SHA1 checksum for a file:

  ```
  shasum {{filename}}
  ```

- Calculate the SHA256 checksum for a file:

  ```
  shasum --algorithm 256 {{filename}}
  ```

- Calculate the SHA512 checksum for multiple files:

  ```
  shasum --algorithm 512 {{filename1}} {{filename2}}
  ```

- Check a file with a list of sums against the directory's files:

  ```
  shasum --check {{list_file}}
  ```

- Calculate the SHA1 checksum from stdin:

  ```
  {{somecommand}} | shasum
  ```

shuf

Generate random permutations.

- Randomize the order of lines in a file and output the result:

  ```
  shuf {{filename}}
  ```

- Only output the first n entries of the result:

  ```
  shuf -n {{n}} {{filename}}
  ```

- Write output to another file:

  ```
  shuf -o {{another_filename}} {{filename}}
  ```

- Generate random numbers in range:

  ```
  shuf -i {{low}}-{{high}}
  ```

shutdown

Shutdown and reboot the system.

- Power off (halt) immediately:

  ```
  shutdown -h now
  ```

- Reboot immediately:

  ```
  shutdown -r now
  ```

- Reboot in 5 minutes:

  ```
  shutdown -r +{{5}} &
  ```

- Shutdown at 1:00 pm (Uses 24h clock):

  ```
  shutdown -h 13:00
  ```

- Cancel a pending shutdown/reboot operation:

  ```
  shutdown -c
  ```

smbclient

FTP-like client to access SMB/CIFS resources on servers.

- Connect to a share (user will be prompted for password; `exit` to quit the session):

```
smbclient {{//server/share}}
```

- Connect with a different username:

```
smbclient {{//server/share}} --user {{username}}
```

- Connect with a username and password:

```
smbclient {{//server/share}} --user {{username%password}}
```

- Download a file from the server:

```
smbclient {{//server/share}} --directory {{path/to/folder}}
--command "get {{file.txt}}"
```

- Upload a file to the server:

```
smbclient {{//server/share}} --directory {{path/to/folder}}
--command "put {{file.txt}}"
```

snap

Tool for managing the "snap" self-contained software packages.

Similar to what `apt` is for ".deb".

- Search for a package:

  ```
  snap find {{package_name}}
  ```

- Install a package:

  ```
  snap install {{package_name}}
  ```

- Display basic information about installed snap software:

  ```
  snap list
  ```

- Uninstall a package:

  ```
  snap remove {{package_name}}
  ```

- Check for recent snap changes in the system:

  ```
  snap changes
  ```

sort

Sort lines of text files.

- Sort a file in ascending order:

  ```
  sort {{filename}}
  ```

- Sort a file in descending order:

  ```
  sort -r {{filename}}
  ```

- Sort a file using numeric rather than alphabetic order:

  ```
  sort -n {{filename}}
  ```

- Sort the passwd file by the 3rd field, numerically:

  ```
  sort -t: -k 3n /etc/passwd
  ```

- Sort human-readable numbers (in this case the 5th field of `ls -lh`):

  ```
  ls -lh | sort -h -k 5
  ```

ss

Utility to investigate sockets.

- Show all TCP/UDP/RAW/UNIX sockets:

```
ss -a {{-t|-u|-w|-x}}
```

- Filter TCP sockets by states, only/exclude:

```
ss {{state/exclude}} {{bucket/big/connected/
synchronized/...}}
```

- Show all TCP sockets connected to the local HTTPS port (443):

```
ss -t src :{{443}}
```

- Show all TCP sockets along with processes connected to a remote ssh port:

```
ss -pt dst :{{ssh}}
```

- Show all UDP sockets connected on specific source and destination ports:

```
ss -u 'sport == :{{source_port}} and dport == :
{{destination_port}}'
```

- Show all TCP IPv4 sockets locally connected on the subnet 192.168.0.0/16:

```
ss -4t src {{192.168/16}}
```

ssh-add

Manage loaded ssh keys in the ssh-agent.

Ensure that ssh-agent is up and running for the keys to be loaded in it.

- Add the default ssh keys in "~/.ssh" to the ssh-agent:

  ```
  ssh-add
  ```

- Add a specific key to the ssh-agent:

  ```
  ssh-add {{path/to/private_key}}
  ```

- List fingerprints of currently loaded keys:

  ```
  ssh-add -l
  ```

- Delete a key from the ssh-agent:

  ```
  ssh-add -d {{path/to/private_key}}
  ```

- Delete all currently loaded keys from the ssh-agent:

  ```
  ssh-add -D
  ```

sshuttle

Transparent proxy server that tunnels traffic over an SSH connection.

Doesn't require admin, or any special setup on the remote SSH server.

- Forward all IPv4 TCP traffic via a remote SSH server:

```
sshuttle --remote={{username}}@{{sshserver}} {{0.0.0.0/0}}
```

- Forward all IPv4 TCP and DNS traffic:

```
sshuttle --dns --remote={{username}}@{{sshserver}}
{{0.0.0.0/0}}
```

- Use the tproxy method to forward all IPv4 and IPv6 traffic:

```
sudo sshuttle --method=tproxy --remote={{username}}@
{{sshserver}} {{0.0.0.0/0}} {{::/0}} --exclude=
{{your_local_ip_address}} --exclude={{ssh_server_ip_address}}
```

stat

Display file and filesystem information.

- Show file properties such as size, permissions, creation and access dates among others:

  ```
  stat {{file}}
  ```

- Same as above but in a more concise way:

  ```
  stat -t {{file}}
  ```

- Show filesystem information:

  ```
  stat -f {{file}}
  ```

- Show only octal file permissions:

  ```
  stat -c "%a %n" {{file}}
  ```

- Show owner and group of the file:

  ```
  stat -c "%U %G" {{file}}
  ```

- Show the size of the file in bytes:

  ```
  stat -c "%s %n" {{file}}
  ```

strace

Troubleshooting tool for tracing system calls.

- Start tracing a specific process by its PID:

  ```
  strace -p {{pid}}
  ```

- Trace a process and filter output by system call:

  ```
  strace -p {{pid}} -e {{system call name}}
  ```

- Count time, calls, and errors for each system call and report a summary on program exit:

  ```
  strace -p {{pid}} -c
  ```

- Show the time spent in every system call:

  ```
  strace -p {{pid}} -T
  ```

- Start tracing a program by executing it:

  ```
  strace {{program}}
  ```

- Start tracing file operations of a program:

  ```
  strace -e trace=file {{program}}
  ```

sv

Control a running runsv service.

- Start a service:

  ```
  sudo sv up {{path/to/service}}
  ```

- Stop a service:

  ```
  sudo sv down {{path/to/service}}
  ```

- Get service status:

  ```
  sudo sv status {{path/to/service}}
  ```

sysctl

List and change kernel runtime variables.

- Show all available variables and their values:

  ```
  sysctl -a
  ```

- Set a changeable kernel state variable:

  ```
  sysctl -w {{section.tunable}}={{value}}
  ```

- Get currently open file handlers:

  ```
  sysctl fs.file-nr
  ```

- Get limit for simultaneous open files:

  ```
  sysctl fs.file-max
  ```

- Apply changes from /etc/sysctl.conf:

  ```
  sysctl -p
  ```

systemctl

Control the systemd system and service manager.

- List failed units:

  ```
  systemctl --failed
  ```

- Start/Stop/Restart/Reload a service:

  ```
  systemctl start/stop/restart/reload {{unit}}
  ```

- Show the status of a unit:

  ```
  systemctl status {{unit}}
  ```

- Enable/Disable a unit to be started on bootup:

  ```
  systemctl enable/disable {{unit}}
  ```

- Mask/Unmask a unit, prevent it to be started on bootup:

  ```
  systemctl mask/unmask {{unit}}
  ```

- Reload systemd, scanning for new or changed units:

  ```
  systemctl daemon-reload
  ```

systemd-analyze

Show timing details about the boot process of units (services, mount points, devices, sockets).

- List time of each unit to start up:

  ```
  systemd-analyze blame
  ```

- Print a tree of the time critical chain of units:

  ```
  systemd-analyze critical-chain
  ```

tcpflow

Capture TCP traffic for debugging and analysis.

- Show all data on the given interface and port:

  ```
  tcpflow -c -i {{eth0}} port {{80}}
  ```

timedatectl

Control the system time and date.

- To check the current system clock time:

  ```
  timedatectl
  ```

- To set the local time of the system clock directly:

  ```
  timedatectl set-time {{"yyyy-MM-dd hh:mm:ss"}}
  ```

- To list available timezones:

  ```
  timedatectl list-timezones
  ```

- To change timezones:

  ```
  timedatectl set-timezone {{timezone}}
  ```

- To enable Network Time Protocol (NTP) syncing:

  ```
  timedatectl set-ntp on
  ```

timeout

Run a command with a time limit.

- Run `sleep 10` and kill it, if it's running after 3 seconds:

  ```
  timeout {{3s}} {{sleep 10}}
  ```

- Specify the signal to be sent to the command after the time limit expires. (By default, TERM is sent):

  ```
  timeout --signal {{INT}} {{5s}} {{sleep 10}}
  ```

tomb

Manage encrypted storage folders that can be safely transported and hidden in a filesystem.

- Create a new tomb with an initial size of 100MB:

```
tomb dig -s {{100}} {{encrypted_folder.tomb}}
```

- Create a new key file that can be used to lock a tomb; user will be prompted for a password for the key:

```
tomb forge {{encrypted_folder.tomb.key}}
```

- Initialize and lock an empty tomb using a key made with `forge`:

```
tomb lock {{encrypted_folder.tomb}} -k
{{encrypted_folder.tomb.key}}
```

- Mount a tomb (by default in /media) using its key, making it usable as a regular filesystem folder:

```
tomb open {{encrypted_folder.tomb}} -k
{{encrypted_folder.tomb.key}}
```

- Close a tomb (fails if the tomb is being used by a process):

```
tomb close {{encrypted_folder.tomb}}
```

- Forcefully close all open tombs, killing any applications using them:

```
tomb slam all
```

- List all open tombs:

```
tomb list
```

top

Display dynamic real-time information about running processes.

- Start top:

  ```
  top
  ```

- Do not show any idle or zombie processes:

  ```
  top -i
  ```

- Show only processes owned by given user:

  ```
  top -u {{user_name}}
  ```

- Show only the processes with the given PID(s), passed as a comma-separated list. (Normally you wouldn't know PIDs off hand. This example picks the PIDs from the process name):

  ```
  top -p $(pgrep -d ',' {{process_name}})
  ```

- Get help about interactive commands:

  ```
  ?
  ```

tree

Show the contents of the current directory as a tree.

- Show files and directories up to 'num' levels of depth (where 1 means the current directory):

  ```
  tree -L {{num}}
  ```

- Show directories only:

  ```
  tree -d
  ```

- Show hidden files too:

  ```
  tree -a
  ```

- Print the tree without indentation lines, showing the full path instead (use `-N` to not escape whitespace and special characters):

  ```
  tree -i -f
  ```

- Print the size of each node next to it, in human-readable format:

  ```
  tree -s -h
  ```

- Filter the tree using a wildcard (glob) pattern:

  ```
  tree -P {{*.txt}}
  ```

- Ignore entries that match a wildcard (glob) pattern:

  ```
  tree -I {{*.txt}}
  ```

ufw

Uncomplicated Firewall.

Frontend for iptables aiming to make configuration of a firewall easier.

- Enable ufw:

  ```
  ufw enable
  ```

- Disable ufw:

  ```
  ufw disable
  ```

- Show ufw rules, along with their numbers:

  ```
  ufw status numbered
  ```

- Allow incoming traffic on port 5432 on this host with a comment identifying the service:

  ```
  ufw allow {{5432}} comment {{"Service"}}
  ```

- Allow only TCP traffic from 192.168.0.4 to any address on this host, on port 22:

  ```
  ufw allow proto {{tcp}} from {{192.168.0.4}} to {{any}} port {{22}}
  ```

- Deny traffic on port 80 on this host:

  ```
  ufw deny {{80}}
  ```

- Deny all UDP traffic to port 22:

  ```
  ufw deny proto {{udp}} from {{any}} to {{any}} port {{22}}
  ```

- Delete a particular rule. The rule number can be retrieved from the `ufw status numbered` command:

  ```
  ufw delete {{rule_number}}
  ```

ulimit

Get and set user limits.

- Get the properties of all the user limits:

  ```
  ulimit -a
  ```

- Get hard limit for the number of simultaneously opened files:

  ```
  ulimit -H -n
  ```

- Get soft limit for the number of simultaneously opened files:

  ```
  ulimit -S -n
  ```

- Set max per-user process limit:

  ```
  ulimit -u 30
  ```

umask

Manage the read/write/execute permissions that are masked out (i.e. restricted) for newly created files by the user.

- Display the current mask in octal notation:

  ```
  umask
  ```

- Display the current mask in symbolic (human-readable) mode:

  ```
  umask -S
  ```

- Change the mask symbolically to allow read permission for all users (the rest of the mask bits are unchanged):

  ```
  umask {{a+r}}
  ```

- Set the mask (using octal) to restrict no permissions for the file's owner, and restrict all permissions for everyone else:

  ```
  umask {{077}}
  ```

uname

Print details about the current machine and the operating system running on it.

Note: for additional information about the operating system, try the `lsb_release` command.

- Print hardware-related information: machine and processor:

  ```
  uname -mp
  ```

- Print software-related information: operating system, release number, and version:

  ```
  uname -srv
  ```

- Print the nodename (hostname) of the system:

  ```
  uname -n
  ```

- Print all available system information (hardware, software, nodename):

  ```
  uname -a
  ```

unexpand

Convert spaces to tabs.

- Convert blanks in each file to tabs, writing to standard output:

  ```
  unexpand {{file}}
  ```

- Convert blanks to tabs, reading from standard output:

  ```
  unexpand
  ```

- Convert all blanks, instead of just initial blanks:

  ```
  unexpand -a {{file}}
  ```

- Convert only leading sequences of blanks (overrides -a):

  ```
  unexpand --first-only {{file}}
  ```

- Have tabs a certain number of characters apart, not 8 (enables -a):

  ```
  unexpand -t {{number}} {{file}}
  ```

update-alternatives

A convenient tool for maintaining symbolic links to determine default commands.

- Add a symbolic link:

```
sudo update-alternatives --install {{path/to/symlink}}
{{command_name}} {{path/to/command_binary}} {{priority}}
```

- Configure a symbolic link for "java":

```
sudo update-alternatives --config {{java}}
```

- Remove a symbolic link:

```
sudo update-alternatives --remove {{java}} {{/opt/java/
jdk1.8.0_102/bin/java}}
```

- Display information about a specified command:

```
update-alternatives --display {{java}}
```

update-rc.d

Install and remove services which are System-V style init script links.

Init scripts are in the /etc/init.d/.

- Install a service:

  ```
  update-rc.d {{mysql}} defaults
  ```

- Enable a service:

  ```
  update-rc.d {{mysql}} enable
  ```

- Disable a service:

  ```
  update-rc.d {{mysql}} disable
  ```

- Forcibly remove a service:

  ```
  update-rc.d -f {{mysql}} remove
  ```

useradd

Create a new user.

- Create new user:

  ```
  useradd {{name}}
  ```

- Create new user with a default home directory:

  ```
  useradd --create-home {{name}}
  ```

- Create new user with specified shell:

  ```
  useradd --shell {{/path/to/shell}} {{name}}
  ```

- Create new user belonging to additional groups (mind the lack of whitespace):

  ```
  useradd --groups {{group1,group2}} {{name}}
  ```

- Create new system user without a home directory:

  ```
  useradd --no-create-home --system {{name}}
  ```

userdel

Remove a user.

- Remove a user and their home directory:

  ```
  userdel -r {{name}}
  ```

usermod

Modifies a user account.

- Change a user's name:

  ```
  usermod -l {{newname}} {{user}}
  ```

- Add user to supplementary groups (mind the whitespace):

  ```
  usermod -a -G {{group1,group2}} {{user}}
  ```

- Create a new home directory for a user and move their files to it:

  ```
  usermod -m -d {{/path/to/home}} {{user}}
  ```

vgcreate

Create volume groups combining multiple mass-storage devices.

- Create a new volume group called vg1 using the `/dev/sda1` device:

  ```
  vgcreate {{vg1}} {{/dev/sda1}}
  ```

- Create a new volume group called vg1 using multiple devices:

  ```
  vgcreate {{vg1}} {{/dev/sda1}} {{/dev/sdb1}} {{/dev/sdc1}}
  ```

vncserver

Launches a VNC (Virtual Network Computing) desktop.

- Launch a VNC Server on next available display:

  ```
  vncserver
  ```

- Launch a VNC Server with specific screen geometry:

  ```
  vncserver --geometry {{width}}x{{height}}
  ```

- Kill an instance of VNC Server running on a specific display:

  ```
  vncserver --kill :{{display_number}}
  ```

wall

Write a message on the terminals of users currently logged in.

- Send a message:

  ```
  echo "{{message}}" | wall
  ```

- Send a message from a file:

  ```
  wall {{file}}
  ```

- Send a message with timeout (default 300):

  ```
  wall -t {{seconds}} {{file}}
  ```

watch

Execute a command repeatedly, and monitor the output in full-screen mode.

- Monitor files in the current folder:

  ```
  watch {{ls}}
  ```

- Monitor disk space and highlight the changes:

  ```
  watch -d {{df}}
  ```

- Monitor "node" processes, refreshing every 3 seconds:

  ```
  watch -n {{3}} "{{ps aux | grep node}}"
  ```

whatis

Display one-line descriptions from manual pages.

- Display a description from a man page:

  ```
  whatis {{command}}
  ```

- Don't cut the description off at the end of the line:

  ```
  whatis --long {{command}}
  ```

- Display descriptions for all commands matching a glob:

  ```
  whatis --wildcard {{net*}}
  ```

- Search man page descriptions with a regular expression:

  ```
  whatis --regex '{{wish[0-9]\.[0-9]}}'
  ```

whereis

Locate the binary, source, and manual page files for a command.

- Locate binary, source and man pages for ssh:

  ```
  whereis {{ssh}}
  ```

- Locate binary and man pages for ls:

  ```
  whereis -bm {{ls}}
  ```

- Locate source of gcc and man pages for git:

  ```
  whereis -s {{gcc}} -m {{git}}
  ```

- Locate binaries for gcc in /usr/bin/ only:

  ```
  whereis -b -B {{/usr/bin/}} -f {{gcc}}
  ```

wodim

Command (aliased as `cdrecord` on some systems) for recording data to CDs or DVDs.

Some invocations of wodim can cause destructive actions, such as erasing all the data on a disc.

- Display optical drives available to `wodim`:

  ```
  wodim --devices
  ```

- Record ("burn") an audio-only disc:

  ```
  wodim dev=/dev/{{optical_drive}} -audio {{track*.cdaudio}}
  ```

- Burn a file to a disc, ejecting the disc once done (some recorders require this):

  ```
  wodim -eject dev=/dev/{{optical_drive}} -data {{file.iso}}
  ```

- Burn a file to the disc in an optical drive, potentially writing to multiple discs in succession:

  ```
  wodim -tao dev=/dev/{{optical_drive}} -data {{file.iso}}
  ```

wpa_cli

Add and configure wlan interfaces.

- Scan for available networks:

  ```
  wpa_cli scan
  ```

- Show scan results:

  ```
  wpa_cli scan_results
  ```

- Add a network:

  ```
  wpa_cli add_network {{number}}
  ```

- Set a network's SSID:

  ```
  wpa_cli set_network {{number}} ssid "{{SSID}}"
  ```

- Enable network:

  ```
  wpa_cli enable_network {{number}}
  ```

- Save config:

  ```
  wpa_cli save_config
  ```

write

Write a message on the terminal of a specified logged in user (ctrl-C to stop writing messages).

Use the `who` command to find out all terminal_ids of all active users active on the system.

- Send a message to a given user on a given terminal id:

  ```
  write {{username}} {{terminal_id}}
  ```

- Send message to "testuser" on terminal "/dev/tty/5":

  ```
  write {{testuser}} {{tty/5}}
  ```

- Send message to "jhondoe" on pseudo terminal "/dev/pts/5":

  ```
  write {{jhondoe}} {{pts/5}}
  ```

x11vnc

A VNC server that will enable VNC on an existing display ser.

By default, the server will automatically terminate once all clients disconnect from it.

- Launch a VNC server that allows multiple clients to connect:

  ```
  x11vnc -shared
  ```

- Launch a VNC server in view-only mode, and which won't terminate once the last client disconnects:

  ```
  x11vnc -forever -viewonly
  ```

- Launch a VNC server on a specific display and screen (both starting at index zero):

  ```
  x11vnc -display :{{display}}.{{screen}}
  ```

- Launch a VNC server on the third display's default screen:

  ```
  x11vnc -display :{{2}}
  ```

- Launch a VNC server on the first display's second screen:

  ```
  x11vnc -display :{{0}}.{{1}}
  ```

xclip

X11 clipboard manipulation tool, similar to `xsel`.

Handles the X primary and secondary selections, plus the system clipboard (`Ctrl + C`/ `Ctrl + V`).

- Copy the output from a command to the X11 primary selection area (clipboard):

```
echo 123 | xclip
```

- Copy the output from a command to a given X11 selection area:

```
echo 123 | xclip -selection {{primary|secondary|clipboard}}
```

- Copy the contents of a file to the system clipboard, using short notation:

```
echo 123 | xclip -sel clip
```

- Copy the contents of a file into the system clipboard:

```
xclip -sel clip {{input_file.txt}}
```

- Paste the contents of the X11 primary selection area to the console:

```
xclip -o
```

- Paste the contents of the system clipboard to the console:

```
xclip -o -sel clip
```

- Paste the contents of the system clipboard into a file:

```
xclip -o -sel clip > {{output_file.txt}}
```

xdotool

Command line automation for X11.

- Retrieve the X-Windows window ID of the running Firefox window(s):

  ```
  xdotool search --onlyvisible --name {{firefox}}
  ```

- Click the right mouse button:

  ```
  xdotool click {{3}}
  ```

- Get the id of the currently active window:

  ```
  xdotool getactivewindow
  ```

- Focus on the window with id of 12345:

  ```
  xdotool windowfocus --sync {{12345}}
  ```

- Type a message, with a 500ms delay for each letter:

  ```
  xdotool type --delay {{500}} "Hello world"
  ```

- Press the enter key:

  ```
  xdotool key {{KP_Enter}}
  ```

xeyes

Display eyes on the screen that follow the mouse cursor.

- Launch xeyes on the local machine's default display:

  ```
  xeyes
  ```

- Launch xeyes on a remote machine's display 0, screen 0:

  ```
  xeyes -display {{remote_host}}:{{0}}.{{0}}
  ```

xinput

List available input devices, query information about a device and change input device settings.

- List all input devices:

  ```
  xinput list
  ```

- Disconnect an input from its master:

  ```
  xinput float {{id}}
  ```

- Reattach an input as slave to a master:

  ```
  xinput reattach {{id}} {{master_id}}
  ```

xman

Manual page viewer for X Window System.

- Start xman in three-button window:

  ```
  xman
  ```

- Open the manual page output stored in a given file:

  ```
  xman -helpfile {{filename}}
  ```

- Show both manual page and directory:

  ```
  xman -bothshown
  ```

xrandr

Set the size, orientation and/or reflection of the outputs for a screen.

- Display the current state of the system (known screens, resolutions, ...):

  ```
  xrandr --query
  ```

- Disable disconnected outputs and enable connected ones with default settings:

  ```
  xrandr --auto
  ```

- Change the resolution and update frequency of DisplayPort 1 to 1920x1080, 60Hz:

  ```
  xrandr --output {{DP1}} --mode {{1920x1080}} --rate {{60}}
  ```

- Set the resolution of HDMI2 to 1280x1024 and put it on the right of DP1:

  ```
  xrandr --output {{HDMI2}} --mode {{1280x1024}} --right-of
  {{DP1}}
  ```

- Disable the VGA1 output:

  ```
  xrandr --output {{VGA1}} --off
  ```

xsel

X11 selection and clipboard manipulation tool.

- Use a command's output as input of the clip[b]oard (equivalent to `Ctrl + C`):

  ```
  echo 123 | xsel -ib
  ```

- Use the contents of a file as input of the clipboard:

  ```
  cat {{file}} | xsel -ib
  ```

- Output the clipboard's contents into the terminal (equivalent to `Ctrl + V`):

  ```
  xsel -ob
  ```

- Output the clipboard's contents into a file:

  ```
  xsel -ob > {{file}}
  ```

- Clear the clipboard:

  ```
  xsel -cb
  ```

- Output the X11 primary selection's contents into the terminal (equivalent to a mouse middle-click):

  ```
  xsel -op
  ```

xsetwacom

Command line tool to change settings for Wacom pen tablets at runtime.

- List all the available wacom devices. The device name is in the first column:

  ```
  xsetwacom list
  ```

- Set Wacom area to specific screen. Get name of the screen with `xrandr`:

  ```
  xsetwacom set "{{device name}}" MapToOutput {{screen}}
  ```

- Set mode to relative (like a mouse) or absolute (like a pen) mode:

  ```
  xsetwacom set "{{device name}}" Mode "{{Relative|Absolute}}"
  ```

- Rotate the input (useful for tablet-PC when rotating screen) by 0|90|180|270 degrees from "natural" rotation:

  ```
  xsetwacom set "{{device name}}" Rotate {{none|half|cw|ccw}}
  ```

- Set button to only work when the tip of the pen is touching the tablet:

  ```
  xsetwacom set "{{device name}}" TabletPCButton "on"
  ```

yaourt

Arch Linux utility for building packages from the Arch User Repository.

- Synchronize and update all packages (including AUR):

  ```
  yaourt -Syua
  ```

- Install a new package (includes AUR):

  ```
  yaourt -S package-name
  ```

- Remove a package and its dependencies (includes AUR packages):

  ```
  yaourt -Rs package-name
  ```

- Search the package database for a keyword (including AUR):

  ```
  yaourt -Ss package-name
  ```

- List installed packages, versions, and repositories (AUR packages will be listed under the repository name 'local'):

  ```
  yaourt -Q
  ```

yum

Package management utility for RHEL, Feodra, and CentOS (for older versions).

- Synchronize list of packages and versions available. This should be run first, before running subsequent yum commands:

  ```
  yum update
  ```

- Install a new package:

  ```
  yum install {{package}}
  ```

- Install a new package and assume yes to all questions (also works with update, great for automated updates):

  ```
  yum -y install {{package}}
  ```

- Find the package that provides a particular command:

  ```
  yum provides {{command}}
  ```

- Remove a package:

  ```
  yum remove {{package}}
  ```

- Upgrade installed packages to newest available versions:

  ```
  yum upgrade
  ```

zenity

Display dialogs from the command line/shell scripts.

Return user-inserted values or 1 if error.

- Display the default question dialog:

  ```
  zenity --question
  ```

- Display an info dialog displaying the text "Hello!":

  ```
  zenity --info --text="{{Hello!}}"
  ```

- Display a name/password form and output the data separated by ";":

  ```
  zenity --forms --add-entry="{{Name}}" --add-
  password="{{Password}}" --separator="{{;}}"
  ```

- Display a file selection form in which the user can only select directories:

  ```
  zenity --file-selection --directory
  ```

- Display a progress bar which updates its message every second and show a progress percent:

  ```
  {{(echo "#1"; sleep 1; echo "50"; echo "#2"; sleep 1; echo
  "100")}} | zenity --progress
  ```

zramctl

Setup and control zram devices.

Use `mke2fs` or `mkswap` to format zram devices to partitions.

- Check if zram is enabled:

  ```
  lsmod | grep -i zram
  ```

- Enable zram with 2 devices (use `zramctl` to configure the devices further):

  ```
  sudo modprobe zram num_devices={{2}}
  ```

- Find and initialise the next free zram device to a 2GB virtual drive using LZ4 compression:

  ```
  sudo zramctl --find --size {{2GB}} --algorithm {{lz4}}
  ```

- List currently initialised devices:

  ```
  zramctl
  ```

zypper

SUSE & openSUSE package management utility.

- Synchronize list of packages and versions available:

  ```
  zypper refresh
  ```

- Install a new package:

  ```
  zypper install {{package}}
  ```

- Remove a package:

  ```
  zypper remove {{package}}
  ```

- Upgrade installed packages to newest available versions:

  ```
  zypper update
  ```

- Search package via keyword:

  ```
  zypper search {{keyword}}
  ```