SSH/Xterm and Common Commands

The recommended way to access Batocera's terminal is through SSH from another computer on the same local network (so that you get nice features like command history and ASCII coloration). First, make sure that SSH is enabled on your Batocera machine (it is enabled by default): check /userdata/system/batocera.conf and make sure system.ssh.enabled=1 (with no leading #) is present. Then, use your preferred SSH tool to connect:

- Linux and MacOSX and Windows 10 users can use command line ssh from a terminal (or the Command Prompt under Windows 10). You just need to enter ssh root@batocera or equivalent in your terminal/command prompt.
- Windows users are advised to use PuTTY or Kitty to connect via SSH. Under Translation, set the Remote character set: as “Use font encoding”.

The only user on a Batocera system is root. The typical address to attempt an SSH connection to the Batocera machine would be root@batocera.local or just root@batocera. If none of those work, you can connect to root@192.168.0.42, where 192.168.0.42 would be the IP address that you get from MAIN MENU → NETWORK SETTINGS in the Batocera interface.

- The username is root
- The default password is linux

For example, this is how Putty should look when SSH'ing in:
You can click “Save” after everything is set up to create a double-clickable profile for your convenience.

You'll then need to enter the default password, `linux`. While typing your password in, no asterisks will appear. That is how Linux does things.

If you do not want guest access, it is strongly advised to turn on “Enforce Security” from **MAIN MENU > SYSTEM SETTINGS > SECURITY** and to change the password as well.

Turning on **Enforce Security** will also require your username and password to access the network share. Despite this, keep in mind that **Batocera is not a secure operating system** and exposing it to a public network is at your own risk.

You'll know that you've successfully SSH'd into Batocera with if you see a screen similar to this:
SSH/Xterm and Common Commands

You can also access the local terminal on the Batocera machine itself by opening Xterm from the applications menu in the file manager ([F1] on the system list), however this can only be done on x86/x86_64 platforms. You can also open a new TTY session from anywhere by pressing [Ctrl]+[Alt]+[F5] (you will be asked to login, refer to above). To return to Batocera after doing this, press [Ctrl]+[Alt]+[F2]. In Batocera v31 and lower, this shortcut is [Ctrl]+[Alt]+[F3]. In Batocera v31 and lower, it is not recommended to use this terminal due to garbled text input; only use it in emergency situations.

Whenever you are using a command-line, try to understand what the commands you are using do exactly. Even if it doesn't seem like it, certain commands can be used in harmful ways. For example, putting simply `> empty.txt` without any command will create an empty file called empty.txt in the current working directory. That's fine, right? If you already had a useful file with this name, it'll be empty after this command.

Even more dangerously, a short command like `rm -rf /` or `::` could destroy your entire hard drive without prompt!

Password-less authentication

It is possible to have Batocera remember your computer's key such that you will no longer need to...
type your password in every time to log into SSH. Very useful for developers.

1. On your client computer (the one you intend to SSH into Batocera with), open a terminal and run the following:

```
ssh-keygen
```

○ You will be asked a series of questions. For the path, leave it at its default (just press [Enter] to skip). Then you will be asked for a password, this is for if you want to enable password protection for the key itself. Since that defeats the point of this, just press [Enter] again without typing anything to continue. In summary: just mash [Enter].

2. After that, then run the following:

```
ssh-copy-id -i ~/.ssh/id_rsa.pub root@<my_IPaddress>/hostname>
```

○ Some operating systems may use a different default path to store their public SSH keys in other than ~/.ssh/id_rsa.pub, change this accordingly.

3. SSH into Batocera (you will need to enter your password one last time) and run the following command:

```
mv /userdata/system/ssh/authorized_keys
/userdata/system/.ssh/authorized_keys
```

### Basic SSH commands

Once you are connected to Batocera, you can use most standard Unix commands.

> When we say most, we mean mostly the GNU coreutils. Batocera is a lean build-root Linux based system, which does not have a lot of the commands or packages that other distributions typically do. A lot of commands aimed at Ubuntu or Mint for example won’t work here.

Some command-line tools have an --help option describing how to use the program, for example, typing `cp --help` will print how the command is used in the command-line interface.

```
```

Usage: `cp [OPTIONS] SOURCE... DEST`

Copy SOURCE(s) to DEST

```
-a    Same as -dpR
-R,-r  Recurse
-d,-P  Preserve symlinks (default if -R)
-L    Follow all symlinks
-H    Follow symlinks on command line
```
Most command-line tools have a syntax similar to `<program> -<option flags> <parameters of program> <path/to/input/file> <path/to/output/file>`, but not all of them. It's worth reading their manual before using them.

The default working directory is the HOME folder of Batocera at `/userdata/system`. This will appear as `~` in your terminal. You can check what directory you are currently in with `pwd`.

Using the command line with paths of files can be confusing at first, there are two types of paths:

- **Absolute paths**: they will always be the same ones regardless of the current directory you are in, and they start with a `/` character; `/userdata/saves` for example.

- **Relative paths**: they are relative to your current position. For example, if you are in the `/userdata` directory, and you use `nano system/batocera.conf`, you will execute the command `nano` on the file `batocera.conf` located in the `/userdata/system` folder.

If a path, or a filename, contains special characters or spaces, you will need to put either single quotes `'` or double quotes `"` around it.

Most commands can be immediately halted with `[Ctrl]+[C]`. Please note that doing this may corrupt data if that program is in the middle of editing a file.

But if `[Ctrl]+[C]` is used, then how do you copy text from a SSH session? Simple: highlight the text and it will automatically be copied to your host system's clipboard. To paste, right click.

**Basic file usage**

Batocera includes a powerful command-line explorer called Midnight commander. This will offer a

- `mc`: starts the command-line file explorer Midnight Commander, this tool can be used to move, copy, delete, rename, and edit files, as well as create folders, make symbolic links, and change individual file permissions. This utility also supports a mouse input.

If you'd like to instead manipulate files using standard Unix commands, here is a cheat-sheet:

- `pwd`: tells you the current directory you are in [Print Working Directory], eg. running `cd /userdata` then `pwd` will output `/userdata`
- `cd`: changes the current working directory [Change Directory], eg. `cd /userdata/roms` puts you in the `/userdata/roms` folder
- `cp`: copies a given file or folder to another path [CoPy], eg. `cp /userdata/system/batocera.conf /userdata/batocera.conf` will create a copy of
batocera.conf in the /userdata folder.
- `du -sh`: gives you the size of the specified element [Disk Usage, Specified, Human-readable], eg. `du -sh /userdata/roms/snes` gives you the size on disk of the snes roms folder (If you use Batocera on x86, or access it from a windows computer with the file manager, the “size” and “size on disk” may not be the same, especially for .wine games. This command lets you see the actual size on disk)
- `ls`: lists the files and folders present in the current directory [List]. eg. `ls` while in `/userdata/` will output `bios cheats decorations extractions ...` etc.
- `mkdir`: creates a directory [Make Directory], eg. `mkdir content` will create a directory called `content` in the current working directory.
- `mv`: moves a given file to another path [Move], eg. `mv /userdata/roms/gb/game.zip /userdata/roms/gbc` will move the file `game.zip` from gb to gbc; the `mv` command can also be used to rename files, eg. `mv /userdata/roms/gb/game.zip /userdata/roms/gbc/gb_game.zip` would rename the file in the same directory.
- `nano`: opens a command-line text editor for the specified file, eg. `nano /userdata/system/batocera.conf` opens the file `batocera.conf` to edit it; for more info see this link.
- `rmdir`: deletes a directory if it is empty [Remove Directory], eg. `rmdir content` will delete the directory content if it is empty.
- `rm`: deletes a specified file [Remove], eg. `rm invaders-201226-124223.png` will erase the file `invaders-201226-124223.png` in the current working directory.
- `rm -r`: deletes a directory and all the files it contains [Remove, Recursive], use with caution as it has no prompt!
- `unzip`: decompresses a given .zip file in the working directory, eg. `unzip file.zip` will extract all the data in file.zip in the current working directory. You can ask to extract it elsewhere using the `-d` option [Directory], for example, `unzip file.zip -d uncompressed` will extract all the data in file.zip inside a subdirectory called uncompressed.

**Batocera store**

The **content downloader**, essentially. This was introduced in Batocera v29 (Fix Me! or close enough to it).

- `batocera-store list`: list all available packages (pre-configured games from the store)
- `batocera-store install <package>`: install a package
- `batocera-store list-repositories`: list all currently configured repositories to fetch data from.
- `batocera-store refresh`: refresh the store list
- `batocera-store update`: update all installed packages to their latest available versions
- `batocera-store clean`: clear the store cache
- `batocera-store clean-all`: clear the store cache and package files

**Debugging**

Most of these commands require `export DISPLAY=:0.0` to be run first before they can work.

The following commands can be used to debug your Linux-based operating system:

- `aplay -l`: returns the list of playback hardware devices [Audio Player, List] (can be used to
debug audio issues on PC).

- `xrandr`: returns the list of available displays [X Window Resize AND Rotate] and their reported resolutions (can be used to debug video issues on PC). More info on this page.
- `blkid`: returns the list of mountable drives connected to the machine. More info on the external storage page.
- `htop`: basically task manager.
- `pidof`: gives a list of processes identifiers (PID) for a running process name [Process IDentification OF], for example `pidof retroarch` returns a number when a retroarch-based emulator is running.
- `kill [PID]`: kills a process with a given PID, for example if `pidof retroarch` returned 640, then running `kill 640` would terminate the retroarch process.

Unique to Batocera:

- `/etc/init.d/S31emulationstation stop`: kills EmulationStation.
- `batocera-es-swissknife --restart`: starts/restarts EmulationStation (can [Ctrl] + [C] without closing ES to get back to terminal).
- `/etc/init.d/S31emulationstation start`: starts/restarts EmulationStation ([Ctrl] + [C] will bring back control of the terminal without closing ES).
- `batocera-info`: prints out the info seen when logging in.
- `batocera-settings`: autonomously make edits to configuration files, see its own page for usage.
- `batocera-resolution listModes`: Shows a list of the available display modes as they appear in EmulationStation's Video mode option.
- `batocera-padinfo`: prints info about the current connected pads (v32 has this functionality broken, pads use a different input driver now?)
- `batocera-support`: create a Batocera support tarball

**BPYTOP** is a more sophisticated system resource monitor powered by Python.
To install it:

```sh
python -m ensurepip --upgrade
pip3 install bpytop --upgrade
batocera-save-overlay
```

This needs to be done every update.

**Batocera resolution**

Most of these commands require `export DISPLAY=:0.0` to be run first before they can work.

- `batocera-resolution listModes`: Shows a list of commands related to resolution/display.
- `batocera-resolution listModes`: Shows a list of the available display modes as they appear in EmulationStation's **Video mode** option.
- `batocera-resolution setMode <mode>`: set a mode shown in the list, eg. `batocera-resolution listModes max-1920×1080`
- `batocera-resolution currentMode`: show the name of the current modeline being used (modeline includes setting the resolution, refresh rate and timings).
- `batocera-resolution currentResolution`: show the current resolution being sent to the output.
- `batocera-resolution listOutputs`: list all the available and connected outputs.
- `batocera-resolution setOutput <output>`: switch to the specified `<output>`, eg. `batocera-resolution setOutput HDMI-1`
- `batocera-resolution minTomaxResolution <width>x<height>`: (outdated from v32 and higher, use `max-1920×1080` as a setMode instead) force a maximum resolution; if the configuration or video mode attempts to exceed this, bring it back down to this specified resolution, eg. `batocera-resolution minTomaxResolution 1280×720`
- `batocera-resolution forceMode <horizontal>x<vertical>:<refresh>`: create a custom modeline and force it on the current display (can result in no display if incompatible settings are used), eg. `batocera-resolution forceMode 1920×1080:60`

**Batocera ES swissknife (dev tools)**

Batocera features some tools to aid developers. Run `batocera-es-swissknife [FLAG]` to use them. You can run `batocera-es-swissknife --help` to see the current list of flags, but here is a copy of that (last updated Batocera v32):

```
--restart will RESTART EmulationStation only
--kodi will startup KODI Media Center stopping ES
--reboot will REBOOT whole system
--shutdown will SHUTDOWN whole system
--emukill will exit any running EMULATORS
--espid checks if EmulationStation is currently active
    This number is the real PID of the binary!
    If the output is 0, then ES isn't active
--emupid to check if an Emulator is running
```

https://wiki.batocera.org/
This number is just the PID of emulatorlauncher.py
If output is 0 then there is no emulator active!

--arch   Shows current architecture running
--version Shows current version of BATOCERA running
--update Shows possible update for your install
default: stable, you can type --update beta

--overlay will try to backup your overlay file
--remount toggle write access to <dir>, default /boot
   This switch can have serious effects for your setup
--reset-ra will set all RA settings to default

Formatting tools

VERY DANGEROUS

- batocera-format --help: summons an angry, sentient tuba
- batocera-format listDisks: list all the partitions and disks currently available to format.
- batocera-format listFstypes: list the available formats Batocera can format a disk to.
- batocera-format format <disk> <fstype>: destroy a whole disk and format it to <fstype>
- batocera-format format INTERNAL <fstype>: special exception format, only format the userdata partition of the internal drive to a particular <fstype>.

Internet functionalities

- batocera-upgrade: lets you update batocera using the command-line with the correct URL, see Manual upgrades/downgrades for more details.
- batocera-install listDisks: lists the current disks available to install Batocera onto (also visible with the Install Batocera on a new disk option in ES).
- batocera-install listArchs: downloads the list of current stable architectures that Batocera has available.
- batocera-install install <disk> <arch|file>: downloads and installs the latest stable version of Batocera or a specified file onto the target disk. Be careful, as this completely destroys all existing data on that disk! If using a local file, that file must be in /userdata/system/installs.
- batocera-install listFiles: lists all the files found in /userdata/system/installs.
- pacman -Ss: lets you search through the pacman packages using the command-line [PACkage MANager, Sync, Search], see Batocera Package Manager (pacman) for more infos.
- pacman -S: lets you install a package using its name [PACkage MANager, Sync], see Batocera Package Manager (pacman) for more info.
- pacman -Rsd: lets you remove a package using its name [PACkage MANager, Remove, recurSive, skip Dependancies checks], see Batocera Package Manager (pacman) for more infos.
- pacman -Scc: clears the entire cache of the pacman manager [PACkage MANager, Sync, clear Cache (extra c forces a complete clear)], see Batocera Package Manager (pacman) for more infos.
• **pacman -Sy**: Updates the pacman database [PACkage MANager, Sync (extra y refreshes the database)], see Batocera Package Manager (pacman) for more infos.

## Logging to a file

Sometimes, you might prefer having the output of a command inside a separated text file instead of reading through the command line interface. To do that you can use the > and >> symbols followed by the path of a filename.

For example, the command `ls` gives you the list of all files and folders in your current working directory, if you want to gather this inside a text file named `list-files.txt` at `/userdata/system`, you simply need to use:

- `ls > /userdata/system/list-files.txt`, the command will return nothing, and instead the files will be listed inside a new file called `list-files.txt` located in `/userdata/system`. Running the command again will replace the content of `list-files.txt`.
- `ls >> /userdata/system/list-files.txt` however will add the output of the command to the existing file, without removing the previous infos.

You can also pipe the output to another program by using the vertical line character (|). For example, `dmesg | less` will let you see a scrollable list of the output with the [Up]/[Down] arrow keys; `dmesg | more` will let you see a full page of the output at a time, moving forward with [Spacebar]. Both of these examples can be quit by pressing [Q].

## Miscellaneous scripts

- **batocera-screenshot**: Saves a **screenshot** of the current screen in the `/userdata/screenshots` folder.
- **batocera-record**: Starts recording the screen; [Ctrl]+[C] to stop.
- **batocera-overclock list**: shows current available overclocking options (RPi and s922 only)
- **batocera-overclock set <value>**: set and save the selected overclock

Overclocking your hardware could cause irreversible damage and/or erratic behavior, this is at your own risk. If any issues occur after setting this, this should be the first thing to be returned to default.

- **batocera sync list**: list storage devices that are capable of being synced to
- **batocera sync <storage UID>**: uses rsync to sync the current userdata to the batocera/ folder on the selected storage device (NTFS not supported, FAT systems don't support syncing Bluetooth settings)
- **batocera-timezone get**: show the current configured timezone
- **batocera-timezone detect**: attempt to automatically guess your timezone
- `batocera-timezone set <timezone>`: manually set a timezone, timezones follow the 
  `<country/continent>/<city/region>` format, eg. `batocera-timezone set 
  Europe/Malta` or `etc/GMT+9`

## Custom aliases

While not mandatory, it is good to know you can create aliases of commands, they let you launch a command with a simple keyword. One way to do this can be done by creating a text file in 
`/userdata/system`, that file must be named `.profile`: it will contain aliases commands and will execute them when Batocera is launched, so be careful of what you do.

The alias command syntax is as follow:

```
alias customname='du -sh /userdata/system/batocera.conf'
```

In this example, once batocera has been restarted, entering `customname` in the command line will do the same as entering `du -sh /userdata/system/batocera.conf` (in this example, it basically returns the size of the batocera.conf file in the command-line interface).

A more useful, yet complicated example, would be to try and use ffmpeg to save a screenshot of Batocera in the screenshots folder for x86_64 devices, as devices other than Raspberry Pi's don't have an equivalent of raspi2png. In this example I call the custom alias “pc2jpeg”.

```
alias pc2jpeg='ffmpeg -hide_banner -loglevel error -f x11grab -i :0.0 -frames:v 1 /userdata/screenshots/$(date +%y-%m-%d_%H-%M-%S).jpg'
```

This commands does the following:

- `ffmpeg`: an command-line encoder included inside batocera.
- `-hide_banner`: prevents ffmpeg from printing the copyright notice when running this command.
- `-loglevel error`: will only alert in the command-line if there is an actual error preventing the command from working.
- `-f x11grab`: uses `x11grab` to capture the screen 0 (this lets you use the screen as the source).
- `-frames:v 1`: the number of video frames to capture.
- `/userdata/screenshots/$(date +%y-%m-%d%H%M%S).jpg`: the full path to the file we want to save.

```
  $(date +%y-%m-%d%H%M%S): executes the date +%y-%m-%d%H%M%S command and reurns its result in the command line (for example, it will return 20-12-31_21-52-19 the 31st december of 2020 at 21h52m19s), so the path before would be read as /userdata/screenshots/20-12-31_21-52-19.jpg.
```

So by using the `pc2jpeg` custom command, a JPEG file will be created with a filename based off the date of the screenshot.

💡 This functionality has since been integrated into the command `batocera-screenshot`. 
This example remains here to show you the syntax of alias.
Troubleshooting

Visit the relevant section in the troubleshooting page for further help.

From:
https://wiki.batocera.org/ - Batocera.linux - Wiki

Permanent link:
https://wiki.batocera.org/access_the_batocera_via_ssh

Last update: 2022/03/20 06:44