dzo

Install, update and restart DayZ servers on one host, with rootless podman and btrfs snapshots.

reads | changes something | destroys data

1 Install the package

Creates the `dayz` user and `/var/lib/dzo`. Put that directory on btrfs if you want snapshots. Everything else runs as `dayz`.

`sudo apt install ./dzo_<version>_amd64.deb`	install, or upgrade
`sudo -iu dayz`	work as the service user

2 Point dzo at your site repository

The site repository holds every instance's settings. Create an empty git repository, then set it in `/etc/dzo/config.yaml`:

```
site:
  remote: git@host:you/dayz-site.git
  branch: main
```

`dzo setup`	directories, container images, site checkout
`dzo site validate`	load the site, resolve every instance

3 Log in to Steam, download the server

`dzo steam login --user <account>`	password and Steam Guard, asked once
`dzo product install dayz-stable`	about 3 GB (`dayz-experimental` works the same)

4 Describe the instance, then push it

Add `instances/chernarus/instance.yaml` and a `serverDZ.cfg` next to it. Set a join password in the config.

```
name: chernarus
product: dayz-stable
map: dayzOffline.chernarusplus   # Bohemia's mission
ports: {game: 2302, rcon: 2303, query: 27016}
network: host
mods: []
restarts: {schedule: ["*-*-* 00/4:00"]}   # every 4 h
```

`dzo site pull`	fetch what you pushed

5 Create it and start it

`dzo instance render chernarus --dry-run`	preview what will be written
`dzo instance create chernarus`	data directory, mission, first render
`dzo instance apply chernarus`	write the units and timers
`dzo start chernarus`	returns when the server answers Steam queries
`dzo status`	state, players, health, restarts

6 Add mods

`dzo mod add 1559212036 --instance chernarus`	install a workshop mod, add it to the list
`dzo mod add <id> --instance chernarus --server`	loaded with -servermod (clients don't get it)
`dzo site commit -m "add mods" --push`	`mod add` edits your checkout, keep it in git
`dzo restart chernarus --minutes 5`	announce, lock, kick, restart with the mods

Where things live

`/etc/dzo/config.yaml`	host settings
`/var/lib/dzo/site`	your site checkout
`…/instances/<n>`	one server: `mpmissions`, `profiles`, `storage`
`…/cache`	server builds and mods, kept as generations
`…/snapshots`	btrfs backups
`…/logs`	archived profile logs
`…/secrets`	Steam session, RCon passwords

Mission from another repo

For a map mod or your own fork, replace `map` and add `mission_source` in `instance.yaml`:

```
map: empty.deerisle
mission_source:
  git: <repository url>
  ref: main
  path: empty.deerisle
fallback_mission:
  dayzOffline.chernarusplus
```

`path` is the mission folder inside the repo. `ref` is a branch, tag or commit; pin one so updates stay your choice. `fallback_mission` supplies files the map lacks. Add the map's mod too.

`dzo mission update <n>` fetches a newer ref and shows the plan; `dzo instance render <n>` applies it.

Ports

Every instance needs its own **game**, **rcon** and **query** port. RCon listens on localhost only; open the other two in the firewall.

Run it

Find the job, copy the command. <n> is the instance name.

Start, stop, look

`dzo status [<n>]`
state, players, health, pending updates

`dzo start <n>`
waits until the server answers

`dzo stop <n>`
asks over RCon, then stays stopped

`dzo logs <n> -f`
follow the server console

`dzo events <n> -f`
follow kills, joins, chat, raids

`dzo instance ack-failure <n>`
allow starts again after a failed render or a crash loop

Restart without annoying players

`dzo restart <n> --minutes 30 --lock 3`
announce every minute, lock 3 minutes before, kick, restart

`dzo restart <n> --now`
no countdown

`dzo restart <n> --cancel`
stop the countdown, unlock

`dzo instance apply <n>`
after changing `restarts.schedule`: rewrite the timers

Mods and updates

`dzo instance mods list <n>`
the list; its order decides which mod wins a merge

`dzo instance mods move <id> --before <other>`
change that order

`dzo mod remove <id> --instance <n>`
drop a mod from the list

`dzo update check [--apply]`
hourly by timer; `policy: auto` restarts for you

`dzo mod refresh <id> --force`
download again after a broken download

New server build

A new build is only reported. You switch each instance yourself.

`dzo product update dayz-stable`
download the new build

`dzo instance upgrade <n> --build <id> --dry-run`
render plan for that build

`dzo instance upgrade <n> --build <id>`
announce, stop, snapshot, switch, start. A failed switch reverts

Mission and config

`dzo mission status <n>`
source, age, what the next render changes

`dzo mission update <n>`
fetch a newer mission, show the plan

`dzo instance render <n>`
apply in place, only files dzo manages

`dzo mission rollback <n> --list`
managed files from file history

Backups

Snapshots are taken before upgrades, mod updates and wipes.

`dzo backup create <n>`
snapshot now

`dzo backup list <n>`
what exists, and why

`dzo backup diff <n> <id>`
files that changed since then

`dzo backup pin <n> <id>`
keep it, whatever the retention says

RCon and logs

`dzo rcon console <n>`
interactive, with connects and chat

`dzo rcon exec --instance <n> players`
one command

`dzo rcon rotate <n> --restart`
new RCon password, read at the next start

`dzo logs rotate <n> --dry-run`
which profile logs would be archived

`dzo logs archive <n> --since 7d`
list the archive

Look inside

`dzo shell <n>`
debug container with the instance's mounts, server not started

`dzo exec <n> ls /dayz/profiles`
a command in the running container

Web, API and metrics

`dzo token create web --role admin --web`
token for `dzo web` (save it as `web.token`), then `dzo units sync`

`systemctl --user status dzo-{serve,web,exporter}`
API, web UI (`:8081`), metrics (`:9464/metrics`)

People and the first admin

`dzo user grant <steam64> --role admin --username ann`
first admin; `--instance <n>` limits it

`dzo user password ann`
password login if Steam sign-in fails; 2nd factor at first login

`dzo user disable <steam64>`
block the login; also `list`, `revoke`, `delete`, `reset` (lost second factor)

These delete data

They ask you to confirm (`--yes` skips it). `wipe`, `reinit` and `restore` take a snapshot first.

`dzo wipe <n>`
the world (persistence), not the mission

`dzo mission reinit <n>`
recreate the live mission from the original

`dzo instance upgrade <n> --build <id> --wipe`
switch builds and wipe the world

`dzo restore <n> <id> [--path …]`
go back to a snapshot, all or one path

`dzo instance remove <n>`
data directory and, unless `--keep-snapshots`, its snapshots

Upgrading dzo itself: `apt install ./dzo_<version>_amd64.deb`. Running servers keep going until their next restart.