Reproducible Linux Installations

on Your Laptop Without Data Loss

…or having multiple Linux installs side-by-side

…or switching Linux distributions without hassle

Holger Fischer (hoo) - hoonetorg - 2026

The Problem

> The installer has a lot of capabilities, but it's *dangerous* if you don't know exactly what you are doing.

Modern OS installers are great at **first installs** on empty disks.

Most have an interactive partition editor,

but:

One wrong click and your data is gone - like that.

It's not the installer's fault. It's

- that humans make mistakes under pressure, at 2am, or
- simply because the layout wasn't obvious.

Many colleagues told me they *accidentally* wiped their data on (re)install.

Many reports on the internet where you can read

"I lost my data on (re)install, even when I tried hard to keep it"

The Strategy: Know Your Disk State

Two modes — decided by what's already on disk:

Situation	Layout
Empty disk / single OS planned	**single-os**
Existing OS(es) detected	**multi-os**

A Python pre-script inspects the real partition table before touching anything.

It aborts if the state doesn't match expectations. - No guessing. No silent data loss.

BUT:

Have you defined your disk layout correctly - otherwise **data loss**

TEST before you install on real hardware

!!! HAVE BACKUPS !!!

Disk Layout: Multi-OS Example

```
1  EFI                 <- preserved, reused
2  MS Reserved         <- untouched
3  Windows             <- untouched
4  WinRecovery         <- untouched
--------- startpartition 5 -----------------------------
5  /boot               <- created or emptied
6  OS                  <- created or emptied
   (LUKS, btrfs)
   subvol: root
   subvol: home
------------------------------------------------------------
7  data                <- never touched on OS install
   (LUKS, btrfs)          created or verified by Ansible
   subvol: holger      -> /home/holger
   subvol: images      -> /var/lib/libvirt/images

8  winexchange         -> exchange data between
   (NTFS, unencrypted)    Windows and Linux
```

Rules

- Partitions 1–4: **never touched**
- Partitions 5+6: **created on first install, emptied on reinstall**
- Partitions after 6: **never touched**

  Ansible creates or verifies the state of these partitions
- EFI is reused → other OSes keep booting
- Distribution swap on reinstall: possible

> Ubuntu → Fedora on same partitions. Windows and data: untouched.

Partition Layout:

4 TB disk - P7 ≈ 3490 GiB

```
Fixed partitions total: 606 GiB. Scale: each `=` ≈ 82 GiB

| Part | Size       | Bar
|------|------------|--------------------------------------------
| 1    | 2 GiB      | =
| 2    | 256 MiB    | =
| 3    | 200 GiB    | ==
| 4    | 2 GiB      | =
| 5    | 2 GiB      | =
| 6    | 200 GiB    | ==
| 7    | ~3490 GiB  | ============================================
| 8    | 200 GiB    | ==
```

1TB disk, 2TB disk and 4TB disk in relation

```
1TB: |1|2|  3 (200G)  |4|5|  6 (200G)  |        7 (~418G)        |  8 (200G)  |

2TB: |1|2| 3G |4|5| 6G |              7 (~1442G)              | 8G |

4TB: |1|2|3G|4|5|6G|                      7 (~3490G)                      |8G|
```

The Builders (1)

Everything runs in **Podman containers**. No special host setup beyond: `podman` · `tar` · `rsync` - `gpg`

`faibuilder` / `ksbuilder`

faibuilder: produces a **bootable disk image** (btrfs, thumbdrive-ready) with the installer (FAI-based)

ksbuilder: produces a an **image** with a partition labeled "**OEMDRV**" which is found by the second connected image containing a standard Fedora/RHEL (and similar) ISO image.

Both builders add an additional partition labeled "**EXTRA**" for additional content:

- Ansible content (encrypted)
- Offline package repos (e.g. Fedora 43 WS — no Anaconda ISO exists, only LIVE ISO)
- Whatever you need

Installations run fully offline. No internet required.

The Builders (2)

Everything runs in **Podman containers**. No special host setup beyond: `podman` · `tar` · `rsync` - `gpg`

`nsbldr` (Ansible builder)

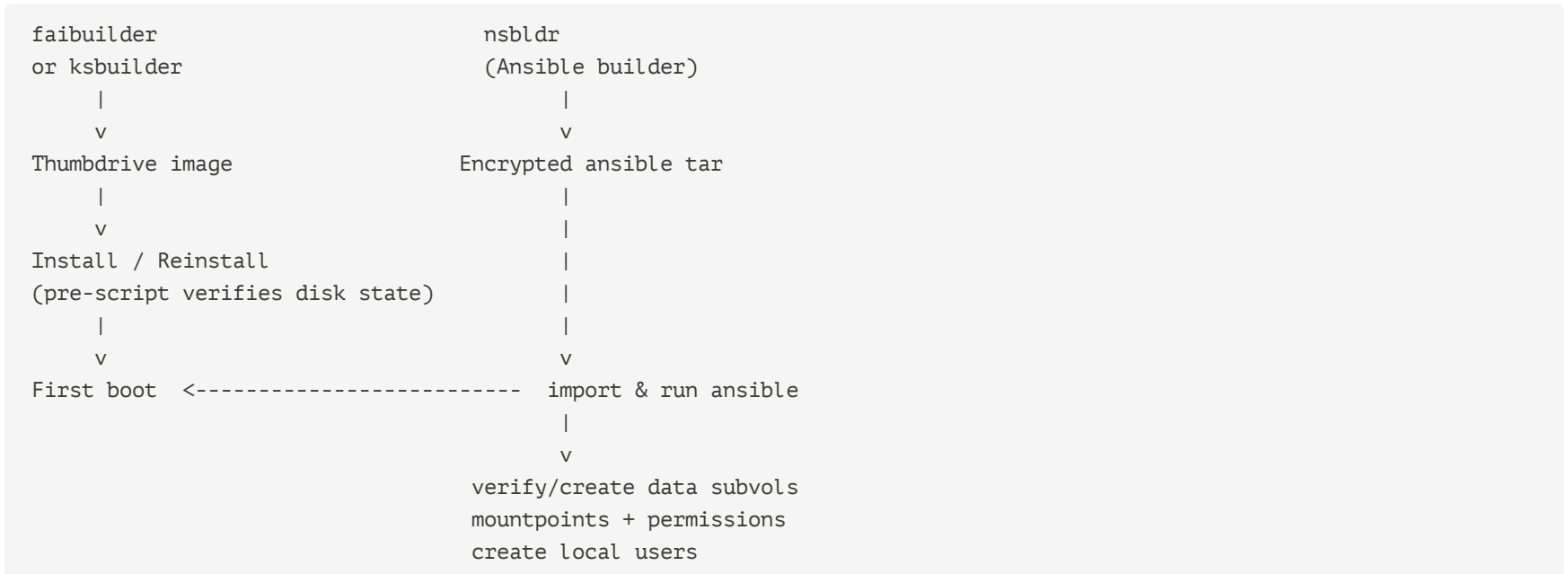
Packages Ansible content into a **GPG-symmetrically encrypted archive**.

Extracted after first boot → run via a dedicated system user (f.e. hoo).

Archive can contains:

- Ansible inventories
- Ansible files (files that should be delivered statically or via template)
- Ansible playbooks
- Ansible roles
- Ansible collections

The Full Workflow



Live Demo

First install → Reinstall with a Distribution swap

Windows and data: untouched throughout

Ask questions while it runs — installs take a few minutes

Takeaways & Links

Even if your setup looks nothing like mine — steal what's useful:

- LUKS + Btrfs subvolumes as a sane data/OS separation
- Pre-script state verification before any destructive action
- Offline-first installer images via Podman
- Agama support: **work in progress**

hoonetorg · github.com/hoonetorg

Slides + code links:

Slides

Source code:

Watch out for existing repos and changes at github.com/hoonetorg.

There will be a repo (not named yet) consolidating all required tools, containing examples.