🚀Proxmox VE → Proxmox Backup Server (PBS) Backup Guide

This guide explains in detail how **Proxmox VE (PVE)** backs up **Virtual Machines (VMs)** and **Containers (CTs)** to **Proxmox Backup Server (PBS)** using a **deduplicating, incremental, block-level backup mechanism**.

Prerequisites

Before setting up PVE → PBS backups, ensure you have:

- 🖥️**Proxmox VE** (v6.4 or later recommended)
- 📦**Proxmox Backup Server** (PBS v2.x or later)
- 🔑SSH / root access to both servers
- 🛡️Reliable network connection between PVE & PBS
- 💾Storage configured on PBS (ZFS, ext4, or xfs)
- 🧰Proxmox VE nodes configured with:
- Correct DNS & hostname resolution
- PBS datastore accessible
- Sufficient storage space

Step 1 – Add PBS Storage to Proxmox VE

1. Login to **Proxmox VE WebUI**
2. Navigate to: **Datacenter → Storage → Add → Proxmox Backup Server**
3. Enter the following:
4. 🗂️**ID:** Name for PBS storage (e.g. `pbs-data`)
5. 🛡️**Server:** PBS IP or FQDN
6. 📂**Datastore:** Datastore name on PBS
7. 🔑**Username:** Example `root@pam`
8. 🔐**Password/API Token:** Your PBS credential
9. Click **Add**

🔗Now PVE is linked with PBS.

Step 2 – Backup Modes

PVE supports multiple modes for VM/CT backup:

- 🩺**Snapshot mode** → Uses storage snapshots (fast, recommended)
- 🗒️**Suspend mode** → Pauses VM/CT during backup
- 🕛**Stop mode** → Shuts down VM/CT during backup

Step 3 – Backup a VM (QEMU/KVM)

- PVE uses the **QEMU Backup API** to read VM disk blocks.
- Workflow:
- Snapshot/suspend the VM
- Split VM disk into fixed-size **chunks (default 4 MB)**
- Compute **SHA256 hash** for each chunk
- Compare with PBS datastore:
    - If chunk exists → Skip upload
    - If new chunk → Upload & store
- Metadata (index file) created mapping VM → chunks

🔒 Data is transferred via **TLS encryption** and can be optionally **client-side encrypted**.

Step 4 – Backup a CT (LXC)

- PVE uses the `vzdump` tool.
- Workflow:
- Take snapshot/suspend/stop container
- Archive container root filesystem + config
- Stream data directly to PBS
- Same chunking, hashing, deduplication as VMs

🚡 Deduplication & Incremental Backups

- **Deduplication** → Identical chunks stored only once across all backups & VMs.
- **Incremental** → After first full backup, only changed chunks are uploaded.
- **Block-level** → Works at raw disk level, not just files.
- **Zstandard Compression** → Reduces backup size.
- **Integrity Checking** → SHA256 ensures data correctness.

Example:

Backup Run	VM Disk Size	Changed Data	Uploaded	Stored on PBS
Day 1	20 GB	20 GB	20 GB	20 GB
Day 2	20 GB	2 GB	2 GB	22 GB
Day 3	20 GB	500 MB	500 MB	22.5 GB

Why Each Chunk is Hashed (SHA256)?

- 🧰 **Deduplication:** Identify identical chunks across backups
- 🛡️ **Integrity Check:** Detect corruption or bit-rot
- 🧮 **Efficiency:** Skip already-known chunks during incremental backups
- 🔒 **Security:** Supports deduplication even with client-side encryption

Example: Two VMs with Ubuntu installed → shared system files → same hash → stored once.

Step 5 – Restore Process

1. Select VM/CT from Proxmox backup list
2. PVE requests required chunks from PBS
3. PBS reads metadata index and streams chunks
4. VM/CT rebuilt exactly as it was at backup time

🚢Benefits

- 💾**Storage savings** (deduplication)
- ⏱️**Faster backups** (incremental)
- 🔒**Secure & encrypted** transfer
- 🔗**Reliable integrity** checks
- 📦**Cross-VM deduplication** (shared base OS chunks stored once)

🗾References

- Proxmox Backup Server Documentation
- Proxmox VE Admin Guide

🏙️Final Notes

This setup ensures **efficient, secure, and space-optimized backups** of all your Proxmox workloads. Always test your restore process regularly to confirm backup integrity.

💡 *Tip: Combine PBS backups with offsite replication for disaster recovery.*