Backup Recipes



Copy-paste recipes for backups that can actually be restored.

Printed from maple.sanriodevgroup.org

Rsync mirror with deletions



A mirror that follows deletions, with a dry run first.

Rsync mirror with deletions

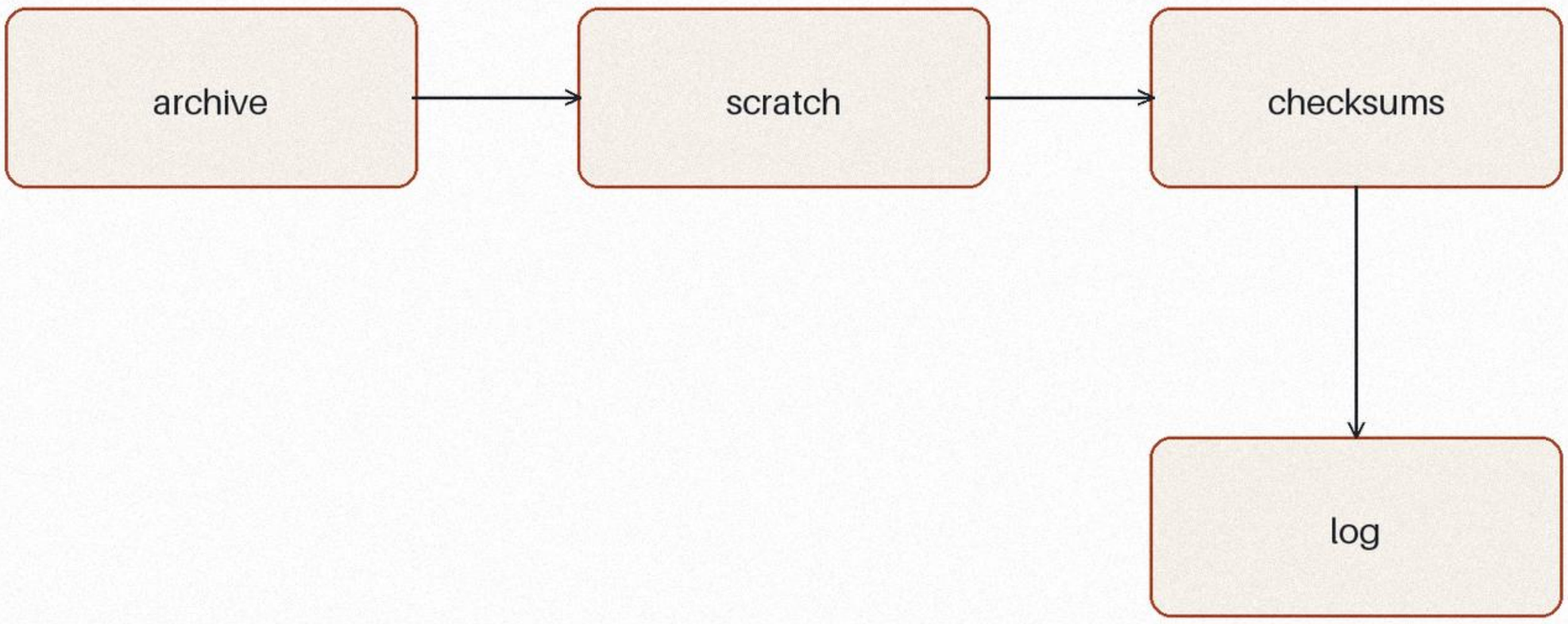


A mirror is not a backup on its own, because it copies mistakes too. It is still the first building block: a second copy that is cheap to keep current.

rsync -aHAX --delete --dry-run /srv/data/ backup:/srv/mirror/ rsync -aHAX --delete /srv/data/ backup:/srv/mirror/

The trailing slashes matter: with them rsync copies the contents of the directory, without them it creates the directory inside the target.

Verifying a restore



A backup is only finished when a restore has been tried.

Verifying a restore



Restore into a scratch directory, compare checksums against the source, and record the date. Doing it monthly is enough to notice a silent failure before it matters.

mkdir -p /tmp/restore && tar --zstd -xf latest.tar.zst -C /tmp/restore (cd /srv && find data -type f -print0 | xargs -0 sha256sum) > /tmp/src.sum (cd /tmp/restore && sha256sum -c /tmp/src.sum)

Rotating tarball archives



Daily archives kept for a week, weeklies for two months.

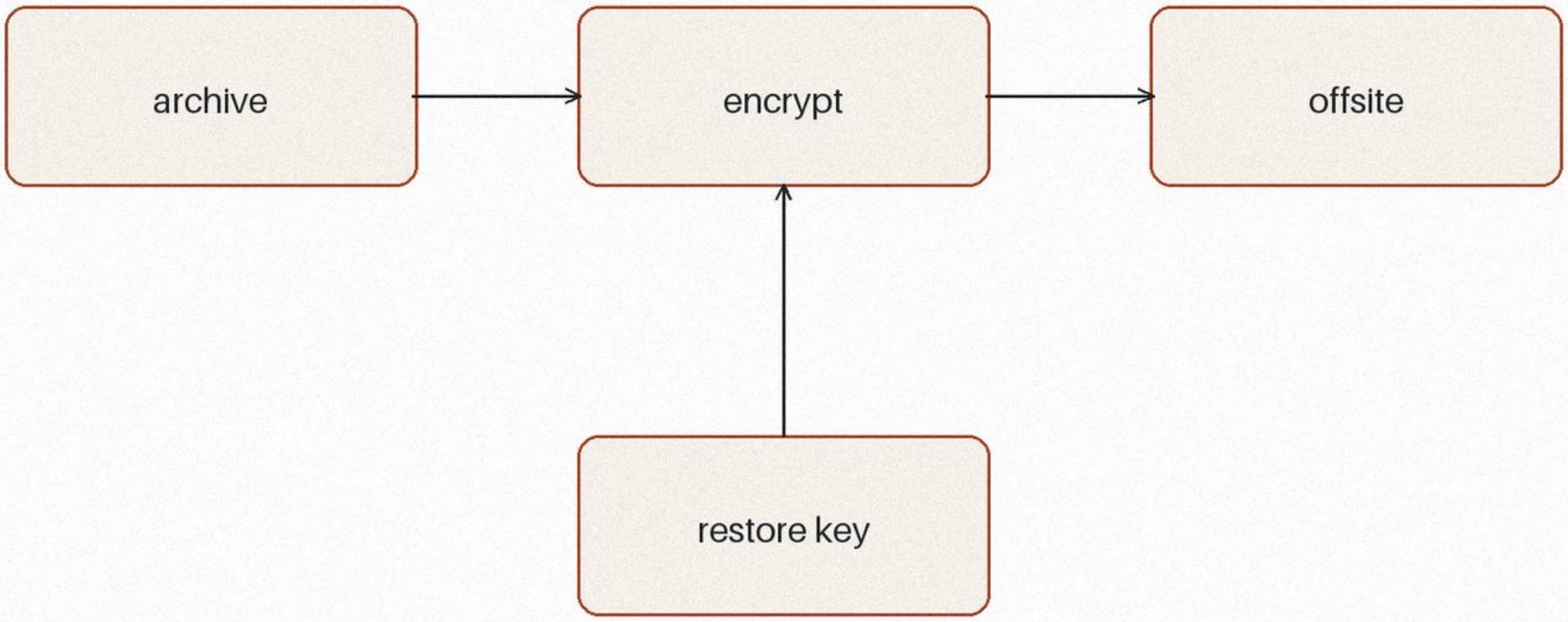
Rotating tarball archives



Dated archives are easy to reason about: one file per day, named by date, removed by age. A weekly copy on Sunday and a monthly copy on the first of the month make a simple grandfather-father-son scheme.

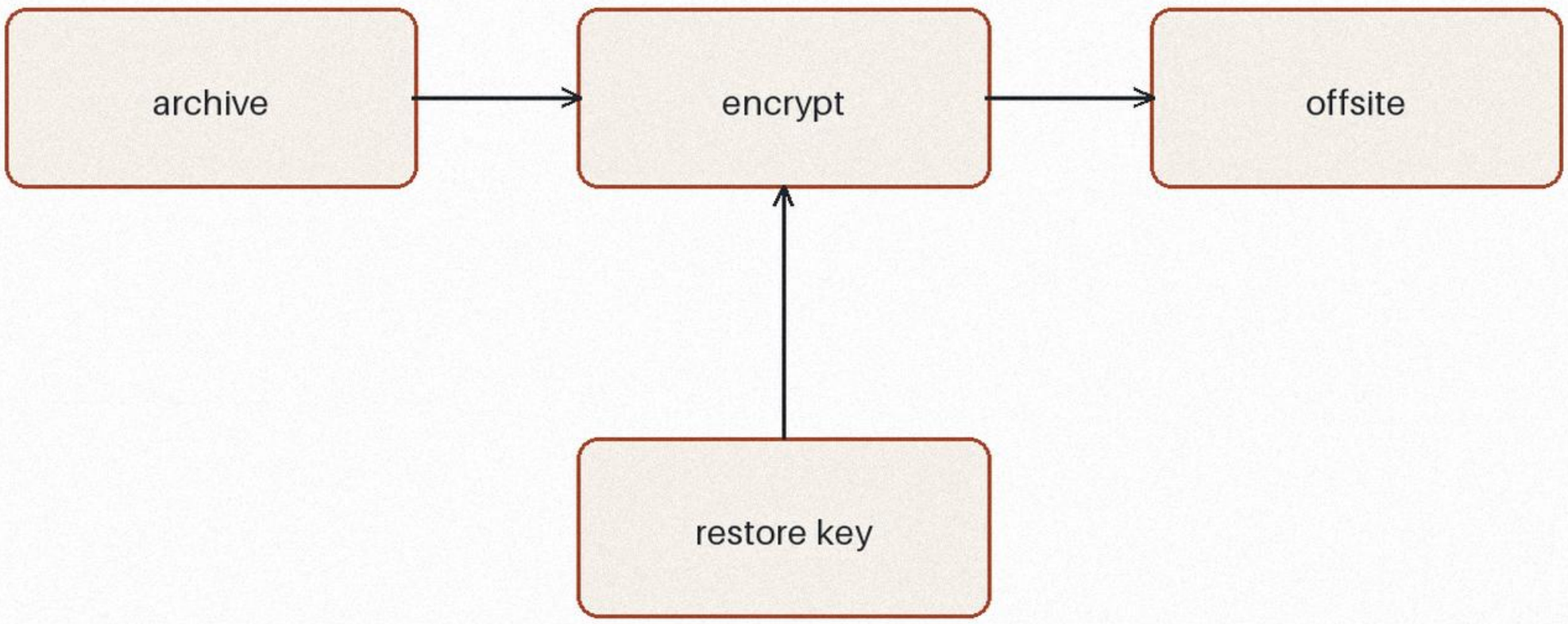
tar --zstd -cf /backups/daily/$(date +%F).tar.zst -C /srv data find /backups/daily -name '*.tar.zst' -mtime +7 -delete

Encrypting an offsite copy



Encrypt before the copy leaves the machine.

Encrypting an offsite copy



Whatever stores the offsite copy should only ever see ciphertext. Public-key encryption keeps the private key off the backup host entirely; only the restore machine needs it.

Keep a printed copy of the key fingerprint next to the restore instructions. A backup that nobody can decrypt is a very expensive random file.