Server & desktop backup solutions

The first backups will take a while. The following ones are much faster, but it depends on how much you change. Only the changes are saved.

rsync

Follow our rsync tutorial first.

The snapshots are stored locally and remotely via rsync daemon.

This backup solution is only recommended for an internal network. Use an encrypted backup method with borg instead.

Dependencies

The script needs `inetutils` for `hostname` command.

```
pacman -S inetutils
```

Credentials

```
echo "$password" > /etc/rsyncd.password
chmod 400 /etc/rsyncd.password
```

Script

Add your details for DAEMONUSER=“” and DAEMONHOST=“”.

```
nano /root/rsnapbackup.sh
```

```
#!/bin/sh

## Based on:
## my own rsync-based snapshot-style backup procedure
## (cc) marcio rps AT gmail.com

# config vars

SRC="/"
SNAP="/root/backup/"
OPTS="--rltogiPhv --stats --delay-updates --delete --chmod=a-w"
EXCL="--exclude-from=/root/backup-filter.rule"
DAEMONUSER=""
```

```
DAEMONHOST=""
HOSTNAME=$(hostname)
MINCHANGES=1

# run this process with real low priority

ionice -c 3 -p $$
renice +12  -p $$

# List and save installed packages
pacman -Qn | awk '{ print $1 }' > /root/pkglist

# sync

rsync $OPTS $EXCL $SRC $SNAP/latest >> $SNAP/rsync.log

# check if enough has changed and if so
# make a hardlinked copy named as the date

COUNT=$( wc -l $SNAP/rsync.log|cut -d" " -f1 )
if [ $COUNT -gt $MINCHANGES ] ; then
        DATETAG=$(date +%Y-%m-%d-%H:%M)
        if [ ! -e $SNAP/$DATETAG ] ; then
                cp -al $SNAP/latest $SNAP/$DATETAG
                chmod u+w $SNAP/$DATETAG
                mv $SNAP/rsync.log $SNAP/$DATETAG
               chmod u-w $SNAP/$DATETAG
         fi
fi

rsync -avAXHP --delete --password-file=/etc/rsyncd.password $SNAP
rsync://$DAEMONUSER@$DAEMONHOST/archive/backup/$HOSTNAME
```

```
chmod +x /root/rsnapbackup.sh
```

Exclude folder and files

This is an example. Add anything you don't need to backup. And change home $USER.

```
nano /root/backup-filter.rule
```

```
/dev/*
/proc/*
/sys/*
/tmp/*
/run/*
/mnt/*
/media/*
/lost+found
```

```
# root user
/root/backup/*
/root/.cache/*
# Home user
/home/$USER/.cache/*
```

borg

Follow our borg tutorial first.

The snapshots are stored remotely via ssh.

Script

Don't forget to create the borg repo first and add the credentials to the script.

```
borg init --encryption=keyfile-blake2 --make-parent-dirs
ssh://username@remote.host.address:$port>/~/backups/borg/{hostname}
```

Add your excluded folders/files `–exclude '/home/*/.cache/*' \` and under `::'{hostname}-{now}' \` add folders/files you want to backup.

```
#!/bin/sh

# Setting this, so the repo does not need to be given on the commandline:
export BORG_REPO=ssh://username@example.com:2022/~/backups/borg/{hostname}

# See the section "Passphrase notes" for more infos.
export BORG_PASSPHRASE='XYZl0ngandsecurepa_55_phrasea&&123'

# some helpers and error handling:
info() { printf "\n%s %s\n\n" "$( date )" "$*" >&2; }
trap 'echo $( date ) Backup interrupted >&2; exit 2' INT TERM

info "Starting backup"

# Backup the most important directories into an archive named after
# the machine this script is currently running on:

borg create                         \
    --verbose                       \
    --filter AMEhsx                 \
    --list                          \
    --stats                         \
    --progress                      \
    --verbose                       \
    --show-version                  \
    --show-rc                       \
```

```
    --compression zstd,11           \
    --exclude-caches                \
    --exclude '/home/*/.cache/*'    \
    --exclude '/var/tmp/*'          \
                                    \
    ::'{hostname}-{now}'            \
    /etc                            \
    /home                           \
    /root                           \
    /var                            \

backup_exit=$?

info "Pruning repository"

# Use the `prune` subcommand to maintain 7 daily, 4 weekly and 6 monthly
# archives of THIS machine. The '{hostname}-' prefix is very important to
# limit prune's operation to this machine's archives and not apply to
# other machines' archives also:

borg prune                          \
    --list                          \
    --prefix '{hostname}-'          \
    --show-rc                       \
    --keep-daily    7               \
    --keep-weekly   4               \
    --keep-monthly  6               \
    --keep-yearly   1               \

prune_exit=$?

# use highest exit code as global exit code
global_exit=$(( backup_exit > prune_exit ? backup_exit : prune_exit ))

if [ ${global_exit} -eq 0 ]; then
    info "Backup and Prune finished successfully"
elif [ ${global_exit} -eq 1 ]; then
    info "Backup and/or Prune finished with warnings"
else
    info "Backup and/or Prune finished with errors"
fi

exit ${global_exit}
```

Crontab - rsync and borg

Follow our crontab tutorial first and add the following for your root user:

```
@daily /root/rsnapbackup.sh
```

```
@daily /root/bsnapbackup.sh
```

- @yearly
- @annually
- @monthly
- @weekly
- @daily
- @hourly
- @reboot

Syncthing

Follow our Syncthing tutorial for both devices (backup server and your data device).

Add device

Add the backup server to your client under Remote Devices.

Add folder

- Add a folder under Folder and select the folder you want to backup under General.
- Select your backup server under Sharing.
- Under File Versioning you could add Staggered File Versioning which gives you more certainty, but have a look at https://docs.syncthing.net/users/versioning.html and choose what suits you best.
- Also check Advanced and Folder type and again choose what suits you best. For example, Keepass can be used with Send & Receive if you want sync your database on both devices.