

# Backup

## Rsnapshot

ist ein einfach zu bedienendes Backupprogramm. [Rsnapshot](#) basiert auf rsync, es ist einfach zu bedienen und eignet sich sehr gut für Backups auf USB Festplatten.

### ein anderes einfaches Backup-Skript

```
#!/bin/sh
NOW=$(date +"%d-%m-%Y")
# set mysql login info
MUSER="MySQL-Username"           # Username
MPASS="MySQL-SERVER-PASSWORD"   # Password
MHOST="MySQL-SERVER-IP-ADDRESS" # Server Name
# guess binary names
MYSQL="$(which mysql)"
MYSQLDUMP="$(which mysqldump)"
GZIP="$(which gzip)"

# get all db names
DBS="$(($MYSQL -u $MUSER -h $MHOST -p$MPASS -Bse 'show databases')"
for db in $DBS
do
    FILE=mysql-$db.$NOW-$(date +"%T").gz
    $MYSQLDUMP -u $MUSER -h $MHOST -p$MPASS $db | $GZIP -9> $FILE
done
```

### Verzeichnisse sichern

```
#!/bin/bash
# Simple backup with rsync
# local-mode, tossh-mode, fromssh-mode

SOURCES=(/root /etc /home /boot /lampp)
TARGET="/media/backup"

# edit or comment with "#"
#LISTPACKAGES=listdebianpackages           # local-mode and tossh-mode
MONTHROTATE=monthrotate                    # use DD instead of YYMMDD

RSYNCCONF=(--delete)
#MOUNTPOINT="/media/daten"                 # check local mountpoint
#MAILREC="user@domain"

#SSHUSER="sshuser"
#FROMSSH="fromssh-server"
```

```

#TOSSH="tossh-server"
SSHPORT=22

### do not edit ###

MOUNT="/bin/mount"; FGREP="/bin/fgrep"; SSH="/usr/bin/ssh"
LN="/bin/ln"; ECHO="/bin/echo"; DATE="/bin/date"; RM="/bin/rm"
DPKG="/usr/bin/dpkg"; AWK="/usr/bin/awk"; MAIL="/usr/bin/mail"
CUT="/usr/bin/cut"; TR="/usr/bin/tr"; RSYNC="/usr/bin/rsync"
LAST="last"; INC="--link-dest=$TARGET/$LAST"

LOG=$0.log
$DATE> $LOG

if [ "${TARGET:${#TARGET}-1:1}" != "/" ]; then
    TARGET=$TARGET/
fi

if [ "$LISTPACKAGES" ] && [ -z "$FROMSSH" ]; then
    $ECHO "$DPKG --get-selections | $AWK '!/deinstall|purge|hold/'|$CUT -f1 |
$TR '\n' ' ' >> $LOG
    $DPKG --get-selections | $AWK '!/deinstall|purge|hold/'|$CUT -f1 |$TR '\n'
' ' >> $LOG 2>&1
fi

if [ "$MOUNTPOINT" ]; then
    MOUNTED=$(($MOUNT | $FGREP "$MOUNTPOINT"));
fi

if [ -z "$MOUNTPOINT" ] || [ "$MOUNTED" ]; then
    if [ -z "$MONTHROTATE" ]; then
        TODAY=$(($DATE +%y%m%d)
    else
        TODAY=$(($DATE +%d)
    fi

    if [ "$SSHUSER" ] && [ "$SSHPORT" ]; then
        S="$SSH -p $SSHPORT -l $SSHUSER";
    fi

    for SOURCE in "${SOURCES[@]}"
    do
        if [ "$S" ] && [ "$FROMSSH" ] && [ -z "$TOSSH" ]; then
            $ECHO "$RSYNC -e \"$S\" -avR \"$FROMSSH:$SOURCE\" ${RSYNCCONF[@]}
$TARGET$TODAY $INC">> $LOG
            $RSYNC -e "$S" -avR "$FROMSSH:\"$SOURCE\" " "${RSYNCCONF[@]}"
"$TARGET"$TODAY $INC">> $LOG 2>&1
            if [ $? -ne 0 ]; then
                ERROR=1
            fi
        fi
    fi

```

```

    if [ "$S" ] && [ "$TOSSH" ] && [ -z "$FROMSSH" ]; then
        $ECHO "$RSYNC -e \"$S\" -avR \"$SOURCE\" ${RSYNCCONF[@]}
\"$TOSSH:$TARGET$TODAY\" $INC ">> $LOG
        $RSYNC -e "$S" -avR "$SOURCE" "${RSYNCCONF[@]}"
"$TOSSH:\"$TARGET\"$TODAY" $INC>> $LOG 2>&1
        if [ $? -ne 0 ]; then
            ERROR=1
        fi
    fi
    if [ -z "$S" ]; then
        $ECHO "$RSYNC -avR \"$SOURCE\" ${RSYNCCONF[@]} $TARGET$TODAY $INC">>
$LOG
        $RSYNC -avR "$SOURCE" "${RSYNCCONF[@]}" "$TARGET"$TODAY $INC>> $LOG
2>&1
        if [ $? -ne 0 ]; then
            ERROR=1
        fi
    fi
done

if [ "$S" ] && [ "$TOSSH" ] && [ -z "$FROMSSH" ]; then
    $ECHO "$SSH -p $SSHPORT -l $SSHUSER $TOSSH $LN -nsf $TARGET$TODAY
$TARGET$LAST">> $LOG
    $SSH -p $SSHPORT -l $SSHUSER $TOSSH "$LN -nsf \"$TARGET\"$TODAY
\"$TARGET\"$LAST">> $LOG 2>&1
    if [ $? -ne 0 ]; then
        ERROR=1
    fi
fi
if ( [ "$S" ] && [ "$FROMSSH" ] && [ -z "$TOSSH" ] ) || ( [ -z "$S" ] );
then
    $ECHO "$LN -nsf $TARGET$TODAY $TARGET$LAST">> $LOG
    $LN -nsf "$TARGET"$TODAY "$TARGET"$LAST>> $LOG 2>&1
    if [ $? -ne 0 ]; then
        ERROR=1
    fi
fi
else
    $ECHO "$MOUNTPOINT not mounted">> $LOG
    ERROR=1
fi
$DATE>> $LOG
if [ -n "$MAILREC" ]; then
    if [ $ERROR ];then
        $MAIL -s "Error Backup $LOG" $MAILREC <$LOG
    else
        $MAIL -s "Backup $LOG" $MAILREC <$LOG
    fi
fi

```

# rsnapshot-Tagessicherung mit Überwachung und Fehlermail

## Ziel

Die tägliche rsnapshot-Sicherung soll automatisch gestartet und überwacht werden. Wenn die Sicherung fehlschlägt oder bereits ein Sicherungslauf aktiv ist, soll automatisch eine E-Mail an [webmaster@waldhofer.com](mailto:webmaster@waldhofer.com) gesendet werden.

Die Lösung verwendet ein Shell-Skript mit:

- \* Start der Tagessicherung
- \* Protokollierung in eine Logdatei
- \* Fehlererkennung über den Rückgabecode
- \* Mailversand im Fehlerfall
- \* Sperre gegen Doppelstarts mit flock

---

## Wichtiger Hinweis

Die Tagessicherung ist bereits im Skript integriert. Es ist **kein zusätzlicher Cronjob für rsnapshot daily** notwendig.

Das Skript enthält den eigentlichen Sicherungsaufruf:

```
usr/bin/rsnapshot daily>> "$LOGFILE" 2>&1
```

Es genügt daher **ein einziger Cronjob**, der nur das Überwachungsskript startet.

---

## Warum "flock" verwenden?

Mit flock wird verhindert, dass ein zweiter Sicherungslauf startet, wenn der erste noch aktiv ist.

Vorteile:

- \* keine Doppelstarts
- \* saubere Sperrlogik
- \* robust unter Ubuntu
- \* besser als einfache selbst gebaute PID-/Lockfile-Prüfungen

---

## Skript anlegen

Datei erstellen:

```
sudo nano /usr/local/bin/rsnapshot_daily_monitor.sh
```

Inhalt:

```
#!/bin/bash

set -u

LOGFILE="/var/log/rsnapshot_daily.log"\
MAILTO="webmaster@waldhofer.com"\
HOSTNAME="$(hostname -f 2>/dev/null || hostname)"\
LOCKFILE="/var/run/rsnapshot_daily.lock"

send_error_mail() {\
    local subject="$1"\
    local body="$2"

    /usr/sbin/sendmail "$MAILTO" <<EOF\
Subject: $subject\
From: root@$HOSTNAME\
To: $MAILTO

$body\
EOF\
}

{\
    flock -n 9 || {\
        DATE_NOW="$(date '+%Y-%m-%d %H:%M:%S')"\
        echo "==== $DATE_NOW LOCK AKTIV: rsnapshot daily laeuft bereits\n====">> "$LOGFILE"\
        send_error_mail \
            "[FEHLER] rsnapshot bereits aktiv auf $HOSTNAME" \
            "Der rsnapshot-Job wurde nicht gestartet, weil bereits ein Lauf\naktiv ist.

Host: $HOSTNAME\
Zeit: $DATE_NOW\
Lockdatei: $LOCKFILE\
Logdatei: $LOGFILE"\
        exit 1\
    }

    DATE_NOW="$(date '+%Y-%m-%d %H:%M:%S')"\
    echo "==== Start rsnapshot daily: $DATE_NOW ==== ">> "$LOGFILE"

    /usr/bin/rsnapshot daily>> "$LOGFILE" 2>&1\
    RETVAL=$?

    DATE_END="$(date '+%Y-%m-%d %H:%M:%S')"
```

```
if [ $RETVAL -eq 0 ]; then\  
    echo "==== Ende rsnapshot daily erfolgreich: $DATE_END ====">>>  
"$LOGFILE"\  
else\  
    echo "==== Ende rsnapshot daily FEHLER ($RETVAL): $DATE_END ====">>>  
"$LOGFILE"\  
    send_error_mail \  
        "[FEHLER] rsnapshot Backup auf $HOSTNAME" \  
        "Die rsnapshot-Sicherung ist fehlgeschlagen."  
  
Host: $HOSTNAME\  
Zeit: $DATE_END\  
Rueckgabecode: $RETVAL\  
Logdatei: $LOGFILE  
  
Die letzten 50 Zeilen aus dem Log:  
  
$(tail -n 50 "$LOGFILE")\  
fi  
  
exit $RETVAL\  
} 9>"$LOCKFILE"
```

---

## Skript ausführbar machen

```
sudo chmod +x /usr/local/bin/rsnapshot_daily_monitor.sh
```

---

## Cronjob einrichten

Root-Crontab bearbeiten:

```
sudo crontab -e
```

Eintrag hinzufügen:

```
0 1 * * * /usr/local/bin/rsnapshot_daily_monitor.sh
```

Damit wird jeden Tag um 01:00 Uhr:

- \* das Skript gestartet
- \* die Tagessicherung ausgeführt
- \* das Ergebnis protokolliert
- \* im Fehlerfall eine Mail versendet

## Wichtige Regel

Ein alter direkter Cronjob mit `rsnapshot daily` darf **nicht zusätzlich** aktiv sein.

Also **nicht beides gleichzeitig**:

```
0 1 * * * /usr/bin/rsnapshot daily\  
0 1 * * * /usr/local/bin/rsnapshot_daily_monitor.sh
```

Richtig ist nur:

```
0 1 * * * /usr/local/bin/rsnapshot_daily_monitor.sh
```

## Funktion testen

Skript manuell starten:

```
sudo /usr/local/bin/rsnapshot_daily_monitor.sh  
echo $?
```

Bedeutung:

- \* 0 = Sicherung erfolgreich
- \* 1 oder anderer Wert ungleich 0 = Fehler erkannt

Logdatei ansehen:

```
sudo tail -n 50 /var/log/rsnapshot_daily.log
```

## rsnapshot separat prüfen

Falls das Skript Fehler liefert, zuerst `rsnapshot` selbst testen:

Konfiguration prüfen:

```
\\  
sudo /usr/bin/rsnapshot configtest\  
echo $?\\
```

Tagessicherung direkt starten:

```
\\  
sudo /usr/bin/rsnapshot daily\\
```

```
echo $?\
```

---

## Mailversand testen

Testmail senden:

```
\
echo -e "Subject: Testmail\n\nDas ist ein Test." | sudo /usr/sbin/sendmail
webmaster@waldhofer.com\
```

Wenn die Mail ankommt, funktioniert der Versand über smtp bzw. sendmail.

---

## Logdatei

Die Logdatei ist:

```
\
/var/log/rsnapshot_daily.log\
```

Dort stehen unter anderem:

- \* Startzeit des Sicherungslaufs
  - \* Ende des Sicherungslaufs
  - \* Erfolg oder Fehler
  - \* eventuelle Fehlermeldungen von rsnapshot
- 

## Typische Fehlerquellen

- \* Fehler in /etc/rsnapshot.conf
  - \* Zielverzeichnis nicht erreichbar
  - \* fehlende Berechtigungen
  - \* zu wenig Speicherplatz
  - \* SSH-Probleme bei entfernten Sicherungszielen
  - \* fehlerhafte Mail-Konfiguration
  - \* bereits laufender zweiter Sicherungsjob
- 

## Zusammenfassung

Diese Lösung kombiniert:

- \* Tagessicherung

- \* Überwachung
- \* Logging
- \* Fehlermail
- \* Schutz vor Doppelstarts

in **einem einzigen Skript**.

Daher reicht **ein einziger Cronjob** völlig aus.

From:

<http://wiki.waldhofer.at/> - **Wiki von Franz**

Permanent link:

<http://wiki.waldhofer.at/doku.php?id=ubuntu:backup&rev=1775983844>

Last update: **2026/04/12 10:50**

