Automatizare Backup în Linux

Timp lucru – 1.5 -2 ore

1. Introducere

Acest tutorial te va învăța să creezi scripturi de backup automate în Linux, de la nivel de bază până la scripturi mai avansate pentru administrarea serverelor.

1. Ce vei învăța:

   Planificarea backup-ului
   Diferența tar vs rsync
   Scripturi Bash
   Programare CRON
   Monitorizare

!!! *Acest tutorial este pentru admini începători Linux cu cunoştinţe de bază Bash.*

2. Pas 1: Decizi Ce Backup

Identifică datele importante.

Tipuri de Date

Documente personale:

/home/user/Documents
/home/user/Pictures
/home/user/Projects

Configurări server:

/etc/nginx/, /var/www/html/
MySQL dumps
/etc/ssh/

3. Pas 2: Unde Backup

Locația e critică.

Opțiuni Stocare

Partiție: /backup
Disk extern: /mnt/backup_disk
NAS: /mnt/nas/
Remote: ssh

!!! *NU face backup pe același disk fizic!*

Structură

```
/backup/
├── daily/
├── weekly/
└── logs/
```

4. Pas 3: Metoda

4.1 tar

Arhivare completă într-un .tar.gz
Avantaje:
- Un singur fișier
- Compresie bună
- Portabil

Când:
Backup periodic (zilnic/săptămânal)

```
tar -czf backup.tar.gz /source

-c=create, -z=gzip, -f=file
```

4.2 rsync

Sincronizare incrementală
Avantaje:
- Doar ce s-a schimbat
- Rapid
- Păstrează structura

Când:
Backup frecvent, seturi mari

```
rsync -av /source/ /backup/

-a=archive, -v=verbose
```

5. Pas 4: Scripturi

5.1 Script SIMPLU - Documente

```
#!/bin/bash

SOURCE="/home/$USER/Documents"
BACKUP="/backup/daily"
DATE=$(date +%Y-%m-%d_%H-%M-%S)
FILE="docs_$DATE.tar.gz"

mkdir -p "$BACKUP"
echo "Start backup..."
tar -czf "$BACKUP/$FILE" "$SOURCE"

if [ $? -eq 0 ]; then
  echo "✓ Succes: $FILE"
  ls -lh "$BACKUP/$FILE"
else
  echo "  Eroare!"
  exit 1
fi
```

Explicații:

• SOURCE - ce salvezi

• BACKUP - unde salvezi

• DATE - timestamp

• $? - exit code (0=ok)

!!! *Modifică căile cu valorile tale!*

Utilizare:

```
nano backup_docs.sh
chmod ug+x backup_docs.sh
./backup_docs.sh
```

5.2 Script AVANSAT - Server

```
#!/bin/bash

BACKUP_ROOT="/backup"
DATE=$(date +%Y-%m-%d)
LOG="$BACKUP_ROOT/logs/backup_$DATE.log"
MAX=7  # păstrează 7 zile

log() {
  echo "[$(date)] $1" | tee -a "$LOG"
}

check_space() {
  local avail=$(df -BG "$BACKUP_ROOT" | awk 'NR==2{print $4}' | sed 's/G//')
  if [ "$avail" -lt 5 ]; then
    log "EROARE: Spațiu insuficient!"
    exit 1
  fi
}

mkdir -p "$BACKUP_ROOT/{daily,logs,mysql}"
log "===== START BACKUP ====="
check_space

# Nginx
log "Backup nginx..."
tar -czf "$BACKUP_ROOT/daily/nginx_$DATE.tar.gz" /etc/nginx/ 2>>$LOG
[ $? -eq 0 ] && log "✓ Nginx OK" || log "  Nginx FAIL"

# Website
log "Backup website..."
rsync -av --delete /var/www/html/ "$BACKUP_ROOT/daily/www_$DATE/" >>$LOG 2>&1
[ $? -eq 0 ] && log "✓ Website OK" || log "  Website FAIL"

# MySQL
if command -v mysqldump &>/dev/null; then
  log "Backup MySQL..."
  mysqldump -u root --all-databases >$BACKUP_ROOT/mysql/db_$DATE.sql 2>>$LOG
  [ $? -eq 0 ] && gzip $BACKUP_ROOT/mysql/db_$DATE.sql && log "✓ MySQL OK"
fi

# Rotație
find $BACKUP_ROOT/daily -type f -mtime +$MAX -delete
log "✓ Cleanup vechi ($MAX zile)"
log "===== END ====="
du -sh $BACKUP_ROOT/* | tee -a $LOG
```

Funcții:

• log() - scrie în console + fișier
• check_space() - verifică 5GB disponibil
• Rotație - șterge >7 zile

!!! *Pentru MySQL: configurează /root/.my.cnf cu user/password!*

6. Pas 5: Testare

Test în director temporar:

```
mkdir -p /tmp/{test_src,test_bkp}
echo "test" > /tmp/test_src/file.txt

# Modifică script temporar:
SOURCE="/tmp/test_src"
BACKUP="/tmp/test_bkp"

# Rulează
./backup.sh

# Verifică
ls -lh /tmp/test_bkp/
tar -tzf /tmp/test_bkp/*.tar.gz
```

Test Restaurare

```
mkdir /tmp/restore
tar -xzf /backup/backup.tar.gz -C /tmp/restore/
ls -la /tmp/restore/
```

!!! *Testează restaurarea ÎNAINTE de producție!*

7. Pas 6: CRON

CRON rulează automat comenzi.

Sintaxă

```
┌─ minut (0-59)
│ ┌─ oră (0-23)
│ │ ┌─ zi (1-31)
│ │ │ ┌─ lună (1-12)
│ │ │ │ ┌─ zi săpt (0-7)
│ │ │ │ │
* * * * * comandă

Exemple:
0 2 * * * /root/backup.sh     # Zilnic 2 AM
0 3 * * 0 /root/backup.sh     # Duminică 3 AM
0 */6 * * * /root/backup.sh   # La 6 ore
```

Configurare

```
crontab -e   # editează
crontab -l   # listează

# Adaugă:
0 2 * * * /home/user/backup.sh >> /var/log/backup.log 2>&1
```

!!! *Folosește căi ABSOLUTE în cron!*

Troubleshooting:

```
systemctl status cron
grep CRON /var/log/syslog
ls -l /path/script.sh  # verifică +x
```

8. Pas 7: Monitorizare

Verifică regulat!

Zilnic (5 min)

```
ls -lh /backup/daily/ | grep $(date +%Y-%m-%d)
tail -20 /backup/logs/backup_$(date +%Y-%m-%d).log
```

Săptămânal (15 min)

```
find /backup/daily -type f -mtime -7 -ls
du -sh /backup/*
tar -tzf /backup/daily/file.tar.gz >/dev/null && echo "OK"
```

Script Verificare

```
#!/bin/bash
BKP="/backup/daily"
RECENT=$(find $BKP -mtime -1 | wc -l)

if [ $RECENT -eq 0 ]; then
  echo "⚠ ATENȚIE: Niciun backup recent!"
  exit 1
else
  echo "✓ $RECENT backup-uri recente"
fi

echo "Ultimele 3:"
ls -lht $BKP | head -4
```

!!! *CEL MAI IMPORTANT: testează RESTAURAREA lunar!*

9. Top 10 Erori + Soluții

1. Permission denied

```
tar: Cannot open: Permission denied
```

Soluție:

```
sudo ./script.sh
sudo chmod 755 /directory
```

2. No space left

```
Cannot write: No space left
```

Soluție:

```
df -h /backup
find /backup -mtime +14 -delete
```

3. CRON nu rulează

```
Script corect dar nu rulează
```

Soluție:

```
systemctl status cron
# Folosește căi absolute!
0 2 * * * /full/path/script.sh >>log 2>&1
```

4. tar leading /

```
Removing leading / warning
```

Soluție:

```
# Normal, nu-i eroare!
# SAU:
cd /home && tar -czf backup.tar.gz user/
```

5. rsync failed times

```
failed to set times
```

Soluție:

```
rsync -av --no-perms --no-times /src /dst
```

6. MySQL access denied

```
1045: Access denied
```

Soluție:

```
# Crează /root/.my.cnf:
[client]
user=root
password=pass

chmod 600 /root/.my.cnf
```

7. Script fără output

```
Rulează dar nu creează fișier
```

Soluție:

```
set -x  # debug
```

```
echo "SOURCE: $SOURCE"
[ ! -d "$SOURCE" ] && echo "NU EXISTĂ!" && exit 1
```

8. Backup durează mult

```
Blochează minute/ore
```

Soluție:

```
tar --exclude=Videos --exclude=.cache -czf backup.tar.gz /src
# SAU rulează noaptea: 0 2 * * *
```

9. Arhivă coruptă

```
gzip: not in gzip format
```

Soluție:

```
# Verifică IMEDIAT:
tar -tzf file.tar.gz
if [ $? -eq 0 ]; then echo OK; fi

# Adaugă în script verificare automată!
```

10. CRON prea multe emails

```
Email pentru fiecare output
```

Soluție:

```
# Redirectează:
0 2 * * * /script.sh >/dev/null 2>&1
# SAU doar la erori:
0 2 * * * /script.sh || mail -s "FAIL" admin@
```

10. Rezumat

Pași:

2. 1. Identifică date importante
3. 2. Alege locație sigură
4. 3. tar (complet) sau rsync (incremental)
5. 4. Scrie script (simplu → avansat)
6. 5. Testează manual + restaurare!
7. 6. Programează CRON
8. 7. Monitorizează regulat

Regula 3-2-1

!!! *3 copii, 2 media diferite, 1 offsite (altă locație fizică)*

Comenzi Esențiale

```
# BACKUP
tar -czf bkp.tar.gz /src/
rsync -av /src/ /dst/

# VERIFICARE
tar -tzf bkp.tar.gz
df -h /backup

# RESTAURARE
tar -xzf bkp.tar.gz -C /restore/

# CRON
crontab -e
systemctl status cron

# MONITORING
tail -f /var/log/backup.log
find /backup -mtime -7
```

Resurse:

• tar: man tar, gnu.org/software/tar
• rsync: man rsync, rsync.samba.org
• cron: crontab.guru

!!! *Un backup netestat = niciun backup! Testează restaurarea LUNAR!*