MagicMirror – Backup auf NAS

Ziel

Der MagicMirror-Ordner wird auf ein NAS gesichert.

Es gibt zwei Varianten:

- **Manuelles Backup:** Start bei Bedarf per Befehl
- **Automatisches Backup:** jeden Sonntag um 01:00 Uhr

Eigenschaften:

- Backup als komprimiertes `tar.gz`
- `node_modules` werden nicht gesichert
- SHA-256-Prüfsumme zur Integritätskontrolle
- NAS muss tatsächlich erreichbar und gemountet sein
- unvollständige Backups bleiben als `_INCOMPLETE` erhalten
- automatische Backups: maximal **4 Stück**
- Erfolg/Fehler wird über **ioBroker → Telegram** gemeldet

Verwendete Pfade

Funktion	Pfad / Ort
Manuelles Backup-Script	`/home/pi/bin/backup_magicmirror.sh`
Wöchentliches Backup-Script	`/home/pi/bin/backup_magicmirror_weekly.sh`
ioBroker-Zugangsdaten	`/home/pi/.config/magicmirror-backup/iobroker.conf`
Backup-Logs	`/home/pi/magicmirror-backup-logs/`
systemd-Service	`/etc/systemd/system/magicmirror-backup-weekly.service`
systemd-Timer	`/etc/systemd/system/magicmirror-backup-weekly.timer`
NAS-Zugangsdaten	`/etc/samba/credentials-nasbackup`
NAS-Mount-Konfiguration	`/etc/fstab`
NAS-Mountpunkt	`/mnt/nasbackup`
Manuelle Backups	`/mnt/nasbackup/Manuelle_Backups/`
Automatische Backups	`/mnt/nasbackup/Automatische_Backups/`
ioBroker Telegram-Script	`ioBroker → Skripte → JavaScript`
ioBroker Backup-Datenpunkte	`0_userdata.0.MagicMirror.Backup.*`

1. NAS auf dem Raspberry Pi einbinden

Pakete installieren

```
sudo apt update
```

```
sudo apt install cifs-utils
```

Gruppe anlegen

```
sudo groupadd nasusers
sudo usermod -aG nasusers <PI_USER>
```

Danach neu anmelden oder neu starten.

UID und GID ermitteln:

```
id -u <PI_USER>
getent group nasusers
```

Mountpunkt anlegen

```
sudo mkdir -p /mnt/nasbackup
```

NAS-Zugangsdaten speichern

```
sudo mkdir -p /etc/samba
sudo nano /etc/samba/credentials-nasbackup
```

Inhalt:

```
username=<NAS_USER>
password=<NAS_PASSWORT>
```

Datei schützen:

```
sudo chown root:root /etc/samba/credentials-nasbackup
sudo chmod 600 /etc/samba/credentials-nasbackup
```

/etc/fstab

Vorher sichern:

```
sudo cp /etc/fstab /etc/fstab.backup
sudo nano /etc/fstab
```

Eintrag:

```
//<NAS_HOST>/<NAS_SHARE> /mnt/nasbackup cifs
credentials=/etc/samba/credentials-
nasbackup,vers=3.0,iocharset=utf8,uid=<UID>,gid=<GID>,file_mode=0660,dir_mod
e=0770,_netdev,nofail,x-systemd.automount,x-systemd.mount-timeout=10s 0 0
```

systemd neu laden:

```
sudo systemctl daemon-reload
sudo systemctl start mnt-nasbackup.automount
```

Status:

```
systemctl status mnt-nasbackup.automount --no-pager
```

Erwartet:

```
Active: active (waiting)
```

NAS testen

Mount auslösen:

```
ls -la /mnt/nasbackup
```

Prüfen:

```
findmnt /mnt/nasbackup
df -h /mnt/nasbackup
```

Schreibtest:

```
touch /mnt/nasbackup/test.txt
rm /mnt/nasbackup/test.txt
```

2. Manuelles MagicMirror-Backup

Script anlegen

```
mkdir -p ~/bin
nano ~/bin/backup_magicmirror.sh
```

Inhalt:

```
#!/usr/bin/env bash
```

```
SOURCE="$HOME/MagicMirror"
NAS="/mnt/nasbackup"
BACKUP_BASE="$NAS/Manuelle_Backups"
LOGDIR="$HOME/magicmirror-backup-logs"

# Anpassen:
EXPECTED_SOURCE="//<NAS_HOST>/<NAS_SHARE>"


check_nas() {

    if ! timeout 15s stat "$NAS" >/dev/null 2>&1; then
        echo "FEHLER: NAS ist nicht erreichbar."
        return 1
    fi

    MOUNT_INFO="$(findmnt -T "$NAS" -n -o FSTYPE,SOURCE | tail -n 1)"
    FSTYPE="$(awk '{print $1}' <<< "$MOUNT_INFO")"
    MOUNT_SOURCE="$(awk '{print $2}' <<< "$MOUNT_INFO")"

    if [[ "$FSTYPE" != "cifs" || "$MOUNT_SOURCE" != "$EXPECTED_SOURCE" ]];
then
        echo "FEHLER: NAS ist nicht korrekt eingebunden."
        return 1
    fi

    return 0
}


if [[ "${1:-}" == "--worker" ]]; then

    FINAL_DEST="$2"
    TEMP_DEST="${FINAL_DEST}_INCOMPLETE"
    ARCHIVE="$TEMP_DEST/MagicMirror.tar.gz"

    exec 9>/tmp/magicmirror-backup.lock

    if ! flock -n 9; then
        echo "FEHLER: Es läuft bereits ein Backup."
        exit 1
    fi

    echo "========================================="
    echo "MagicMirror Backup"
    echo "Start:  $(date)"
    echo "Quelle: $SOURCE"
    echo "Ziel:   $FINAL_DEST"
    echo "========================================="
```

```
if ! check_nas; then
    echo "BACKUP ABGEBROCHEN: NAS nicht verfügbar."
    exit 2
fi

mkdir -p "$TEMP_DEST" || exit 3

echo "Erstelle Archiv ..."

tar \
    --create \
    --gzip \
    --dereference \
    --exclude='MagicMirror/node_modules' \
    --exclude='MagicMirror/node_modules/*' \
    --exclude='*/node_modules' \
    --exclude='*/node_modules/*' \
    --file="$ARCHIVE" \
    --directory="$HOME" \
    MagicMirror

RESULT=$?

if [[ $RESULT -ne 0 ]]; then
    echo "BACKUP FEHLGESCHLAGEN: tar Exit-Code $RESULT"
    exit "$RESULT"
fi

echo "Prüfe gzip ..."

gzip -t "$ARCHIVE" || exit 4

echo "Prüfe TAR ..."

tar -tzf "$ARCHIVE" >/dev/null || exit 5

echo "Erzeuge SHA-256 ..."

(
    cd "$TEMP_DEST" || exit 1
    sha256sum MagicMirror.tar.gz > MagicMirror.tar.gz.sha256
    sha256sum -c MagicMirror.tar.gz.sha256
) || exit 6

mv "$TEMP_DEST" "$FINAL_DEST" || exit 7

echo
echo "========================================="
echo "BACKUP ERFOLGREICH"
echo "Ende: $(date)"
```

```
    echo "Archiv: $FINAL_DEST/MagicMirror.tar.gz"
    du -h "$FINAL_DEST/MagicMirror.tar.gz"
    echo "========================================="

    exit 0
fi


if [[ ! -d "$SOURCE" ]]; then
    echo "FEHLER: $SOURCE existiert nicht."
    exit 1
fi

echo "Prüfe NAS ..."

if ! check_nas; then
    echo "BACKUP NICHT GESTARTET."
    echo "Es werden keine Daten lokal nach /mnt/nasbackup geschrieben."
    exit 1
fi

mkdir -p "$BACKUP_BASE"
mkdir -p "$LOGDIR"

STAMP="$(date '+%Y-%m-%d_%H-%M-%S')"
DEST="$BACKUP_BASE/MagicMirror_$STAMP"
LOG="$LOGDIR/MagicMirror_$STAMP.log"

echo "NAS ist erreichbar."
echo "Backup wird im Hintergrund gestartet."
echo "Ziel: $DEST"
echo "Log:  $LOG"

nohup "$0" --worker "$DEST" >"$LOG" 2>&1 < /dev/null &

echo "PID: $!"
echo "Log ansehen mit:"
echo "tail -f \"$LOG\""
```

Ausführbar machen:

```
chmod +x ~/bin/backup_magicmirror.sh
```

Manuelles Backup starten

```
~/bin/backup_magicmirror.sh
```

Neuestes Log ansehen:

```
tail -f "$(ls -t ~/magicmirror-backup-logs/*.log | head -1)"
```

`Ctrl+C` beendet nur die Log-Anzeige.

Ein Backup ist erfolgreich, wenn im Log steht:

```
BACKUP ERFOLGREICH
```

und der Backup-Ordner **kein** `_INCOMPLETE` enthält.

3. ioBroker für Backup-Meldungen

REST-API

In ioBroker den Adapter **REST API** installieren.

Empfohlene Einstellungen:

- Authentifizierung: **Ein**
- Keine GUI: **Ein**
- Commands erlauben: **Aus**
- Admin-Commands erlauben: **Aus**
- API nur im internen Netzwerk erreichbar machen
- kein Port-Forwarding ins Internet

Benutzer und Gruppe

ioBroker-Benutzer anlegen:

```
magicmirror
```

Gruppe anlegen:

```
magicmirror_backup
```

Benutzer der Gruppe hinzufügen.

Benötigte Gruppenrechte:

- Objekte: Auflisten + Lesen
- Zustände: Auflisten + Lesen + Schreiben
- HTTP: Ein
- Execute/SendTo: Aus

Datenpunkte

Folgende Datenpunkte werden benötigt:

```
0_userdata.0.MagicMirror.Backup.Status
0_userdata.0.MagicMirror.Backup.Message
0_userdata.0.MagicMirror.Backup.LastRun
0_userdata.0.MagicMirror.Backup.BackupPath
```

Wichtig:

Der Benutzer `magicmirror` sollte Besitzer dieser vier Datenpunkte sein.

Beispiel auf dem ioBroker-Host:

```
iobroker object chown magicmirror magicmirror_backup
0_userdata.0.MagicMirror.Backup.Status
iobroker object chown magicmirror magicmirror_backup
0_userdata.0.MagicMirror.Backup.Message
iobroker object chown magicmirror magicmirror_backup
0_userdata.0.MagicMirror.Backup.LastRun
iobroker object chown magicmirror magicmirror_backup
0_userdata.0.MagicMirror.Backup.BackupPath
```

Telegram-JavaScript

Im ioBroker-JavaScript-Adapter:

```
const BASE = '0_userdata.0.MagicMirror.Backup';

createState(`${BASE}.Status`, 'idle', {
    type: 'string',
    role: 'text',
    read: true,
    write: true
});

createState(`${BASE}.Message`, '', {
    type: 'string',
    role: 'text',
    read: true,
    write: true
});

createState(`${BASE}.LastRun`, '', {
    type: 'string',
    role: 'text',
```

```
    read: true,
    write: true
});

createState(`${BASE}.BackupPath`, '', {
    type: 'string',
    role: 'text',
    read: true,
    write: true
});


on({
    id: `${BASE}.Status`,
    change: 'ne'
}, obj => {

    const status = String(obj.state.val);

    if (status !== 'success' && status !== 'error') return;

    const message = String(getState(`${BASE}.Message`).val || '');
    const lastRun = String(getState(`${BASE}.LastRun`).val || '');
    const backupPath = String(getState(`${BASE}.BackupPath`).val || '');

    let text;

    if (status === 'success') {
        text = `□ MagicMirror Backup erfolgreich\n\n${message}`;
    } else {
        text = `□ MagicMirror Backup FEHLGESCHLAGEN\n\n${message}`;
    }

    text += `\n\nZeit: ${lastRun}`;

    if (backupPath) {
        text += `\nBackup: ${backupPath}`;
    }

    sendTo('telegram.0', text);
});
```

4. Zugangsdaten für ioBroker auf dem Pi

```
mkdir -p ~/.config/magicmirror-backup
nano ~/.config/magicmirror-backup/iobroker.conf
```

Inhalt:

```
IOBROKER_URL='http://<IOBROKER_HOST>:<REST_PORT>'
IOBROKER_USER='magicmirror'
IOBROKER_PASSWORD='<PASSWORT>'
```

Datei schützen:

```
chmod 600 ~/.config/magicmirror-backup/iobroker.conf
```

5. Automatisches wöchentliches Backup

Script

```
nano ~/bin/backup_magicmirror_weekly.sh
```

Inhalt:

```
#!/usr/bin/env bash

BACKUP_SCRIPT="$HOME/bin/backup_magicmirror.sh"

NAS="/mnt/nasbackup"
AUTO_BASE="$NAS/Automatische_Backups"

LOGDIR="$HOME/magicmirror-backup-logs"

CONFIG="$HOME/.config/magicmirror-backup/iobroker.conf"

KEEP_BACKUPS=4

STATE_BASE="0_userdata.0.MagicMirror.Backup"


if [[ ! -r "$CONFIG" ]]; then
    echo "FEHLER: $CONFIG fehlt."
    exit 1
fi

source "$CONFIG"


set_iobroker_state() {

    local ID="$1"
    local VALUE="$2"
```

```
    curl \
        --fail \
        --silent \
        --show-error \
        --max-time 10 \
        --user "${IOBROKER_USER}:${IOBROKER_PASSWORD}" \
        --get \
        --data-urlencode "value=${VALUE}" \
        "${IOBROKER_URL}/v1/state/${ID}" \
        >/dev/null
}


notify_iobroker() {

    local STATUS="$1"
    local MESSAGE="$2"
    local BACKUP_PATH="${3:-}"

    set_iobroker_state "${STATE_BASE}.Message" "$MESSAGE" || return 1
    set_iobroker_state "${STATE_BASE}.BackupPath" "$BACKUP_PATH" || return 1
    set_iobroker_state "${STATE_BASE}.LastRun" "$(date -Iseconds)" || return
1

    # Status zuletzt setzen: dieser löst Telegram aus.
    set_iobroker_state "${STATE_BASE}.Status" "$STATUS"
}


mkdir -p "$LOGDIR"

STAMP="$(date '+%Y-%m-%d_%H-%M-%S')"

DEST="$AUTO_BASE/MagicMirror_Auto_$STAMP"

LOG="$LOGDIR/MagicMirror_Auto_$STAMP.log"


set_iobroker_state "${STATE_BASE}.Status" "running" || true


"$BACKUP_SCRIPT" --worker "$DEST" >"$LOG" 2>&1

RESULT=$?


if [[ $RESULT -eq 0 ]]; then

    SIZE="$(du -h "$DEST/MagicMirror.tar.gz" | awk '{print $1}')"

    mapfile -t BACKUPS < <(
```

```
        find "$AUTO_BASE" \
            -mindepth 1 \
            -maxdepth 1 \
            -type d \
            -name 'MagicMirror_Auto_*' \
            ! -name '*_INCOMPLETE' \
            -printf '%f\n' \
            | sort -r
    )

    if (( ${#BACKUPS[@]} > KEEP_BACKUPS )); then

        for ((i=KEEP_BACKUPS; i<${#BACKUPS[@]}; i++)); do
            rm -rf -- "$AUTO_BASE/${BACKUPS[$i]}"
        done

    fi


    MESSAGE="Automatisches MagicMirror-Backup erfolgreich. Größe: ${SIZE}.
Es werden ${KEEP_BACKUPS} automatische Backups aufbewahrt."

    notify_iobroker "success" "$MESSAGE" "$DEST" || true

    exit 0
fi


MESSAGE="Automatisches MagicMirror-Backup fehlgeschlagen. Exit-Code:
${RESULT}. Log: ${LOG}"

notify_iobroker "error" "$MESSAGE" "${DEST}_INCOMPLETE" || true

exit "$RESULT"
```

Ausführbar machen:

```
chmod +x ~/bin/backup_magicmirror_weekly.sh
```

Manueller Test:

```
~/bin/backup_magicmirror_weekly.sh
```

Bei Erfolg:

- Backup liegt unter `Automatische_Backups`
- Telegram-Meldung wird gesendet
- Exit-Code ist `0`

6. systemd-Timer

Service

```
sudo nano /etc/systemd/system/magicmirror-backup-weekly.service
```

```
[Unit]
Description=Wöchentliches MagicMirror Backup auf NAS
Wants=network-online.target
After=network-online.target

[Service]
Type=oneshot
User=<PI_USER>
Group=<PI_USER>
Environment=HOME=/home/<PI_USER>
WorkingDirectory=/home/<PI_USER>

ExecStart=/home/<PI_USER>/bin/backup_magicmirror_weekly.sh

TimeoutStartSec=infinity
```

Timer

```
sudo nano /etc/systemd/system/magicmirror-backup-weekly.timer
```

```
[Unit]
Description=MagicMirror Backup jeden Sonntag um 01:00 Uhr

[Timer]
OnCalendar=Sun *-*-* 01:00:00
Persistent=true
AccuracySec=1min

[Install]
WantedBy=timers.target
```

Aktivieren

```
sudo systemctl daemon-reload
sudo systemctl enable --now magicmirror-backup-weekly.timer
```

Kontrolle:

```
systemctl list-timers --all | grep magicmirror
```

Timer-Status:

```
systemctl status magicmirror-backup-weekly.timer --no-pager
```

Erwartet:

```
Active: active (waiting)
```

Service manuell testen

```
sudo systemctl start magicmirror-backup-weekly.service
```

Danach:

```
systemctl status magicmirror-backup-weekly.service --no-pager
```

7. Backups prüfen

SHA-256

```
cd /mnt/nasbackup/Manuelle_Backups/MagicMirror_YYYY-MM-DD_HH-MM-SS
sha256sum -c MagicMirror.tar.gz.sha256
```

Erwartet:

```
MagicMirror.tar.gz: OK
```

Archivinhalt ansehen

```
tar -tzf MagicMirror.tar.gz | less
```

8. Wiederherstellung

Auf gleichem System

Archiv prüfen:

```
sha256sum -c MagicMirror.tar.gz.sha256
```

Dann:

```
tar -xzf MagicMirror.tar.gz -C ~
```

Auf einem neuen Raspberry Pi

Bei einem neuen Pi sollte MagicMirror und Node.js zunächst frisch installiert werden.

Aus dem Backup werden anschließend insbesondere übernommen:

- `config/config.js`
- `css/custom.css`
- Drittmodule und deren Konfiguration
- eigene Bilder, Skripte und Dateien

`node_modules` werden bewusst nicht wiederhergestellt.

Die Abhängigkeiten werden auf dem neuen System neu installiert.

9. Alltag

Manuelles Backup

```
~/bin/backup_magicmirror.sh
```

Backup-Log

```
tail -f "$(ls -t ~/magicmirror-backup-logs/*.log | head -1)"
```

Automatische Backups

Jeden Sonntag:

```
01:00 Uhr
```

Es werden maximal:

```
4 automatische Backups
```

aufbewahrt.

Timer prüfen

```
systemctl list-timers --all | grep magicmirror
```

NAS prüfen

```
findmnt /mnt/nasbackup
df -h /mnt/nasbackup
```

From:
https://wiki.mahlen.eu/ - **Smart-Home Wiki**

Permanent link:
https://wiki.mahlen.eu/doku.php?id=magicmirror:magicmirror_tipps:backup&rev=1791408750

Last update: **07.10.2026**