

# Mikrotik

- [RouterOS](#)
  - [MikroTik RouterOS als Layer-2 Switch konfigurieren](#)
  - [MAC-Address Tabelle anzeigen lassen in RouterOS](#)
- [Hardware](#)
  - [LCD-Display "Standby" deaktivieren](#)
- [RouterOS Skripte](#)
  - [RouterOS Backup per E-Mail versenden](#)
- [CHR \(Cloud Hosted Router\)](#)
  - [CHR Router in einer Proxmox VM installieren](#)

# RouterOS

RouterOS

# MikroTik RouterOS als Layer-2 Switch konfigurieren

Im Anhang befindet sich die Anleitung.

# MAC-Address Tabelle anzeigen lassen in RouterOS

## Einleitung

In diesem Beitrag beschreibe ich kurz, wie wir in **RouterOS** uns die **MAC-Address Tabelle** anzeigen lassen können.

## Tabelle anzeigen

Um die Tabelle uns anzeigen zu lassen, setzen wir den folgenden Befehl in der CLI ab:

```
/interface bridge host print
```

Über **Winbox** befindet sich die Tabelle unter **Bridge / Hosts**. Dort können wir dann nach Hosts filtern.

# Hardware

# LCD-Display "Standby" deaktivieren

## Standby deaktivieren

Bei der Konfiguration des **MikroTik Routers** hat man vielleicht ja den Wunsch, dass das **LCD-Display** nicht in den "Standby" Modus geht. Dafür muss im Terminal nur der nachstehende Befehl eingegeben werden.

```
/lcd set backlight-timeout=never
```

# RouterOS Skripte

# RouterOS Backup per E-Mail versenden

## Einleitung

Anbei ein **RouterOS Skript** um ein Backup von der **Gerätekonfiguration** zu erstellen, und dieses im Anschluss per E-Mail zu versenden. Das lokal gespeicherte Backup wird im Anschluss dann wieder gelöscht.

```
:local fileName "backup_${/system identity get name}.backup"
/log info "Backup erstellt! ($fileName)"
/system backup save name=$fileName
:delay 5s

/tool e-mail send file=$fileName to="backup@mail.de" from="MikroTik<mikrotik@mail.de>"
body="Backup im Anhang" subject="Backup ${/system identity get name}" server="<Mail-Server
IP>" port=25
/log info "Backup versendet!"
:delay 5s;
/file remove $fileName
```

Wenn dieses Skript automatisch ausgeführt werden soll, kann über den nachfolgenden Befehl ein wiederkehrender **Task** erstellt werden. Dabei muss das Skript aber den Namen `send_backup` haben.

```
add interval=1d name=send_backup_email on-event=send_backup policy=\
ftp,reboot,read,write,policy,test,password,sniff,sensitive,romon \
start-date=may/13/2024 start-time=02:00:00
```

## Kurzanleitung

Um das schnell umsetzen, einfach den folgenden Befehl absetzen und dann die entsprechenden Parameter des E-Mailservers ändern:

```
/system/script
add dont-require-permissions=no name=send_backup owner=admin policy=\
ftp,reboot,read,write,policy,test,password,sniff,sensitive,romon source=":lo\
```



```
cal fileName "\"backup_\[ /system identity get name].backup\" \"r\
\n/log info \"Backup erstellt! (\$fileName)\" \"r\
\n/system backup save name=\$fileName\r\
\n:delay 5s\r\
\n\r\
\n/tool e-mail send file=\$fileName to=\"backup@mail.de\" from=\"\
MikroTik<mikrotik@mail.de>\" body=\"Backup im Anhang\" subject=\"Bac\
kup \[ /system identity get name]\" server=\"<Server-IP>\" port=25\r\
\n/log info \"Backup versendet!\" \"r\
\n:delay 5s;\r\
\n/file remove \$fileName"
/system scheduler
add interval=1d name=send_backup_email on-event=send_backup policy=\
ftp,reboot,read,write,policy,test,password,sniff,sensitive,romon \
start-date=may/13/2024 start-time=02:00:00
```

# CHR (Cloud Hosted Router)

CHR (Cloud Hosted Router)

# CHR Router in einer Proxmox VM installieren

In diesem kleinen Artikel geht es kurz darum, wie wir eine virtuelle Maschine unter Proxmox erstellen können, welche den **CHR (MikroTik Cloud Hosted Router)** ausführt.

Der CHR unterstützt die X86\_64-Bit-Architektur und kann daher *Baremetal* oder innerhalb eines Hypervisors (z.B. VMware, Xen, Hyper-V, Proxmox) installiert werden. Beim CHR gibt es 4 "Lizenzlevel". Innerhalb jeden Lizenzlevels besitzen wir alle Funktionen eines MikroTik Routers. Die Lizenzen unterscheiden sich ausschließlich in der Geschwindigkeit der Schnittstellen.

Lizenzlevel	Geschwindigkeit pro Schnittstelle	Preis
free	1 Mbit	Kostenlos
p1	1 Gbit	Einmalig 45 \$
p10	10 Gbit	Einmalig 95 \$
p-unlimited	Kein Limit	Einmalig 250 \$

## Installation

Im ersten Schritt öffnen wir die Verwaltungskonsole unseres Proxmox Servers. Dazu öffnen wir die folgende Internetseite und melden uns mit unseren Anmeldedaten an:

```
https://<IP-Adresse>:8006
```

Dort angekommen, öffnen wir die Shell unseres **Proxmox-Servers**.

Natürlich kann die **Shell** auch über eine **SSH-Verbindung** geöffnet werden.

## Virtuelle Maschine erstellen

Im ersten Schritt müssen wir die virtuelle Maschine erstellen. Dabei müssen wir die ID anpassen, auf eine ID, welche noch nicht vergeben ist.

```
qm create <VM-ID> --name RouterOS --net0 virtio,bridge=vmbr0 --memory 1024 --sockets 1 --cores 2
```

**Info:** Falls unsere Virtuelle Maschine mehr Ressourcen erhalten soll, oder an eine andere bridge angeschlossen werden soll, muss der Befehl natürlich entsprechend angepasst werden.

## CHR Image herunterladen und installieren

In diesem Schritt werden wir das Image des CHR herunterladen und an unsere VM anbinden. Mit dem unten stehenden Befehl laden wir das Image in der Version 7.14.3 herunter und entpacken diese.

```
wget https://download.mikrotik.com/routeros/7.14.3/chr-7.14.3.img.zip
unzip chr-7.14.3.img.zip
```

Im Anschluss wird das Image als VM-Disk importiert und an unsere virtuelle Maschine geknüpft und die Festplatte etwas vergrößert.

```
qm importdisk <VM-ID> chr-7.14.3.img local-lvm
qm set <VM-ID> --scsihw virtio-scsi-pci --scsi0 local-lvm:vm-<VM-ID>-disk-0
qm resize <VM-ID> scsi0 +1896M
```

## Einstellungen für die virtuelle Maschine

Im letzten Schritt stellen wir noch ein paar Optionen für die virtuelle Maschine ein und starten die virtuelle Maschine zum Abschluss.

```
qm set <VM-ID> --boot order=scsi0
qm set <VM-ID> --agent 1
qm set <VM-ID> --tablet 0
qm start <VM-ID>
```

Die virtuelle Maschine sollte jetzt starten und man kann sich per **Winbox** oder **HTTP** mit dem Router verbinden.