Unifi - Protect

Alles rund um Sicherheitslösungen von Ubiquiti

- Unifi Protect im Docker Container in einer arm64 Debian VM installieren
- Werkseinstellungen - UNVR
- Unifi-Protect - Fehlerseite an Error occoured
- Unifi Protect im Docker Container in einer amd64 Debian VM installieren mit qemu arm64
- Unifi G4 Pro (Standalone Mode) Zeitstempel ändern
- Gruppen

Unifi Protect im Docker Container in einer arm64 Debian VM installieren

Beschreibung:

Unifi UNVR im Docker Conatiner auf einem ARm64 Host. Wir können einmal das Image aus dem Docker Hub Pullen oder uns auch ein eigenes Image bauen. Erstmal die Variante mit dem Pullen.

Vorraussetzung:

Einmal ein arm64 Host in dem Docker installiert ist.
Eine zweite HDD mindesten 128 GB groß. würde aber eine Größe wählen die größer ist. Unbedingt eine Partioion und ext4 so kann sie on the fly vergrößert werden.

Grundinstallation Docker und Einbungung Festplatte

Installation Docker, als root per ssh einloggen

```
apt install docker.io gdisk curl
```

Nun eine weitere Festplatte hinzufügen und diese Partitionieren wenn nicht schon geschehen
Mit lsbk die neue Festplatte ermitteln

```
oot@unnvr:~# lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda      8:0    0    32G  0 disk
├─sda1   8:1    0   512M  0 part /boot/efi
```

```
├─sda2   8:2    0  30.5G  0 part /
└─sda3   8:3    0   976M  0 part [SWAP]
sdb      8:16   0   256G  0 disk
sr0     11:0    1 329.3M  0 rom
```

sdb ist die gewünschte Festplatte

nun partitonieren

```
gdisk /dev/sdb

dann o eingeben für neues layout
dann n eingeben für eine neue partition

alles durchentern

nun w für schreiben
und exit
```

Nun Dateisystem ext4 erzeugen auf sdb1

```
mkfs.ext4 /dev/sdb1
```

Nun ein verzeichnis für den mount point erstellen

```
mkdir /unvr
```

Nun die part UUID auselsen

```
blkid /dev/sdb1
/dev/sdb1: UUID="a396f93a-305e-4412-9040-5c51c9203f78" BLOCK_SIZE="4096" TYPE="ext4"
PARTLABEL="Linux filesystem"
PARTUUID="c9c72c32-250f-4b08-b74f-0aadc682d313"
```

Nun haben wir die UUID. Mit dieser serstellen wir einen automount in der fstab.

```
nano /etc/fstab
```

Dort tragen wir unsere UUID ein dann das Mountverzeichnis das Dateisystem

```
...
UUID=a396f93a-305e-4412-9040-5c51c9203f78  /unvr      ext4    defaults,errors=remount-ro 0       1
...
```

testen des mount points.
Wenns geklappt hat können wir mit

```
mount /unvr
```

die Festplatte mounten.
Das Ergebnis ob gemountet sehen wir durch lsblk

```
lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda      8:0    0    32G  0 disk
├─sda1   8:1    0   512M  0 part /boot/efi
├─sda2   8:2    0  30.5G  0 part /
└─sda3   8:3    0   976M  0 part [SWAP]
sdb      8:16   0   256G  0 disk
└─sdb1   8:17   0   256G  0 part /unvr
sr0     11:0    1 329.3M  0 rom
```

Die Grub config um diesen Eintrag erweitern.

```
/etc/default/grub
```

Inhalt:

```
GRUB_CMDLINE_LINUX_DEFAULT="quiet systemd.unified_cgroup_hierarchy=0"


Sollten schon andere parameter drin stehen diese logischerweise hinzufügen und nicht ersetzten...
```

nun update grub ausführen

```
update-grub
```

Neustarten. Achtung bei einem neustart könnte die sda / sdb reihenfolge anders sein.

```
reboot
```

Erstellen der Docker instanz durch pullen von dockerhub

Folgendes script laufen lassen, die Pfade sind angepasst auf /unvr
Soll der conateiner automatisch gestartet werden an den Befehl noch

```
--restart always hinter -d
dranhängen
```

Hiernochmals darauf achten das bei STOARGE_DISK=/dev/sdb1 oder sdb2 die richtige partion angegeben wird. Denn nach dem neustart können diese sich ändern

```
docker run -d --name unifi-protect  \
    --privileged \
    --tmpfs /run \
    --tmpfs /run/lock \
    --tmpfs /tmp \
    -v /sys/fs/cgroup:/sys/fs/cgroup:ro \
    -v /unvr/srv:/srv \
    -v /unvr/data:/data \
    -v /unvr/persistent:/persistent \
    --network host \
    -e STORAGE_DISK=/dev/sda1 \
    -p 443:443 \
    -p 80:80 \
    -p 7446:7446 \
    -p 7443:7443 \
    -p 8443:8443 \
    markdegroot/unifi-protect-arm64
```

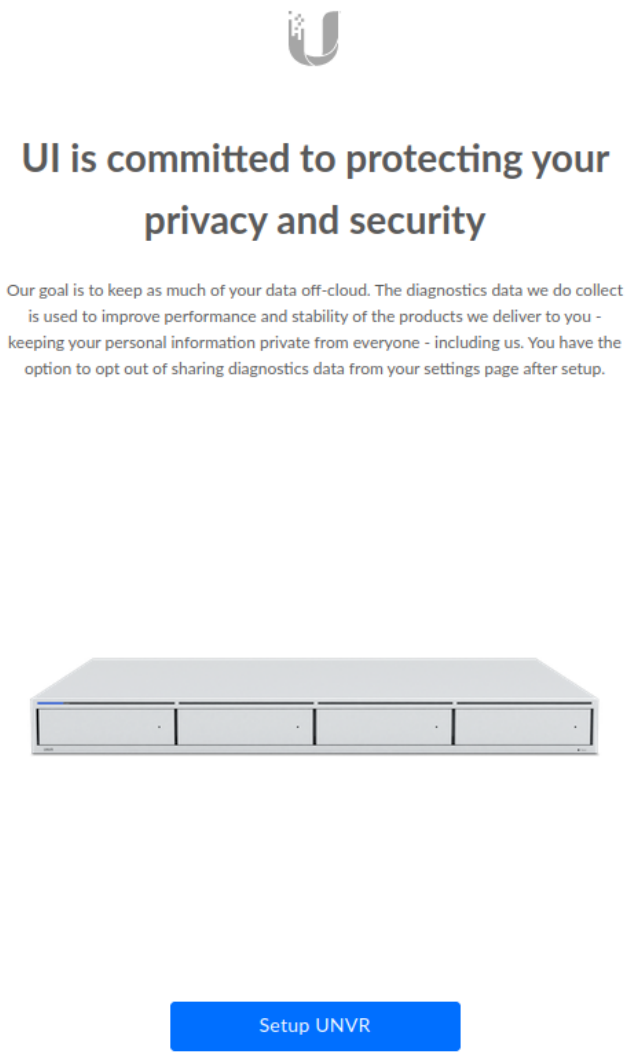
Mit Autostart Flag

```
docker run -d --restart always --name unifi-protect  \
    --privileged \
    --tmpfs /run \
    --tmpfs /run/lock \
    --tmpfs /tmp \
    -v /sys/fs/cgroup:/sys/fs/cgroup:ro \
    -v /unvr/srv:/srv \
    -v /unvr/data:/data \
    -v /unvr/persistent:/persistent \
    --network host \
    -e STORAGE_DISK=/dev/sda1 \
    -p 443:443 \
```

```
-p 80:80 \
-p 7446:7446 \
-p 7443:7443 \
-p 8443:8443 \
markdegroot/unifi-protect-arm64
```

Nun 9-10 Minuten warten

Dann unter https://ip:443 aufrufbar



Nun sind wir fertig.

Sollte Unifi Prtotect noch ewig beim Status starten stehen.
Dann einloggen mittels

```
docker exec -it unifi-protect /bin/bash
```

Nun den unifi-core neustarten

```
systemctl restart unifi-core
```

Fehler:

Failed to create /init.scope control group:

Wenn wir folgende Fehler bekommenr (or any systemd error):

```
Failed to create /init.scope control group: Read-only file system
Failed to allocate manager object: Read-only file system
[!!!!!!] Failed to allocate manager object.
Exiting PID 1...
```

Die Grub config um diesen Eintrag erweitern.

```
/etc/default/grub
```

Inhalt:

```
GRUB_CMDLINE_LINUX_DEFAULT="quiet systemd.unified_cgroup_hierarchy=0"


Sollten schon andere parameter drin stehen diese logischerweise hinzufügen und nicht ersetzten...
```

nun update grub ausführen

```
update-grub
```

Certiface not Found

```
journalctl -u unifi-protect
```

Ausgabe:

```
[Error: ENOENT: no such file or directory, open '/data/unifi-core/config/unifi-core.crt'] {
Jul 07 10:18:36 unvr node[220735]:   errno: -2,
Jul 07 10:18:36 unvr node[220735]:   code: 'ENOENT',
```

```
Jul 07 10:18:36 unvr node[220735]:   syscall: 'open',
Jul 07 10:18:36 unvr node[220735]:   path: '/data/unifi-core/config/unifi-core.crt'
Jul 07 10:18:36 unvr node[220735]: }
Jul 07 10:18:36 unvr node[220735]: Unifi core certificate not found
```

Werkseinstellungen - UNVR

Um auf Werkseinstellungen zurückzusetzten:

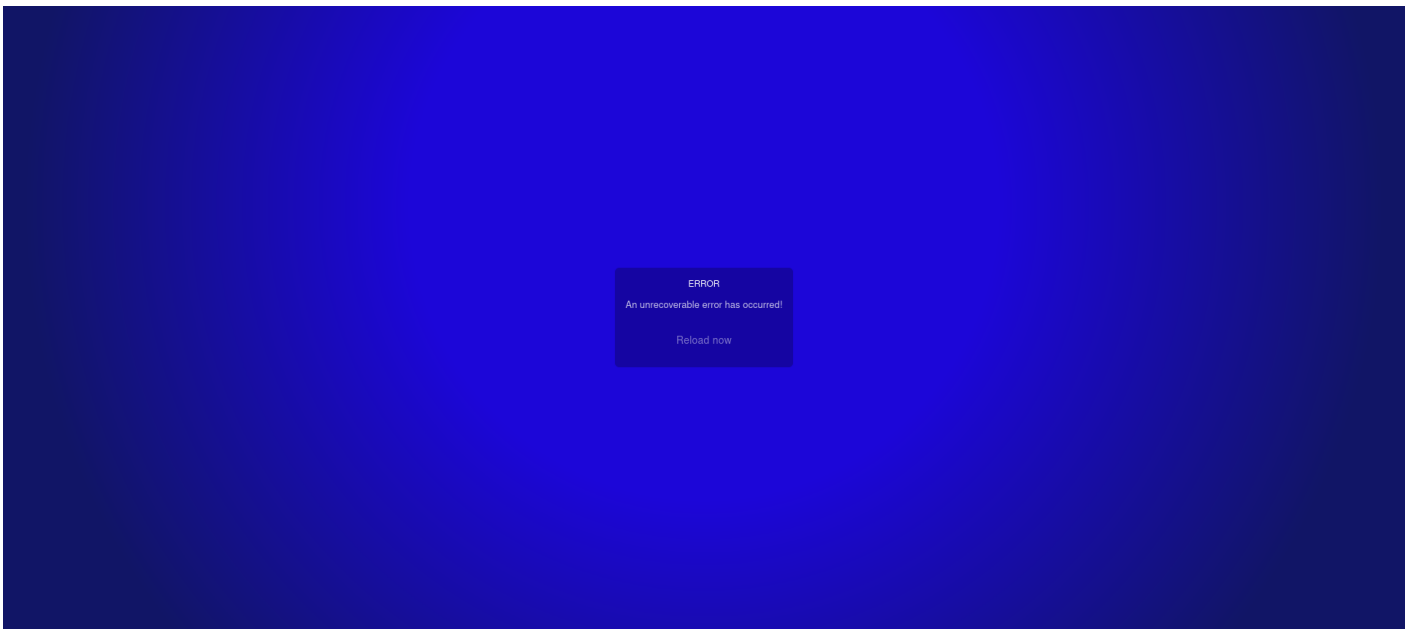
1. Reset Knopf 10 -15 Sekunden gedrückt halten
2. Oder per ssh einloggen, standard credentials
   user : ubnt
   Pass : ubnt
   Dann

   ```
   ubnt-systool reset2defaults
   ```

   ausführen um Werkseinstellungen durchzuführen

Unifi-Protect - Fehlerseite an Error occoured

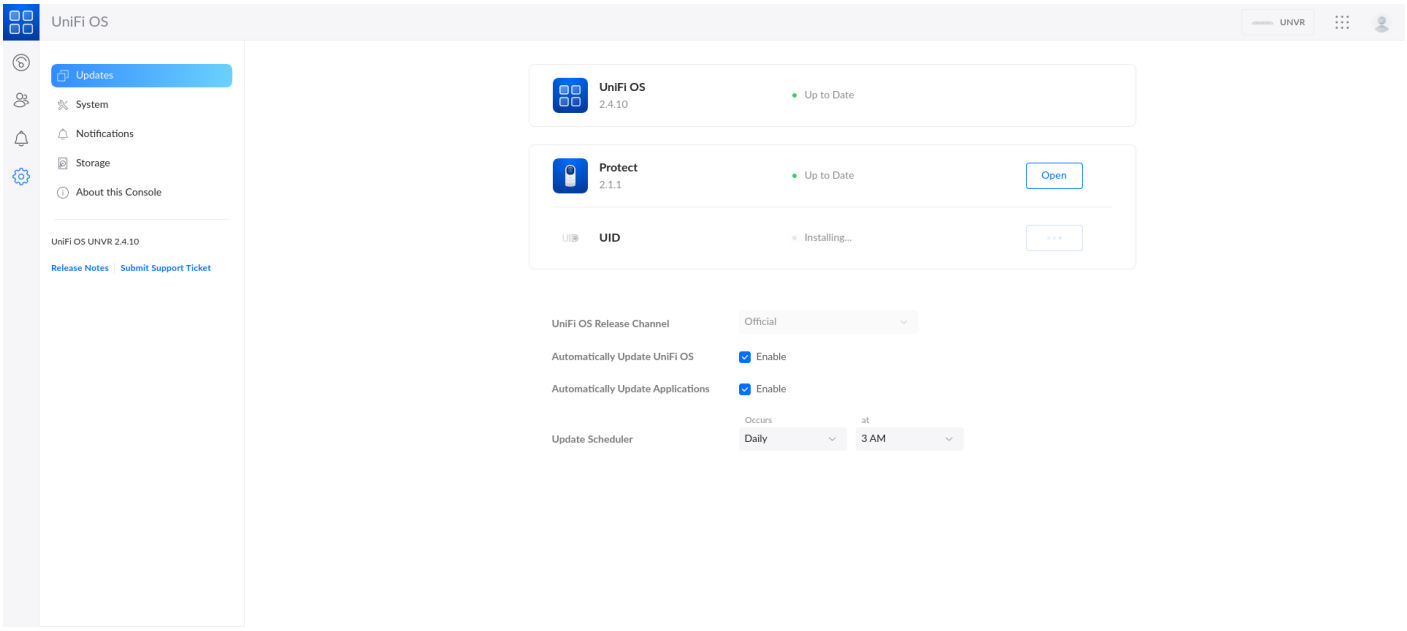
Feherwenn die Seite aufgerufen wird über https://ip:7443



Ab Fimrware version 2.0 wird das Dashboard über https://ip/protect aufgerufen
Ist die Firmware version niedriger dann liegt es warscheinlich an den nächsten schritten:

Ist das Gerät neu und noch nicht eingerichtet, ist das der falsche Port.

Über https://ip:443 wird das Unifi OS eingerichtet dazu, das Unifi protect und UID.
Zu finden unter Settings -> Updates



Im journalctl -u unifi-protect.service
Sieht das ganze dann so aus

```
journalctl -u unifi-protect.service
-- Logs begin at Fri 2022-07-22 15:37:25 CEST, end at Fri 2022-07-22 15:48:32 CEST. --
Jul 22 15:39:27 UniFi-NVR systemd[1]: Starting UniFi Protect...
Jul 22 15:39:27 UniFi-NVR su[3483]: Successful su for postgres by root
Jul 22 15:39:27 UniFi-NVR su[3483]: + ??? root:postgres
Jul 22 15:39:27 UniFi-NVR su[3483]: pam_unix(su:session): session opened for user postgres by (uid=0)
Jul 22 15:39:28 UniFi-NVR su[3483]: pam_unix(su:session): session closed for user postgres
Jul 22 15:39:33 UniFi-NVR node12[3547]: CONFIG LOADED : /usr/share/unifi-protect/app/config/config.json
Jul 22 15:39:33 UniFi-NVR node12[3547]: Certificate /data/unifi-protect/data/unifi-protect.crt not found....
generating
Jul 22 15:39:33 UniFi-NVR node12[3547]: Using unifi core certificate
Jul 22 15:39:33 UniFi-NVR node12[3547]: {
Jul 22 15:39:33 UniFi-NVR node12[3547]:   crt: '/data/unifi-core/config/unifi-core.crt',
Jul 22 15:39:33 UniFi-NVR node12[3547]:   key: '/data/unifi-core/config/unifi-core.key',
```

```
Jul 22 15:39:33 UniFi-NVR node12[3547]:   deviceCrt: '/data/unifi-protect/data/devices.crt',
Jul 22 15:39:33 UniFi-NVR node12[3547]:   deviceKey: '/data/unifi-protect/data/devices.key'
Jul 22 15:39:33 UniFi-NVR node12[3547]: }
Jul 22 15:39:34 UniFi-NVR node12[3547]: Device certificate /data/unifi-protect/data/devices.crt not found.... generating
Jul 22 15:40:28 UniFi-NVR node12[3547]: Fri, 22 Jul 2022 13:40:28 GMT sequelize deprecated String based operators are now deprecated. Please use Symbol based operators for better security, read more at http://docs.sequelizejs.com/manual/t
Jul 22 15:40:42 UniFi-NVR sudo[4674]: unifi-protect : TTY=unknown ; PWD=/usr/share/unifi-protect ; USER=root ; COMMAND=/sbin/ubnt-tools id
Jul 22 15:40:42 UniFi-NVR sudo[4674]: pam_unix(sudo:session): session opened for user root by (uid=0)
Jul 22 15:40:42 UniFi-NVR sudo[4674]: pam_unix(sudo:session): session closed for user root
Jul 22 15:40:42 UniFi-NVR node12[3547]: (node:3547) MaxListenersExceededWarning: Possible EventEmitter memory leak detected. 11 save listeners added to [DecalObject]. Use emitter.setMaxListeners() to increase limit
Jul 22 15:40:42 UniFi-NVR systemd[1]: Started UniFi Protect.
Jul 22 15:40:47 UniFi-NVR node12[3547]: (node:3547) [DEP0005] DeprecationWarning: Buffer() is deprecated due to security and usability issues. Please use the Buffer.alloc(), Buffer.allocUnsafe(), or Buffer.from() methods instead.
root@UniFi-NVR:~#
```

Dienst stoppen

```
service unifi-protect stop
```

Zum benutzer postgre wechseln

```
su - postgres
```

In die Datenbank einloggen

```
psql -p 5433
```

Auflisten ob es den Benutzer unifi-protect gibt

```
\du
```

Ausgabe:

```
postgres=# \du
                                   List of roles
  Role name   |                         Attributes                         | Member of
--------------+------------------------------------------------------------+-----------
 postgres     | Superuser, Create role, Create DB, Replication, Bypass RLS | {}
 unifi-protect | Create DB                                                  | {}
```

Datenbank löschen

```
drop database "unifi-protect";
```

Und wenn es den Benutzer gab, den Benutzer löschen

```
drop user "unifi-protect";
```

Die SQL Console verlassen

```
\q
```

Als Benutzer postgres wieder abmelden

```
exit
```

Unifi protect wieder starten

```
service unifi-protect start
```

Unifi Protect im Docker Container in einer amd64 Debian VM installieren mit qemu arm64

Beschreibung:

Unifi UNVR im Docker Conatiner auf einem x86/x864 Host. Wir können einmal das Image aus dem Docker Hub Pullen oder uns auch ein eigenes Image bauen. Erstmal die Variante mit dem Pullen.

Vorraussetzung:

Einmal ein x86/x64 Host in dem Docker installiert ist.
Eine zweite HDD mindesten 128 GB groß. würde aber eine Größe wählen die größer ist. Unbedingt eine Partioion und ext4 so kann sie on the fly vergrößert werden.

Grundinstallation Docker und Einbungung Festplatte

Installation Docker, als root per ssh einloggen

```
apt install docker.io gdisk curl
#Installieren der quemu erweiterungen für arm64
apt-get install qemu qemu-system-arm binfmt-support qemu-user-static # Install the qemu packages
docker run --rm --privileged multiarch/qemu-user-static --reset -p yes # This step will execute the registering scripts
```

Nun eine weitere Festplatte hinzufügen und diese Partitionieren wenn nicht schon geschehen
Mit lsbk die neue Festplatte ermitteln

```
oot@unnvr:~# lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda      8:0    0    32G  0 disk
├─sda1   8:1    0   512M  0 part /boot/efi
├─sda2   8:2    0  30.5G  0 part /
└─sda3   8:3    0   976M  0 part [SWAP]
sdb      8:16   0   256G  0 disk
sr0     11:0    1 329.3M  0 rom
```

sdb ist die gewünschte Festplatte

nun partitonieren

```
gdisk /dev/sdb

dann o eingeben für neues layout
dann n eingeben für eine neue partition

alles durchentern

nun w für schreiben
und exit
```

Nun Dateisystem ext4 erzeugen auf sdb1

```
mkfs.ext4 /dev/sdb1
```

Nun ein verzeichnis für den mount point erstellen

```
mkdir /unvr
```

Nun die part UUID auselsen

```
blkid /dev/sdb1
/dev/sdb1: UUID="a396f93a-305e-4412-9040-5c51c9203f78" BLOCK_SIZE="4096" TYPE="ext4"
PARTLABEL="Linux filesystem"
PARTUUID="c9c72c32-250f-4b08-b74f-0aadc682d313"
```

Nun haben wir die UUID. Mit dieser serstellen wir einen automount in der fstab.

```
nano /etc/fstab
```

Dort tragen wir unsere UUID ein dann das Mountverzeichnis das Dateisystem

```
...
UUID=a396f93a-305e-4412-9040-5c51c9203f78  /unvr      ext4    defaults,errors=remount-ro 0       1
...
```

testen des mount points.
Wenns geklappt hat können wir mit

```
mount /unvr
```

die Festplatte mounten.
Das Ergebnis ob gemountet sehen wir durch lsblk

```
lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda      8:0    0    32G  0 disk
├─sda1   8:1    0   512M  0 part /boot/efi
├─sda2   8:2    0  30.5G  0 part /
└─sda3   8:3    0   976M  0 part [SWAP]
sdb      8:16   0   256G  0 disk
└─sdb1   8:17   0   256G  0 part /unvr
sr0     11:0    1 329.3M  0 rom
```

Die Grub config um diesen Eintrag erweitern.

```
nano /etc/default/grub
```

Inhalt:

```
GRUB_CMDLINE_LINUX_DEFAULT="quiet systemd.unified_cgroup_hierarchy=0"


Sollten schon andere parameter drin stehen diese logischerweise hinzufügen und nicht ersetzten...
```

nun update grub ausführen

```
update-grub
```

Neustarten. Achtung bei einem neustart könnte die sda / sdb reihenfolge anders sein.

```
reboot
```

Netzwerkkarten Namen anpassen sonst kann man sich nicht mit einem Cloud Knto von Unifi anmelden. Der Anme muss enp0s2 lauten

```
nano /etc/systemd/network/99-enp0s2.link
```

Die MACAdress mit der MAc Adresse unserer Netzwerkkarte austauschen.
Diese bekommt man mit

```
ip a
```

raus

Inhalt:

```
[Match]
MACAddress=xx:xx:xx:xx:xx:xx

[Link]
Name=enp0s2

[Network]
DHCP=yes
```

nun

```
update-initramfs -u
```

ausführen und neustarten.

Erstellen der Docker instanz durch pullen von dockerhub

Folgendes script laufen lassen, die Pfade sind angepasst auf /unvr
Soll der conateiner automatisch gestartet werden an den Befehl noch

```
--restart always hinter -d
dranhängen
```

Hiernochmals darauf achten das bei STOARGE_DISK=/dev/sdb1 oder sdb2 die richtige partion angegeben wird. Denn nach dem neustart können diese sich ändern

```
docker run -d --name unifi-protect  \
    --privileged \
    --tmpfs /run \
    --tmpfs /run/lock \
    --tmpfs /tmp \
    -v /sys/fs/cgroup:/sys/fs/cgroup:ro \
    -v /unvr/srv:/srv \
    -v /unvr/data:/data \
    -v /unvr/persistent:/persistent \
    --network host \
    -e STORAGE_DISK=/dev/sda1 \
    -p 443:443 \
    -p 80:80 \
    -p 7446:7446 \
    -p 7443:7443 \
    -p 8443:8443 \
    snowsnoot/unifi-unvr:latest
```

Mit Autostart Flag

```
docker run -d --restart always --name unifi-protect  \
    --privileged \
    --tmpfs /run \
    --tmpfs /run/lock \
    --tmpfs /tmp \
    -v /sys/fs/cgroup:/sys/fs/cgroup:ro \
    -v /unvr/srv:/srv \
    -v /unvr/data:/data \
    -v /unvr/persistent:/persistent \
    --network host \
    -e STORAGE_DISK=/dev/sda1 \
    -p 443:443 \
    -p 80:80 \
    -p 7446:7446 \
```

```
    -p 7443:7443 \
    -p 8443:8443 \
    snowsnoot/unifi-unvr:latest
```

Nun 9-15 Minuten warten

mit folgendem Befehl kann man sich in den Container einloggen und zum beispiel den service status sich anzuschauen.

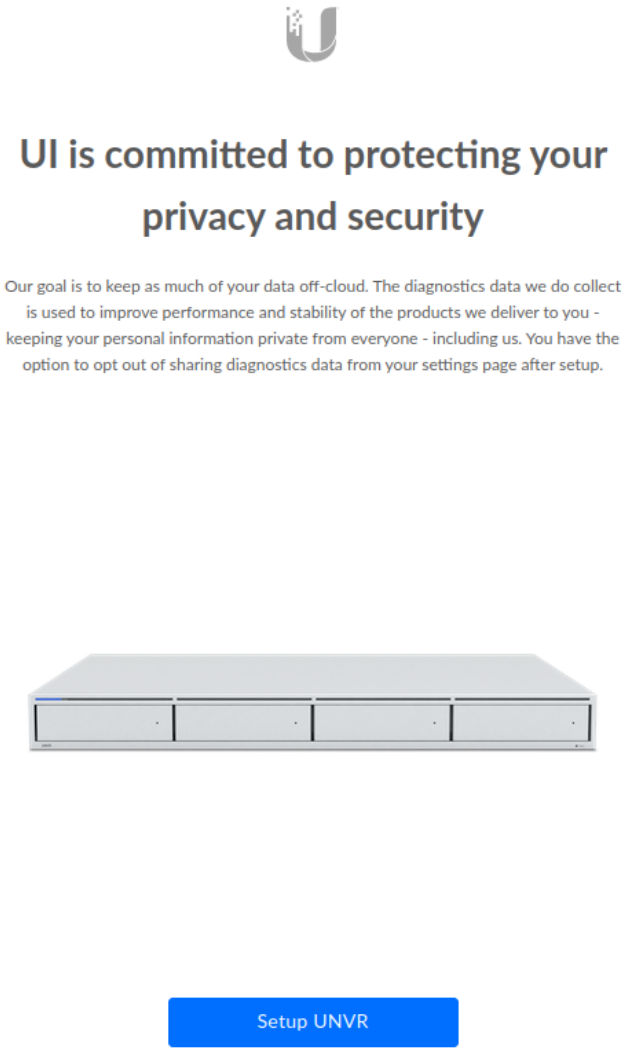
```
docker exec -it unifi-protect bash
```

Nun

```
service unifi-protect status



oder



journalctl -u unifi-protect
```

Nach 9-15 Minuten sollte die Website aufrufbar sein

Dann unter https://ip:443 aufrufbar



Nun sind wir fertig.

Sollte Unifi Prtotect noch ewig beim Status starten stehen.
Dann einloggen mittels

```
docker exec -it unifi-protect /bin/bash
```

Nun den unifi-core neustarten

```
systemctl restart unifi-core
```

Fehler:

Failed to create /init.scope control group:

Wenn wir folgende Fehler bekommenr (or any systemd error):

```
Failed to create /init.scope control group: Read-only file system
Failed to allocate manager object: Read-only file system
[!!!!!!] Failed to allocate manager object.
Exiting PID 1...
```

Die Grub config um diesen Eintrag erweitern.

```
/etc/default/grub
```

Inhalt:

```
GRUB_CMDLINE_LINUX_DEFAULT="quiet systemd.unified_cgroup_hierarchy=0"


Sollten schon andere parameter drin stehen diese logischerweise hinzufügen und nicht ersetzten...
```

nun update grub ausführen

```
update-grub
```

Certiface not Found

```
journalctl -u unifi-protect
```

Ausgabe:

```
[Error: ENOENT: no such file or directory, open '/data/unifi-core/config/unifi-core.crt'] {
Jul 07 10:18:36 unvr node[220735]:   errno: -2,
Jul 07 10:18:36 unvr node[220735]:   code: 'ENOENT',
Jul 07 10:18:36 unvr node[220735]:   syscall: 'open',
Jul 07 10:18:36 unvr node[220735]:   path: '/data/unifi-core/config/unifi-core.crt'
Jul 07 10:18:36 unvr node[220735]: }
Jul 07 10:18:36 unvr node[220735]: Unifi core certificate not found
```

Unifi G4 Pro (Standalone Mode) Zeitstempel ändern

Beschreibung:

Wird eine Unifi Kamera in Standolane betrieben, kann man das Format des Zeitstempels nicht ändern, zumindest nicht über die GUI.

Hier eine Anleitung wie wir dieses per SSH ändern können.

Abänderung:

Per ssh auf die Kamera einloggen. Standard
user: ubnt
pass : ubnt

```
ssh ubnt@192.168.178.68
The authenticity of host '192.168.178.68 (192.168.178.68)' can't be established.
ECDSA key fingerprint is SHA256:woZlLB1CI9VRk7g9+AqFQ+lzeil1N9Rvgdnct2Hj6r0.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.178.68' (ECDSA) to the list of known hosts.
ubnt@192.168.178.68's password:
```

Nach dem wir eingeloggt sind, auf das wurzelverzeichnis / wechseln und nach der config siuchen. Ist jedesmal nach firmware wieder anders.

```
cd /
find -name config
```

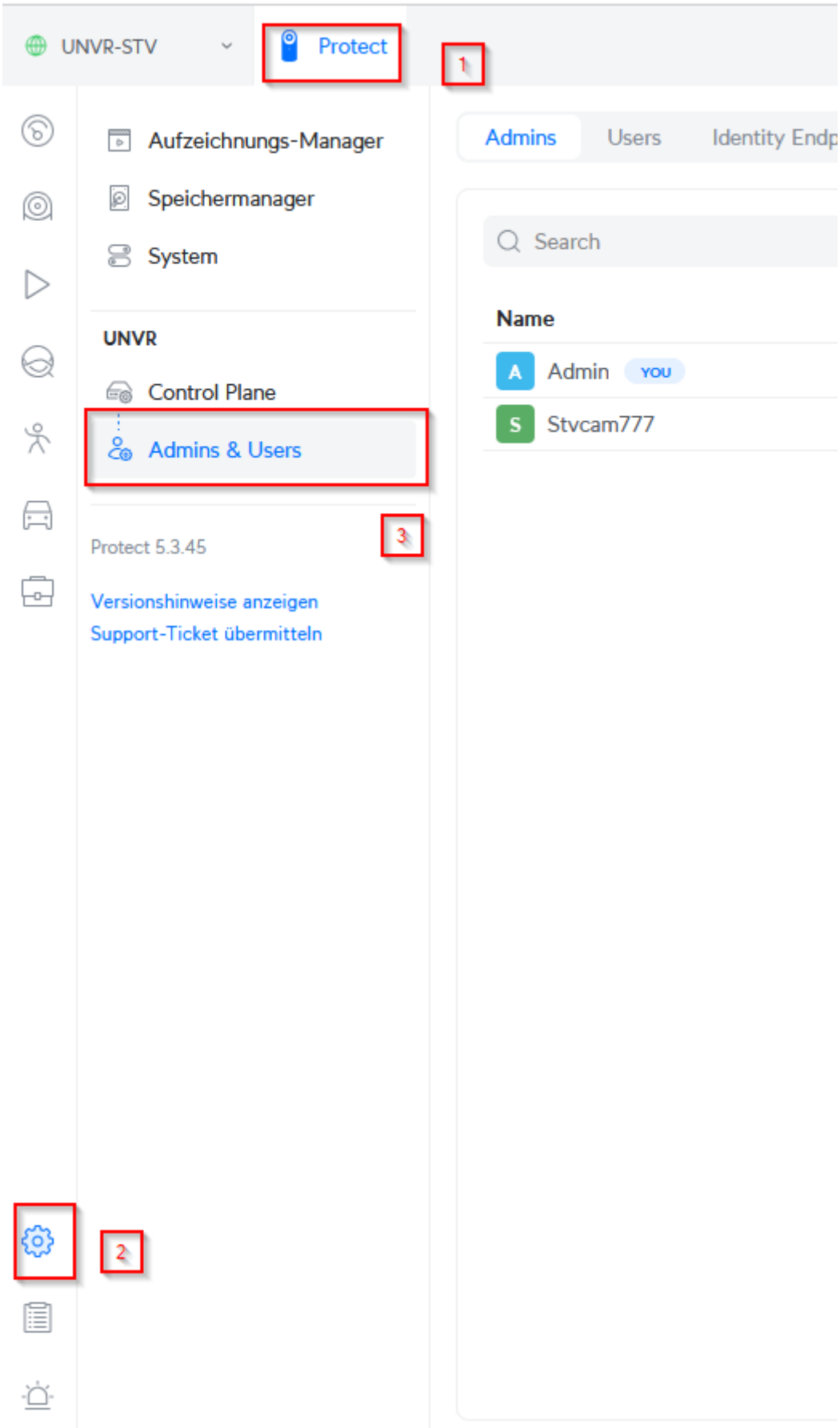
Gruppen

Beschreibung:

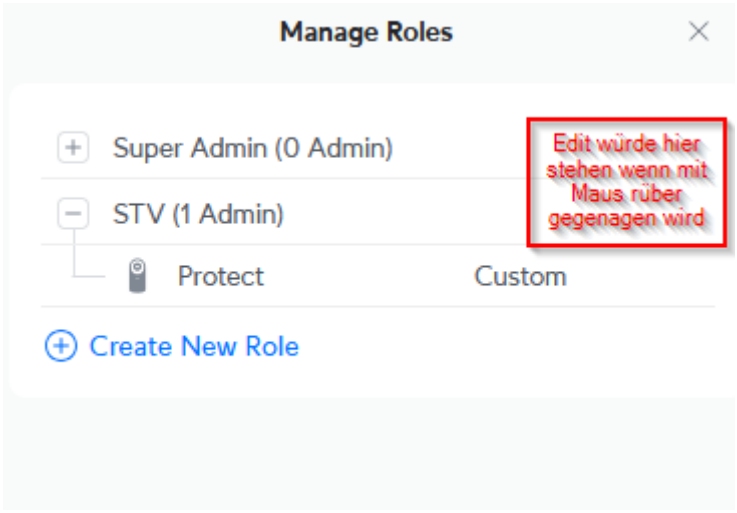
Um Verschiedenen Benutzern verschiedene Kameras zuzuweisen müssen unter Protect in Admin und Roles Gruppen definiert werden.
Diesen gruppen können dann Kameras hinzugefügt werden

Einrichtung:

Auf Protect -> Dann unten das Zahnrad für Einstellungen -> Admin & Users



Nun Auf Manage Roles klicken um vorhandene Rollen zu editieren


Manage Roles
Super Admin (0 Admin)
STV (1 Admin)
Protect
Custom
Create New Role
Edit würde hier stehen wenn mit Maus rüber gegenagen wird