# Elbrus Analytics - Bereitstellungshandbuch

Tobias Schmidt

July 25, 2022

# 1 Server Infrastruktur

## 1.1 SSH-Zugriff vorbereiten

Listing 1: Updaten vorhandener Packages.

```
root@server:~$ yum update
```

Listing 2: Installieren des 'ssh' Packages.

```
root@server:~$ yum install -y openssh-server
```

Listing 3: Starten des 'sshd' Services.

```
root@server:~$ systemctl start sshd
```

Listing 4: Aktivieren des 'sshd' Services.

```
root@server:~$ systemctl enable sshd
```

Listing 5: Aktivieren des 'sshd' Services.

```
root@server:~$ systemctl enable sshd
```

Listing 6: Anlegen des Users Elbrus.

```
root@server:~$ useradd elbrus
```

Listing 7: Hinzufügen des Users Elbrus zu der Gruppe 'wheel'.

```
root@server:~$ usermod -aG wheel elbrus
Changing password for user elbrus.
New password:
Retype new password:
passwd: all authentication tokens updated successfully
root@server:~$
```

## 1.2 Initiale Server Konfiguration

Listing 8: Setzen der Zeitzone auf 'Europa/Wien'.

```
elbrus@server:~$ sudo timedatectl set-timezone Europe/Vienna
```

Listing 9: Installieren von dem 'firewalld' Service.

```
elbrus@server:~$ sudo dnf install firewalld
```

### 1.2.1 Node.Js

Listing 10: Installieren des Framworks 'Node.Js'.

```
elbrus@server:~$ sudo dnf -y module install nodejs:12
```

### 1.2.2 Ablagestruktur

Listing 11: Anlegen der Verzeichnissstruktur.

```
elbrus@server:~$ sudo mkdir /var/elbrus
elbrus@server:~$ chown -R elbrus:elbrus /var/elbrus
elbrus@server:~$ cd /var/elbrus
elbrus@server:~/var/elbrus$ mkdir shared keys
elbrus@server:~/var/elbrus$ chmod -R 770 /var/elbrus/shared
```

## 1.3  Git

Listing 12: Installieren von dem VCS 'git'.

```
elbrus@server:~$ sudo yum install -y git
```

### 1.3.1  Git - Erstellen der SSH-Keys

Listing 13: Wechseln des Verzeichnisses.

```
elbrus@server:~$ cd /var/elbrus/keys
```

Listing 14: Erstellen des SSH-keys der für das Herunterladen der 'Database' benötigt wird.

```
elbrus@server:~/var/elbrus/keys$ ssh-keygen -t rsa -b 2048 -f database_key -q -N ""
```

Listing 15: Erstellen des SSH-keys der für das Herunterladen des 'Capture-Device' benötigt wird.

```
elbrus@server:~/var/elbrus/keys$ ssh-keygen -t rsa -b 2048 -f capture_device_key -q -N ""
```

Listing 16: Erstellen des SSH-keys der für das Herunterladen des 'Report-Generator' benötigt wird.

```
elbrus@server:~/var/elbrus/keys$ ssh-keygen -t rsa -b 2048 -f\
report_generator_key -q -N ""
```

Listing 17: Erstellen des SSH-keys der für das Herunterladen des 'SNMP-Managers' benötigt wird.

```
elbrus@server:~/var/elbrus/keys$ ssh-keygen -t rsa -b 2048 -f snmp_manager_key -q -N ""
```

Listing 18: Erstellen des SSH-keys der für das Herunterladen des 'SSH-Managers' benötigt wird.

```
elbrus@server:~/var/elbrus/keys$ ssh-keygen -t rsa -b 2048 -f ssh_manager_key -q -N ""
```

Listing 19: Erstellen des SSH-keys der für das Herunterladen der 'API' benötigt
wird.

```
elbrus@server:~/var/elbrus/keys$ ssh-keygen -t rsa -b 2048 -f api_key -q -N ""
```

Listing 20: Erstellen des SSH-keys der für das Herunterladen des 'Webinterfaces'
benötigt wird.

```
elbrus@server:~/var/elbrus/keys$ ssh-keygen -t rsa -b 2048 -f webinterface_key -q -N ""
```

Bevor mit der Installation vorgefahren werden kann müssen die soeben an-
gelegten SSH-Keys an "keys@Elbrus-Analytics.at" gesendet werden. Bitte beachten
Sie, dass Ihre Email-Adresse Sie als berechtigten Nutzer ausweist.

### 1.3.2   Git - Clonen der Software

Listing 21: Wechseln des Verzeichnisses.

```
elbrus@server:~$ cd /var/elbrus
```

Listing 22: Clonen der Datenbank Software.

```
elbrus@server:~/var/elbrus$ git clone git@github.com:Elbrus-Analytics/database.git\
--config core.sshCommand="ssh -i ./keys/database_key"
```

Listing 23: Clonen der Capture-Device Software.

```
elbrus@server:~/var/elbrus$ git@github.com:Elbrus-Analytics/capture-device.git\
--config core.sshCommand="ssh -i ./keys/capture_device_key"
elbrus@server:~/var/elbrus$ mkdir capture
elbrus@server:~/var/elbrus$ cp -a capture-device/src/. capture
elbrus@server:~/var/elbrus$ rm -rfd capture-device
```

Listing 24: Clonen der Report-Generator Software.

```
elbrus@server:~/var/elbrus$ git clone git@github.com:Elbrus-Analytics/\
report-generator.git --config core.sshCommand="ssh -i ./keys/report_generator_key"
```

Listing 25: Clonen der SNMP-Manager Software.

```
elbrus@server:~/var/elbrus$ git@github.com:Elbrus-Analytics/snmp-manager.git\
--config core.sshCommand="ssh -i ./keys/snmp-manager_key"
```

Listing 26: Clonen der SSH-Manager Software.

```
elbrus@server:~/var/elbrus$ git@github.com:Elbrus-Analytics/ssh-manager.git\
--config core.sshCommand="ssh -i ./keys/ssh-manager_key"
```

Listing 27: Clonen der API Software.

```
elbrus@server:~/var/elbrus$ git@github.com:Elbrus-Analytics/api.git\
--config core.sshCommand="ssh -i ./keys/api_key"
```

Listing 28: Clonen der Packet-Importer Software

```
elbrus@server:~/var/elbrus$ git@github.com:Elbrus-Analytics/webinterface.git\
--config core.sshCommand="ssh -i ./keys/webinterface_key"
```

## 1.4 Python

### 1.4.1 1 - Automatische Installation

Listing 29: Kopieren des Github Repositorys 'report-generator'.

```
elbrus@server:~$ cd /var/elbrus
elbrus@server:~/var/elbrus$ git clone https://github.com/\
Elbrus-Analytics/report-generator.git
```

Listing 30: Ausführen des 'pythonSourceInstall.sh' Scripts.

```
elbrus@server:~/var/elbrus$ bash report-generator/pythonSourceInstall.sh
```

### 1.4.2 2 - Manuel Installation

Listing 31: Installieren von benötigten Packeten und Abhängigkeiten.

```
elbrus@server:~$ sudo dnf install gcc openssl-devel bzip2-devel\
libffi-devel zlib-devel wget make -y
```

Listing 32: Extrahieren der installierent Dateien.

```
elbrus@server:~$ tar -xf Python-3.10.2.tar.xz
```

Listing 33: Wechseln zu source Verzeichniss. Und ausführen des Konfigurations Scripts.

```
elbrus@server:~$ cd Python-3.10.0 && ./configure --enable-optimizations
```

Listing 34: Starten des build Prozesses.

```
elbrus@server:~Python-3.10.0$ cd make -j $(nproc)
```

Listing 35: Installieren von Python.

```
elbrus@server:~Python-3.10.0$ sudo make install
```

### 1.4.3 Upgrade von 'pip'

Listing 36: Upgraden von 'pip'.

```
elbrus@server:~$ /usr/local/bin/python3.10 -m pip install --upgrade pip
```

## 1.5 Rust

Listing 37: Installieren von GNU Compiler Collection.

```
elbrus@server:~$ sudo dnf install gcc -y
```

Listing 38: Installieren von Rust.

```
elbrus@server:~$ curl --proto '=https' --tlsv1.2 -sSf \
https://sh.rustup.rs/ | sh

...

default host triple: x86_64-unknown-linux-gnu
default toolchain: stable (default)
profile: default
modify PATH variable: yes

1) Proceed with installation (default)
2) Customize installation
3) Cancel installation
>1

...

stable-x86_64-unknown-linux-gnu installed - rustc 1.62.1 (e092d0b6b 2022-07-16)


Rust is installed now. Great!

To get started you may need to restart your current shell.
This would reload your PATH environment variable to include
Cargo's bin directory ($HOME/.cargo/bin).

To configure your current shell, run:
source "$HOME/.cargo/env"
elbrus@server:~$
```

Listing 39: Laden der Variablen aus dem Terminal Profil.

```
elbrus@server:~$ source ~/.profile
```

Listing 40: Hinzufügen des Befehls Cargo zu dem Pfad.

```
elbrus@server:~$ source ~/.cargo/env
```

## 1.6 SSH-Keys

Weil der pcap-importer und der report-generator auf zwei verschiedenen Server liegen könnten, muss für die Kommunikation zwischen jenen Sever SSH-Funktionieren.

Dieser Schritt kann übersprungen werden wenn alles auf einem Server installiert wird.

### 1.6.1 Capture-Server

Listing 41: Anlegen der SSH-Keys.

```
elbrus@server:~$ mkdir -p /var/elbrus/shared/.ssh/
elbrus@server:~$ ssh-keygen -t ecdsa -b 256 -f \
/var/elbrus/shared/.ssh/id_report_generator_connection -N ''
```

Listing 42: Übertragen der SSH-Keys auf den Database-Server.

```
elbrus@server:~$ ssh-copy-id -i \
/var/elbrus/shared/.ssh/id_report_generator_connection.pub \
elbrus@10.0.76.220
```

### 1.6.2 Database-Server

Listing 43: Anlegen der SSH-Keys.

```
elbrus@server:~$ mkdir -p /var/elbrus/shared/.ssh/
elbrus@server:~$ ssh-keygen -t ecdsa -b 256 -f \
/var/elbrus/shared/.ssh/id_capture_connection -N ''
```

Listing 44: Übertragen der SSH-Keys auf den Capture-Server.

```
elbrus@server:~$ ssh-copy-id -i \
/var/elbrus/shared/.ssh/id_capture_connection.pub \
elbrus@10.0.76.217
```

# 2 Datenbank

## 2.1 Voraussetzungen

Listing 45: Hinzufügen des PostgreSQL Drittanbieter-Repository, um die neuesten PostgreSQL-Pakete zu erhalten.

```
elbrus@server:~$ sudo yum install\
https://download.postgresql.org/pub/repos/yum/reporpms/\
EL-$(rpm -E %{rhel})-x86_64/pgdg-redhat-repo-latest.noarch.rpm
```

Listing 46: Erstellen des Timescale repository.

```
elbrus@server:~$ sudo tee /etc/yum.repos.d/timescale_timescaledb.repo <<EOL
[timescale_timescaledb]
name=timescale_timescaledb
baseurl=https://packagecloud.io/timescale/timescaledb/el/$(rpm -E %{rhel})/\$basearch
repo_gpgcheck=1
gpgcheck=0
enabled=1
gpgkey=https://packagecloud.io/timescale/timescaledb/gpgkey
sslverify=1
sslcacert=/etc/pki/tls/certs/ca-bundle.crt
metadata_expire=300
EOL
```

Listing 47: Updaten der lokalen Package-Liste.

```
elbrus@server:~$ sudo yum update
```

Listing 48: Installieren von TimescaleDB.

```
elbrus@server:~$ sudo dnf -qy module disable postgresql
elbrus@server:~$ sudo dnf install postgresql14 postgresql14-server -y
elbrus@server:~$ sudo dnf install timescaledb-2-postgresql-14 -y
```

## 2.2 Umgebung Konfigurieren

Listing 49: Initialisieren der Datenbank.

```
elbrus@server:~$ sudo /usr/pgsql-14/bin/postgresql-14-setup initdb
```

Listing 50: Verknüpfen von 'postgresql' Serive Start mit Serverstart sowie den Service starten.

```
elbrus@server:~$ sudo systemctl enable postgresql-14
elbrus@server:~$ sudo systemctl start postgresql-14
```

Listing 51: var/lib/pgsql/14/data/postgresql.conf - Ändern der folgenden Zeilen.

```
- #shared_preload_libraries = ''
+ shared_preload_libraries = 'timescaledb'

- #listen_addresses = 'localhost'
+ listen_addresses = '*'
```

Listing 52: var/lib/pgsql/14/data/ph_hba.conf - Ändern der folgenden Zeilen.

```
# TYPE      DATABASE      USER      ADDRESS      METHOD
+ host        elbrus        elbrus      0.0.0.0/0      trust
```

Listing 53: Anpassen der Datenbank Einstellungen auf die Server Hardware.

```
elbrus@server:~$ sudo timescaledb-tune --pg-config=/usr/\
pgsql-14/bin/pg_config --yes
```

Listing 54: Neustarten des Services um Änderungen zu übernehmen.

```
elbrus@server:~$ sudo systemctl restart postgresql-14
```

## 2.3    Erstellen der Elbrus-Datenbank

Listing 55: Verbinden mit dem interaktiven Terminal von 'postgres'.

```
elbrus@server:~$ sudo su postgres -c psql
```

Im folgenden Text sind markierte Abschnitte Variablen, welche im darunter-liegen SQL gerändert werden können, was aus Sichertsgründen dringend emp-fohlen wird.

1. Die Datenbank elbrus anlegen

2. Die Zeitzone auf Europe/Vienna setzen

3. Den User elbrus mit dem Passwort elbrus123! anlegen

4. Dem User alle rechte auf die voher erstellte Datenbank geben

Listing 56: Aufführen von SQL Befehlen.

```
CREATE DATABASE elbrus;
ALTER DATABASE elbrus SET timezone TO 'Europe/Vienna';
CREATE USER elbrus PASSWORD 'elbrus123!';
GRANT ALL ON DATABASE elbrus TO elbrus;
```

Listing 57: Wechseln zu erstellter Datenbank.

```
\c elbrus
```

Listing 58: Hinzufügen der TimescaleDB Erweiterung.

```
CREATE EXTENSION IF NOT EXISTS timescaledb;
exit
```

Listing 59: Anlegen der benötigten Tabellen duch das ausführen von 'init.sql'.

```
elbrus@server:~/var/elbrus$ psql -U elbrus -d elbrus -f \
database/sql/init.sql
```

# 3 Aufzeichnen der Daten

## 3.1 Voraussetzungen

Listing 60: Installieren von 'tcpdump' für das aufzeichnen von Daten.

```
elbrus@server:~$ sudo dnf install tcpdump
```

Listing 61: Anlegen eines Users der Berechtigungen zum ausführen von 'tcpdump' erhält.

```
elbrus@server:~$ sudo useradd aragog
```

Listing 62: Zuweisen von 'tcpdump' zu der Gruppe 'aragog'.

```
elbrus@server:~$ sudo chgrp aragog /usr/sbin/tcpdump
```

Listing 63: Ändern der Berechtigungen auf 'tcpdump'.

```
elbrus@server:~$ chmod 750 /usr/sbin/tcpdump
elbrus@server:~$ sudo setcap cap_net_raw,cap_net_admin=eip \
/usr/sbin/tcpdump
```

## 3.2 Umgebung Konfigurieren

### 3.2.1 1 - Mit Setup Script

Listing 64: Ausführen des 'install.sh' Scripts.

```
elbrus@server:~/var/elbrus$ bash capture/install.sh
Do you want to proceed with setup of the 'capture-device'? (y/n) y

Where should the log be stored (dir) [/var/elbrus/shared/log]:
Where is the elb-capture-postrotate.sh stored [/var/elbrus/capture/elb-capture-
  postrotate.sh]:
Where is the shared config stored [/var/elbrus/shared/.config]:

Should the log be stored at '/var/elbrus/shared/log' ?
Is the 'elb-capture-postrotate.sh' stored at '/var/elbrus/capture/elb-capture-postrotate
  .sh' ?
Is the shared config stored at '/var/elbrus/shared/.config' ? (y/n/exit) y
#global
SHAREDCONFIG=/var/elbrus/shared/.config

#paths
POSTROTATESCRIPT=/var/elbrus/capture/elb-capture-postrotate.sh
LOGFILE=/var/elbrus/shared/log

#settings
TIMEPERCAPTURE=900
MAXFILES=10
INTERFACE=eth0
Cleaning up...
elbrus@server:~/var/elbrus$
```

### 3.2.2 2 - Ohne Setup Script

Listing 65: Anhand von '.env.example' eigene '.env' Datei anlegen.

```
1    #global
2    SHAREDCONFIG=/var/elbrus/shared/.config
3
4    #path
5    POSTROTATESCRIPT=/var/elbrus/capture/elb-capture-postrotate.sh
6    LOGFILEDIR=/var/elbrus/capture/capture"-$(date +"%Y-%U")".log
7
8    #settings
9    TIMEPERCAPTURE=900
10   MAXFILES=10
11   INTERFACE=eth0
```

## 3.3 Der Systemd Service

Listing 66: capture.service.example - Die Variable 'WorkingDirectory', Die Variable 'User' sowie die Variable 'ExecStopPost' anpassen.

```
 3      ...
 4      #job is starting immediatly after the start action has been called
 5      Type=simple
 6      #the user to execute the script
 7      User=aragog
 8      #the working directory
 9      WorkingDirectory=/var/elbrus/capture
10      #which script should be executed
11      ExecStart=/bin/bash elb-capture.sh
12      #when the script should restart
13      Restart=on-failure
14      #set the restart timeout
15      RestartSec=5
16      #which script should be executed when the service stops
17      ExecStopPost=/bin/bash elb-capture-log.sh
18
19      [Install]
20      ...
```

Listing 67: Kopieren des Serviceprogrammes.

```
elbrus@server:~/var/elbrus$ sudo cp capture/capture.service.example\
/etc/systemd/system/capture.service
```

Listing 68: Neuladen des 'systemctl' Deamons.

```
elbrus@server:~/var/elbrus$ sudo systemctl daemon-reload
```

Listing 69: Aktivieren des Serviceprogrammes.

```
elbrus@server:~/var/elbrus$ sudo systemctl enable capture.service
```

Listing 70: Starten des Serviceprogrammes.

```
elbrus@server:~/var/elbrus$ sudo systemctl start capture.service
```

# 4 Packet Capture Importer

## 4.1 Umgebung Konfigurieren

### 4.1.1 1 - Mit Setup Script

Listing 71: Ausführen des 'install.sh' Scripts.

```
elbrus@server:~$ cd /var/elbrus
elbrus@server:~/var/elbrus$ sudo bash database/importer/pcap-importer/\
install.sh
Do you want to proceed? (y/n) y

Where is the shared config stored [/var/elbrus/shared/.config]:
Where should the 'pcap-importer' (dir) be stored [/var/elbrus/pcap-importer]:

Would you like to store the 'pcap-importer' at '/var/elbrus/pcap-importer' ?
Is the shared config stored at '/var/elbrus/shared/.config' ? (y/n/exit) y
Submodule 'importer/pcap-importer/pcap-analyzer' (https://github.com/rusticata/pcap-
  analyzer.git) registered for path 'importer/pcap-importer/pcap-analyzer'
Cloning into '/var/elbrus/database/importer/pcap-importer/pcap-analyzer'...
Submodule path 'importer/pcap-importer/pcap-analyzer': checked out '26
  abc0b0f4d9b2f0e6a72a62e694cd60ae6b6011'
Start Building ... (this may take a while)
Compiling proc-macro2 v1.0.38
Compiling unicode-xid v0.2.3
Compiling syn v1.0.93
...
Compiling libpcap-tools v0.1.0 (/var/elbrus/database/importer/pcap-importer/pcap-
  analyzer/libpcap-tools)
Compiling tokio-postgres v0.7.6
Compiling pcap-importer v0.1.0 (/var/elbrus/database/importer/pcap-importer)
Finished release [optimized] target(s) in 1m 38s
Cleaning up...
elbrus@server:~/var/elbrus$
```

### 4.1.2  2 - Ohne Setup Script

Listing 72: Anhand von '.env.example' eigene '.env' Datei anlegen.

```
1    #global
2    SHAREDCONFIG=/var/elbrus/shared/.config
```

Listing 73: Wechseln des Besitzers von Database.

```
elbrus@server:~/var/elbrus$ sudo chown -R elbrus:elbrus /var/elbrus/database/
```

Listing 74: Updaten der git Submodule.

```
elbrus@server:~/var/elbrus$ sudo git -C database submodule update --init
```

Listing 75: Kompilieren des 'pcap-importers'.

```
elbrus@server:~/var/elbrus$ cargo build --release --manifest-path \
database/importer/pcap-importer/Cargo.toml
```

Listing 76: Kopieren des 'pcap-importers' in ein eigenes Verzeichniss.

```
elbrus@server:~/var/elbrus$ sudo mkdir -p /var/elbrus/pcap-importer
elbrus@server:~/var/elbrus$ sudo mv database/importer/pcap-importer/target/\
release/pcap-importer /var/elbrus/pcap-importer/pcap-importer
```

# 5 Report Generator

## 5.1 Umgebung Konfigurieren

### 5.1.1 1 - Mit Setup Script

Listing 77: Ausführen des 'install.sh' Scripts.

```
elbrus@server:~/var/elbrus$ bash report-generator/install.sh
Do you want to proceed with setup of the 'report-generator'? (y/n) y

Where is the shared config stored [/var/elbrus/shared/.config]:

Is the shared config stored at '/var/elbrus/shared/.config' ? (y/n/exit) y
Install dependencies ...

...

elbrus@server:~/var/elbrus$
```

### 5.1.2 2 - Ohne Setup Script

Listing 78: Anhand von '.env.example' eigene '.env' Datei anlegen.

```
#global
SHAREDCONFIG=/var/elbrus/shared/.config
```

Listing 79: Installieren von fehlenden python3 Packages.

```
elbrus@server:~/var/elbrus$ pip3 install -r \
report-generator/requirements.txt
```

# 6  Globale Konfiguration

Listing 80: Anhand von '.config.example' eigene '.config' Datei in '/var/elbrus/shared' anlegen.

```
1    #database settings
2    DB_HOST=10.0.76.220
3    DB_PORT=5432
4    DB_NAME=elbrus
5    DB_USER=elbrus
6    DB_PASSWORD=elbrus123!
7
8    #paths
9    PCAPFOLDER=/var/elbrus/shared/traces
10   IMPORTERPATH=/var/elbrus/pcap-importer/pcap-importer
11   REPORTERPATH=/var/elbrus/report-generator/src/main.py
```

# 7 SNMP Manager

## 7.1 Umgebung Konfigurieren

### 7.1.1 1 - Mit Setup Script

Listing 81: Ausführen des 'install.sh' Scripts.

```
elbrus@server:~$ cd /var/elrbus
elbrus@server:~/var/elrbus$ bash snmp-manager/src/install.sh
Do you want to proceed with setup of the 'snmp-manager'? (y/n) y

Where should the log be stored (dir) [/var/elbrus/shared/log]:
Where is the shared config stored [/var/elbrus/shared/.config]:

Should the log be stored at '/var/elbrus/shared/log' ?
Is the shared config stored at '/var/elbrus/shared/.config' ? (y/n/exit) y
#global
SHAREDCONFIG=/var/elbrus/shared/.config

#paths
LOGFILEDIR=/var/elbrus/snmp-manager/log
Install dependencies ...

...

Cleaning up...
elbrus@server:~/var/elrbus/snmp-manager$
```

### 7.1.2 2 - Ohne Setup Script

Listing 82: Anhand von '.env.example' eigene '.env' Datei anlegen.

```
17    #global
18    SHAREDCONFIG=/var/elbrus/shared/.config
19
20    #paths
21    LOGFILEDIR=/var/elbrus/shared/log
```

## 7.2 Der Systemd Service

Listing 83: snmp-manager.service.example - Die Variable 'WorkingDirectory'
sowie die Variable 'User' anpassen.

```
 5    ...
 6    #job is starting immediatly after the start action has been called
 7    Type=simple
 8    #the user to execute the script
 9    User=elbrus
10    #the working directory
11    WorkingDirectory=/var/elbrus/snmp-manager/src
12    #which script should be executed
13    ExecStart=/bin/bash elb-snmp-manager.sh
14    ...
```

Listing 84: Kopieren des Serviceprogrammes.

```
elbrus@server:~/var/elbrus$ sudo cp snmp-manager/src/snmp-manager.service\
.example /etc/systemd/system/snmp-manager.service
```

Listing 85: Kopieren des Zeitplanungsprogrammes.

```
elbrus@server:~/var/elbrus$ sudo cp snmp-manager/src/snmp-manager.timer\
.example /etc/systemd/system/snmp-manager.timer
```

Listing 86: Neuladen des 'systemctl' Deamons.

```
elbrus@server:~/var/elbrus$ sudo systemctl daemon-reload
```

Listing 87: Aktivieren des Serviceprogrammes.

```
elbrus@server:~/var/elbrus$ sudo systemctl enable snmp-manager.service
```

Listing 88: Aktivieren des Zeitplanungsprogrammes.

```
elbrus@server:~/var/elbrus$ sudo systemctl enable snmp-manager.timer
```

Listing 89: Starten des Zeitplanungsprogrammes.

```
elbrus@server:~/var/elbrus$ sudo systemctl start snmp-manager.timer
```

# 8 SSH Manager

## 8.1 Umgebung Konfigurieren

### 8.1.1 1 - Mit Setup Script

Listing 90: Ausführen des 'install.sh' Scripts.

```
elbrus@server:~$ cd /var/elbrus
elbrus@server:~$ bash ssh-manager/src/install.sh
Do you want to proceed with the setup of the 'ssh-manager'? (y/n) y
we will proceed

Where do you want the ssh config replies to be stored (dir) [/var/elbrus/
shared/ssh-configs]
Where is the 'main.py' file stored [/var/elbrus/ssh-manager/src/main.py]
Where is the shared config stored [/var/elbrus/shared/.config]

Do you want to store the config files at '/var/elbrus/shared/ssh-configs'?
Is your 'main.py' stored at '/var/elbrus/ssh-manager/src/main.py'?
Is the shared config stored at '/var/elbrus/shared/.config'? (y/n/exit) y
we will proceed

#global
SHAREDCONFIG=/var/elbrus/shared/.config

#values regarding the jumpserver:
#IP, PORT and USER values must be set!
#depending on the usage you can set either:
#   -PASS and KEYFILE: the keyfile is used, the pass is interpreted as the
passphrase
#   -only KEYFILE: the keyfile is used
#   -only PASS: the password is used as is regular credentials
JUMPSERVER_IP=
JUMPSERVER_PORT=
JUMPSERVER_USER=
JUMPSERVER_PASS=
SSH_KEYFILE=

#paths
CONFIGPATH=/var/elbrus/shared/ssh-configs
MAINPATH=/var/elbrus/ssh-manager/src/main.py

Do you want to run the setup script? (y/n/exit) y
...
Initialized empty Git repository in /var/elbrus/shared/ssh-configs/.git/
info: created config folder!
Install dependencies ...
...
Cleaning up...
elbrus@server:~/ssh-manager$
```

Listing 91: Ergänzen der fehlenden Werten in '.env'.

```
1    #values regarding the jumpserver:
2    #IP, PORT and USER values must be set!
3    #depending on the usage you can set either:
4    #    -PASS and KEYFILE: the keyfile is used, the pass is interpreted as the
     passphrase
5    #    -only KEYFILE: the keyfile is used
6    #    -only PASS: the password is used as is regular credentials
7    JUMPSERVER_IP=
8    JUMPSERVER_PORT=
9    JUMPSERVER_USER=
10   JUMPSERVER_PASS=
11   SSH_KEYFILE=
```

### 8.1.2   2 - Ohne Setup Script

Listing 92: Anhand von '.env.example' eigene '.env' Datei anlegen.

```
1    #global
2    SHAREDCONFIG=/var/elbrus/shared/.config
3
4    #values regarding the jumpserver:
5    #IP, PORT and USER values must be set!
6    #depending on the usage you can set either:
7    #    -PASS and KEYFILE: the keyfile is used, the pass is interpreted as the
     passphrase
8    #    -only KEYFILE: the keyfile is used
9    #    -only PASS: the password is used as is regular credentials
10   JUMPSERVER_IP=
11   JUMPSERVER_PORT=
12   JUMPSERVER_USER=
13   JUMPSERVER_PASS=
14   SSH_KEYFILE=
15
16   #paths
17   CONFIGPATH=/var/elbrus/shared/ssh-configs
18   MAINPATH=/var/elbrus/ssh-manager/src/main.py
```

Listing 93: Ausführen des Scripts zur Initialisierung des VCS Verzeichnisses.

```
elbrus@server:~ssh-manager$ bash src/setup.sh
```

Listing 94: Installieren von fehlenden python3 Packages.

```
elbrus@server:~ssh-manager$ pip3 install -r requirements.txt
```

## 8.2 Der Systemd Service

Listing 95: ssh-manager.service.example - Die Variable 'WorkingDirectory' sowie die Variable 'User' anpassen.

```
 5    ...
 6    #job is starting immediatly after the start action has been called
 7    Type=simple
 8    #the user to execute the script
 9    User=elbrus
10    #the working directory
11    WorkingDirectory=/var/elbrus/ssh-manager/src/
12    #which script should be executed
13    ExecStart=/bin/bash routine.sh
14    ...
```

Listing 96: Kopieren des Serviceprogrammes.

```
elbrus@server:~$ cp src/ssh-manager.service.example\
/etc/systemd/system/ssh-manager.service
```

Listing 97: Kopieren des Zeitplanungsprogrammes.

```
elbrus@server:~$ cp src/ssh-manager-schedule.timer.example\
/etc/systemd/system/ssh-manager-schedule.timer
```

Listing 98: Neuladen des 'systemctl' Deamons.

```
elbrus@server:~$ systemctl daemon-reload
```

Listing 99: Aktivieren des Serviceprogrammes.

```
elbrus@server:~$ systemctl enable ssh-manager.service
```

Listing 100: Aktivieren des Zeitplanungsprogrammes.

```
elbrus@server:~$ systemctl enable ssh-manager-schedule.timer
```

Listing 101: Starten des Zeitplanungsprogrammes.

```
elbrus@server:~$ systemctl start ssh-manager-schedule.timer
```

# 9 API

## 9.1 Voraussetzungen

Listing 102: Nachinstallieren der Abhängigkeiten.

```
elbrus@server:~/var/elbrus/api$ npm install
```

Listing 103: Installieren von 'pm2'.

```
elbrus@server:~/var/elbrus/api$ npm install -g pm2
```

## 9.2 Umgebung Konfigurieren

Listing 104: Anhand von '.env.example' eigene '.env' Datei anlegen.

```
1   # Application Name
2   APP_NAME=Elbrus-API
3
4   # Port number
5   PORT=3000
6
7   # BASE URL
8   BASE=https://localhost:3000
9
10  # URL of DB
11  DB_USER=
12  DB_HOST=
13  DB_DATABASE=
14  DB_PASSWORD=
15  DB_PORT=
16
17  # JWT
18  JWT_SECRET=thisisasamplesecret
19  JWT_ACCESS_EXPIRATION_MINUTES=30
20  JWT_REFRESH_EXPIRATION_DAYS=30
21
22  # SMTP configuration options for the email service
23  SMTP_HOST=
24  SMTP_PORT=
25  SMTP_USERNAME=
26  SMTP_PASSWORD=
27  EMAIL_FROM=
28  EMAIL_NAME=
```

## 9.3 Inbetriebnahme

Listing 105: Starten der API.

```
elbrus@server:~/var/elbrus/api$ pm2 start ecosystem.config.json
```

Die API läuft in folge automatisch im Hintergrund.

1. **APP_NAME** wird rein als beschreibender Name genutzt und kann so belassen werden.

2. **PORT** beschreibt den TCP Port auf dem die Applikation laufen soll.

3. **BASE** ist der Wert der Basis URL auf welche zugegriffen wird. Hier muss der Port auch angegeben werden!

4. **DB_USER** ist der benutzername des DBMS Benutzers, über welchen der Zugriff auf die Datenbank läuft.

5. **DB_HOST** ist der hostname/ip-adresse des Servers welcher die Datenbank hostet.

6. **DB_DATABASE** beschreibt den Namen der Datenbank selber.

7. **DB_PASSWORD** ist das Passwort des DBMS Benutzers, über welchen der Zugriff auf die Datenbank läuft.

8. **DB_PORT** ist der TCP Port des Servers welcher die Datenbank hostet.

9. **JWT_SECRET** ist das Passwort mit dem alle JWT Tokens ausgestellt werden.

10. **JWT_ACCESS_EXPIRATION_MINUTES** gibt die Dauer der Gültigkeit eines Access-Tokens an (in Minuten)

11. **JWT_REFRESH_EXPIRATION_DAYS** gibt die Dauer der Gültigkeit eines Refresh-Tokens an (in Tagen)

12. **SMTP_HOST** ist der hostname/ip-adresse des EMail Servers

13. **SMTP_PORT** ist der TCP Port des EMail Servers für SMTP

14. **SMTP_USERNAME** ist der username des Benutzers zum einloggen in den EMail Account

15. **SMTP_PASSWORD** ist das passwort des Benutzers zum einloggen in den EMail Account

16. **EMAIL_FROM** gibt die Email adresse an, von welcher gesendet werden soll.

17. **EMAIL_NAME** gibt den Namen an, welcher dem Empfänger angezeigt werden soll.

# 10 Webinterface