Elbrus Analytics - Bereitstellungshandbuch

Tobias Schmidt

August 16, 2022

Bekanntes Problem: Beim herauskopieren von Befehlen wird das Apostrophe Zeichen falsch kopiert und führt zu Eingabe Störungen. Lösung: Apostrophe Zeichen des kopierten Befehls händisch im Terminal mit Apostrophe Zeichen austauschen.

1 Server Infrastruktur

1.1 SSH-Zugriff vorbereiten

Listing 1: Updaten vorhandener Packages.

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

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: Anlegen des Users Elbrus.

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

Listing 6: Hinzufügen des Users Elbrus zu der Gruppe ’wheel’.

```
root@server:~$ usermod -aG wheel elbrus
```

Listing 7: Ändern des Passwords für den User Elbrus.

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

Listing 8: Wechseln zu User elbrus.

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

1.2 Initiale Server Konfiguration

Listing 9: Setzen der Zeitzone auf ’Europa/Wien’.

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

Listing 10: Installieren von dem ’firewalld’ Service.

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

1.2.1 Node.Js

Listing 11: Installieren des Framworks ’Node.Js’.

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

1.2.2 Ablagestruktur

Listing 12: Anlegen der Verzeichnissstruktur.

```
elbrus@server:~$ sudo mkdir /var/elbrus/shared/keys
elbrus@server:~$ sudo chown -R elbrus:elbrus /var/elbrus
elbrus@server:~$ cd /var/elbrus
elbrus@server:/var/elbrus$ chmod -R 777 /var/elbrus/shared
```

1.3 Git

Listing 13: Installieren von dem VCS 'git'.

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

1.3.1 Git - Erstellen der SSH-Keys

Listing 14: Wechseln des Verzeichnisses.

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

Listing 15: 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 16: 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 17: 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 18: 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 19: 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 20: Erstellen des SSH-keys der für das Herunterladen des ’Uptime-Monitors’ benötigt wird.

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

Listing 21: Erstellen des SSH-keys der für das Herunterladen des ’geo session finders’ benötigt wird.

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

Listing 22: Erstellen des SSH-keys der für das Herunterladen des ’office365 analyzers’ benötigt wird.

```
elbrus@server:/var/elbrus/keys$ ssh-keygen -t rsa -b 2048 \
-f office365_analyzer_key -q -N ""
```

Listing 23: 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 24: 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 angelegten 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 25: Wechseln des Verzeichnisses.

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

Listing 26: Clonen der Datenbank Software.

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

Listing 27: Clonen der ’Capture-Device’ Software.

```
elbrus@server:/var/elbrus$ git clone git@github.com:Elbrus-Analytics/capture-device.git\
 --config core.sshCommand="ssh -i ./shared/keys/capture_device_key"
```

Listing 28: Clonen der ’Report-Generator’ Software.

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

Listing 29: Clonen der ’SNMP-Manager’ Software.

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

Listing 30: Clonen der ’SSH-Manager’ Software.

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

Listing 31: Clonen der ’Uptime-Monitor’ Software.

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

Listing 32: Clonen der ’geo session finders’ Software.

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

Listing 33: Clonen der ’office365-analyzer’ Software.

```
elbrus@server:/var/elbrus/keys$ git clone git@github.com:Elbrus-Analytics/\
office365-analyzer.git --config core.sshCommand="ssh -i ./shared/keys/office365_analyzer_key"
```

Listing 34: Clonen der API Software.

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

Listing 35: Clonen der Packet-Importer Software

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

1.4 Python

1.4.1 1 - Automatische Installation

Listing 36: Ausführen des ’pythonSourceInstall.sh’ Scripts.

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

1.4.2 2 - Manuel Installation

Listing 37: Installieren von benötigten Packeten und Abhängigkeiten.

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

Listing 38: Herunterladen der Source Datei.

```
elbrus@server:/var/elbrus$ wget https://www.python.org/ftp/python/\
3.10.0/Python-3.10.0.tar.xz
```

Listing 39: Extrahieren der installierten Datei.

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

Listing 40: Wechseln zu source Verzeichniss. Und ausführen des Konfigurations Scripts.

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

Listing 41: Starten des build Prozesses.

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

Listing 42: Installieren von Python.

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

Listing 43: Löschen der kompremierten Python Datei.

```
elbrus@server:/var/elbrus/Python-3.10.0$ cd .. && rm Python-3.10.0.tar.xz
```

1.4.3 Upgrade von ’pip’

Listing 44: Upgraden von ’pip’.

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

1.5 Rust

Listing 45: Installieren von GNU Compiler Collection.

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

Listing 46: 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 47: Laden der Variablen aus dem Terminal Profil.

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

Listing 48: Hinzufügen des Befehls Cargo zu dem Pfad.

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

2 Datenbank

2.1 Voraussetzungen

Listing 49: 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 50: 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 51: Updaten der lokalen Package-Liste.

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

Listing 52: 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 53: Initialisieren der Datenbank.

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

Listing 54: 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 55: /var/lib/pgsql/14/data/postgresql.conf - Ändern der folgenden Zeilen.

```
- #shared_preload_libraries = ’’
+ shared_preload_libraries = ’timescaledb’

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

Listing 56: /var/lib/pgsql/14/data/pg_hba.conf - Ändern der folgenden Zeilen.

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

Listing 57: Anpassen der Datenbank Einstellungen auf die Server Hardware.

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

Listing 58: Neustarten des Services um Änderungen zu übernehmen.

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

2.3 Erstellen der Elbrus-Datenbank

Listing 59: Verbinden mit dem interaktiven Terminal von 'postgres'.

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

Im folgenden Text sind markierte Abschnitte Variablen, welche im darunterliegen SQL geändert werden können, was aus Sichertsgründen dringend empfohlen 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 60: Auführen 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 61: Wechseln zu erstellter Datenbank.

```
\c elbrus
```

Listing 62: Hinzufügen der TimescaleDB Erweiterung.

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

Listing 63: 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 Globale Konfiguration

Listing 64: Anhand von '.config.example' eigene '.config' Datei in '/var/elbrus/shared' anlegen.

```
#database settings
DB_HOST=localhost
DB_PORT=5432
DB_NAME=elbrus
DB_USER=elbrus
DB_PASSWORD=elbrus123!

#paths
PCAPFOLDER=/var/elbrus/shared/traces
IMPORTERPATH=/var/elbrus/pcap-importer/pcap-importer
REPORTERPATH=/var/elbrus/report-generator/src/main.py
```

```
elbrus@server:/var/elbrus$ sudo chown elbrus:elbrus /var/elbrus/shared/.config
elbrus@server:/var/elbrus$ sudo chmod 776 /var/elbrus/shared/.config
```

4 Aufzeichnen der Daten

4.1 Umgebung Konfigurieren

4.1.1 1 - Mit Setup Script

Listing 65: Ausführen des 'install.sh' Scripts.

```
elbrus@server:~$ cd /var/elbrus
elbrus@server:/var/elbrus$ sudo bash capture-device/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-device/scripts/elb-
  capture-postrotate.sh]:
Where is the shared config stored [/var/elbrus/shared/.config]:
Where should the capture-device be stored [/var/elbrus/capture-device/target/release/
  aragog]:

Should the log be stored at '/var/elbrus/shared/log' ?
Is the 'elb-capture-postrotate.sh' stored at '/var/elbrus/capture-device/scripts/elb-
  capture-postrotate.sh' ?
Is the shared config stored at '/var/elbrus/shared/.config' ?
Should the capture-device be stared at '/var/elbrus/capture-device/target/release/aragog
  ' ? (y/n/exit) y

...

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

#path
POSTROTATESCRIPT=/var/elbrus/capture-device/target/release/scripts/elb-capture-
  postrotate.sh
LOGFILEDIR=/var/elbrus/capture/
CAPTUREPATH=/var/elbrus/capture-device/target/release/aragog

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

4.1.2 2 - Ohne Setup Script

Listing 66: Anlegen eines Users der Berechtigungen zum ausführen des ’capture-device’ erhält.

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

Listing 67: Hinzufügen von User ’aragog’ zu Gruppe ’elbrus’.

```
elbrus@server:~$ sudo usermod -aG elbrus aragog
```

Listing 68: Wechseln des Owners & der Berechtigung auf ’/var/elbrus/capture-device/’

```
elbrus@server:/var/elbrus$ sudo chown -R aragog:aragog /var/elbrus/capture-device/
elbrus@server:/var/elbrus$ sudo chmod -R 770 /var/elbrus/capture-device/
elbrus@server:/var/elbrus$ sudo chmod 777 /var/elbrus/capture-device/
elbrus@server:/var/elbrus$ sudo chmod 777 /var/elbrus/capture-device/install.sh
```

Listing 69: Anhand von ’.env.example’ eigene ’.env’ Datei anlegen.

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

#path
POSTROTATESCRIPT=/var/elbrus/capture/elb-capture-postrotate.sh
LOGFILEDIR=/var/elbrus/capture/
CAPTUREPATH=/var/elbrus/capture-device/target/release/aragog

#settings
TIMEPERCAPTURE=900
MAXFILES=10
INTERFACE=eth0
```

Listing 70: Kompilieren des ’capture-device’.

```
elbrus@server:~$ sudo ln -s /usr/lib64/libpcap.so.1 /usr/lib64/libpcap.so
elbrus@server:~$ cd /var/elbrus/capture-device
elbrus@server:/var/elbrus/capture-device$ cargo build --release
```

Listing 71: Erteilen der Berechtigung zum Aufzeichenen von Packages.

```
elbrus@server:/var/elbrus/capture-device$ cd /var/elbrus
elbrus@server:/var/elbrus$ sudo chmod 750 \
/var/elbrus/capture-device/target/release/aragog
elbrus@server:/var/elbrus$ sudo setcap cap_net_raw,cap_net_admin=eip \
/var/elbrus/capture-device/target/release/aragog
```

4.2 Der Systemd Service

Listing 72: capture.service.example - Die Variable ’WorkingDirectory’, Die Variable ’User’ sowie die Variable ’ExecStopPost’ anpassen.

```
...
#job is starting immediatly after the start action has been called
Type=simple
#the user to execute the script
User=aragog
#the working directory
WorkingDirectory=/var/elbrus/capture
#which script should be executed
ExecStart=/bin/bash elb-capture.sh
#when the script should restart
Restart=on-failure
#set the restart timeout
RestartSec=5
#which script should be executed when the service stops
ExecStopPost=/bin/bash elb-capture-log.sh

[Install]
...
```

Listing 73: Kopieren des Serviceprogrammes.

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

Listing 74: Neuladen des ’systemctl’ Deamons.

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

Listing 75: Aktivieren des Serviceprogrammes.

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

Listing 76: Starten des Serviceprogrammes.

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

5 Packet Capture Importer

5.1 Umgebung Konfigurieren

5.1.1 1 - Mit Setup Script

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

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

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: Updaten der git Submodule.

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

Listing 80: Kompilieren des 'pcap-importers'.

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

Listing 81: Kopieren des 'pcap-importers' in ein eigenes Verzeichniss.

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

6 Report Generator

6.1 Umgebung Konfigurieren

6.1.1 1 - Mit Setup Script

Listing 82: Ausführen des ’install.sh’ Scripts.

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

6.1.2 2 - Ohne Setup Script

Listing 83: Anhand von ’.env.example’ eigene ’.env’ Datei anlegen.

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

Listing 84: Installieren von fehlenden python3 Packages.

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

6.2 Der Systemd Service

Listing 85: report-generator.service.example - Die Variable ’WorkingDirectory’ sowie die Variable ’User’ anpassen.

```
...
#job is starting immediatly after the start action has been called
Type=simple
#the user to execute the script
User=elbrus
#the working directory
WorkingDirectory=/var/elbrus/report-generator/src/
#which script should be executed
ExecStart=python3 main.py
...
```

Listing 86: Kopieren des Serviceprogrammes.

```
elbrus@server:/var/elbrus$ sudo cp report-generator/src/report-generator.service\
.example /etc/systemd/system/report-generator.service
```

Listing 87: Kopieren des Zeitplanungsprogrammes.

```
elbrus@server:/var/elbrus$ sudo cp report-generator/src/report-generator-schedule.timer\
.example /etc/systemd/system/report-generator-schedule.timer
```

Listing 88: Neuladen des ’systemctl’ Deamons.

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

Listing 89: Aktivieren des Serviceprogrammes.

```
elbrus@server:/var/elbrus$ sudo systemctl enable report-generator.service
```

Listing 90: Aktivieren des Zeitplanungsprogrammes.

```
elbrus@server:/var/elbrus$ sudo systemctl enable report-generator-schedule.timer
```

Listing 91: Starten des Zeitplanungsprogrammes.

```
elbrus@server:/var/elbrus$ sudo systemctl start report-generator-schedule.timer
```

7 SNMP Manager

7.1 Umgebung Konfigurieren

7.1.1 1 - Mit Setup Script

Listing 92: Ausführen des ’install.sh’ Scripts.

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

7.1.2 2 - Ohne Setup Script

Listing 93: Anhand von ’.env.example’ eigene ’.env’ Datei anlegen.

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

#paths
LOGFILEDIR=/var/elbrus/shared/log
```

7.2 Der Systemd Service

Listing 94: snmp-manager.service.example - Die Variable 'WorkingDirectory' sowie die Variable 'User' anpassen.

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

Listing 95: Kopieren des Serviceprogrammes.

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

Listing 96: Kopieren des Zeitplanungsprogrammes.

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

Listing 97: Neuladen des 'systemctl' Deamons.

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

Listing 98: Aktivieren des Serviceprogrammes.

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

Listing 99: Aktivieren des Zeitplanungsprogrammes.

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

Listing 100: Starten des Zeitplanungsprogrammes.

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

8 SSH Manager

8.1 Umgebung Konfigurieren

8.1.1 1 - Mit Setup Script

Listing 101: 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 102: Ergänzen der fehlenden Werten in '.env'.

```
#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=
```

8.1.2 2 - Ohne Setup Script

Listing 103: Anhand von '.env.example' eigene '.env' Datei anlegen.

```
#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
```

Listing 104: Ausführen des Scripts zur Initialisierung des VCS Verzeichnisses.

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

Listing 105: Installieren von fehlenden python3 Packages.

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

8.2 Der Systemd Service

Listing 106: ssh-manager.service.example - Die Variable ’WorkingDirectory’ sowie die Variable ’User’ anpassen.

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

Listing 107: Kopieren des Serviceprogrammes.

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

Listing 108: Kopieren des Zeitplanungsprogrammes.

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

Listing 109: Neuladen des ’systemctl’ Deamons.

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

Listing 110: Aktivieren des Serviceprogrammes.

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

Listing 111: Aktivieren des Zeitplanungsprogrammes.

```
elbrus@server:/var/elbrus$ sudo systemctl enable ssh-manager-schedule.timer
```

Listing 112: Starten des Zeitplanungsprogrammes.

```
elbrus@server:/var/elbrus$ sudo systemctl start ssh-manager-schedule.timer
```

9 Uptime Monitor

9.1 Umgebung Konfigurieren

9.1.1 1 - Mit Setup Script

Listing 113: Ausführen des ’install.sh’ Scripts.

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

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

Is the shared config stored at ’/var/elbrus/shared/.config’? (y/n/exit) y
we will proceed

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

#config
INITIALPING=1
STATISTICPING=10
Cleaning up...
elbrus@server:~$
```

9.1.2 2 - Ohne Setup Script

Listing 114: Anhand von ’.env.example’ eigene ’.env’ Datei anlegen.

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

#config
# Initial pings to see if device is alive
INITIALPING=1
# Pings to get the availability statistic
STATISTICPING=10
```

9.2 Der Systemd Service

Listing 115: uptime_monitor.service.example - Die Variable 'WorkingDirectory' sowie die Variable 'User' anpassen.

```
...
#job is starting immediatly after the start action has been called
Type=simple
#the user to execute the script
User=elbrus
#the working directory
WorkingDirectory=/var/elbrus/uptime_monitor
#which script should be executed
ExecStart=/bin/bash uptime_monitor.sh
...
```

Listing 116: Kopieren des Serviceprogrammes.

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

Listing 117: Kopieren des Zeitplanungsprogrammes.

```
elbrus@server:/var/elbrus$ sudo cp uptime_monitor/uptime_monitor-schedule.timer.example \
/etc/systemd/system/uptime_monitor-schedule.timer
```

Listing 118: Neuladen des 'systemctl' Deamons.

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

Listing 119: Aktivieren des Serviceprogrammes.

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

Listing 120: Aktivieren des Zeitplanungsprogrammes.

```
elbrus@server:/var/elbrus$ sudo systemctl enable uptime_monitor-schedule.timer
```

Listing 121: Starten des Zeitplanungsprogrammes.

```
elbrus@server:/var/elbrus$ sudo systemctl start uptime_monitor-schedule.timer
```

10 Geo Session finder

10.1 Umgebung Konfigurieren

10.1.1 1 - Mit Setup Script

Listing 122: Ausführen des ’install.sh’ Scripts.

```
elbrus@server:~$ cd /var/elbrus
elbrus@server:~$ bash geo_session_finder/src/install.sh
Do you want to proceed with setup of the ’geo session finder’? (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/shared/log
Install dependencies ...

...

elbrus@server:~$
```

10.1.2 2 - Ohne Setup Script

Listing 123: Installieren der Abhängigkeiten.

```
elbrus@server:~$ pip3 install -r geo_session_finder/requirements.txt
```

Listing 124: Anhand von ’.env.example’ eigene ’.env’ Datei anlegen.

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

#paths
LOGFILEDIR=/var/elbrus/shared/log
```

10.2 Der Systemd Service

Listing 125: geo-session-finder.service.example - Die Variable 'WorkingDirectory' sowie die Variable 'User' anpassen.

```
...
#job is starting immediatly after the start action has been called
Type=simple
#the user to execute the script
User=elbrus
#the working directory
WorkingDirectory=/var/elbrus/geo_session_finder/src
#which script should be executed
ExecStart=/bin/bash elb-geo-session-finder.sh
...
```

Listing 126: Kopieren des Serviceprogrammes.

```
elbrus@server:/var/elbrus$ sudo cp geo_session_finder/src/geo-session-finder\
.service.example /etc/systemd/system/geo-session-finder.service
```

Listing 127: Kopieren des Zeitplanungsprogrammes.

```
elbrus@server:/var/elbrus$ sudo cp geo_session_finder/src/geo-session-finder-schedule\
.timer.example /etc/systemd/system/geo-session-finder-schedule.timer
```

Listing 128: Neuladen des 'systemctl' Deamons.

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

Listing 129: Aktivieren des Serviceprogrammes.

```
elbrus@server:/var/elbrus$ sudo systemctl enable geo-session-finder.service
```

Listing 130: Aktivieren des Zeitplanungsprogrammes.

```
elbrus@server:/var/elbrus$ sudo systemctl enable geo-session-finder-schedule.timer
```

Listing 131: Starten des Zeitplanungsprogrammes.

```
elbrus@server:/var/elbrus$ sudo systemctl start geo-session-finder-schedule.timer
```

11 office365

11.1 Umgebung Konfigurieren

11.1.1 1 - Mit Setup Script

Listing 132: Ausführen des ’install.sh’ Scripts.

```
elbrus@server:~$ cd /var/elbrus
elbrus@server:~$ bash office365-analyzer/src/install.sh
Do you want to proceed with setup of the ’office365-analyzer’? (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/shared/log

#ms url
MS_URL=https://endpoints.office.com/endpoints/worldwide?clientrequestid=b10c5ed1-bad1-445
f-b386-b919946339a7
Install dependencies ...

...

elbrus@server:~$
```

11.1.2 2 - Ohne Setup Script

Listing 133: Anhand von ’.env.example’ eigene ’.env’ Datei anlegen.

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

#paths
LOGFILEDIR=/var/elbrus/shared/log

#ms url
MS_URL=https://endpoints.office.com/endpoints/worldwide?clientrequestid=
 b10c5ed1-bad1-445f-b386-b919946339a7
```

11.2 Der Systemd Service

Listing 134: uptime_monitor.service.example - Die Variable 'WorkingDirectory' sowie die Variable 'User' anpassen.

```
...
#job is starting immediatly after the start action has been called
Type=simple
#the user to execute the script
User=elbrus
#the working directory
WorkingDirectory=/var/elbrus/office365-analyzer/src
#which script should be executed
ExecStart=/bin/bash elb-office365-get-endpoints.sh
...
```

Listing 135: Kopieren des Serviceprogrammes.

```
elbrus@server:/var/elbrus$ sudo cp office365-analyzer/src/office365-get-endpoints\
.service.example /etc/systemd/system/office365-get-endpoints.service
```

Listing 136: Kopieren des Zeitplanungsprogrammes.

```
elbrus@server:/var/elbrus$ sudo cp office365-analyzer/src/office365-get-endpoints\
-schedule.timer.example /etc/systemd/system/office365-get-endpoints-schedule.timer
```

Listing 137: Neuladen des 'systemctl' Deamons.

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

Listing 138: Aktivieren des Serviceprogrammes.

```
elbrus@server:/var/elbrus$ sudo systemctl enable office365-get-endpoints.service
```

Listing 139: Aktivieren des Zeitplanungsprogrammes.

```
elbrus@server:/var/elbrus$ sudo systemctl enable office365-get-endpoints-schedule.timer
```

Listing 140: Starten des Zeitplanungsprogrammes.

```
elbrus@server:/var/elbrus$ sudo systemctl start office365-get-endpoints-schedule.timer
```

12 API

12.1 Voraussetzungen

Listing 141: Installieren von ’pm2’.

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

Listing 142: Nachinstallieren der Abhängigkeiten.

```
elbrus@server:/var/elbrus$ cd api
elbrus@server:/var/elbrus/api$ sudo npm install
```

12.2 Umgebung Konfigurieren

Listing 143: Anhand von '.env.example' eigene '.env' Datei anlegen.

```
# Application Name
APP_NAME=Elbrus-API

# Port number
PORT=3000

# BASE URL
BASE=https://localhost:3000

# URL of DB
DB_USER=
DB_HOST=
DB_DATABASE=
DB_PASSWORD=
DB_PORT=

# JWT
JWT_SECRET=thisisasamplesecret
JWT_ACCESS_EXPIRATION_MINUTES=30
JWT_REFRESH_EXPIRATION_DAYS=30

# SMTP configuration options for the email service
SMTP_HOST=
SMTP_PORT=
SMTP_USERNAME=
SMTP_PASSWORD=
EMAIL_FROM=
EMAIL_NAME=
```

12.3 Inbetriebnahme

Listing 144: 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.

13 Webinterface