Elbrus Analytics - Bereitstellungshandbuch

Tobias Schmidt

August 21, 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 Nginx

Listing 12: Installieren der Webserver-Software 'Nginx'.

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

1.2.3 Pm2

Listing 13: Installieren von 'pm2'.

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

1.2.4 Ablagestruktur

Listing 14: Anlegen der Verzeichnissstruktur.

```
elbrus@server:~$ sudo mkdir -p /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 15: Installieren von dem VCS 'git'.

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

1.3.1 Git - Erstellen der SSH-Keys

Listing 16: Wechseln des Verzeichnisses.

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

Listing 17: 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 18: Erstellen des SSH-Keys der für das Herunterladen der Kernsoftware 'tabby' benötigt wird.

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

Listing 19: 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 20: 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 21: 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 22: 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 23: 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 24: 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 25: 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 26: Wechseln des Verzeichnisses.

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

Listing 27: Clonen der Datenbank Software.

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

Listing 28: Clonen der Kernsoftware 'tabby'.

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

Listing 35: Clonen der 'Webinterface' Software

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

Listing 56: Anpassen der Datenbank Einstellungen auf die Server Hardware.

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

Listing 57: Neustarten des Services um Änderungen zu übernehmen.

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

2.3 Erstellen der Elbrus-Datenbank

Listing 58: 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 gerändert 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 59: 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 60: Wechseln zu erstellter Datenbank.

```
\c elbrus
```

Listing 61: Hinzufügen der TimescaleDB Erweiterung.

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

Listing 62: 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 63: 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!
```

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

4 Kernsoftware - Tabby

4.1 1 - Mit Setup Script

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

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

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

Is the shared config stored at '/var/elbrus/shared/.config' ?
Should the capture-device be stored at '/var/elbrus/tabby' ?
Should the log be stored at '/var/elbrus/shared/log' ?
Should the traces be stored at '/var/elbrus/shared/traces' ? (y/n/exit) y

info: a listing of your interfaces follows

...

Which inteface should be captured (name)? eth0if81

Should 'eth0if81' be captured? (y/n/exit) y
info: the capturing interface can be changed in the environment file.

...

Compiling proc-macro2 v1.0.38
Compiling unicode-xid v0.2.3
Compiling syn v1.0.93
Compiling libpcap-tools v0.1.0
Compiling tokio-postgres v0.7.6
Compiling pcap-importer v0.1.0

...

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

4.2 2 - Ohne Setup Script

4.2.1 Umgebung Konfigurieren

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

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

#paths
PCAPFOLDER=/var/elbrus/shared/traces
LOGFILEDIR=/var/elbrus/shared/log
CAPTUREPATH=/var/elbrus/tabby/tabby

#settings
TIMEPERCAPTURE=600
TIMEPERIMPORT=300
TIMEPERREPORT=900
MAXFILES=12
INTERFACE=eth0if81
```

Listing 66: Kompilieren der Kernsoftware.

```
elbrus@server:/var/elbrus$ sudo /root/.cargo/bin/cargo build --release \
--manifest-path="/var/elbrus/tabby/Cargo.toml"
```

Listing 67: Verschieben der kompilierten Software.

```
elbrus@server:/var/elbrus$ mv tabby/target/release/tabby tabby/tabby
```

Listing 68: Verschieben der kompilierten Software.

```
elbrus@server:/var/elbrus$ mv tabby/target/release/tabby tabby/tabby
```

Listing 69: Anlegen des Benutzers 'tabby'.

```
elbrus@server:/var/elbrus$ sudo useradd -G elbrus tabby
```

Listing 70: Verändern der berechtigungen auf Ordner 'tabby'.

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

Listing 71: Anlegen des Log-Verzeichnisses.

```
elbrus@server:/var/elbrus$ mkdir -p /var/elbrus/shared/log
```

Listing 72: Verändern der Berechtigung das Log-Verzeichnisses.

```
elbrus@server:/var/elbrus$ chmod 777 -R /var/elbrus/shared/log
```

Listing 73: Verändern der Berechtigung der kompilierten Software.

```
elbrus@server:/var/elbrus$ sudo chmod 750 tabby/tabby
elbrus@server:/var/elbrus$ sudo setcap cap_net_raw,cap_net_admin=eip tabby/tabby
```

4.2.2 Der Systemd Service

Listing 74: tabby.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=tabby
10  #the working directory
11  WorkingDirectory=/var/elbrus/tabby/
12  #which script should be executed
13  ExecStart=/bin/bash tabby.sh
14  ...
```

Listing 75: tabby-error-handler.service.example - Die Variable 'WorkingDirectory' sowie die Variable 'User' anpassen.

```
5   ...
6   [Service]
7   Type=oneshot
8   User=tabby
9   WorkingDirectory=/var/elbrus/tabby/
10  ExecStart=/bin/bash tabby-log.sh
11  ...
```

Listing 76: Kopieren des Serviceprogrammes.

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

Listing 77: Kopieren des Errorhandlers.

```
elbrus@server:/var/elbrus$ sudo cp tabby/tabby-error-handler.service.example \
/etc/systemd/system/tabby-error-handler.service
```

Listing 78: Neuladen des 'systemctl' Deamons.

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

Listing 79: Aktivieren des Serviceprogrammes.

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

Listing 80: Starten des Serviceprogrammes.

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

5 SNMP Manager

5.1 1 - Mit Setup Script

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

```
elbrus@server:~$ cd /var/elbrus
elbrus@server:/var/elbrus$ sudo 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 should the 'snmp-manager' be stored [/var/elbrus/snmp-manager]:
Where is the shared config stored [/var/elbrus/shared/.config]:

Should the log be stored at '/var/elbrus/shared/log' ?
Should the 'snmp-manager' be stored at '/var/elbrus/snmp-manager'?
Is the shared config stored at '/var/elbrus/shared/.config' ? (y/n/exit) y

Success! .env file was created
Success! systemd service was automatically deployed,

Installing dependencies ...

...

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

5.2 2 - Ohne Setup Script

5.2.1 Umgebung Konfigurieren

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

```
1  #global
2  SHAREDCONFIG=/var/elbrus/shared/.config
3
4  #paths
5  LOGFILEDIR=/var/elbrus/shared/log
```

5.2.2 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-schedule.timer\
.example /etc/systemd/system/snmp-manager-schedule.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-schedule.timer
```

Listing 89: Starten des Zeitplanungsprogrammes.

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

6 SSH Manager

6.1 1 - Mit Setup Script

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

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

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

Do you want to store the config files at '/var/elbrus/shared/ssh-configs'?
Do you want to store the 'ssh-manager' at '/var/elbrus/ssh-manager'?
Is the shared config stored at '/var/elbrus/shared/.config'? (y/n/exit) y

info: Be aware if you skip you will have to enter the values manually in
 '/var/elbrus/ssh-manager/.env'
Do you want to configure the jumphost settings (y/skip/exit): y
info: While configuring the credentials you can choose if you want to configure
) only the ssh-key   (leave password blank)   <-- only ssh keyfile is used to connect
) only the password  (leave ssh-key blank)    <-- only password is used to connect
) both                                        <-- keyfile is used with password as
 passphrase


What is your Jumpserver IP (e.g. 1.2.3.4): 1.2.3.4
On which Port is the Jumpserver listening (e.g. 22): 22
What is your Jumpserver Username (e.g. elbrus): elburs
What is your Jumpserver Password (e.g. ******): elbrus
Where is your ssh-key located (e.g. /var/elbrus/shared/elbrus_key):
Please check the following settings:
HOST: 1.2.3.4
PORT: 22
USER: elburs
PASSWORD: elbrus
SSH-KEY:
are these your settings? (y/n/exit) y
Success! .env file was created, Please fill in the unfilled values.
Success! systemd service was automatically deployed,
info: installing dependencies.
info: installed dependencies.


Success! .env file was created
Success! systemd service was automatically deployed,

Do you want to run the setup script? (y/n/exit) y

...

info: created config folder!
Installing dependencies ...

...

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

6.2 2 - Ohne Setup Script

6.2.1 Umgebung Konfigurieren

Listing 91: 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 92: Ausführen des Scripts zur Initialisierung des VCS Verzeichnisses.

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

Listing 93: Installieren von fehlenden python3 Packages.

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

6.2.2 Der Systemd Service

Listing 94: 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 95: Kopieren des Serviceprogrammes.

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

Listing 96: 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 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 ssh-manager.service
```

Listing 99: Aktivieren des Zeitplanungsprogrammes.

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

Listing 100: Starten des Zeitplanungsprogrammes.

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

7 Geo Session finder

7.1 1 - Mit Setup Script

Listing 101: Ausführen des 'install.sh' Scripts.

```
elbrus@server:~$ cd /var/elbrus
elbrus@server:~$ sudo 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 should the 'geo-session-finder' be stored [/var/elbrus/geo-session-finder]:
Where is the shared config stored [/var/elbrus/shared/.config]:

Should the log be stored at '/var/elbrus/shared/log' ?
Should the 'geo session finder' be stored at '/var/elbrus/geo-session-finder'?
Is the shared config stored at '/var/elbrus/shared/.config' ? (y/n/exit) y

Success! .env file was created
Success! systemd service was automatically deployed,

Installing dependencies ...

...

elbrus@server:~$
```

7.2 2 - Ohne Setup Script

Listing 102: Installieren der Abhängigkeiten.

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

7.2.1 Umgebung Konfigurieren

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

```
1  #global
2  SHAREDCONFIG=/var/elbrus/shared/.config
3
4  #paths
5  LOGFILEDIR=/var/elbrus/shared/log
```

7.2.2 Der Systemd Service

Listing 104: geo-session-finder.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/geo_session_finder/src
12   #which script should be executed
13   ExecStart=/bin/bash elb-geo-session-finder.sh
14   ...
```

Listing 105: 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 106: 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 107: Neuladen des 'systemctl' Deamons.

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

Listing 108: Aktivieren des Serviceprogrammes.

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

Listing 109: Aktivieren des Zeitplanungsprogrammes.

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

Listing 110: Starten des Zeitplanungsprogrammes.

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

8 Uptime Monitor

8.1 Umgebung Konfigurieren

8.1.1 1 - Mit Setup Script

Listing 111: Ausführen des 'install.sh' Scripts.

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

Where is the shared config stored [/var/elbrus/shared/.config]:
Where should the 'uptime-monitor' be stored [/var/elbrus/uptime-monitor]:

Is the shared config stored at '/var/elbrus/shared/.config'?
Should the 'uptime-monitor be stored at '/var/elbrus/uptime-monitor'? (y/n/exit) y

Success! .env file was created
Success! systemd service was automatically deployed,

elbrus@server:~$
```

8.1.2 2 - Ohne Setup Script

Listing 112: Anhand von '.env.example' eigene '.env' Datei anlegen.

```
1  #global
2  SHAREDCONFIG=/var/elbrus/shared/.config
3
4  #config
5  # Initial pings to see if device is alive
6  INITIALPING=1
7  # Pings to get the availability statistic
8  STATISTICPING=10
```

8.2 Der Systemd Service

Listing 113: uptime_monitor.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/uptime_monitor
12   #which script should be executed
13   ExecStart=/bin/bash uptime_monitor.sh
14   ...
```

Listing 114: Kopieren des Serviceprogrammes.

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

Listing 115: Kopieren des Zeitplanungsprogrammes.

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

Listing 116: Neuladen des 'systemctl' Deamons.

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

Listing 117: Aktivieren des Serviceprogrammes.

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

Listing 118: Aktivieren des Zeitplanungsprogrammes.

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

Listing 119: Starten des Zeitplanungsprogrammes.

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

9 office365

9.1 Umgebung Konfigurieren

9.1.1 1 - Mit Setup Script

Listing 120: Ausführen des 'install.sh' Scripts.

```
elbrus@server:~$ cd /var/elbrus
elbrus@server:~$ sudo bash office365-analyzer/src/install.sh
Do you want to proceed with setup of the 'office365-analyzer'? (y/n) y

Where should the 'office365-analyzer' be stored (dir) [/var/elbrus/office365-analyzer]:
Where should the log be stored (dir) [/var/elbrus/shared/log]:
Where is the shared config stored [/var/elbrus/shared/.config]:

Should the 'office365-analyzer' be stored at '/var/elbrus/office365-analyzer' ?
Should the log be stored at '/var/elbrus/shared/log' ?
Is the shared config stored at '/var/elbrus/shared/.config' ? (y/n/exit) y

Success! .env file was created
Success! systemd service was automatically deployed,

info: installing dependencies.

...

info: installed dependencies.

elbrus@server:~$
```

9.1.2 2 - Ohne Setup Script

Listing 121: Anhand von '.env.example' eigene '.env' Datei anlegen.

```
1  #global
2  SHAREDCONFIG=/var/elbrus/shared/.config
3
4  #paths
5  LOGFILEDIR=/var/elbrus/shared/log
6
7  #ms url
8  MS_URL=https://endpoints.office.com/endpoints/worldwide?clientrequestid=
    b10c5ed1-bad1-445f-b386-b919946339a7
```

9.2 Der Systemd Service

Listing 122: uptime_monitor.service.example - Die Variable 'WorkingDirectory' sowie die Variable 'User' anpassen.

```
4    ...
5    #job is starting immediatly after the start action has been called
6    Type=simple
7    #the user to execute the script
8    User=elbrus
9    #the working directory
10   WorkingDirectory=/var/elbrus/office365-analyzer/src
11   #which script should be executed
12   ExecStart=/bin/bash elb-office365-get-endpoints.sh
13   ...
```

Listing 123: 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 124: 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 125: Neuladen des 'systemctl' Deamons.

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

Listing 126: Aktivieren des Serviceprogrammes.

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

Listing 127: Aktivieren des Zeitplanungsprogrammes.

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

Listing 128: Starten des Serviceprogrammes & Zeitplanungsprogrammes.

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

10 API

10.1 1 - Mit Setup Script

Listing 129: Ausführen des 'install.sh' Scripts.

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

Where should the 'api' be stored [/var/elbrus/api]:
Where is the shared config stored [/var/elbrus/shared/.config]:

Is the shared config stored at '/var/elbrus/shared/.config'?
Do you want to store the 'api' at '/var/elbrus/api'? (y/n/exit) y

info: installing dependencies

...

On which Port should the api run ['3000']? 3000
On which URL is the webinterface running? (e.g. http://1.2.3.4:80/) http://1.2.3.4:80/

Is your webinterface running on 'http://1.2.3.4:80/'?
Should the api run on port '3000'? (y/n/exit) y

info: Be aware if you skip you will have to enter the values manually in
'/var/elbrus/api/.env' and run 'pm2 restart elb-api'
Do you want to configure the api email settings (y/skip/exit): y
What is your SMTP Host (e.g. smtp.gmail.com): smtp.gmail.com
What is your SMTP Port (e.g. 465): 465
What is your SMTP Username (e.g. elbrus): elbrus
What is your SMTP Password (e.g. ******): elbrus
Who should send the email (e.g. info@gmail.com): info@gmail.com
Which sender should be displayed (e.g ELBRUS SYSTEM): ELBRUS SYSTEM
Please check the following settings:
HOST: smtp.gmail.com
PORT: 465
USER: elbrus
PASSWORD: elbrus
EMAIL: info@gmail.com
DISPLAYNAME: ELBRUS SYSTEM
are these your settings? (y/n/exit) y

Success! .env file was created

...

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

10.2 2 - Ohne Setup Script

10.3 Voraussetzungen

Listing 130: Nachinstallieren der Abhängigkeiten.

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

10.4 Umgebung Konfigurieren

Listing 131: 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=
```

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.

10.5 Inbetriebnahme

Listing 132: Starten der API.

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

11 Webinterface

11.1 1 - Mit Setup Script

Listing 133: Ausführen des 'install.sh' Scripts.

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

        Where should the 'webinterface' be stored [/var/elbrus/webinterface]:

        Do you want to store the 'webinterface' at '/var/elbrus/webinterface'? (y/n/exit)
         y

        info: installing dependencies:

        info: installing nginx
        info: installed nginx

        What is your baseurl (eg. 'http://1.2.3.4:3000/v1/' )? http://1.2.3.4/3000/v1/

        Is your baseurl 'http://1.2.3.4/3000/v1/'? (y/n/exit) y


        Success! .env file was created.


        info: building frontend

        > elbrus-webinterface@0.1.0 build /var/elbrus/webinterface
        > vue-cli-service build


        .:  Building for production...

        ...

        info: built frontend
        info: overwriting nginx config. Your previous nginx config is stored as
        'nginx.conf.be'.
        info: starting nginx.
        Success: webinterface is now running at 'http://your-ip:80/'.
elbrus@server:/var/elbrus$
```

11.2 2 - Ohne Setup Script

11.2.1 Umgebung Konfigurieren

Listing 134: Anhand von '.env.example' eigene '.env' Datei anlegen.

```
1    VUE_APP_BASEURL=http://localhost:3000/v1/
```

Listing 135: Compilieren des Webinterface.

```
elbrus@server:~$ cd /var/elbrus/webinterface
elbrus@server:/var/elbrus/webinterface$ sudo npm run build
```

Listing 136: Kopieren des kompilierten Webinterfaces in der Ordner des Webservers.

```
elbrus@server:/var/elbrus/webinterface$ sudo cp -r dist/ /usr/share/nginx/html/
```

Listing 137: /etc/nginx/nginx.conf - Ändern der folgenden Zeilen.

```
- server {
- listen 80 default_server;
- listen [::]:80 default_server;
- server_name _;
- root /usr/share/nginx/html;
- # Load configuration files for the default server block.
- include /etc/nginx/default.d/*.conf;
- location / {
- }

+ server {
+ listen 80 default_server;
+ listen [::]:80 default_server;
+ server_name _;
+ root /usr/share/nginx/html/dist;
+ # Load configuration files for the default server block.
+ include /etc/nginx/default.d/*.conf;
+ location / {
+ }

- error_page 404 404.html;
- location = 40x.html {
- }

+ error_page 404 = @elbrus;
+ location @elbrus {
+ root /usr/share/nginx/html/dist;
+ try_files $uri /index.html =502;
+ }
```

11.3 Der Systemd Service

Listing 138: Aktivieren des Webservers.

```
elbrus@server:/var/elbrus/webinterface$ sudo systemctl enable nginx
```

Listing 139: Starten des Webservers.

```
elbrus@server:/var/elbrus/webinterface$ sudo systemctl systemctl start nginx
```