Install BloodHound CE under Kali Linux 2024.4

breachar.medium.com/install-bloodhound-ce-under-kali-linux-2024-4-2a68feebdb62

breachar December 30, 2024

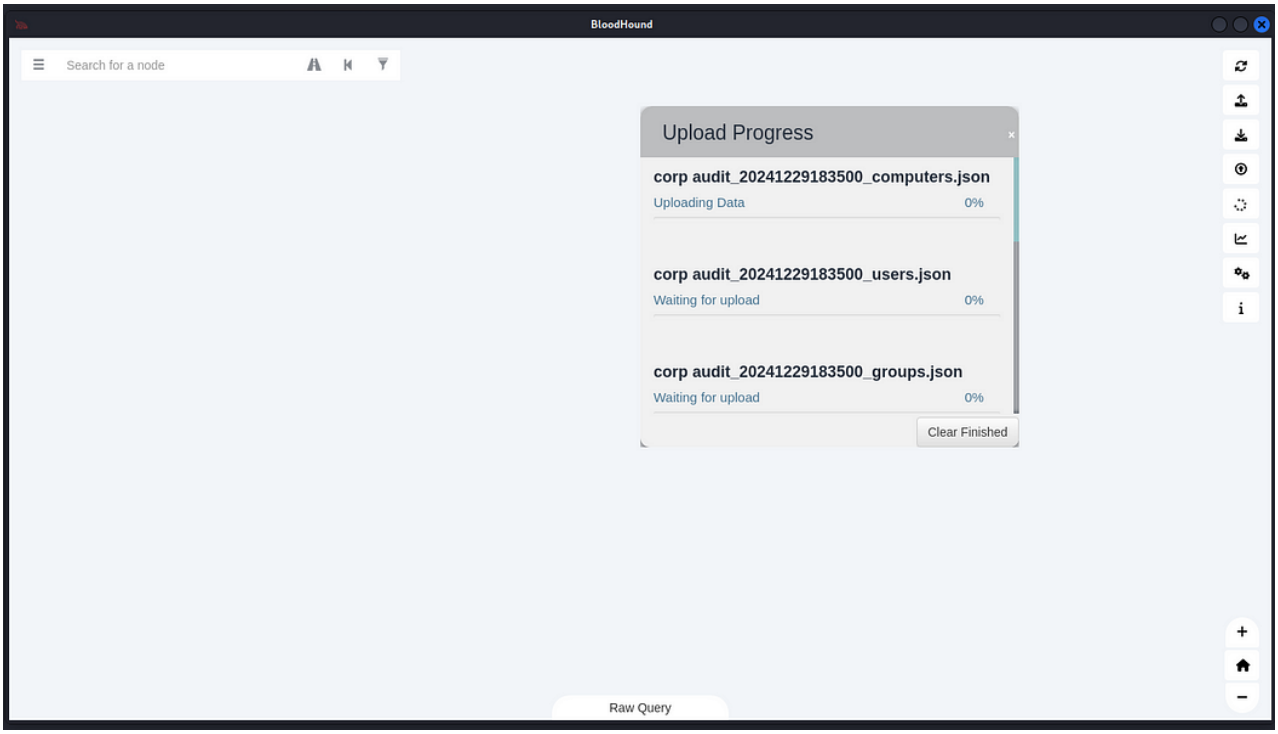
breachar

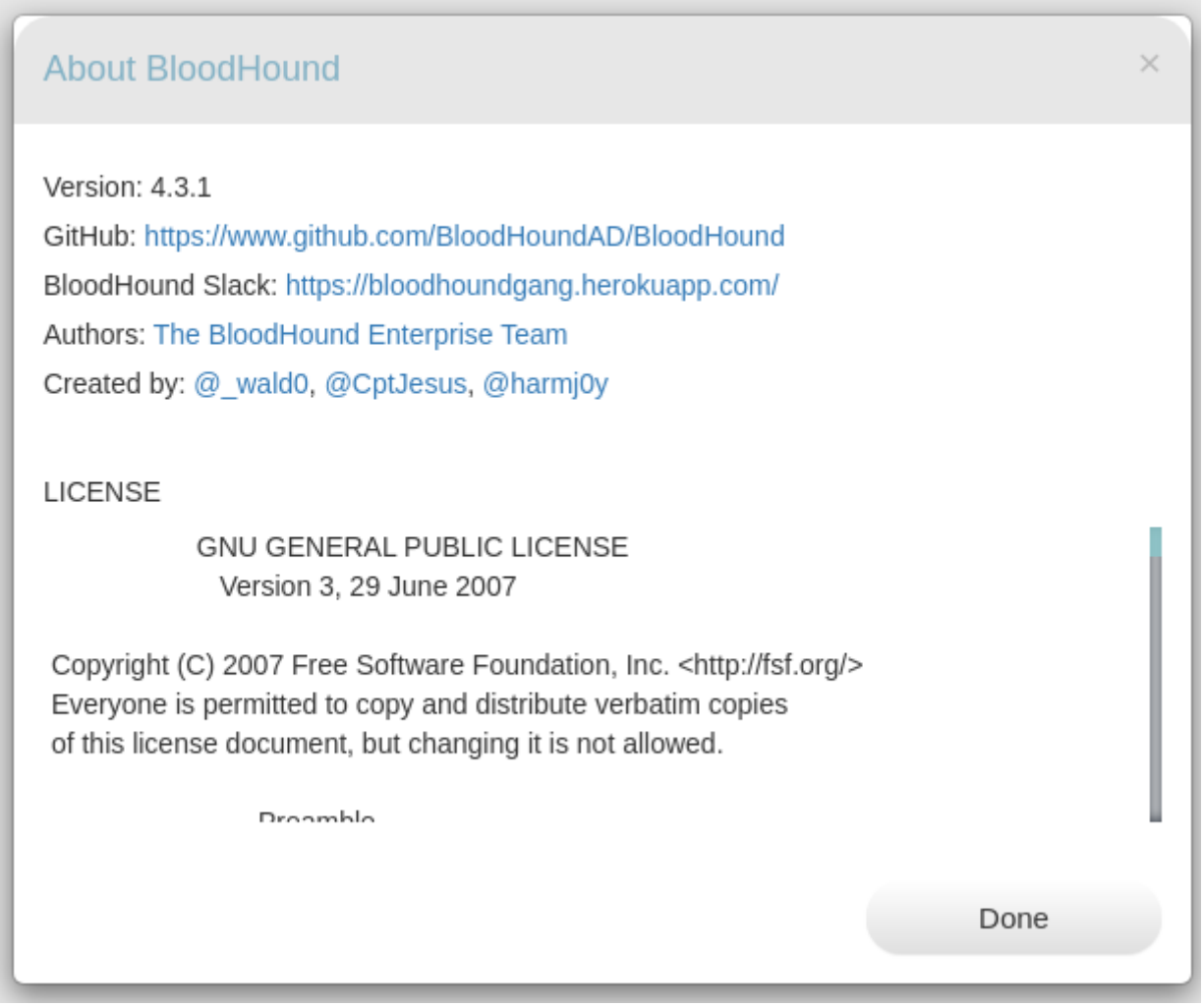
Kali Linux 2024.4 comes with BloodHound version 4.3.1. Unfortunately, the error that BloodHound always hangs at 0% when importing data occurs very frequently.

Bug Report: BloodHound Upload Stuck at 0% · Issue #724 · SpecterOps/BloodHound-Legacy

Description: When attempting to upload SharpHound JSON files into BloodHound, the upload process gets stuck at 0%, and…

github.com



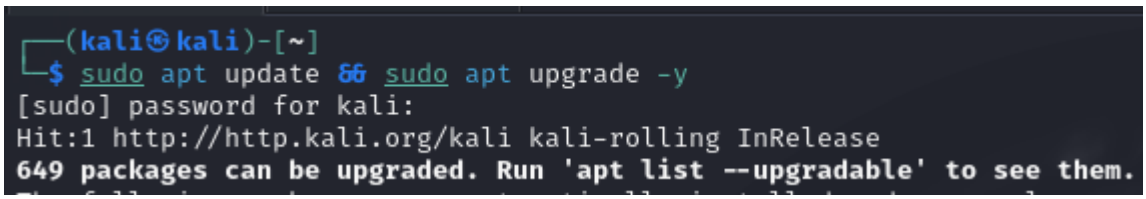


If you would like to know what requirements you need to operate BloodHound CE, you can read about them here: https://github.com/SpecterOps/BloodHound

Lets begin

Firstly, we update our Kali:
This step may take longer if you are using a new Kali VM / freshly installed Kali or have not upgraded for a while.

```
sudo apt update && sudo apt upgrade -y
```



Then we install docker.io:

```
sudo apt install docker.io
```

```
┌──(kali㉿kali)-[~]
└─$ sudo apt install docker
Completing package
docker-clean      docker-cli        docker-doc        docker.io         docker-registry
```

Now we need to install docker-composer:
Unfortunately, this does not work with apt.

You can load docker compose in the terminal as follows:

```
sudo curl -L "https://github.com/docker/compose/releases/download/v2.32.1/docker-compose--" -o /usr/local/bin/docker-compose
```

```
┌──(kali㉿kali)-[~]
└─$ sudo curl -L "https://github.com/docker/compose/releases/download/v2.32.1/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100 61.6M  100 61.6M    0     0  10.9M      0  0:00:05  0:00:05 --:--:-- 12.7M
```

> At the time of writing, this was the current version. If this is no longer the case, you can search for the latest docker compose version here: https://github.com/docker/compose/releases

Next, we make docker-compose executable

```
sudo  +x /usr/local/bin/docker-compose
```

```
┌──(kali㉿kali)-[~]
└─$ sudo chmod +x /usr/local/bin/docker-compose
```

Finally, we check the version of docker-compose:

```
docker-compose --version
```

```
┌──(kali㉿kali)-[~]
└─$ docker-compose --version
Docker Compose version v2.32.1
```

Now we install BloodHound

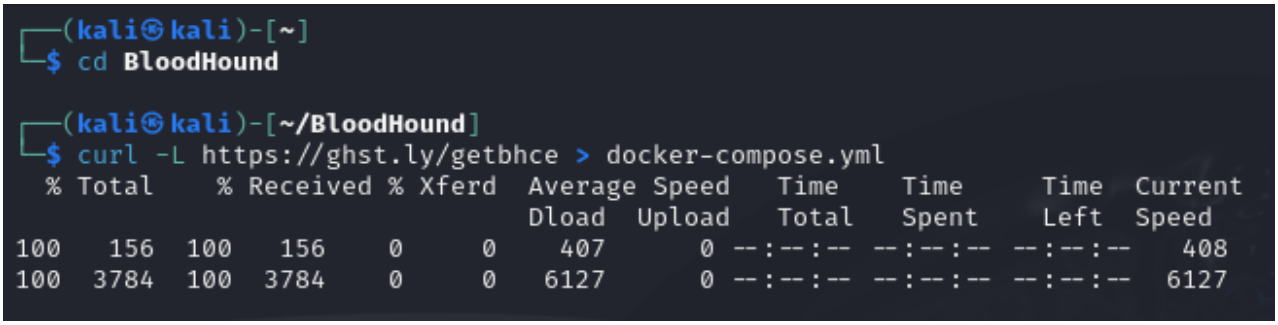
First create a new folder called BloodHound:

```
BloodHound
```

```
┌──(kali㉿kali)-[~]
└─$ mkdir BloodHound

┌──(kali㉿kali)-[~]
└─$ ls
BloodHound  Desktop  Documents  Downloads  Music  Pictures  Public  sharedFolder  Templates  Videos
```

Next, we cd into the newly created folder and download the BloodHound docker-compose.yml:

```
curl -L https://ghst.ly/getbhce > docker-compose.yml
```



My docker-compose.yml has the following content:

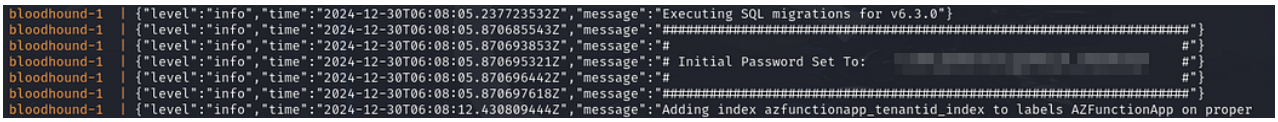
```
# Copyright 2023 Specter Ops, Inc.
#
# Licensed under the Apache License, Version 2.0
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#
# SPDX-License-Identifier: Apache-2.0



services:
  app-db:
    image: docker.io/library/postgres:16
    environment:
      - PGUSER=${POSTGRES_USER:-bloodhound}
      - POSTGRES_USER=${POSTGRES_USER:-bloodhound}
      - POSTGRES_PASSWORD=${POSTGRES_PASSWORD:-bloodhoundcommunityedition}
      - POSTGRES_DB=${POSTGRES_DB:-bloodhound}
# Database ports are disabled by default. Please change your database password to
something secure before uncommenting
# ports:
#   - 127.0.0.1:${POSTGRES_PORT:-5432}:5432
    volumes:
      - postgres-data:/var/lib/postgresql/data
    healthcheck:
test:
        [
"CMD-SHELL",
"pg_isready -U  -d  -h 127.0.0.1 -p 5432"
        ]
      interval: 10s
timeout: 5s
      retries: 5
      start_period: 30s



  graph-db:
    image: docker.io/library/neo4j:4.4
    environment:
      - NEO4J_AUTH=${NEO4J_USER:-neo4j}/${NEO4J_SECRET:-
bloodhoundcommunityedition}
      - NEO4J_dbms_allow__upgrade=${NEO4J_ALLOW_UPGRADE:-true}
# Database ports are disabled by default. Please change your database password to
something secure before uncommenting
    ports:
      - 127.0.0.1:${NEO4J_DB_PORT:-7687}:7687
      - 127.0.0.1:${NEO4J_WEB_PORT:-7474}:7474
    volumes:
```

```
      - ${NEO4J_DATA_MOUNT:-neo4j-data}:/data
    healthcheck:
test:
        [
"CMD-SHELL",
"wget -O /dev/null -q http://localhost:7474 || exit 1"
        ]
      interval: 10s
timeout: 5s
      retries: 5
      start_period: 30s



  bloodhound:
    image: docker.io/specterops/bloodhound:${BLOODHOUND_TAG:-latest}
    environment:
      -
bhe_disable_cypher_complexity_limit=${bhe_disable_cypher_complexity_limit:-false}
      - bhe_enable_cypher_mutations=${bhe_enable_cypher_mutations:-false}
      - bhe_graph_query_memory_limit=${bhe_graph_query_memory_limit:-2}
      - bhe_database_connection=user=${POSTGRES_USER:-bloodhound}
password=${POSTGRES_PASSWORD:-bloodhoundcommunityedition} dbname=${POSTGRES_DB:-
bloodhound} host=app-db
      - bhe_neo4j_connection=neo4j://${NEO4J_USER:-neo4j}:${NEO4J_SECRET:-
bloodhoundcommunityedition}@graph-db:7687/
### Add additional environment variables you wish to use here.
### For common configuration options that you might want to use environment
variables for, see `.env.example`
### example: bhe_database_connection=${bhe_database_connection}
### The left side is the environment variable you're setting for bloodhound, the
variable on the right in `${}`
### is the variable available outside of Docker
    ports:
### Default to localhost to prevent accidental publishing of the service to your
outer networks
### These can be modified by your .env file or by setting the environment
variables in your Docker host OS
      - ${BLOODHOUND_HOST:-127.0.0.1}:${BLOODHOUND_PORT:-8080}:8080
### Uncomment to use your own bloodhound.config.json to configure the application
# volumes:
#   - ./bloodhound.config.json:/bloodhound.config.json:ro
    depends_on:
      app-db:
        condition: service_healthy
      graph-db:
        condition: service_healthy

                                                    volumes:  neo4j-data:
postgres-data:
```

Now we run

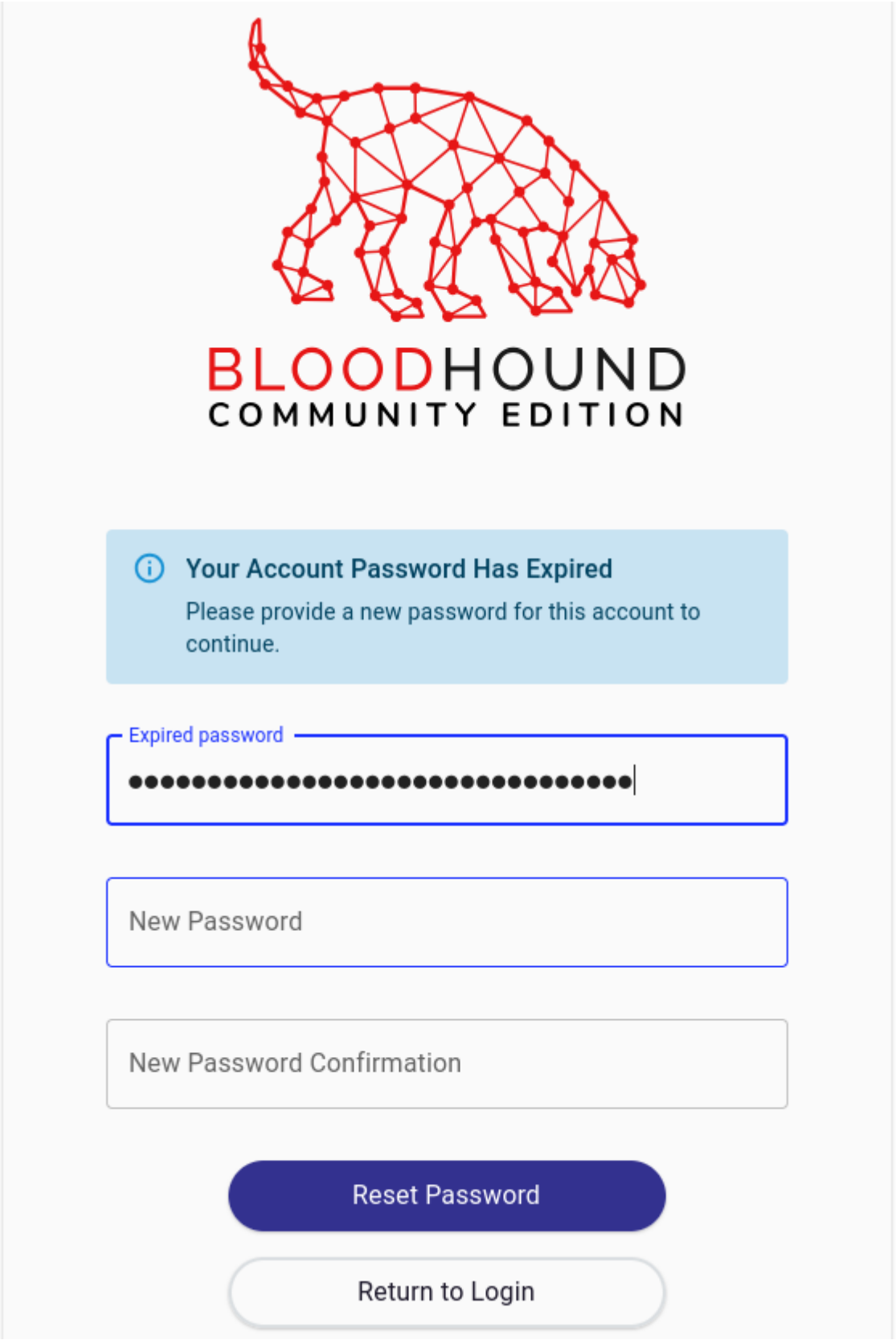
```
sudo docker-compose pull && sudo docker-compose up
```

BloodHound displays the randomly generated initial password in the terminal:

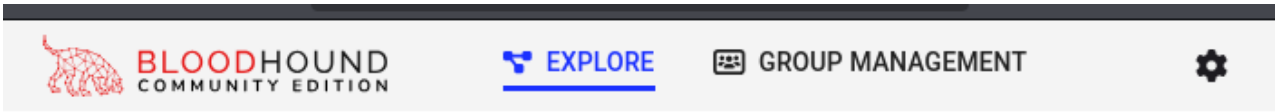


You can now log in to BloodHound at http://localhost:8080/ui/ with the username **admin** and **your password**:

After the first login you have to change your password:



Done! BloodHound is now successfully installed. Now all you have to do is import your data and you're ready to go.



No Data Available

It appears that no data has been uploaded yet. See our Data Collection documentation to learn how to start collecting data.
If you have files available from a SharpHound or AzureHound collection, please visit the File Ingest page to begin uploading your data.
If you want to test BloodHound with sample data, you may download some from our GitHub Sample Collection GitHub page.