



PHederationTM

Local installation instructions

28 November 2019

CONTEXT

Phederation, a network of centers, is being established with the mutual goal of being able to track treatment outcomes in those centres and generate RWD to create valuable RWE. Each member of the network will sign a contract with Janssen agreeing to conduct studies on their RWE data and share the results.

Phederation is a federated data network of centers to support real world evidence research with a focus on outcomes management. The focus is on sharing analysis methods and results, not the granular data itself. To accommodate this objective, every data source is converted to the same data model (aka a Common Data Model - CDM). This is done locally at the participating centers, and, importantly, the data stays locally.

Since all centers are using the same model, a data analysis script can be defined centrally and sent to each of the participating centers for performing the analysis against their local data. The summary results from the individual centers can then be shared with participants through the central database, and summary analysis and visualizations can be done in the central workbench.

REQUIREMENTS

Hardware

Modern dual core processor (or better)
16 GB RAM (or more)
100 GB free disk space (or more)

Operating system

Windows 10, MacOS or Linux (Ubuntu, CentOS, Debian, ...)

Docker

- Windows: <https://docs.docker.com/docker-for-windows/install/>
- MacOS: <https://docs.docker.com/docker-for-mac/install/>
- Linux: <https://docs.docker.com/install/linux/docker-ce/ubuntu/>

Assign 2 or more CPU's, 8 GB of RAM and 100 GB of disk space to Docker in Docker Desktop.

On Linux Docker compose (v1.24 or higher) should be installed separately.

Docker images for PHederation

A Docker Hub account with read access on the HONEUR Docker image repository (<https://hub.docker.com/u/honeur>) is required.

Please create a Docker hub account or use an existing account and request access by sending a mail to Michel Van Speybroeck (mvspeybr@its.jnj.com)

INSTALLATION COMPONENTS

The PHederation local installation consists of the following components.

PostgreSQL database

The Postgres image for PHederation contains the OHDSI database with 4 predefined schema's:

omopcdm

OMOP CDM (v5.3.1) schema containing the vocabulary.

webapi

Schema used by the backend of Atlas (WebAPI) to store application data

results

Schema used by the backend of Atlas (WebAPI) to store result data

scratch

Empty schema that can be freely used

See <https://www.ohdsi.org/data-standardization/the-common-data-model/> for more info about OMOP CDM

User Management

Optional component that can be used to enable security based on a local user database (as an alternative for integrating with an LDAP server).

User Management is a simple web application to create users, roles and permissions and assign permissions to roles and roles to users.

More information can be found on the PHederation portal:

https://portal.phederation.org/group/phederation/knowledge/-/knowledge_base/user-documentation/user-management-at-the-remote-site

Atlas / WebAPI

Atlas is used to explore data in the OMOP CDM database. It's also used to import cohort definitions that are shared, run them on the local data and export the inclusion results back to the PHederation central site.

More information can be found on the PHederation portal

https://portal.phederation.org/group/phederation/knowledge/-/knowledge_base/user-documentation/atlas-and-webapi-at-the-remote-site) and on the OHDSI website (<https://github.com/OHDSI/Atlas/wiki>)

Zeppelin

Zeppelin is a notebook server that is used to import and run notebooks that are shared. The results after running the analytics can be uploaded to the central side (manually or automatically).

More information can be found on the PHederation portal:

https://portal.phederation.org/group/phederation/knowledge/-/knowledge_base/user-documentation/zeppelin-at-the-remote-site and on the Zeppelin website <https://zeppelin.apache.org/>

INSTALLATION OPTIONS

The PHederation local components can be installed with security enabled or disabled.

When security is disabled, Atlas and Zeppelin can be accessed without login.

When security is required, the installer can choose to integrate with a local LDAP server (if available) or use the security database that comes out of the box with the secure setup.

INSTALLATION EFFORT

Installing the local PHederation components doesn't take more than 15 minutes once a (physical or virtual) host machine - with Docker installed on it - is available.

INSTALLATION STEPS

Standard version

1. Open a terminal window (Command Prompt on Windows)

2. Download the installation script

- a. For Windows

```
curl -L
https://raw.githubusercontent.com/solventrix/Honeur-Setup/master/standard/p
hederation/start-phederation.cmd --output start-phederation.cmd
```

b. For Linux and Mac

```
curl -L
https://raw.githubusercontent.com/solventrix/Honeur-Setup/master/standard/p
hederation/start-phederation.sh --output start-phederation.sh && chmod +x
start-phederation.sh
```

3. Run `start-phederation.sh` (on Linux or Mac) or `start-phederation.cmd` (on Windows)
4. The script will prompt to enter the username and password of your Docker account. Make sure this docker account has read access on the HONEUR images. If you are already logged in to Docker, the script will automatically use the existing credentials.
5. Press Enter to remove the existing webapi, zeppelin and postgres container (if present).
6. After the script has downloaded the docker-compose.yml file, it will prompt you to enter a Fully Qualified Domain Name (FQDN) or IP Address of the host machine. Atlas will only be accessible on the host machine (via localhost) if you accept the default 'localhost' value.
7. The script will prompt you to enter the directory where the Zeppelin notebooks and log files should be stored.
8. The Postgres database, Atlas, Zeppelin will be downloaded and will be started as Docker containers.

Secure version

1. Open a terminal window (Command Prompt on Windows)
2. Download the installation file

a. For Windows

```
curl -L
https://raw.githubusercontent.com/solventrix/Honeur-Setup/master/secure/phe
deration/start-phederation-secure.cmd --output start-phederation.cmd
```

b. For Linux and Mac

```
curl -L
https://raw.githubusercontent.com/solventrix/Honeur-Setup/master/secure/phe
deration/start-phederation-secure.sh --output start-phederation.sh && chmod
+x start-phederation.sh
```

3. Run `start-phederation.sh` (on Linux or Mac) or `start-phederation.cmd` (on Windows)

-
4. The script will prompt you to enter the username and password of your Docker account. Make sure this Docker account has read access on the HONEUR images. If you are already logged in to docker, the script will automatically use the existing credentials.
 5. Press Enter to remove existing webapi, zeppelin and postgres containers (if present).
 6. After the script has downloaded the docker-compose.yml file, it will prompt you to enter a Fully Qualified Domain Name (FQDN) or IP Address of the host machine. Atlas and User Management will only be accessible on the host machine (via localhost) if you accept the default 'localhost' value.
 7. The script will prompt you to enter the directory of where the Zeppelin notebooks and log files should be stored.
 8. The script will prompt to choose between an LDAP or a JDBC connection for authentication. If you choose LDAP, the script will prompt to enter the following LDAP properties:
 - a. security ldap.url: ldap://ldap.forumsys.com:389
 - b. security ldap.system.username: cn=read-onlyadmin,dc=example,dc=com
 - c. security ldap.system.password: password
 - d. security ldap.baseDn: dc=example,dc=com
 - e. security ldap.dn: uid={0},dc=example,dc=com
 1. The script will prompt to enter the following User Management properties:
 - a. usermgmt admin username: admin
 - b. usermgmt admin password: admin
 2. The Postgres database, Atlas, Zeppelin and a User Mgmt applications will be downloaded and will be started as Docker containers

Installation Notes

Downloading the Docker images can take some time.

The following ports are used on the host machine:

- Postgres: 5444
- Atlas/WebAPI: 8080
- User management: 8081
- Zeppelin: 8082

Please contact the PHederation support team if these ports are not available or should be changed.
The database data will be stored in a Docker volume named “pgdata”

HOW TO ACCESS THE PHEDERATION INSTALLATION

After the installation is completed. All HONEUR components are available and can be accessed as follows:

Postgres DB

1. Open a SQL Client (e.g.: <https://dbeaver.io/>)
2. Connect to the Postgres database running on:
Host: hostname or IP address of the host machine
Port: **5444**
Database: **OHDSI**
Username: phederation, Password: phederation (to select, update and delete data) or
Username: phederation_admin, Password: phederation_admin (to make DB changes)
Note: *Please change the default password of the phederation and phederation_admin DB accounts
Changing the password of the phederation and phederation_admin account will not have impact on
the correct working of the installation.*
3. The “omopcdm”, “results”, “webapi” and scratch schemas can be accessed in the database.

Atlas

1. Open a modern browser (e.g. Google Chrome)
2. Navigate to **[http://\[hostname or IP of host machine\]:8080/atlas](http://[hostname or IP of host machine]:8080/atlas)**
3. The Atlas home page will be displayed.

User Management (optional)

1. Open a modern browser (e.g. Google Chrome)
2. Navigate to **[http://\[hostname or IP of host machine\]:8081/usermgmt](http://[hostname or IP of host machine]:8081/usermgmt)**
3. The user management home page will be displayed.

Zeppelin

1. Open a modern browser (e.g. Google Chrome)
2. Navigate to **[http://\[hostname or IP of host machine\]:8082](http://[hostname or IP of host machine]:8082)**

3. The Zeppelin home page will be displayed.

RUNNING THE ETL

The ETL code to map the source data to the OMOP Common Data Model (CDM) will be custom build for each data center. The script to execute the ETL will be provided.

POST ETL INSTALLATION STEPS

Add constraints and indexes to the OMOP CDM tables

After the ETL is successfully executed, it's recommended to add the constraints and indexes to the OMOP CDM tables. It will improve the performance and reduce the risk of corrupt data in the database.

Installation steps

1. Open a terminal window (Command Prompt on Windows)
2. Download the installation file

- a. For Windows

```
curl -L
https://raw.githubusercontent.com/solventrix/Honeur-Setup/master/start-omop
cdm-indexes-and-constraints.cmd --output
start-omopcdm-indexes-and-constraints.cmd
```

- b. For Linux and Mac

```
curl -L
https://raw.githubusercontent.com/solventrix/Honeur-Setup/master/start-omop
cdm-indexes-and-constraints.sh --output
start-omopcdm-indexes-and-constraints.sh && chmod +x
start-omopcdm-indexes-and-constraints.sh
```

1. Run `start-omopcdm-indexes-and-constraints.sh` (on Linux or Mac) or
`start-omopcdm-indexes-and-constraints.cmd` (on Windows)

The program will prompt to enter the username and password of your Docker account. Make sure this Docker account has read access on the HONEUR images. If you are already logged in to Docker, the script will automatically use the existing credentials. Press Enter to remove existing omop-indexes-and-constraints container (if present). After the script has

downloaded the docker-compose.yml and setup-conf/setup.yml files, it will start creating the indexes and constraints on the OMOP CDM tables.

MAINTENANCE

Starting and stopping the Docker Containers

The Docker containers for PHederation will automatically restart when the host machine is restarted. To manually start and stop the containers, open a terminal window and run one of the following commands.

Postgres:

- **to start:** `docker start postgres` **to stop:** `docker stop postgres`

Atlas

- **to start:** `docker start webapi` **to stop:** `docker stop webapi`

User Management

- **to start:** `docker start user-mgmt` **to stop:** `docker stop user-mgmt`

Zeppelin

- **to start:** `docker start zeppelin` **to stop:** `docker stop zeppelin`

Backup and restore of the database

Backup

Download and run the script 'backup-database.sh':

```
curl -L
https://raw.githubusercontent.com/solventrix/Honeur-Setup/master/backup-database.
sh --output backup-database.sh && chmod +x backup-database.sh
```

The backup script will create a tar file name 'postgres_backup_<date_time>.tar.bz2' in the current directory. Creating the backup file can take a long time depending on the size of the database.

Copy the backup file to a save location for long term storage.

Restore

Download the script 'restore-database.sh'

```
curl -L
https://raw.githubusercontent.com/solventrix/Honeur-Setup/master/restore-database
.sh --output restore-database.sh && chmod +x restore-database.sh
```

Run the script and provide the name of the backup file as parameter. The name of the target volume can be provided as a second argument. The backup will be restored in the pgdata volume if the target volume is not provided. The backup file should be present in the folder where the script is executed.

```
./restore-database postgres_backup_<date_time>.tar.bz2
```

Hot snapshot of the database

The database volume can be copied to a new volume (with a different name) to take a snapshot of the current database state. Download the script 'clone-volume.sh'

```
curl -L
https://raw.githubusercontent.com/solventrix/Honeur-Setup/master/clone-volume.sh
--output clone-volume.sh && chmod +x clone-volume.sh
```

Run the script, provide the source volume as first parameter and the target volume as second parameter.

```
./clone-volume.sh pgdata pgdata_snapshot1
```

Accessing log files

Console logs

Docker provides real time access to the applications logs.

- Atlas: `docker logs webapi -f`
- Zeppelin: `docker logs zeppelin -f`
- Postgres: `docker logs postgres -f`

Logs files

The log files of Zeppelin are available on the host machine where the Docker containers are running. The location of the logs files can be configured during the setup. The default location is `/zeppelin/logs`

The logs files of Atlas are available within the Docker container and can be copied to the host machine if needed. The logs files are available in the `/usr/local/tomcat/logs` directory.

Execute the following commands to inspect the log files:

- `docker exec -it webapi /bin/bash`
- `cd logs`
- `tail -200 webapi.log` or `tail catalina.<yyyy-mm-dd>.log`

Execute the following command to copy all log files to a logs directory below the current directory on host machine:

-
- `docker cp webapi:/usr/local/tomcat/logs .`

FURTHER READING

More documentation is available in the user documentation on the PHederation portal:

<https://portal.phederation.org/group/phederation/knowledge>

ADDENDUM: SECURITY SCAN OF DOCKER CONTAINERS

The following images were scanned for vulnerabilities:

- `honeur/postgres`
- `honeur/webapi-atlas`
- `honeur/user-mgmt`
- `honeur/zeppelin`

Vulnerability scanner

Docker bench security is used for the vulnerability scan on the images and containers.

<https://github.com/docker/docker-bench-security>

The Docker Bench for Security is a script that checks for dozens of common best-practices around deploying Docker containers in production. The tests are all automated and are inspired by the CIS Docker Community Edition Benchmark v1.1.0.

Results

Command

```
docker run -it \
--net host --pid host --usersns host \
--cap-add audit_control \
-e DOCKER_CONTENT_TRUST=$DOCKER_CONTENT_TRUST \
-v /var/lib:/var/lib \
-v /var/run/docker.sock:/var/run/docker.sock \
-v /usr/lib/systemd:/usr/lib/systemd \
-v /etc:/etc --label docker_bench_security \
docker/docker-bench-security \
-t honeur/postgres:9.6-5.3.1,honeur/webapi-atlas:2.6.0-1.1.4-secure,honeur/zeppelin:0.8.0,honeur/user-mgmt:1.0.0
```

Output

```
# -----
# Docker Bench for Security v1.3.4
#
# Docker, Inc. (c) 2015-
#
# Checks for dozens of common best-practices around deploying Docker containers in production.
# Inspired by the CIS Docker Community Edition Benchmark v1.1.0.
# -----
Initializing Mon Apr  8 14:35:32 UTC 2019

Looking for image honeur/postgres:9.6-5.3.1
Looking for image honeur/webapi-atlas:2.6.0-1.1.4-secure
Looking for image honeur/zeppelin:0.8.0
Looking for image honeur/user-mgmt:1.0.0
```

[INFO] 4 - Container Images and Build File

```
[WARN] 4.1 - Ensure a user for the container has been created
[WARN] * Running as root: zeppelin
[NOTE] 4.2 - Ensure that containers use trusted base images
[NOTE] 4.3 - Ensure unnecessary packages are not installed in the container
[NOTE] 4.4 - Ensure images are scanned and rebuilt to include security patches
[WARN] 4.5 - Ensure Content trust for Docker is Enabled
[PASS] 4.6 - Ensure HEALTHCHECK instructions have been added to the container image
[INFO] 4.7 - Ensure update instructions are not use alone in the Dockerfile
[INFO] * Update instruction found: [honeur/postgres:9.6-5.3.1 honeur/postgres:latest]
[INFO] * Update instruction found:
[honneur/webapi-atlas:2.6.0-1.1.4-secure honeur/webapi-atlas:latest honeur/webapi:2.6.0-1.1.4-secure]
[INFO] * Update instruction found: [honeur/zeppelin:0.8.0 honeur/zeppelin:latest]
[INFO] * Update instruction found: [honeur/user-mgmt:1.0.0 honeur/user-mgmt:latest]
[NOTE] 4.8 - Ensure setuid and setgid permissions are removed in the images
[INFO] 4.9 - Ensure COPY is used instead of ADD in Dockerfile
[INFO] * ADD in image history: [honeur/postgres:9.6-5.3.1 honeur/postgres:latest]
[INFO] * ADD in image history:
[honneur/webapi-atlas:2.6.0-1.1.4-secure honeur/webapi-atlas:latest honeur/webapi:2.6.0-1.1.4-secure]
[INFO] * ADD in image history: [honeur/zeppelin:0.8.0 honeur/zeppelin:latest]
[INFO] * ADD in image history: [honeur/user-mgmt:1.0.0 honeur/user-mgmt:latest]
[NOTE] 4.10 - Ensure secrets are not stored in Dockerfiles
[NOTE] 4.11 - Ensure verified packages are only Installed
```

[INFO] 5 - Container Runtime

```
[PASS] 5.1 - Ensure AppArmor Profile is Enabled
[PASS] 5.2 - Ensure SELinux security options are set, if applicable
[PASS] 5.3 - Ensure Linux Kernel Capabilities are restricted within containers
[PASS] 5.4 - Ensure privileged containers are not used
[PASS] 5.5 - Ensure sensitive host system directories are not mounted on containers
[PASS] 5.6 - Ensure ssh is not run within containers
[PASS] 5.7 - Ensure privileged ports are not mapped within containers
[NOTE] 5.8 - Ensure only needed ports are open on the container
[PASS] 5.9 - Ensure the host's network namespace is not shared
[WARN] 5.10 - Ensure memory usage for container is limited
[WARN] * Container running without memory restrictions: zeppelin
```

```

[WARN] * Container running without memory restrictions: webapi
[WARN] * Container running without memory restrictions: user-mgmt
[WARN] * Container running without memory restrictions: postgres
[WARN] 5.11 - Ensure CPU priority is set appropriately on the container
[WARN] * Container running without CPU restrictions: zeppelin
[WARN] * Container running without CPU restrictions: webapi
[WARN] * Container running without CPU restrictions: user-mgmt
[WARN] * Container running without CPU restrictions: postgres
[WARN] 5.12 - Ensure the container's root filesystem is mounted as read only
[WARN] * Container running with root FS mounted R/W: zeppelin
[WARN] * Container running with root FS mounted R/W: webapi
[WARN] * Container running with root FS mounted R/W: user-mgmt
[WARN] * Container running with root FS mounted R/W: postgres
[WARN] 5.13 - Ensure incoming container traffic is binded to a specific host interface
[WARN] * Port being bound to wildcard IP: 0.0.0.0 in zeppelin
[WARN] * Port being bound to wildcard IP: 0.0.0.0 in zeppelin
[WARN] * Port being bound to wildcard IP: 0.0.0.0 in webapi
[WARN] * Port being bound to wildcard IP: 0.0.0.0 in user-mgmt
[WARN] * Port being bound to wildcard IP: 0.0.0.0 in postgres
[WARN] 5.14 - Ensure 'on-failure' container restart policy is set to '5'
[WARN] * MaximumRetryCount is not set to 5: zeppelin
[WARN] * MaximumRetryCount is not set to 5: webapi
[WARN] * MaximumRetryCount is not set to 5: user-mgmt
[WARN] * MaximumRetryCount is not set to 5: postgres
[PASS] 5.15 - Ensure the host's process namespace is not shared
[PASS] 5.16 - Ensure the host's IPC namespace is not shared
[PASS] 5.17 - Ensure host devices are not directly exposed to containers
[INFO] 5.18 - Ensure the default ulimit is overwritten at runtime, only if needed
[INFO] * Container no default ulimit override: zeppelin
[INFO] * Container no default ulimit override: webapi
[INFO] * Container no default ulimit override: user-mgmt
[INFO] * Container no default ulimit override: postgres
[PASS] 5.19 - Ensure mount propagation mode is not set to shared
[PASS] 5.20 - Ensure the host's UTS namespace is not shared
[PASS] 5.21 - Ensure the default seccomp profile is not Disabled
[NOTE] 5.22 - Ensure docker exec commands are not used with privileged option
[NOTE] 5.23 - Ensure docker exec commands are not used with user option
[PASS] 5.24 - Ensure cgroup usage is confirmed
[PASS] 5.25 - Ensure the container is restricted from acquiring additional privileges
[PASS] 5.26 - Ensure container health is checked at runtime
[INFO] 5.27 - Ensure docker commands always get the latest version of the image
[WARN] 5.28 - Ensure PIDs cgroup limit is used
[WARN] * PIDs limit not set: zeppelin
[WARN] * PIDs limit not set: webapi
[WARN] * PIDs limit not set: user-mgmt
[WARN] * PIDs limit not set: postgres
[PASS] 5.29 - Ensure Docker's default bridge docker0 is not used
[PASS] 5.30 - Ensure the host's user namespaces is not shared
[PASS] 5.31 - Ensure the Docker socket is not mounted inside any containers

```

Clarifications

WARNINGS

[WARN] 4.1 - Ensure a user for the container has been created

[WARN] * Running as root: zeppelin

The final command or entrypoint of an image should never be run as a root user. If the container should get compromised, root access is granted to the attacker which is never good.

In the following images we prevented this:

- honeur/postgres:9.6-5.3.1
- honeur/webapi-atlas:2.6.0-1.1.4-secure
- honeur/user-mgmt:1.0.0

For Zeppelin we used apache:zeppelin:0.8.0 as the base image and only did configuration changes. Because the base image has a few programs depending on root access in the container, we decided to keep the root user.

[WARN] 4.5 - Ensure Content trust for Docker is Enabled

You can enable content trust in docker setting the environment variable DOCKER_CONTENT_TRUST to 1:

```
export DOCKER_CONTENT_TRUST=1
```

This will tell docker that it only should trust signed images.

All the images delivered by HONEUR are signed. We leave this option disabled during development.

[WARN] 5.10 - Ensure memory usage for container is limited

[WARN] * Container running without memory restrictions: zeppelin

[WARN] * Container running without memory restrictions: webapi

[WARN] * Container running without memory restrictions: user-mgmt

[WARN] * Container running without memory restrictions: postgres

This are runtime warnings and are not related to the HONEUR images. CGroups can be created on the host machine and referenced in the docker containers to avoid this warning.

[WARN] 5.11 - Ensure CPU priority is set appropriately on the container

[WARN] * Container running without CPU restrictions: zeppelin

[WARN] * Container running without CPU restrictions: webapi

[WARN] * Container running without CPU restrictions: user-mgmt

[WARN] * Container running without CPU restrictions: postgres

This are runtime warnings and are not related to the HONEUR images. CGroups can be created on the host machine and referenced in the docker containers to avoid this warning.

[WARN] 5.12 - Ensure the container's root filesystem is mounted as read only

[WARN] * Container running with root FS mounted R/W: zeppelin

[WARN] * Container running with root FS mounted R/W: webapi

[WARN] * Container running with root FS mounted R/W: user-mgmt

[WARN] * Container running with root FS mounted R/W: postgres

Some services on operating systems still work with access to the root filesystem. In our case the root filesystem was used to write process locks to. If we enabled read only for the root filesystem, we could not get the main programs run in the containers.

[WARN] 5.13 - Ensure incoming container traffic is binded to a specific host interface

[WARN] * Port being bound to wildcard IP: 0.0.0.0 in zeppelin

[WARN] * Port being bound to wildcard IP: 0.0.0.0 in zeppelin

[WARN] * Port being bound to wildcard IP: 0.0.0.0 in webapi

[WARN] * Port being bound to wildcard IP: 0.0.0.0 in user-mgmt

[WARN] * Port being bound to wildcard IP: 0.0.0.0 in postgres

0.0.0.0 is a wildcard IP Address. We opted for this instead of the host IP address because our remote system will mostly run on computers with dynamic IP Addresses. It is still possible to change this for servers with a fixed IP address.

[WARN] 5.14 - Ensure 'on-failure' container restart policy is set to '5'

[WARN] * MaximumRetryCount is not set to 5: zeppelin

[WARN] * MaximumRetryCount is not set to 5: webapi

[WARN] * MaximumRetryCount is not set to 5: user-mgmt

[WARN] * MaximumRetryCount is not set to 5: postgres

The restart policy we use is to always restart, not only on failure so MaximumRetryCount does not apply to this policy.

[WARN] 5.28 - Ensure PIDs cgroup limit is used

[WARN] * PIDs limit not set: zeppelin

[WARN] * PIDs limit not set: webapi

[WARN] * PIDs limit not set: user-mgmt

[WARN] * PIDs limit not set: postgres

This are runtime warnings and are not directly related to the HONEUR images. CGroups can be created on host machine and referenced in the docker container to avoid this warning.