Advanced Authentication 6.3

Repo Agent Installation Guide

December 2019

Legal Notices

Contents

About this Book

The Repo Agent Installation guide has been designed for users and describes the system requirements and installation procedure for Repo Agent. Repo Agent enables you to synchronize user data from on-premise AD repository and Advanced Authentication hosted on the cloud environment.

Intended Audience

This book provides information for individuals responsible for understanding administration concepts and implementing a secure, distributed administration model.

1 Overview

The Repo Agent acts as a middleware between the Advanced Authentication server and the organizational repositories. Repo Agent pulls the user data from the LDAP repository and makes this data available to Advanced Authentication based on periodic or on-demand requests. This eases the communication between the Advanced Authentication server and LDAP servers in a hybrid cloud-based environment, during the authentication of users.

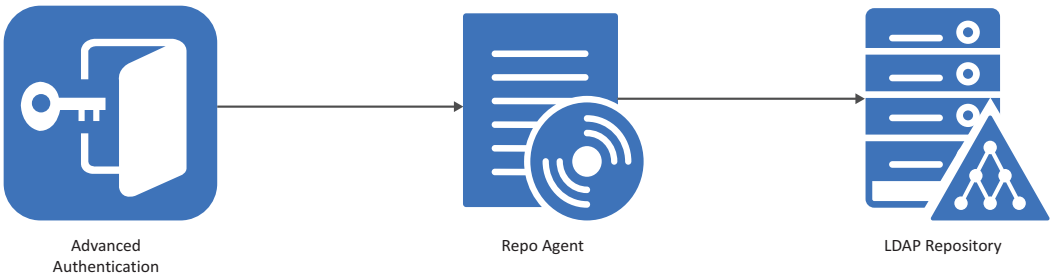
Why Repo Agent?

Previously, Advanced Authentication communicated directly with the LDAP servers to fetch the data of users and groups from the LDAP repository. However, during the authentication, this caused performance issues because of the delays that were caused every time Advanced Authentication interacted with the LDAP repository.

Also, if the Advanced Authentication server is to be hosted on cloud and if an organization may not want to expose their data to the cloud, then an agent is required that can run on-premise and communicate to the Advanced Authentication server on cloud.

To overcome these issues, Advanced Authentication provides the Repo Agent that acts as a middleware between the LDAP repository and Advanced Authentication. The Repo Agent stores the data in an internal database and makes the data available to Advanced Authentication.

Figure 1-1 *Illustrates working of Repo Agent*



2 System Requirements

You must have the administrator privileges to install and configure the Repo Agent.

- Linux host with docker and docker-compose.
  - Minimum Requirement: docker-compose version 1.23.2
  - Minimum Requirement: docker version 18.09.1
- CPU
  - Minimum requirement: 2 Cores CPU
- Memory
  - Minimum requirement: 4 GB of RAM
  - Recommended requirement: 8 GB of RAM
- Hard disk space
  - Minimum requirement: 40 GB
  - Recommended requirement: 60 GB
- IP Ports

  Ensure that the firewall uses the default port 9443
- LDAP Repositories

  The following LDAP repositories are supported:
  - Microsoft Active Directory Services
  - Microsoft Active Directory Lightweight Directory Services
  - NetIQ eDirectory
  - OpenLDAP
  - OpenDJ

  NOTE: Repo Agent does not support the SQL repositories.

  ---

3 Installing and Configuring the Repo Agent

This chapter contains the following sections:

Installing the Repo Agent

1. Create a folder, for example, AuCoreRepoAgent in any valid directory:

   ```
   mkdir AuCoreRepoAgent
   ```

2. After you create the AuCoreRepoAgent, you must create the following script files to run the Repo Agent:

   2a Create a file dockompose with the following content:

   ```
   #!/bin/bash
   pushd config >/dev/null
   docker-compose $*
   popd >/dev/null
   ```

   NOTE: If you create the files on Windows, ensure that you remove the Windows line ending symbol (^M) in the end of each line.

   2b Create a file, for example, aurepa_docker_stats.sh with the following content:

   ```
   #!/bin/bash
   TMP=/tmp/docker-stats
   docker stats --no-stream --format "table
   {{.Name}}\t{{.CPUPerc}}\t{{.MemUsage}}" | grep aurepa | tail -n +2
   >$TMP
   echo " SORT BY NAME"
   cat $TMP | sort -k 1
   echo " SORT BY CPU"
   cat $TMP | sort -k 2
   echo " SORT BY MEM"
   cat $TMP | sort -k 3 -h
   echo " if you want interactive montor, run 'docker stats'"
   ```

   2c Create a file repo.sh with the following content:

   This file helps to start, stop, or restart the services (db, sync, and http) of the Repo Agent.

```
#!/bin/bash
CMD=$1
shift
REPO_NAME=$1
shift
[ -z $REPO_NAME ] && echo "Usage: repo.sh <start|stop|restart>
REPO_NAME" && exit 2
./dockompose $CMD $REPO_NAME-aurepa-sync
./dockompose $CMD $REPO_NAME-aurepa-http
./dockompose $CMD $REPO_NAME-aurepa-db
```

2d Create a file `run_sync.sh` with the following content:

This file helps to manually sync the data of the LDAP repositories. It can be a full sync or a fast sync.

```
#!/bin/bash
# stop parallel sync, if any. run manual sync and start scheduler
again
cat <<EOT >/tmp/run_sync_usage
Usage: run_sync.sh REPO_NANE [command] (command is aurepa.full sync
by default)
Examples:
  run_sync.sh MOON
  run_sync.sh MOON aurepa.fast_sync
  run_sync.sh EARTH aurepa.recreate_db (wipe all data)
  run_sync.sh EARTH aurepa.print_ldap_users (check LDAP
connectivity)
EOT
REPO_NAME=$1
[ -z $REPO_NAME ] && cat /tmp/run_sync_usage && exit 2
COMMAND=$2
[ -z $COMMAND ] && COMMAND=aurepa.full_sync
./dockompose stop  $REPO_NAME-aurepa-sync
./dockompose run --rm  $REPO_NAME-aurepa-sync $COMMAND
./dockompose start $REPO_NAME-aurepa-sync
```

2e Create a file `setup_config_production.sh` with the following content:

This file generates the self-signed certificates, `nginx.conf`, and the docker-compose files.

```
#!/bin/bash
export AUREPA_IMG="mfsecurity/aaf-aurepa:6.2.0.0"
export DOCKER_CONTENT_TRUST=1
export SSL_HOSTNAME=$SSL_HOSTNAME
# Generate docker-compose.yml, nginx and ini file. Generate SSL
certificate, if not provided
[ -z $SSL_HOSTNAME ] && [ ! -f config/etc.nginx/cert.pem ] && \
   echo "Usage: SSL_HOSTNAME=your-server.com ./$(basename
${BASH_SOURCE[0]})" && \
   exit 2
MYDIR=`cd "$( dirname "${BASH_SOURCE[0]}" )" && pwd`
CONF_DIR=$MYDIR/config
docker run --rm \
  -e PYTHONUNBUFFERED=1 \
  -e SSL_HOSTNAME=$SSL_HOSTNAME \
  -e AUREPA_IMG=$AUREPA_IMG \
  -v $CONF_DIR:/mnt/config/  $AUREPA_IMG \
  python /opt/AuRepa/auconfig/setup_config.pyc $CONF_DIR
```

NOTE: Run the command `sudo chmod 755` to set permissions for the above files.

2f Create the following folders in the `AuCoreRepoAgent` folder:

```
mkdir -p config/etc.nginx
mkdir -p config/EXAMPLE1.repo
```

Configuring the Repo Agent

To configure Repo Agent, perform the following:

Setting Up the Config Folder of Repo Agent

The `$AuCoreRepoAgent/config` folder contains the following files:

- `EXAMPLE1.repo`
- `etc.nginx`

You must rename `EXAMPLE1.repo` with the repo name of your repository.

For example, `mv EXAMPLE1.repo/ FOCUS.repo`

NOTE: Repo Name must be same as the NETBIOS name for the Active Directory.

Create the following three files in the FOCUS.repo:

- cron.py: This file allows you to configure the LDAP synchronization.

  For example, the file contains the following format:

```
import schedule, aurepa.scheduler as au
run = au.run
################################
# Schedule, please customize
schedule.every(10).minutes.do(run, command='aurepa.fast_sync')
#schedule.every().saturday.at("00:15").do(run,
command='aurepa.full_sync')
# schedule.every(3).days.do(run, command='aurepa.full_sync')
# Help: see https://schedule.readthedocs.io/en/stable/
KILL_TIMEOUT_MINUTES = 60 * 4  # 4 hours, increase if your full sync may
run longer
# End schedule
################################
# Do not change rest of the file
au.kill_timeout_seconds = KILL_TIMEOUT_MINUTES * 60
au.main_loop()
print(f"This message must not appear. File {__name__} must run
aurepa.scheduler.main_loop() forever")
```

- repo.json: This file helps you configure the LDAP parameters.

  For example, the file contains the following format:

```
{
    "user": "CN=Administrator,CN=Users,DC=focus,DC=com",
    "base_dn": "cn=users,dc=focus,dc=com",
    "password": "sample@12345",
    "ldap_type": 1,
    "ldap_type_help": "(1, 'AD'), (2, 'AD LDS'), (3, 'eDirectory'), (4,
'Other'). This field is ignored",
    "paged_enabled": true,
    "nested_enabled": true,
    "base_dn_one_level": false,
    "group_dn_one_level": false,
    "user_mail_attrs": ["mail", "otherMailbox"],
    "user_name_attrs": ["sAMAccountName", "userPrincipalName"],
    "group_name_attrs": ["sAMAccountName"],
    "user_lookup_attrs": ["sAMAccountName", "userPrincipalName"],
    "group_lookup_attrs": ["sAMAccountName"],
    "user_mobile_phone_attrs": ["mobile", "otherMobile"],
    "custom_attrs": ["info", "pager"],
    "servers": [
        {"name": "1.1.1.1", "port":389,"use_ssl": false},
        {"name": "1.1.1.4", "port":389,"use_ssl": false}
    ]
}
```

NOTE: With custom_attrs, it is possible to return any LDAP attribute from Active Directory. These attributes provide additional information that can be displayed on RADIUS client if the corresponding RADIUS result specification rule exists in the Administration portal.

- `secret.json`: This file helps you to configure the username and password that you must specify during the creation of an external repository in the Advanced Authentication server at Administration portal > Repositories > Add External repo.

  For example, the `secret.json` file contains the following format:

  ```
  {
    "user": "focus",
    "password": "focus"
  }
  ```

Setting Up the Repo Agent for Certificates and Services

You must set up the Repo Agent for generating the self-signed certificates and docker-compose services.

1 To generate a self-signed certificate, run the following command:

  ```
  export SSL_HOSTNAME=<host_server>
  ```

  ```
  ./setup_config_production.sh
  ```

  This generates the self-signed certificates, `nginx.conf`, and docker-compose files that are stored in the `$AuCoreRepoAgent/config` and `$AuCoreRepoAgent/config/etc.nginx` folders.

  A certificate is generated in the following format:

  ```
  -----BEGIN CERTIFICATE-----
  MIIDjjCCAnagAwIBAgIJALEEogxd1k/tMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
  BAYTAkVVMRkwFwYDVQQIDBBTZWxmLVNpZ25lZC1DZXJ0MQwwCgYDVQQKDANBQUYx
  DzANBgNVBAsMBkF1UmVwYTETMBEGA1UEAwwKMTAuNzEuMzIuOTAeFw0xOTAyMTQx
  MjQwMjRaFw0yOTAyMTExMjQwMjRaMFwxCzAJBgNVBAYTAkVVMRkwFwYDVQQIDBBT
  ZWxmLVNpZ25lZC1DZXJ0MQwwCgYDVQQKDANBQUYxDzANBgNVBAsMBkF1UmVwYTET
  MBEGA1UEAwwKMTAuNzEuMzIuOTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoC
  ggEBAJS1vqBE+2jp8KVpJQAI0dg0mRlp/ovzv52CNswgatfdJD/UzK/sr7fEnFY/
  m4C6NZkAscq3zzot+VfoINAQduC6apYj75mrQ7hd8yhjmqsFwIuF/CG9VOOrxNbr
  hQmnsSyfPqYUnD8LCrieQz2U9mQa2TFhExCjkcqJ32M8Q8SKb11pdtfmWdvn8HsS
  1FarqmbhJxNWlYXVVr1XEw/epTTJrlalo+2DEXFMyjTJ5ihliqW8fQ47Gg2piPGN
  3BfE1Xs0ZLxLI0qeXXcr7g6rLWb8BJLCIe4k9DIDMeSyY3Wt5GZsBW2PtJZc5tQq
  0xLl+H/kT+KnzZGUx4RfDah0B0MCAwEAAaNTMFEwHQYDVR0OBBYEFGCay3xXKyZd
  2KAG7+46+9HxcGa0MB8GA1UdIwQYMBaAFGCay3xXKyZd2KAG7+46+9HxcGa0MA8G
  A1UdEwEB/wQFMAMBAf8wDQYJKoZIhvcNAQELBQADggEBADaQS7X9OE/rIm84pCL8
  fBCNelyV1DfdQJk1ZzaIg+QLzLXIHm7pXEjIRqEIVJqIZkdFj4gHdvHxekfZtAX/
  lbVkx3ci6BzjKU/V0oRUxAbwnkU5YIaoklJu9tV/TT2vcynMAV/6o/GLBb29sY0L
  kdScWof5XT4L0+AZvzTFcxfR4ztPCKeIgLswAVdsYaDpw6o45FrJgN+IjKgF+Ge+
  WOTRQjFjiHn2IUBnJLEePg1bs9a25bpO4pqz0wGxyLhXwGUV/6hDGtOI3hxu1kmA
  En+tmxEAasxarjbtXE765KzYfrEgKxoHJUiEKE7KatSIg9REM/oKi4JbhUpQlRsf
  aLI=
  -----END CERTIFICATE-----
  ```

  NOTE: You must upload this certificate in the Administration portal > Repositories > Add External repo while "Creating an External Repository on Advanced Authentication".

2 If you want to upload your own CA certificate, you must place it as `cert.pem` in the `etc.nginx` folder before running the `setup_config_production.sh` file.

NOTE: When the SSL_HOSTNAME is not passed and setup_config_production.sh is executed, a script picks the custom certificates from `etc-config` file and consumes it for nginx. This also creates the two files: `docker-compose.yml` and `aurepa.ini`.

Starting or Stopping the Services of the Repo Agent

Run the following command under the AuCoreRepoAgent directory to start the docker compose services of the Repo Agent:

```
./dockompose up -d
```

Based on the number of repos that are configured, the services are started. Typically, for each repo, the Repo Agent starts three services: db, sync, and http.

The following services are created for **FOCUS**, which is a repository running in the Repo Agent and one single nginx service as a front web-server:

- `config_nginx_1`
- `config_FOCUS-aurepa-db_1`
- `config_FOCUS-aurepa-http_1`
- `config_FOCUS-aurepa-sync_1`

To stop and remove the services of the Repo Agent, run the following command:

```
./dockompose down $* --remove-orphans
```

This cleans or removes the Repo Agent docker services from the host machine.

Syncing the Repository Data to the Repo Agent

To manually sync the data from the LDAP repositories, run the following command:

$AuCoreRepoAgent/`run_sync.sh <REPO_NAME> [aurepa.fast_sync | aurepa.full_sync]`

For example, to do a manual Fast sync for the **FOCUS** repository, run the following command:

$AuCoreRepoAgent/`run_sync.sh FOCUS aurepa.fast_sync`

$AuCoreRepoAgent/`run_sync.sh FOCUS` performs a full sync of the Repo Agent.

NOTE: The Repo Agent fails to sync data with Advanced Authentication when the Repo Name contains spaces.

You can perform the following to validate the syncing of repositories:

- "Checking the Repository LDAP Connectivity Before Syncing" on page 17
- "Checking Repository Information is Synced to the Repo Agent Database" on page 17
- "Cleaning the Repo Agent Database" on page 17

Checking the Repository LDAP Connectivity Before Syncing

Before syncing the repository data, to check the LDAP connectivity and print the users to be synced, run the following command:

```
$AuCoreRepoAgent/run_sync.sh <REPO_NAME> aurepa.print_ldap_users
```

For example, `$AuCoreRepoAgent/run_sync.sh FOCUS aurepa.print_ldap_users`

Checking Repository Information is Synced to the Repo Agent Database

To check all the users and groups information is synced to the Repo Agent database, run the following command:

NOTE: Replace the `REPO_NAME` with the repo name provided in the `$AuCoreRepoAgent/config` directory.

For users:

```
docker exec config_<REPO_NAME>-aurepa-db_1 psql -U postgres -d aurepa -P
pager=off -c "select count(lookup_names) from repa_user"
```

For groups:

```
docker exec config_<REPO_NAME>-aurepa-db_1 psql -U postgres -d aurepa -P
pager=off -c "select count(lookup_names) from repa_group"
```

Cleaning the Repo Agent Database

To delete an invalid user or group information in the Repo Agent database and clean the database without reconfiguring the Repo Agent, run the following command:

```
$AuCoreRepoAgent/run_sync.sh <REPO_NAME> aurepa.recreate_db
```

NOTE: After clean up, you must sync the data for the repositories.

Creating an External Repository on Advanced Authentication

After you install and configure the Repo Agent, you must map the Repo Agent as the external repository on Advanced Authentication.

To add the external repository in Advanced Authentication:

1. Open the Advanced Authentication Administration portal.
2. Click **Repositories > Add External repo**.
3. Specify the following details:
   - **Name**: Name of the repository.

     Name of the repository must be the same as what is defined in the Repo Agent.

     NOTE: Ensure that the repository name does not contain spaces.

- **Username**: Name of the user using the repository.
- **Password**: Password of the repository.

NOTE: The **Username** and **Password** are defined in the `secret.json` file of the Repo Agent. For information about the secret.json file, see "Setting Up the Config Folder of Repo Agent".

4 Add the external repository server configurations:

4a Click **Add Server.**

4b Specify the IP address of the Repo Agent in **Address**.

4c Specify the port number of the external repository server in **Port**. For example, 9443.

4d Save the server credentials.

5 Click **Choose File** to upload the CA certificate for the agent.

This is the self-signed certificate `cert.pem` generated in the `etc.nginx` folder or your own CA certificate used during the configuration of the Repo Agent.

6 Click **Save**.

NOTE: You can perform the synchronization of an external repository only from a Global Master server.

Checking Repository is Synced to the Advanced Authentication Database

After creating the external repository in the Advanced Authentication Administration portal and syncing, to validate whether are all user and group information is synced, perform the following steps:

1 Log in to the Advanced Authentication terminal.

2 Run the following commands:

- To check users:

```
docker exec aaf_audb_1 psql -U root -d aucore_prod -P pager=off -c
"select * from external_user"
```

- To check groups:

```
docker exec aaf_audb_1 psql -U root -d aucore_prod -P pager=off -c
"select * from external_group"
```

Uninstalling the Repo Agent

To uninstall the Repo Agent, run the following commands:

```
./dockompose down -v --remove-orphans
```

```
docker container prune -f
```

```
docker network prune -f
```

NOTE: The above commands removes the unused networks and containers.

4 Troubleshooting

This chapter contains the following topics:

Collecting Logs for Debugging

Run the following command to collect logs of all the Repo services that are configured in the `$AuCoreRepoAgent/config` directory:

```
$AuCoreRepoAgent/dockompose logs -f $*
```

Collecting Statistical Information

To list the statistical information of Repo Agent docker container, for example, the CPU usage, memory, and so on, run the following command:

```
$AuCoreRepoAgent/aurepa_docker_stats.sh
```

Regenerate Self-Signed Certificate or Custom Certificates Used for Repo Agent

To regenerate a new self-signed certificate or use custom certificates for the nginx container, perform the following steps:

1. Delete the existing certificates and nginx configuration files:

   ```
   sudo rm $AuCoreRepoAgent/config/etc.nginx/*.*
   ```

2. Reconfigure the Repo Agent:

   ```
   SSL_HOSTNAME=<Repo_Agent_IP_Hostname>
   ./setup_config_production.sh
   ```

3. Restart the nginx container:

   ```
   $AuCoreRepoAgent/dockompose restart nginx
   ```

NOTE: You must name the custom certificate as `cert.pem`.

Starting or Stopping Repo Agent Services Specific to a Repository

To manage the services of a specific repository, run the following command:

```
$AuCoreRepoAgent/repo.sh <start|stop|restart> REPO_NAME
```