# InterSystems API Manager Version 2.8.1 Guide

2024-05-06

InterSystems Corporation 1 Memorial Drive Cambridge MA 02142 www.intersystems.com

*InterSystems API Manager Version 2.8.1 Guide*
InterSystems IRIS Data Platform 2024-05-06

For Support questions about any InterSystems products, contact:

**InterSystems Worldwide Response Center (WRC)**
Tel: +1-617-621-0700
Tel: +44 (0) 844 854 2917
Email: support@InterSystems.com

# Table of Contents

# 1

# Introduction to the InterSystems API Manager (IAM) Version 2.8.1

InterSystems API Manager (IAM) allows you to take advantage of microservices and APIs that are either exposed or consumed by your InterSystems IRIS® applications. Acting as an API gateway between your InterSystems IRIS servers and applications, it gives you the ability to more effectively monitor and control the traffic of calls between your server-side APIs and your client-side applications. IAM enables you to monitor, control, and govern HTTP-based API traffic.

The more distributed your environment, the more critical it becomes to properly govern and monitor your API traffic. IAM enables you to easily route all your traffic through a centralized gateway and forward API requests to appropriate target nodes.

The InterSystems API Manager is powered by Kong Gateway (Enterprise) Version 2.8.1, an industry-leading API manager from Kong, Inc. The InterSystems Worldwide Response Center (WRC) provides support for IAM including the Kong Gateway (Enterprise). The main source of information about IAM is the Kong Gateway (Enterprise) 2.8.x documentation. InterSystems provides information about installing and upgrading IAM and some details about the IAM environment.

For information on other versions of IAM, see InterSystems API Manager (IAM).

## 1.1 Benefits of Using IAM

Using IAM allows you to do the following:

- Monitor your HTTP-based API traffic and understand who is using your APIs; what are your most popular APIs and which could require a rework.
- Control who is using your APIs and restrict usage in various ways. From simple access restrictions to throttling API traffic and fine-tuning request payloads, you have fine-grained control and can react quickly.
- Protect your APIs with central security mechanisms like OAuth2.0 or Key Token Authentication.
- Onboard third-party developers and provide them with a superb developer experience right from the start by providing a dedicated Developer Portal for their needs.
- Scale your API demands and deliver low-latency responses.
- Provide a uniform API for underlying services that have different APIs.

**Note:** IAM provided with InterSystems IRIS must be used only for APIs that are either provided by InterSystems IRIS or used by InterSystems IRIS. If you want to extend the use of IAM to cover APIs that are independent of InterSystems IRIS, contact InterSystems sales for more information.

## 1.2 Learning About IAM

The IAM Version 2.8.1 documentation provided by InterSystems includes:

- Installing IAM
- Upgrading IAM
- Hands-On with InterSystems API Manager for Developers

For documentation of previous IAM releases, see InterSystems API Manager (IAM).

See the Kong Gateway (Enterprise) 2.8.x documentation for details on using IAM.

For additional information about IAM, see the InterSystems Developer Community article and the InterSystems online learning site. The online learning site has online classes, videos and documents that cover topics on IAM, such as building FHIR applications and best practices. On the online learning site, search for `API Manager` to find these resources.

# 2

# Installing IAM Version 2.8.1

This article tells you how to install the InterSystems API Manager (IAM) from the installation tar file. IAM is provided in container format. You need software that supports the Open Container Initiative (OCI) to be able to install and run the IAM container.

This article uses Docker as an example of software that supports OCI, but not all users may be able to do so. For example, users on systems running Red Hat® Enterprise Linux® will need to use Podman (https://podman.io). Nonetheless, the steps for installing IAM are the same across these utilities, although you will find yourself using different commands depending on which software that supports OCI you are using.

**Note:** In this article, InterSystems IRIS® refers to any InterSystems product based on InterSystems IRIS that supports IAM. This includes InterSystems IRIS for Health and HealthShare® Health Connect.

## 2.1 Downloading IAM

You can download the installation tar file from the InterSystems Worldwide Response Center (WRC) download page: https://wrc.intersystems.com/wrc/coDistGen.csp. To show only the IAM kits, type `IAM` in the Name column. IAM is distributed as a compressed tarball archive. Once you uncompress it and extract the files, you will find the following contents in the distribution kit:

- IAM Docker image, iam-image.tar — do not extract the files from this archive.
- scripts directory with:
  - docker-compose.yml script to start and stop IAM
  - Scripts to setup and test IAM — these optional scripts provide an easy way to start and test IAM. The startup scripts set up the environment variables in the current shell used by IAM. If you do not use the scripts, you need to define these variables some other way.
- readme.txt file with brief instructions for starting IAM. (This article is based on the readme but provides some additional information.)
- EULA files with terms and conditions

## 2.2 Setting Up IAM

To set up IAM, follow these steps:

1. Ensure that your system has the required prerequisites:
    a. Install Docker if your system does not already have it. You must have the Docker engine version 17.04 or later. See Running InterSystems Products in Containers for a brief introduction to containers and Docker.

    b. Install the docker-compose command line interface version 1.12.0 or later if your system does not already have it.

    c. Install the curl utility on the UNIX® system. The script files that start and test IAM use curl.

    d. Ensure that you have a running instance of InterSystems IRIS, InterSystems IRIS for Health, or HealthShare Health Connect that supports IAM:
        - InterSystems IRIS and InterSystems IRIS for Health first supported IAM in version 2019.1.1.
        - HealthShare Health Connect first supported IAM in version 2020.1.

    e. Ensure that your InterSystems IRIS license file specifies “API Management enabled”. If it doesn't, contact InterSystems to obtain a license file that enables IAM.

    f. Download IAM and extract the files as described previously.

2. Enable the IAM user and web application on the InterSystems IRIS instance. The purpose of the IAM user is to allow the setup script to get a copy of the IAM license from the instance of InterSystems IRIS and is only used to access the IAM license information. The IAM user, by default, has only one role, %IAM_API, which provides it with all of the privileges it needs. In the Management Portal for the instance of InterSystems IRIS, InterSystems IRIS for Health, or HealthShare Health Connect:
    a. Select **System Administration** > **Security** > **Applications** > **Web Applications** and select the `/api/iam` web application.

    b. Select the **Enable Application** check box.

    c. Select **Save**.

    d. Select **System Administration** > **Security** > **Users** and select the IAM user.

    e. Enter a new password and select the **User enabled** check box.

    f. Select **Save**.

3. Execute the following command (in the directory where you extracted the IAM archive) to load the IAM image into your local repository:

```
docker load -i iam_image.tar
```

4. Run the setup script and start IAM:
    a. Run the IAM setup script.

    In a UNIX bash shell, enter:

```
source ./scripts/iam-setup.sh
```

    Or in another UNIX shell, enter the equivalent dot command:

```
. ./scripts/iam-setup.sh
```

b. Enter the full image repository, name and tag for your IAM docker image. For example, it could be:

   `intersystems/iam:2.8.1.0-3`

c. Enter the IP address for your InterSystems IRIS instance. If your instance is on your local machine, please use your externally visible local IP address, not `localhost`. If the instance is running in a container, use the IP address of the host environment, not the IP address of the container. To avoid any DNS issues, use the numeric form of the IP address.

d. Enter the web server port for your InterSystems IRIS instance.

e. Enter the password for the IAM user on your InterSystems IRIS instance.

f. Re-enter the password.

g. If you want IAM to request the license from InterSystems IRIS using HTTPS instead of HTTP, provide the full path to your CA Certificate file; otherwise, press **Enter**.

h. With certain InterSystems IRIS configurations, the instance is not accessible by using the instance server name. In these cases, your InterSystems IRIS instance is only accessible via its CSPConfigName URL prefix (see Changing the InterSystems IRIS Server Name in the URL) and you need to provide the prefix with a trailing slash (/) now. If this does not apply, click the Enter key.

i. Confirm your entries.

This script sets the environment variables required by the docker-compose file.

## 2.3 Environment Variables

The following environment variables are used by the docker-compose file. These are set by the startup script. If you do not use the startup script, you must define these variables.

- *ISC_IAM_IMAGE* — contains the repository, name, and tag of the IAM docker image. The docker-compose file uses this variable to access the docker image. The value has the format:

  *repository*`/`*name*`:`*tag*

- *ISC_IRIS_URL* — contains the URL to access the InterSystems IRIS instance to get the IAM license. The docker-compose file defines this environment variable. The value has the format:

  `IAM:`*password@ip-address*`:`*port-number*`/api/iam/license`

  where *password* is the password of the IAM account on the InterSystems IRIS instance and *ip-address*`:`*port-number* are the IP address and web server port of the instance.

- *ISC_CA_CERT* — optionally contains the contents of the CA certificate file for the server running InterSystems IRIS. If local policy requires that HTTPS be used for communication, then this environment variable must contain the contents of the server's CA certificate.

These environment variables are defined in the shell and allow the docker-compose file to access the IAM container and the InterSystems IRIS image. If you are not in the shell where you executed the setup script, these environment variables are not defined. You can either re-run the script or define them in another way.

By default, IAM listens on the following ports:

- `:8000` on which IAM listens for incoming HTTP traffic from your clients, and forwards it to your upstream services.

- `:8443` on which IAM listens for incoming HTTPS traffic. This port has a similar behavior as the `:8000` port, except that it expects HTTPS traffic only. This port can be disabled via the configuration file.

- `:8003` on which IAM listens for Developer Portal GUI traffic — if the Developer Portal is enabled.
- `:8004` on which IAM listens for Developer Portal `/files` traffic — if the Developer Portal is enabled.
- `:8001` on which the Administration API listens.
- `:8444` on which the Administration API listens for HTTPS traffic.
- `:8002` on which IAM listens for management portal GUI traffic.
- `:8445` on which IAM listens for HTTPS management portal GUI traffic.

# 2.4 Start and Test API Manager

The docker-compose.yml file is a convenient way to start and stop IAM in the docker container. To start IAM, navigate to the scripts directory with the docker-compose.yml file and execute the following command to start IAM:

```
docker-compose up -d
```

You can access the user interface at `http://localhost:8002/`.

To test the IAM setup, navigate to the directory *scripts*), and run the *iam-test* script. This script sets up a route and a service in IAM and allows you to check connectivity with your InterSystems IRIS instance.

**Note:** The docker-compose.yml file defines the URLs that are used to access the IAM management portal and the developer portal. To avoid Cross-Origin Resource Sharing (CORS) errors when accessing the IAM management and developer portals, the URLs that you use must match the URLs defined in the docker-compose.yml file in the *KONG_ADMIN_GUI_URL* and *KONG_PORTAL_GUI_HOST* environment variables. The default values of these are `http://localhost:8002` for the management portal and `127.0.0.1:8003` for the developer portal. If you will be using different URLs to access these portals, you must edit the docker-compose file before you start IAM. For details on how Kong Enterprise handles CORS and other DNS issues, see DNS Considerations for Kong Enterprise.

# 2.5 Stop API Manager

To stop the IAM container, navigate to the directory with the docker-compose.yml file and execute the following command:

```
docker-compose down
```

Note that you need to be in the same shell as the one that you ran the setup scripts or you need to define the *ISC_IAM_IMAGE* and *ISC_IRIS_URL* environment variables.

# 2.6 Restart API Manager

To restart the IAM container, navigate to the scripts directory with the docker-compose.yml file and execute the following command to start IAM:

```
docker-compose up -d
```

Note that you need to be in the same shell as the one that you ran the setup scripts or you need to define the *ISC_IAM_IMAGE* and *ISC_IRIS_URL* environment variables.

# 2.7 Troubleshooting IAM Installations

This topic covers some common issues you can run into when installing IAM. We will add additional issues when we discover them.

## 2.7.1 Getting the IAM Logs

If you run into installation problems, it may help to view the install logs. If you are running IAM using the supplied setup scripts with Docker, you can find the logs with:

```
docker logs scripts_iam_1
docker logs scripts_db_1
```

## 2.7.2 Cannot Get IAM License

There are multiple reasons why IAM cannot communicate with the InterSystems IRIS instance and get the license. This problem shows up in the following ways:

- The iam-setup.sh script displays one of the following:
  - The /api/iam web application is disabled. Please enable it before running this script again.
  - Authorization failed. Please make sure to enable the IAM user and reset its password before running this script again. This error may also mean that you entered the wrong password to this script.
  - No content. Either your InterSystems IRIS instance is unlicensed or your license key does not contain an IAM license.
  - Request failed with a 400 status code. You may be trying to use HTTP on an SSL-enabled server port.
  - Couldn't reach InterSystems IRIS at $ip:$port. One or both of your IP and Port are incorrect.
- In some rarer cases, IAM may not succeed in getting the license even if the script reports "Successfully got IAM license!". In this case, the symptom would be "could not decode license JSON: No license found" appearing in the IAM log. This condition could be caused by a network setup where the script running on the host has access to the InterSystems IRIS instance but IAM running in a container does not have access to it.

One other reason that IAM cannot access the InterSystems IRIS instance, is if it is configured so that the instance is not accessible by the default URL with the server name or the /api interface is blocked. In this case, you must specify the *CSPConfigName* URL prefix in the startup script (see Changing the InterSystems IRIS Server Name in the URL).

## 2.7.3 IAM Management Portal or Developer Portal Is Empty

If you enter the URL for the IAM management portal or the developer portal and the portal does not display with content, it is possible that you have entered a URL that may seem correct but is not the URL specified by the docker-compose.yml file. The docker-compose.yml file defines the URLs that are used to access the IAM management portal and the developer portal. To avoid Cross-Origin Resource Sharing (CORS) errors when accessing the IAM management and developer portals, the URLs that you use to access them must match the URLs defined in the docker-compose.yml file in the *KONG_ADMIN_GUI_URL* and *KONG_PORTAL_GUI_HOST* environment variables. The default values of these are `http://localhost:8002` for the management portal and `127.0.0.1:8003` for the developer portal. If you will be using different URLs to access these portals, you must edit the `docker-compose.yml` file before you start IAM. For details on how Kong Enterprise handles CORS and other DNS issues, see DNS Considerations for Kong Enterprise.

# 3

# Upgrading to IAM Version 2.8.1

This topic describes how to upgrade the InterSystems API Manager (IAM) from version 2.3.3.2 to version 2.8.1.

To see what features have been added, see the changelog.

## 3.1 Migration Steps

To migrate from 2.3.3.2 to 2.8.1 (specifically to 2.8.1.0):

1. Download the installation tar file from the InterSystems Worldwide Response Center (WRC) download page: https://wrc.intersystems.com/wrc/coDistGen.csp. To show only the IAM kits, type `IAM` in the **Name** column.
2. Uncompress the file and extract the contents.

   See Installing IAM for a list of the contents.
3. Load the IAM image into your local repository by executing the following command in the directory where you extracted the IAM archive:

   **Docker Compose**

   ```
   docker load -i iam_image.tar
   ```

   **Podman Compose**

   ```
   podman load -i iam_image.tar
   ```
4. Perform the upgrade. To do this, you can create and run a series of docker-compose files or you can migrate manually using instructions at https://docs.konghq.com/gateway/2.8.x/install-and-run/upgrade-enterprise/#migrate-db.

## 3.2 Running a Migration Using Docker Compose

Using docker-compose files to run the migrations can streamline the process.

The following two sections describe how to create the docker-compose files, and then how to run them.

### 3.2.1 Creating the Docker Compose Files

As each IAM cluster is unique, you must create the docker-compose files based on the configuration of your IAM cluster. You should create all these files before you start the migration. Be sure to locate all the docker-compose files in the same directory as the main docker-compose.yml file used to start and stop IAM; this ensures the files share the same network.

As an example, this section describes the process for a single-node IAM cluster:

1. Begin by creating a copy of the main docker-compose.yml file. Name the copy step1.yml.
2. Next, edit step1.yml to perform the first step of the migration: create a new node (`iam-migrations` in the example here) of the target version that points to the same datastore, and run `kong migrations up`. This may look like:

```
version: '3.2'
services:
  iam-migrations:
    image: intersystems/iam:2.8.1.0-3
    command: kong migrations up
    depends_on:
      - db
    environment:
     [...]
    restart: on-failure
    links:
      - db:db
  iam:
    image: intersystems/iam:2.3.3.2-1
    depends_on:
      - db
    environment:
     [...]
    links:
      - db:db
    ports:
     [...]
    restart: on-failure
  db:
    image: postgres:9.6
    [...]
volumes:
  pgdata:
```

3. Then, create another new docker-compose file and name it step2.yml. When run, this file should provision a node of the target version to replace the older node. In the example below, the old node is decommissioned, and a new node is provisioned with the newer version of IAM.

```
version: '3.2'
services:
  iam:
    image: intersystems/iam:2.8.1.0-3
    depends_on:
      - db
    environment:
    [...]
    links:
      - db:db
    ports:
    [...]
    restart: on-failure
  db:
    image: postgres:9.6
    [...]
volumes:
  pgdata:
```

   For multi-node clusters, you will need to create additional docker-compose files to reprovision each node in your cluster. As this is a single-node example, we can move on to the last step of the migration.

4. Finally, create the last docker-compose file, step3.yml. This file should finish the current migration when run.

```
version: '3.2'
services:
  iam-migrations:
    image: intersystems/iam:2.8.1.0-3
    command: kong migrations finish
    depends_on:
      - db
    environment:
    [...]
    restart: on-failure
    links:
      - db:db
  iam:
    image: intersystems/iam:2.8.1.0-3
    depends_on:
      - db
    environment:
    [...]
    links:
      - db:db
    ports:
    [...]
    restart: on-failure
  db:
    image: postgres:9.6
    [...]
volumes:
  pgdata:
```

### 3.2.2 Running the Docker Compose Files

Once all the files are created, run them in sequence. The command to run a given docker-compose file is:

```
docker-compose -f step#.yml up -d
```

where `step#.yml` is the name of the file to run.