

Installing API Management



As of Scheer PAS 21.1, this installation documentation is deprecated. You should not perform an installation with the old Docker images. API Management now is part of the Scheer PAS installation and cannot be installed stand-alone.

Prerequisites

Please consider the following prerequisites regarding **Scheer PAS API Management**.

The software uses [Docker](#) to run.

- The **Docker Community Edition (CE)** (version 18.06 or newer) is sufficient to run the software. The provided Docker containers are Linux containers. Refer to the [Docker documentation pages](#) for more information on supported platforms and how to install Docker.
- As Docker host, we support Linux, namely the following Linux distributions:
 - RedHat/CentOS
 - Ubuntu
- The Docker tool **docker-compose** (version 1.23 or newer) must be installed.

Step 1: Download and Extract the Software

API Management uses Docker to provide a simple setup which is easy to update and scalable if necessary.



Make sure you have network access during the installation, because you have to download images from Docker hub.

1. **Download** the following files provided by Scheer to the folder you want to install API Management to:

```
api-mgmt-<VERSION>.zip
api-mgmt-devportal-<VERSION>.tar API Mgmt 7.6.0
api-mgmt-gateway-<VERSION>.tar
api-mgmt-keycloak-<VERSION>.tar
api-mgmt-ui-<VERSION>.tar
```



Later on, in step 6, you will need file **apiman-default-config.json** from archive api-mgmt-<VERSION>.zip on an API Management client, so copy the ZIP file to your client now (or later).

2. **Extract** api-mgmt-<VERSION>.zip.
Extracting this file will create a folder **api-mgmt** which is required permanently. It contains configuration files with sensitive data such as passwords, so access to this folder should be restricted. The Docker container however must be able to read the data.
3. **Load** the Docker images:

```
docker image load -i api-mgmt-gateway-<VERSION>.tar
```

```
docker image load -i api-mgmt-ui-<VERSION>.tar
```

```
docker image load -i api-mgmt-keycloak-<VERSION>.tar
```

```
docker image load -i api-mgmt-devportal-<VERSION>.tar
```

On this Page:

- [Prerequisites](#)
- [Step 1: Download and Extract the Software](#)
- [Step 2: Configure the Installation Settings](#)
- [Step 3: Prepare the Certificate](#)
 - [Use Official Certificate](#)
 - [Use Self-Signed Certificate](#)
- [Step 4: Configure the Authentication Service \(Keycloak\)](#)
- [Step 5: Start All Services](#)
- [Step 6: Login to API Management](#)
- [Further Configurations](#)

Related Pages:

- [Certificates and Keystores](#)

Related Documentation:

- [Metrics](#)

Docker:


- [Docker documentation](#)




Elasticsearch:




- [Install Elasticsearch with Docker > Notes for production use and defaults](#)
- [Setting the heap size](#)

Step 2: Configure the Installation Settings

Configure the installation settings in the Docker `.env` file as described below. This file resides in folder `api-mgmt/single-host-setup`.

 `.env` is a hidden file.

Setting	Description	Default Value	Since Version
BRIDGE_URL	<p>Provide your BRIDGE hostname(s). You may specify more than one URL as a comma-separated list.</p> <div> Do not use white-spaces in this list: <code>bridge1.acme-corp.com,bridge2.acme-corp.com</code></div>	<code>bridge.acme-corp.com</code>	
BRIDGE_PORT	<p>Provide the BRIDGE port(s). If you have added a list of BRIDGE URLs with setting <code>BRIDGE_URL</code>, the following applies:</p> <ul style="list-style-type: none">• If all BRIDGES have the same port, specify only one port here.• If the BRIDGES have different ports, specify a list of ports as a comma-separated list. The order of ports must correspond to the order of BRIDGE URLs. <div> Do not use white-spaces in this list: <code>18080,18081</code></div>	8080	API Mgmt 7.3.0
BRIDGE_USERNAME	<p>Provide the user name to access your BRIDGE.</p> <p>If you have configured multiple BRIDGES with <code>BRIDGE_URL</code>, these credentials are valid for all BRIDGES. We therefore recommend to use a technical user with permissions limited to role USER.</p>	username	
BRIDGE_PASSWORD	<p>Provide the password to access your BRIDGE.</p> <p>If you have configured multiple BRIDGES with <code>BRIDGE_URL</code>, these credentials are valid for all BRIDGES. We therefore recommend to use a technical user with permissions limited to role USER.</p>	password	
DEV_PORTAL_PORT	<p>Provide the port to access the user interface of the API Management Developer Portal.</p>	8447	API Mgmt 7.6.0
ELASTICSEARCH_JAVA_MEMORY	<p>Provide the amount of memory that Elasticsearch can allocate.</p> <p>2 GB for Elasticsearch is the default value. Please refer to the official Elasticsearch documentation for more information on memory usage and how to determine the needs for your setup (see Install Elasticsearch with Docker > Notes for production use and defaults and Setting the heap size).</p>	2g	
ENDPOINT	<p>Provide the hostname of your API Management HOST.</p> <div> Please note that you cannot use localhost as an endpoint.</div>	<code>api.acme-corp.com</code>	
GATEWAY_PORT	<p>Provide the port to access the published APIs.</p>	8444	
KEYCLOAK_ADMIN_USERNAME	<p>Define a Keycloak user.</p>	username	API Mgmt 7.3.0
KEYCLOAK_ADMIN_PASSWORD	<p>Set a password for the Keycloak user.</p>	password	API Mgmt 7.3.0
KEYCLOAK_PORT	<p>Provide the port to access Keycloak.</p>	8445	

KIBANA_ENCRYPTION_KEY	<p>Set a key to encrypt cookies. You should change the default value to an alphanumeric string with 32 characters. We recommend to use a password generator to generate a random password.</p> <div>  This setting is optional and only necessary if you want to use Kibana to analyze your API metrics. </div>	J844Az3f0s74IiUepVgfX0XqgH1hcUyK	API Mgmt 7.4.0
KIBANA_PORT	<p>Provide the port to access Kibana.</p> <div>  This setting is optional and only necessary if you want to use Kibana to analyze your API metrics. </div>	8446	API Mgmt 7.4.0
MYSQL_PASSWORD	Set a password for the Keycloak database connection.	password	
MYSQL_ROOT_PASSWORD	Set a root password for the MySQL installation.	root_password	
SELF_SIGNED	<p>Specify certificate usage.</p> <ul style="list-style-type: none"> • Leave as true if you are using a self-signed certificate. • Set to false if using a valid certificate from your organization. 	true	
TRUSTSTORE_KEYSTORE_PASSWORD	<p>Provide the password of the Java truststore file that contains the certificates. The password must be at least 6 characters long.</p> <p>This password will also be used if you create a keystore with our script to generate self-signed certificates (see section Use Self-Signed Certificate below).</p> <div>  The password must not contain the ' sign (single quote) and \$ must be escaped like \$\$ because it is a docker-compose specific symbol. </div>	secret	
UI_PORT	Provide the port to access the API Management user interface.	8443	

Step 3: Prepare the Certificate

You need a certificate to establish secure connections between clients and API Management, as well as between the different components of API Management itself. You can use an official certificate, or you can create a self-signed one.

Refer to [Certificates and Keystores](#) for more information on certificate and keystore handling.



We recommend using an official and valid certificate.



Folder **api-mgmt\configs** already contains an example structure of the needed files.



This setup is designed to run on one host only. The SSL KeyStore is shared between all services.

Use Official Certificate

To use your official certificate, proceed as follows:

1. **Copy apiman.jks** and the **tls** files of your certificate (**tls.crt** and **tls.key**) to folder **api-mgmt\configs** (see example structure).

2. **Create** a Java keystore which includes the certificate. The keystore must be secured by a store password. Assign the name **apiman.jks** to the keystore file. Refer to [Certificates and Keystores](#) for some hints on certificate handling.
3. **Update** the following entries in the Docker **.env** file (see table above for details):
 - Update entry `TRUSTSTORE_KEYSTORE_PASSWORD` with the store password.
 - Set entry `SELF_SIGNED` to **false**.

If you have certificates and intermediate certificates, please consult the keycloak documentation [Keycloak Docker image > Setting up TLS\(SSL\)](#).

Use Self-Signed Certificate

If you do not possess a valid certificate, you can create a self-signed one. You can use Java Keytool to do this.

1. **Check the folder permissions.** Docker should be able to write to the folder and execute programs in the folder (Linux folder permission 300).
2. **Generate a keystore.** To do this, run the following command from folder **api-mgmt/single-host-setup** (folder containing the file **docker-compose.yml**):

```
docker-compose run --no-deps --rm --entrypoint '/opt/api-mgmt/create-self-signed-certificates.sh' keycloak
```



Some shells (e.g. git bash) have problems with the path so you may have to escape the slashes with backslash like `\opt\api-mgmt\create-self-signed-certificates.sh`.

3. **Copy** the following generated files from folder **api-mgmt/configs/ self-signed-certificates** to folder **api-mgmt/configs/**:
 - apiman.jks
 - tls.crt
 - tls.key

Step 4: Configure the Authentication Service (Keycloak)

Keycloak is an open source identity and access management solution and is used to create and manage the users of API Management and OAuth2 secured APIs. Before you can start the Docker containers, you need to change some of the Keycloak settings in the Docker configuration file **.env**. The values you need to replace the default values with, are to be obtained from your Keycloak instance.

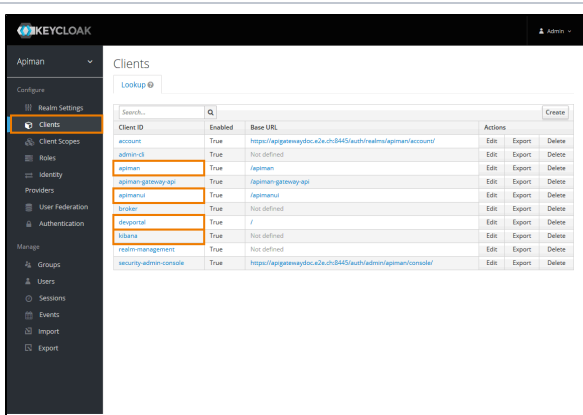
1. **Start Keycloak.** To do this, run the following command from folder **api-mgmt/single-host-setup** (folder containing the file **docker-compose.yml**):

```
docker-compose up keycloak
```

Keycloak has been started when the log reads something like

```
[...] Keycloak 7.3.0 (WildFly Core 6.0.2.Final) started in 50476ms -  
Started 673 of 933 services [...]
```

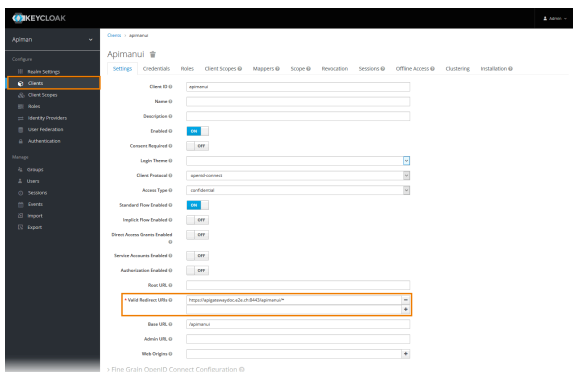
2. **Open your Keycloak URL**, e.g. <https://api.acme-corp.com:8445/auth/admin>, and login to the administration console. To login, use username and password as configured in the **.env** file.
3. **Change the Valid Redirect URIs** of the below listed clients.



Navigate to **Clients**

You will have to change the settings for the following clients:

- apiman
- apimanui
- devportal
- kibana



Open the **Settings** for each of the clients by clicking on the client ID in the list, and change the entry in the **Valid Redirect URIs** to match your setup:

• For client **apiman**, e.g. <https://apiman.acme-corp.com:8443/apiman/>

• For client **apimanui**, e.g. <https://api.acmecorp.com:8443/apimanui/>

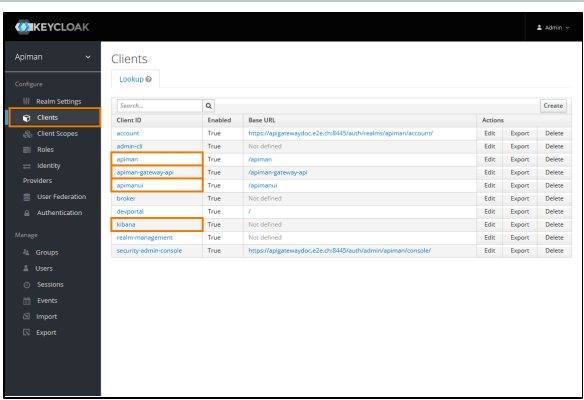
• **F**
o
r
cl
ie
nt
d
e
v
p
o
rt
al
,
e.
g.
ht
tp
s:
//
a
pi
.
a
c
m
e-
c
o
r
p.
c
o
m
:
8
4
4
7
/*

	<ul style="list-style-type: none"> For client kibana, e.g. https://api.acme-corp.com:8446/authorization
--	---

4. Now, **change all default credentials** for clients **apiman** , **apimanui** , **apiman-gateway-api** and **kibana** and copy the new passwords to the Docker configuration file.

Setting	Description	Default Value	Since Version
KEYCLOAK_APIMAN_SECRET	Provide the secret for client apiman generated in Keycloak.	password	
KEYCLOAK_GATEWAY_SECRET	Provide the secret for client apiman-gateway-api generated in Keycloak.	password	
KEYCLOAK_APIMANUI_SECRET	Provide the secret for client apimanui generated in Keycloak.	password	
KEYCLOAK_REALM_PUBLIC_KEY	Provide the realm public key generated in Keycloak.	MIGfMA0GCSqGS Ib3DQEBAQU [...]	
KEYCLOAK_KIBANA_SECRET	Provide the secret for client kibana generated in Keycloak.	password	API Mgmt 7.4.0

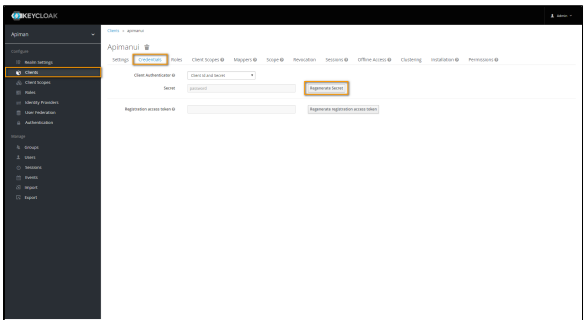
Do this as follows:



Navigate to **Clients**.

You will have to change the settings for the following clients:

- a pi man
 - a pi man-gate way-a pi
3. a pi man ui
4. ki b a na



For each client, go to tab **Credentials** and click **Regenerate Secret**.

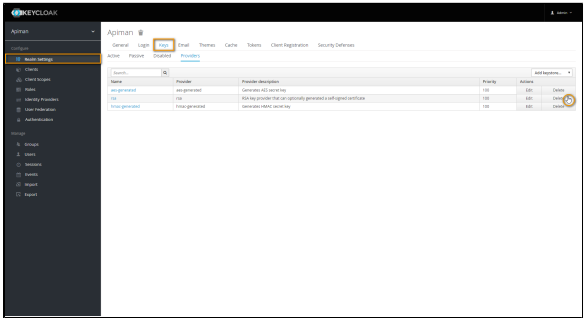
In the Docker `.env` file,

• s
et
th
e
g
e
n
e
r
at
e
d
s
e
cr
et
fo
r
cl
ie
nt
a
p
i
m
an
to
e
nt
ry
K
E
Y
C
L
O
A
K
—
A
P
I
M
A
N
—
S
E
C
R
ET
.

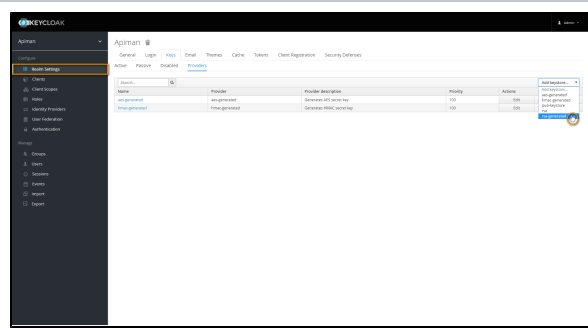
• s
et
th
e
g
e
n
e
r
at
e
d
s
e
cr
et
fo
r
cl
ie
nt
a
p
i
m
a
n
-
g
a
t
e
w
a
y
-
pi
to
e
nt
ry
K
E
Y
C
L
O
A
K
—
G
A
T
E
W
A
Y
—
S
E
C
R
ET
.

• s
et
th
e
g
e
n
e
r
at
e
d
s
e
cr
et
fo
r
cl
ie
nt
a
p
i
m
a
n
ui
to
e
nt
ry
K
E
Y
C
L
O
A
K
—
A
P
I
M
A
N
U
I
—
S
E
C
R
ET
.

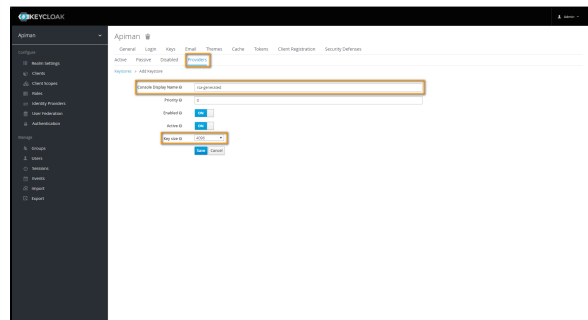
• set the generated secret for client ki b a n a to entry KEYCLOCK — K I B A N A — S E C R E T .



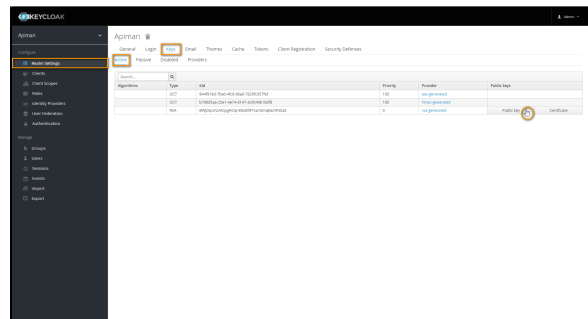
In Realm Settings go to tab Providers and delete provider rsa.



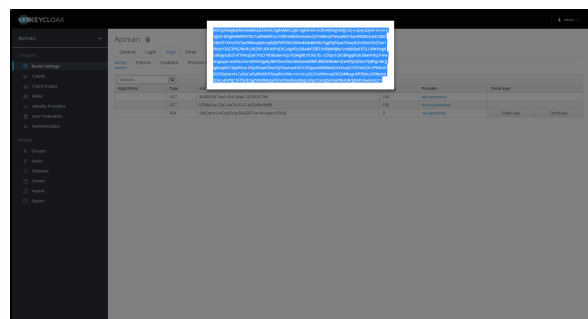
Next, choose **rsa-generated** from the select box **Add Provider**.



In the upcoming dialog, set **Key size** to the maximum available value and **save** your changes.



Go back to tab **Active** and click **Public key**.



The key will be displayed in a separate pop-up window.

Copy the key and paste it to variable **KEY_CLOAK_REALM-PUBLIC_KEY** in the Docker **.env** file.

Step 5: Start All Services

All Keycloak-related settings have been configured now, and you need to stop **docker-compose** and restart all containers.

1. **Stop** the Keycloak container by pressing **Ctrl-C**.
2. **Check** the configuration, if necessary.



You can check, if everything has been configured, with

```
docker-compose config
```

This command will list the Docker configuration and will throw warnings, if something is still missing.

3. Now start the containers. You can start all containers (including Kibana) or all containers except Kibana.

Go to folder **api-mgmt/single-host-setup** (folder that contains the file **docker-compose.yml**).

Start all containers by running the following command:

```
docker-compose up
```

To run the containers in the background, use:

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

To start all containers except Kibana, use:

```
docker-compose up ui gateway
```

or

```
docker-compose up -d ui gateway
```

Step 6: Login to API Management



Starting all services (previous step) may take some time. If the UI is not available yet, just wait a moment.

1. **Open** the URL of your API Management, e.g. <https://api.acme-corp.com:8443> and **log in** with the standard administration user, which is **admin/admin**.
2. **Change** the admin password to a new one of your choice as demanded on first login.
3. **Upload** the initial configuration of API Management.
To do this, go to **Administration > Export/Import** and import file **api-mgmt/configs/bootstrap/apiman-default-config.json** from api-mgmt-<VERSION>.zip. If you did not transfer the file to your client in step1, transfer this file now.



If you want to restore a backup, use the backup file instead of the default configuration (see page [API Management Backup and Restore](#) for details).

4. **Test** the connection between UI and gateway.
To do this, go to **Administration > Manage Gateways** and click on the name of the gateway. Then, click **Test Gateway**.

Edit Gateway

Update this gateway's details, for example when a gateway's endpoint or authentication information has changed..

Gateway Name
The Gateway

Description
This is the default gateway.

Configuration Endpoint
\$apiman.gateway-endpoint.https://gateway.8081/

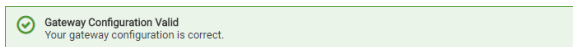
Configuration Endpoint Credentials

Username:
\$apiman.gateway-endpoint.username.gateway

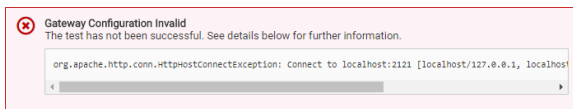
Password:

Test Gateway
Update Gateway
Cancel
Delete Gateway

If the gateway configuration is correct, you will get a success message:



If the configuration is invalid, an error message will be shown including further information about the error itself:



A technical user (**gateway/gateway**) connects the UI of API Management with the gateway. It is highly recommended to change the default passwords for both, the **admin** user and the **gateway** user. Both passwords can be changed in [Keycloak](#).

Further Configurations

- **Advanced Settings**

Please refer to [Advanced API Management Settings](#) for an explanation of some additional advanced settings that can be configured for API Management.

- **Kibana**

If you want to use Kibana for your reports on API Management metrics, you may want to proceed with [Setting up Kibana](#).



However, as Kibana get be setup any time after having installed API Management, we recommend to [configure API Management](#) first and record some API data, before using Kibana.