API Types

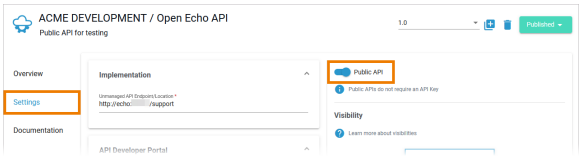
Scheer PAS *API Management* supports the creation and management of two different types of APIs:

- **public APIs**
- **private APIs**

The differences between these two types are often not clear to users, although they are quite simple. In short: While public APIs can be consumed by everyone (assuming no additional security policy has been set), private APIs can only consumed by using an API contract.

In addition to the differences between the two API types, they also have two things in common:

- Both public and private APIs can be protected and further restricted by applying different policies.
- The choice of an API type is not immutable: You can change the API type in the API settings at any time



> However, we do not recommend changing the API type once the API has gone live and is in production, as active clients and contracts may already have been created.

On this Page:

Related Pages:

- The Concepts of API Management
- APIs
- Clients
- Contracts
- Plans
- Developer Portal
  - Consuming a Public API
  - Subscribing to a Private API

Public vs Private API: An Overview

Public APIs

Public APIs are accessible to everyone.
Therefore, public APIs are best suited if you want to make your API accessible to everyone.

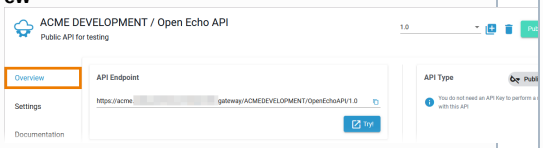
Private APIs

Private APIs need a contract to be used by clients.
They are best when you want to limit access to the API by selected users and customers.

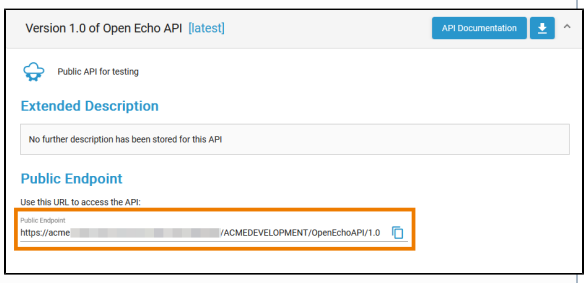
Public APIs do **not require** an **API Key** in order to be called.

You can find the public endpoint for the API displayed:

- in API Management > API details page > tab **Overview**



- in the API Developer Portal > API details. Refer to Consuming a Public API for details.

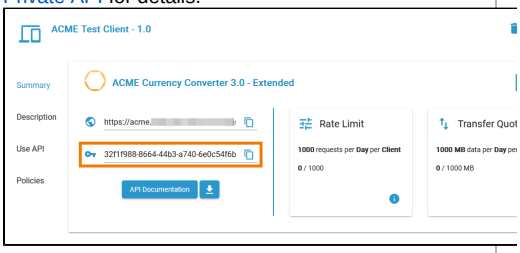


Private APIs **require** an **API Key** in order to be called.

You can find the API Key displayed:

- in API Management > API details page > tab **Contracts**
- in API Management > Client details page > tabs **Overview** and **Contracts**

- in the API Developer Portal > **My Clients** > Contract with the related API. Refer to Subscribing to a Private API for details.



Advantages of Public APIs

- It is very easy to consume a public API - you just need to know its public endpoint.
- Clients do not need to register for a public API: Neither a client nor a contract are necessary.
- Compared to a private API, a public API requires less configuration.

Disadvantages of Public APIs

- You have no control over the users of your API (unless the API is further restricted by policies).
- Since a private API is publicly accessible, there is a higher risk of misuse.
- Regarding the available metrics, you cannot distinguish who the caller of your API was.

Common Use Cases for Public APIs

You should use public APIs...

- ... if you want to provide your API to everyone.
- ... if you want to enable internal access without (too many) restrictions.

Advantages of Private APIs

- Access to private APIs can be controlled in a very fine-grained manner, e.g. per approval workflow (refer to Handling Approval Requests for details).
- Different access levels can be mapped by assigning restricted usage plans to your API.
- You have insight into various metrics that can be displayed per client (refer to Metrics for details).
- In addition to the policies assigned to the API, you can also assign policies to each client (refer to Policies > Policy Chain for details). This enables further changes or restrictions, e.g. different authentication methods for customers.

Disadvantages of Private APIs

- To consume a private API, a client and a contract must be created.
- Compared to a public API, a private API requires more complex configuration.

Common Use Cases for Private APIs

You should use private APIs...

- ... if you want to grant access to selected customers.
- ... if you want to monetize your APIs.

API Contracts and API Keys

Only public APIs can be accessed by any consumer. The only way for a client to consume a private API is by using an API contract. An API contract is a link between a client and an API through a plan offered by that API.

API contracts can only be created between clients and published APIs which are offered through at least one plan. An API contract cannot be created between a client and a public API.

When a client version is created, the system generates a unique API Key. This key is unique per client version and the same for all contracts of this version. All requests made to the API by a client through the gateway must include this API Key to identify the used client version.

You can forward the X-API-Key to the service using the **API Key policy**. However, you cannot define your own value for the X-API-Key, since the gateway uses the key to identify the clients.

However, **the API Key is not a security feature**! API Keys are not encrypted and visible:

- in the request header,
- to people who have access to API Management metrics/the Log Analyzer,
- in the logs of the integration component (Bridge) if you are using the **API Key** policy.

So, API Keys need to be handled in a secure way - otherwise attackers may be able to use the API Key to gain access to your system.

As per definition, API Keys are used to identify technical clients only and, subsequently, to apply related policies. **Do not use API Keys to authenticate users.**

Authentication should always be implemented via a dedicated security policy (refer to Policy Configuration > Security Policies and API Security: Authentication and Authorization).