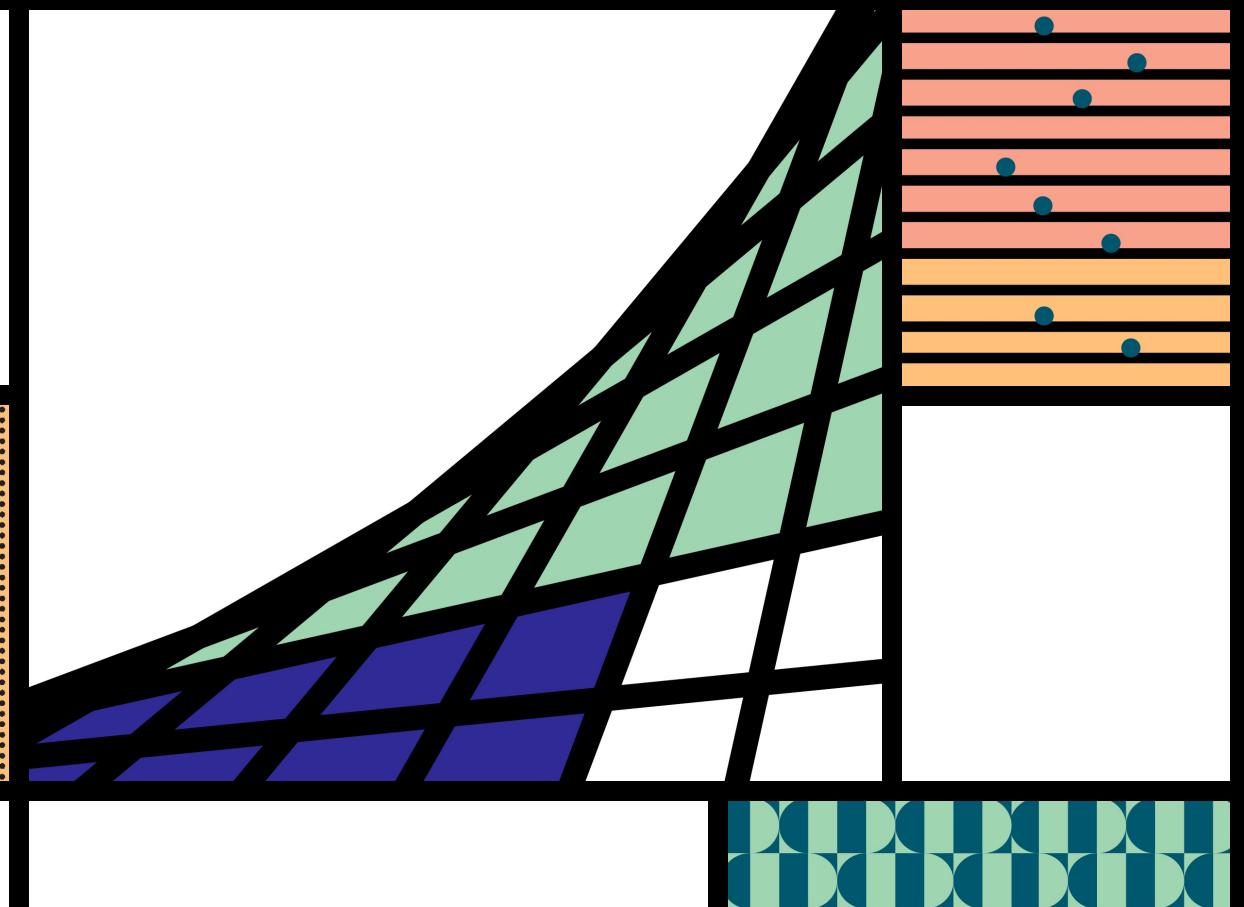
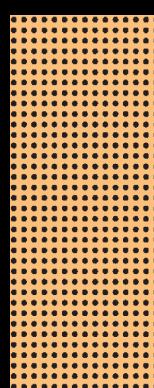
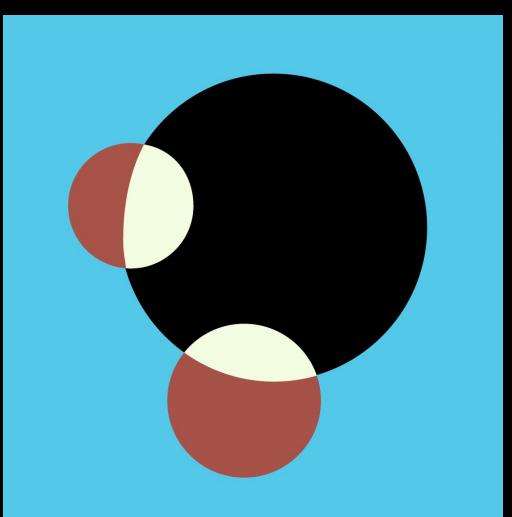
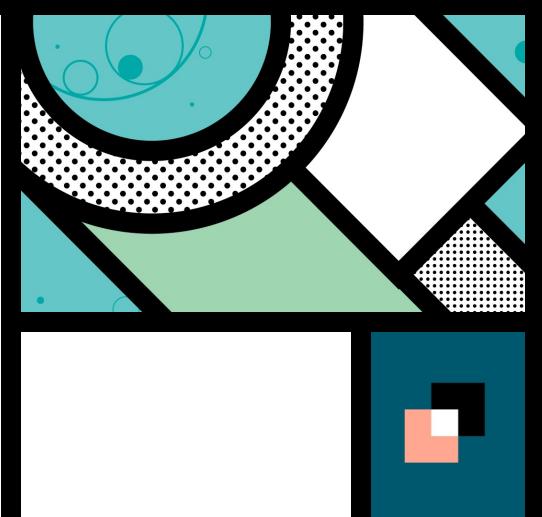


IAM Training



Guillaume Rongier

SALES ENGINEER



Index

Introduction IAM

Installation

- IRIS.key
- /api/iam/licence
- User IAM

IRIS and IAM

- Docker-compose
- Dockerfile
 - Multistage
- IAM Startup
 - Bootstrap
 - Run

First proxy

- Service/Roue
 - Test
- Plugin
 - Limite rate
 - Security
 - ACL

Documentation/Dev portal

- Publish a spec
- Tester
- Secure Doc
- Secure IAM



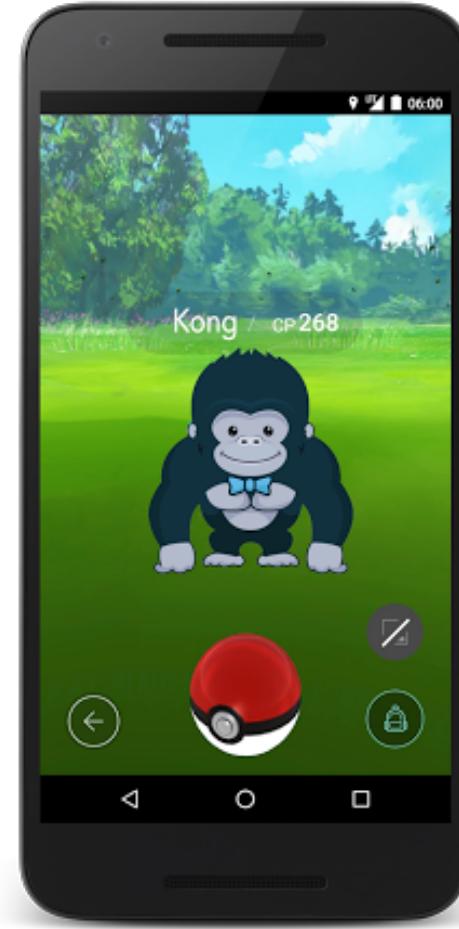
Index

Options :

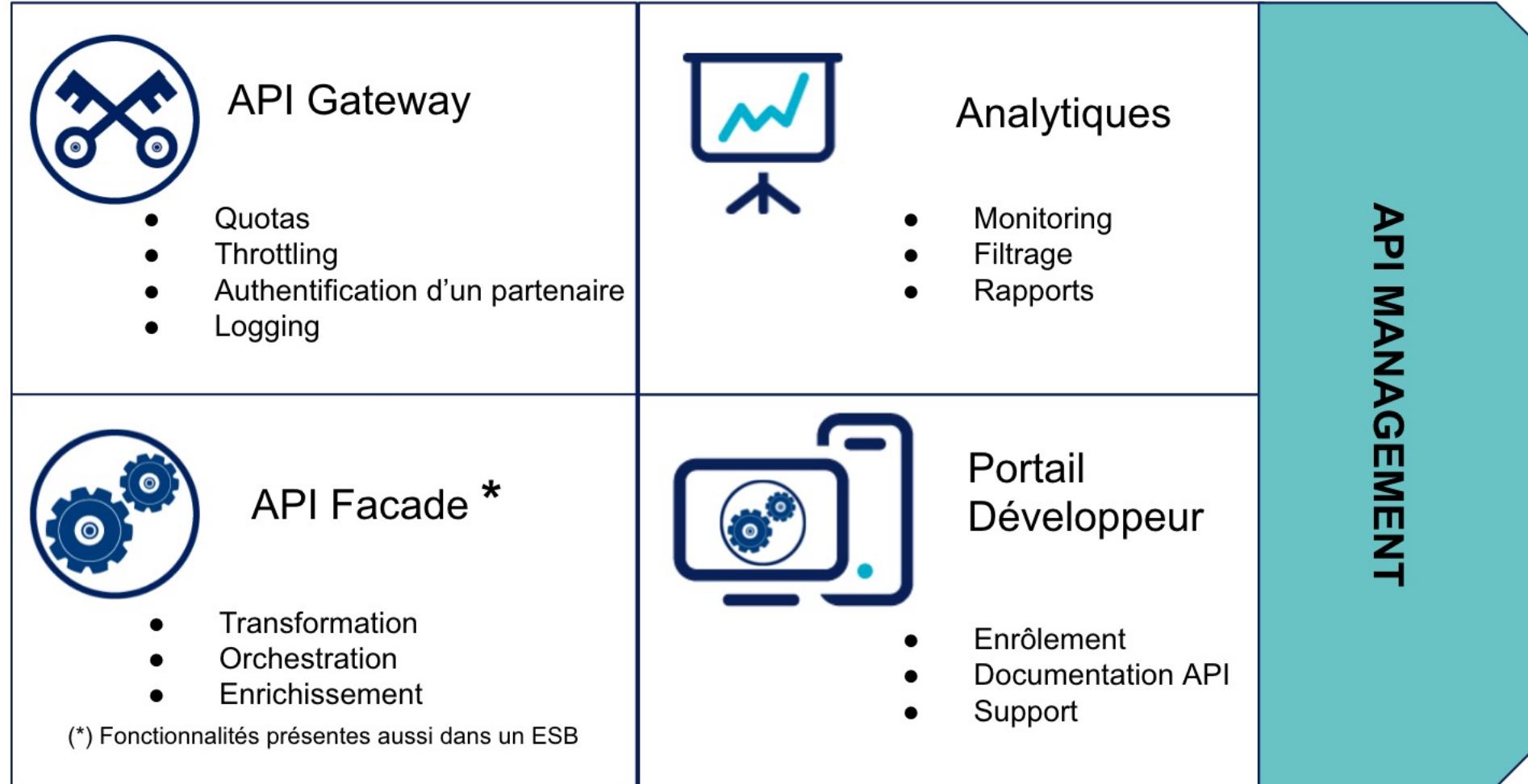
- Import a custom plugin
- Build a custom plugin
- DevOps



Introduction

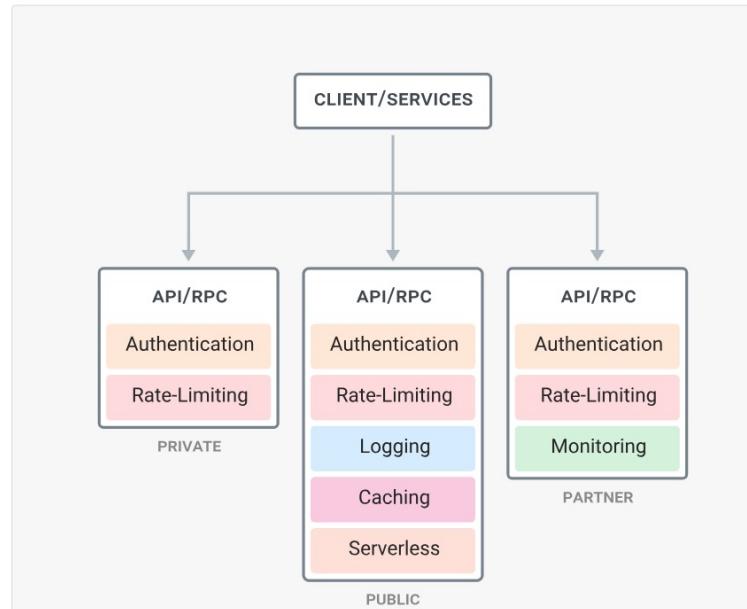


Introduction to IAM aka KONG



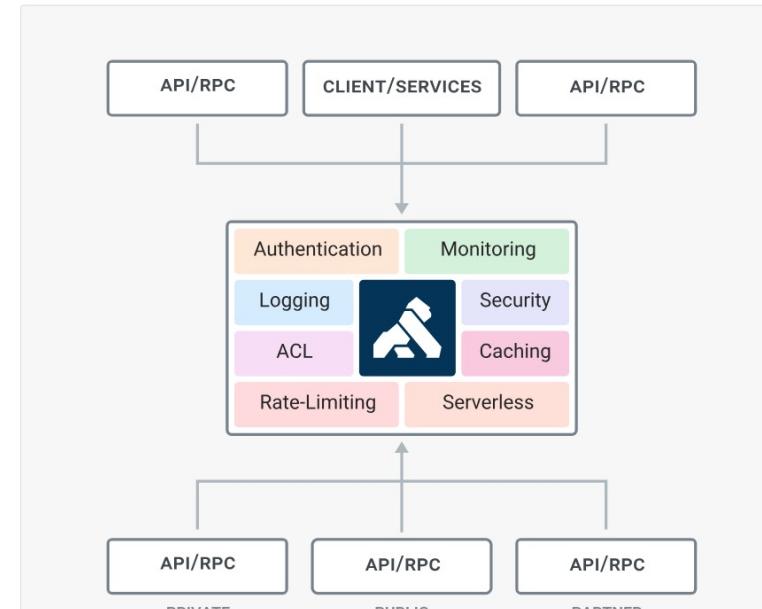
Kong way

The Redundant Old Way



- ✗ Common functionality is duplicated across multiple services
- ✗ Systems tend to be monolithic and hard to maintain
- ✗ Difficult to expand without impacting other services
- ✗ Productivity is inefficient because of system constraints

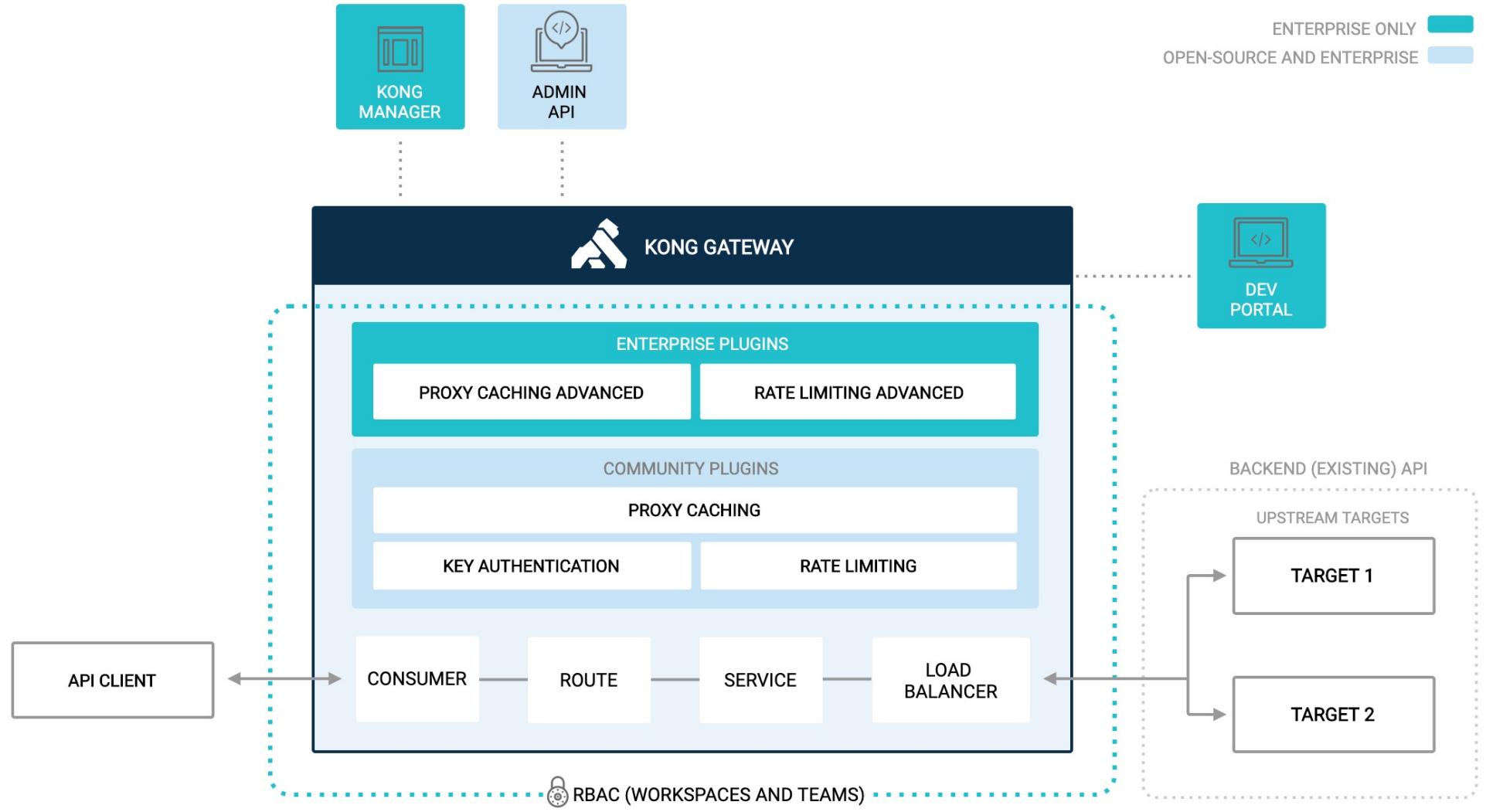
The Kong Way



- ✓ Kong orchestrates common functionality
- ✓ Build efficient distributed architectures ready to scale
- ✓ Expand functionality from one place with a simple command
- ✓ Focus on your product and let Kong do the REST



Kong overview



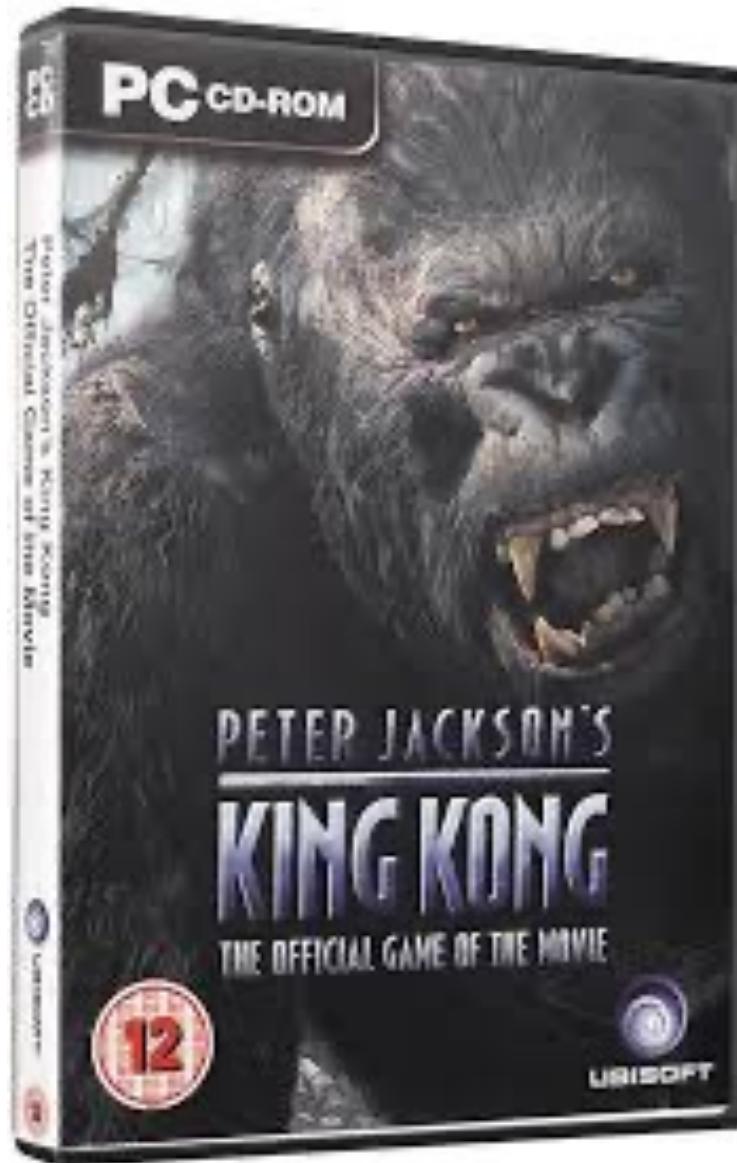
IAM Configuration

The screenshot shows the InterSystems API Management interface. The top navigation bar includes links for Workspaces, Dev Portals, Vitals, Teams, and Service Map. The left sidebar has a 'default' workspace selected. The main content area is titled 'default Workspace' and displays various metrics and charts related to API traffic and consumer activity.

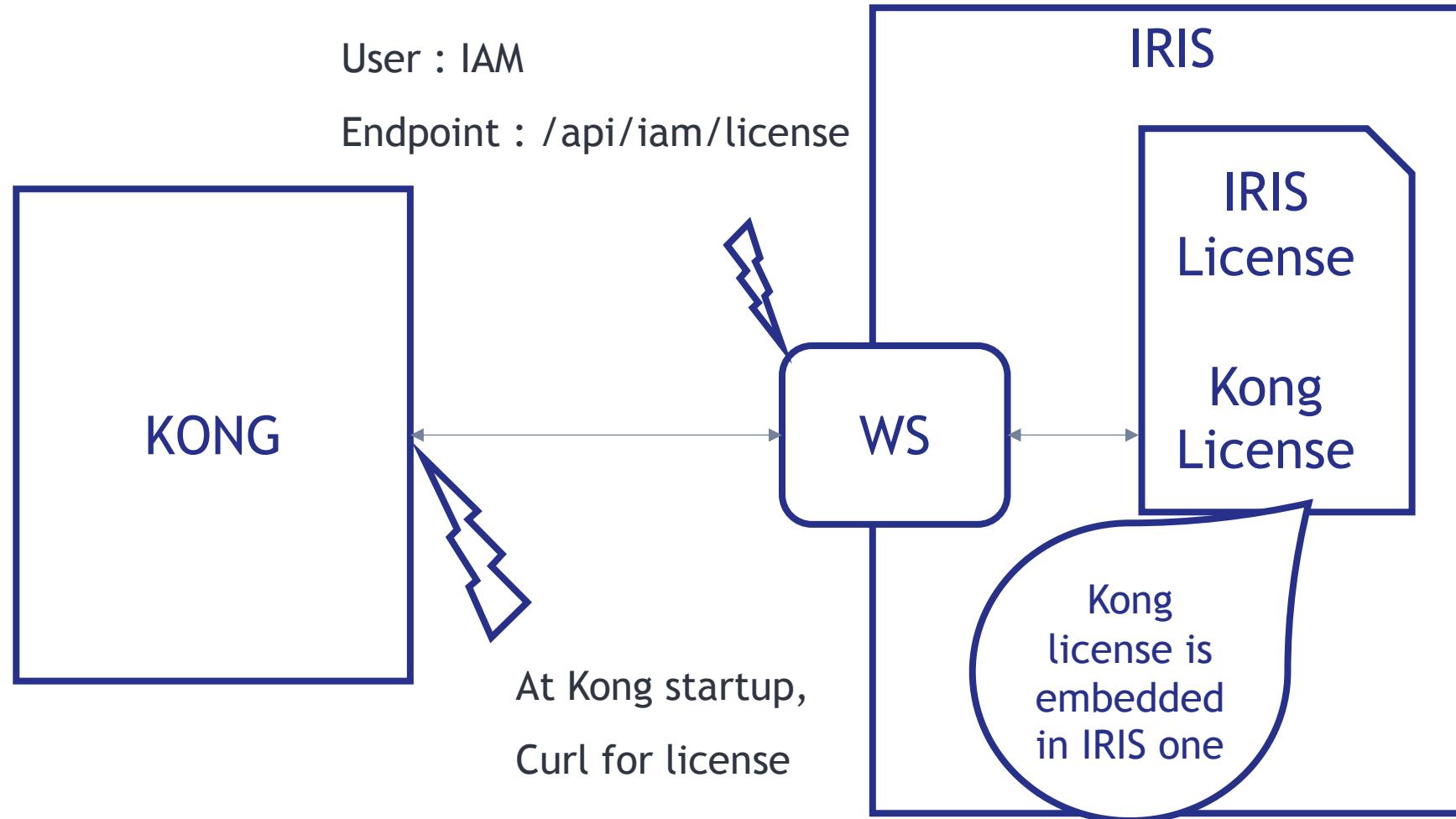
```
curl -i -X POST \
--url http://localhost:8001/apis/nooi-store/plugins/ \
--data 'name=key-auth'
```

Adresse:port du serveur Les APIs L'API cible Ses plugins
Nom du plugin

Installation



How kong work's with IRIS



Pre-requis

Git

Docker

Iris.key

<https://container.intersystems.com/>

Iam docker image : containers.intersystems.com/intersystems/iam:1.5.0.9-4

Iris docker image : containers.intersystems.com/intersystems/irishealth:2020.4.0.524.0



Git clone / fork

Template available here :

<https://github.com/grongierisc/iam-training>

Main -> to follow training

Training -> corrections

What's in ?

An IRIS back-end with a simple rest api with pre-install data.

<http://localhost:52773/swagger-ui/index.html#/>



What we have to do, to install kong ?

Change IRIS community edition to a licensed one.

- containers.intersystems.com/intersystems/irishealth:2020.4.0.524.0
- Add iris.key in key folder

```
ARG
IMAGE=containers.intersystems.com/intersystems/irishealth:2020.4.0.524.0

FROM $IMAGE as iris-iam

COPY key/iris.key
/usr/irissys/mgr/iris.key

COPY iris-iam.script /tmp/iris-iam.script

RUN iris start IRIS \
&& iris session IRIS < /tmp/iris-iam.script \
&& iris stop IRIS quietly

FROM iris-iam
... <- old dockerfile
```

What we have to do, to install kong ?

Change IRIS community edition to a licensed one.

- containers.intersystems.com/intersystems/irishealth:2020.4.0.524.0
- Add iris.key in key folder

Enable IAM endpoint and USER in IRIS

- Build a new IRIS image with this new config
- Create an iris-jam.script
 - Add key
 - Enable user
 - Enable endpoint

```
zn "%SYS"  
wri"Create web application ...",!  
set webName = "/api/iam"  
set webProperties("Enabled") = 1  
set status =  
##class(Security.Applications).Modify(webName,  
.webProperties)  
write:'status $system.Status.DisplayError(status)  
write "Web application "_webName_" was updated!",!  
set userProperties("Enabled") = 1  
set userName = "IAM"  
Do  
##class(Security.Users).Modify(userName,.userPrope  
rties)  
write "User "_userName_" was updated!",!  
halt
```

What we have to do, to install kong ?

Change IRIS community edition to a licensed one.

- containers.intersystems.com/intersystems/irishealth:2020.4.0.524.0
- Add iris.key in key folder

Enable IAM endpoint and USER in IRIS

- Build a new IRIS image with this new config
- Create an `iris-iam.script`
 - Add key
 - Enable user
 - Enable endpoint

Add IAM in docker-compose

- db
 - Postgres database for IAM
- iam-migration
 - Bootstrap the database
- iam
 - Actual IAM instance
- A volume for data persistent

```
iam-migrations: crafter db rld,jwt-crafter failure
image: ISC_IRIS_URL: KONG_DATABASE: ISC_IRIS_URL: db:
intersystems/ia IAM:${IRIS_PASS postgres IAM:${IRIS_PASS image:
m- WORD}@iris:5277 KONG_PG_DATABASE WORD}@iris:5277 postgres:9.6
custom:1.5.0.9- 3/api/iam/licen E: 3/api/iam/licen environment:
4 se ${KONG_PG_DATABASE} se POSTGRES_DB:
build: restart: on- ASE:-iam} volumes: ${KONG_PG_DATABASE}
context: iam failure KONG_PG_HOST: - ./iam:/iam ASE:-iam}
dockerfile: links: db links: POSTGRES_PASSWORD
dockerfile - db:db KONG_PG_PASSWORD - db:db RD:
command: kong iam D: ports: ${KONG_PG_PASSWORD}
migrations image: ${KONG_PG_PASSWORD} - target: 8000 ORD:-iam}
bootstrap up intersystems/ia ORD:-iam} published: 8000 POSTGRES_USER:
depends_on: m- KONG_PG_USER: protocol: tcp ${KONG_PG_USER}:
- db custom:1.5.0.9- ${KONG_PG_USER}: - target: 8001 -iam}
environment: 4 -iam} published: 8001 volumes:
KONG_DATABASE: build: KONG_PROXY_ACCE protocol: tcp -
postgres context: iam SS_LOG: - target: 8002 'pgdata:/var/li
KONG_PG_DATABASE dockerfile: /dev/stdout published: 8002 b/postgresql/da
E: dockerfile KONG_PROXY_ERROR protocol: tcp ta'
${KONG_PG_DATABASE depends_on: R_LOG: - target: 8003 healthcheck:
ASE:-iam} - db /dev/stderr published: 8003 test: ["CMD",
KONG_PG_HOST: environment: KONG_PORTAL: protocol: tcp "pg_isready",
db KONG_ADMIN_ACCE 'on' - target: 8004 "-U",
KONG_PG_PASSWORD SS_LOG: KONG_PORTAL_GUI published: 8004 "${KONG_PG_USER}
D: /dev/stdout _PROTOCOL: http protocol: tcp :-iam"]
${KONG_PG_PASSWORD} KONG_ADMIN_ERROR KONG_PORTAL_GUI - target: 8443 interval: 30s
ORD:-iam} R_LOG: _HOST: published: 8443 timeout: 30s
KONG_PG_USER: /dev/stderr '127.0.0.1:8003 protocol: tcp retries: 3
${KONG_PG_USER}: KONG_ADMIN_LIST ' - target: 8444 restart: on-
-iam} EN: KONG_ADMIN_GUI_ published: 8444 failure
KONG_CASSANDRA_ '0.0.0.0:8001' URL: protocol: tcp stdin_open:
CONTACT_POINTS: KONG_ANONYMOUS_ http://localhost:8003 - target: 8445 true
db REPORTS: 'off' t:8002 published: 8445 tty: true
KONG_PLUGINS: KONG_CASSANDRA_KONG_PLUGINS: protocol: tcp volumes:
bundled,jwt- CONTACT_POINTS: bundled,hellowo restart: on- pgdata:
```

Option : IRIS_PASSWORD

How to paramètre IRIS_PASSWORD for Kong and IRIS.

We can use .env file to do so, then use \${env_var} to use our new password.

```
build:  
context: .  
dockerfile: dockerfile  
args:  
- IRIS_PASSWORD=${IRIS_PASSWORD}  
  
-----  
  
RUN echo "${IRIS_PASSWORD}" >  
/tmp/password.txt &&  
/usr/irissys/dev/Container/changePa  
ssword.sh /tmp/password.txt
```

Test It !

```
docker-compose -f "docker-compose.yml" up -d --build
```

Go to <http://localhost:8002> ← Kong/IAM portal

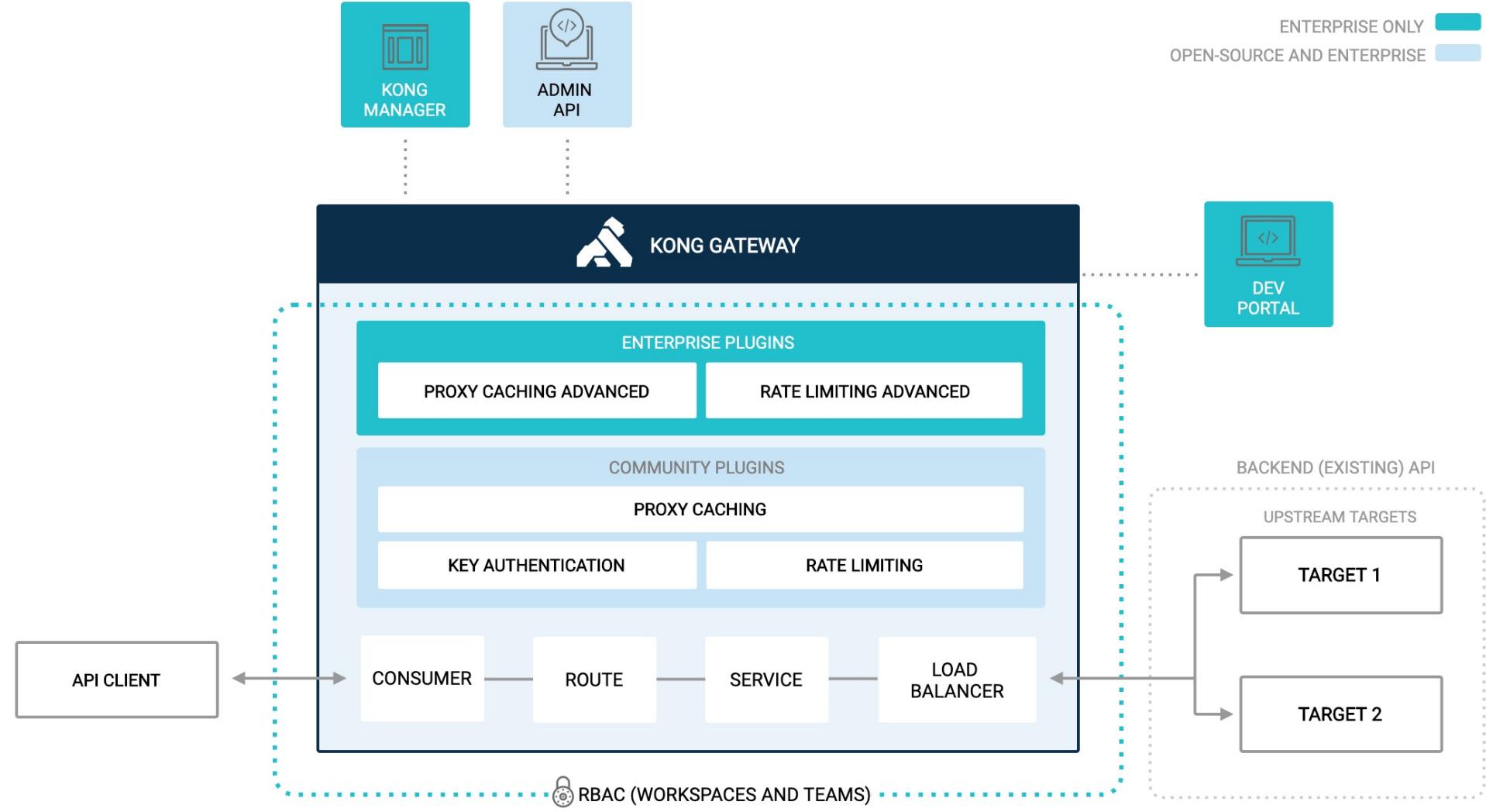
Port	Protocol	Description
:8000	HTTP	Takes incoming HTTP traffic from Consumers, and forwards it to upstream Services.
:8443	HTTPS	Takes incoming HTTPS traffic from Consumers, and forwards it to upstream Services.
:8001	HTTP	Admin API. Listens for calls from the command line over HTTP.
:8444	HTTPS	Admin API. Listens for calls from the command line over HTTPS.
:8002	HTTP	Kong Manager (GUI). Listens for HTTP traffic.
:8445	HTTPS	Kong Manager (GUI). Listens for HTTPS traffic.
:8003	HTTP	Dev Portal. Listens for HTTP traffic, assuming Dev Portal is enabled.
:8446	HTTPS	Dev Portal. Listens for HTTPS traffic, assuming Dev Portal is enabled.
:8004	HTTP	Dev Portal /files traffic over HTTP, assuming the Dev Portal is enabled.
:8447	HTTPS	Dev Portal /files traffic over HTTPS, assuming the Dev Portal is enabled.



First Service / Route / Plugin



First proxy



Create Service

Services >

Create Service

[View docs](#)

Name

crud

Add using URL

URL

http://iris:52773/crud

Add using Protocol, Host and Path

> [View 6 Advanced Fields](#)

[Create](#)

[Cancel](#)

Create service

```
curl -i -X POST \
--url
http://localhost:8001/services/
\
--data 'name=crud' \
--data
'url=http://iris:52773/crud/'
```



Create Route

Routes >

Create Route

[View docs](#)

Service

crud - a8e5aba8-d57a-4e5d-acd7-5f21b506cb7d

Name

Enter a Route Name

Protocols

http

Host(s) ③

Enter a Host

Method(s) ③

Enter a Method

Path(s)

/crud/persons/all



+ Add Path

Create route

```
curl -i -X POST \
```

```
--url
```

```
http://localhost:8001/services/  
crud/routes \
```

```
--data 'name=crud-route' \
```

```
--data 'paths=/persons/*' \
```

```
--data 'strip_path=false'
```



Test It !

Legacy

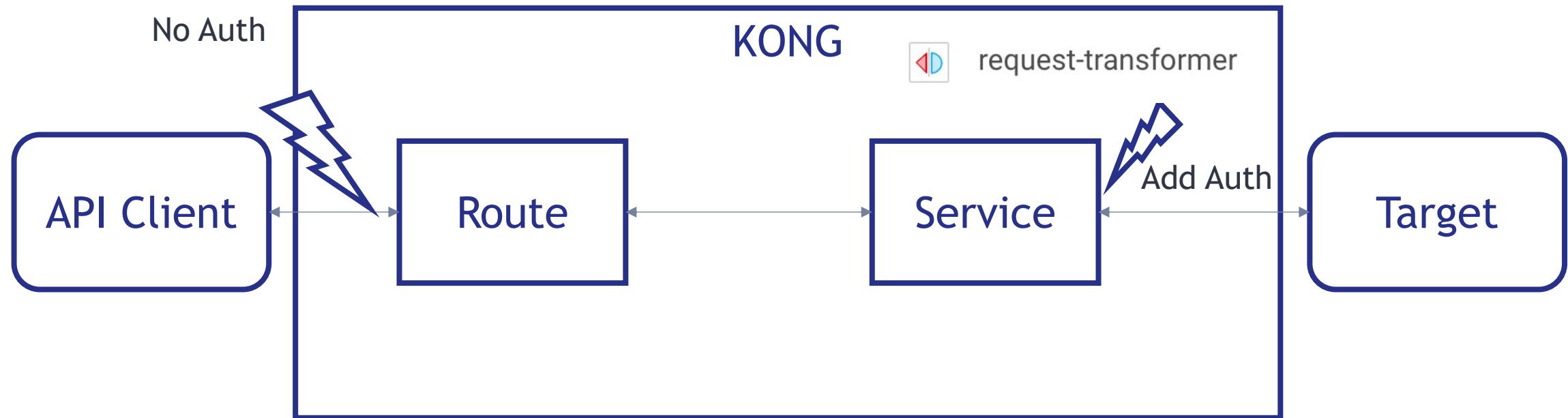
```
curl -i --location --request  
GET  
'http://localhost:52773/crud/  
persons/all' \  
--header 'Authorization:  
Basic U3VwZXJVc2Vy0lNZUw=='
```

KONG

```
curl -i --location --request  
GET  
'http://localhost:8000/person  
s/all' \  
--header 'Authorization:  
Basic U3VwZXJVc2Vy0lNZUw=='
```



Step Two, auto - authenticate



Step Two, auto - authenticate

Plugins >

Update request-transformer plugin

This plugin is Enabled

Global
All Services & Routes in this workspace

Scoped
Specific Services and/or Routes in this workspace

Service ②

Route ②

Consumer ②

Tags ②

... Add another tag

```
# Create plugin
curl -i -X POST \
--url
http://localhost:8001/services/crud
/plugins \
--data 'name=request-transformer' \
--data
'config.add.headers=Authorization:Basic U3VwZXJVc2Vy0lNZUw==' \
--data
'config.replace.headers=Authorization:Basic U3VwZXJVc2Vy0lNZUw=='
```

Test It !

Legacy

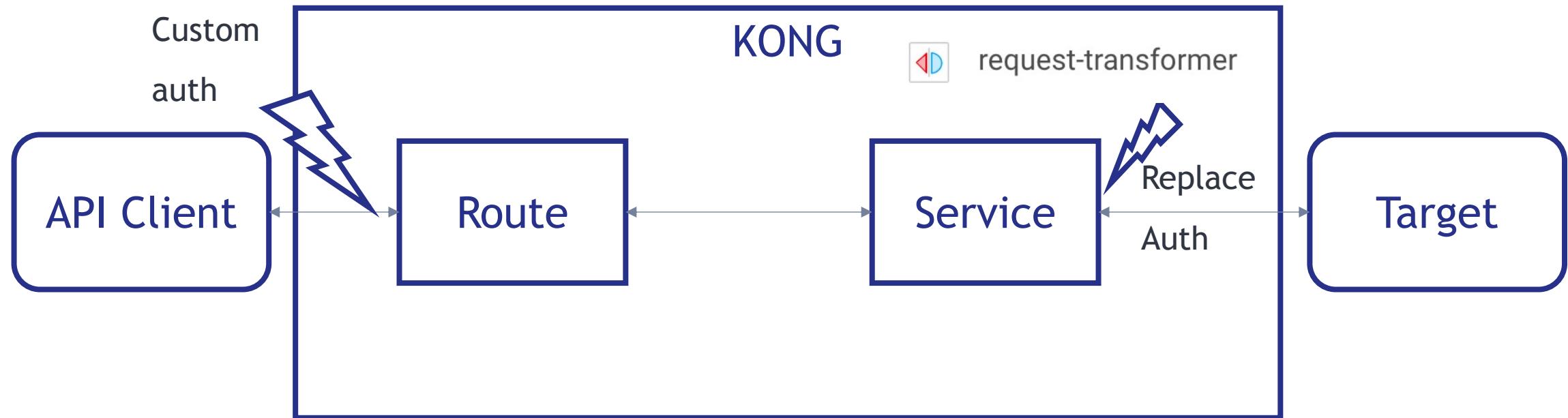
```
curl -i --location --request  
GET  
'http://localhost:52773/crud/  
persons/all'
```

KONG

```
curl -i --location --request  
GET  
'http://localhost:8000/person  
s/all'
```



Step Three, Add our own authentication



Add Consumers

Anonymous

Consumers >

Create Consumer

[View docs](#)

A unique **Username** or **Custom ID** is required.

Username

Custom Id

Tags ⓘ
Enter list of tags
e.g. tag1, tag2, tag3

Create **Cancel**

User

Consumers >

Create Consumer

[View docs](#)

A unique **Username** or **Custom ID** is required.

Username

Custom Id

Tags ⓘ
Enter list of tags
e.g. tag1, tag2, tag3

Create **Cancel**

Add consumer anonymous

```
curl -i -X POST \
--url http://localhost:8001/consumers/ \
--data "username=anonymous" \
--data "custom_id=anonymous"
```

Add consumer user

```
curl -i -X POST \
--url http://localhost:8001/consumers/ \
--data "username=user" \
--data "custom_id=user"
```



Add Basic auth plugin

Create new basic-auth plugin

This plugin is Enabled

Global
All Services & Routes in this workspace

Scoped
Specific Services and/or Routes in this workspace

Service ②

Select a Service

Route ②

afefe836-b9be-49a8-927a-1324a8597a9c

Tags ②

Enter list of tags

e.g. taa1, taa2, taa3

```
# Enable basic auth for service
curl -i -X POST
http://localhost:8001/routes/cr
ud-route/plugins \
--data "name=basic-auth" \
--data
"config.anonymous=5cc8dee4-
066d-492e-b2f8-bd77eb0a4c86" \
--data
"config.hide_credentials=false"
```



Add ACL plugin

Plugins > Select Plugin >

Create new acl plugin

This plugin is Enabled

Global
All Services & Routes in this workspace

Scoped
Specific Services and/or Routes in this workspace

Tags ②

Enter list of tags

e.g. tag1, tag2, tag3

Config.Blacklist

Config.Hide Groups Header

Config.Whitelist

user

Enable ACL

```
curl -i -X POST  
http://localhost:8001/routes/cr  
ud-route/plugins \  
--data "name=acl" \  
--data "config.whitelist=user"
```



Configure USER with ACL and credentials

The screenshot shows the InterSystems API Management interface. The left sidebar has sections for Workspaces, Dev Portals, Vitals, Teams, Service Map, API Gateway (selected), Consumers (selected), Basic Authentication, and Dev Portal. The main area shows a table for 'Consumers' with one row:

id	group	created_at
668b3a6b...	user	2021-03-16 16:52:48 +0100

Below this is a table for 'Basic Authentication' with one row:

id	username	password	created_at
1de213a2...	user	e1426991a7546c00be9020a803e49cc288f5659f	2021-03-16 16:52:59 +0100

Add consumer group

```
curl -i -X POST \
--url
http://localhost:8001/consumers/use
r/acls \
--data "group=user"
```

Add consumer credentials

```
curl -i -X POST
http://localhost:8001/consumers/use
r/basic-auth \
--data "username=user" \
--data "password=user"
```

Test It !

Legacy

```
curl -i --location --request  
GET  
'http://localhost:52773/crud/  
persons/all' \  
--header 'Authorization:Basic  
dXNlcjp1c2Vy'
```

KONG

```
curl -i --location --request  
GET  
'http://localhost:8000/person  
s/all' \  
--header 'Authorization:Basic  
dXNlcjp1c2Vy'
```



Exercises

1. Enable Unauthenticated user
2. Limite rate by 2 calls per minutes to Unauthenticated user



Exercises

Consumers › 5cc8dee4 ›

Create Access Control Group

Group

user

The arbitrary group name to associate to the consumer.

Tags

Create

Cancel

+ * I wish this page would...

Add consumer group

```
curl -i -X POST \
--url
http://localhost:8001/consumers
/anonymous/acls \
--data "group=user"
```



Exercises

Consumers >

anonymous

created_at	2021-03-16 16:38:29 +0100
custom_id	anonymous
id	5cc8dee4-066d-492e-b2f8-bd77eb0a4c86
tags	
type	proxy
username	anonymous

Activity Credentials Plugins

No Plugins Enabled On This Consumer

Plugins allow you to extend Kong's capabilities with features like rate limiting, authentication, and logging.

Add Plugin

```
# Add rate limite consumer
curl -i -X POST \
--url
http://localhost:8001/consumers
/anonymous/plugins \
--data "name=rate-limiting" \
--data
"config.limit_by=consumer" \
--data "config.minute=2"
```



Dev Portal



Overview

The Kong Developer Portal provides :

- a single source of truth for all developers
- intuitive content management for documentation
- streamlined developer onboarding
- role-based access control (RBAC)

The screenshot shows the Kong Developers portal homepage. At the top right are links for Catalog, Login, and Sign Up. The main header features the Kong logo and the text "Kong Developers". Below the header is a large blue background with white wavy lines, containing the text "Built with Kong" and "Empowering developers across the globe.", along with a green "Create a Developer Account" button. In the center, there are two service cards: one for "httpbin.org" (version v0.9.3) and one for "Swagger Petstore" (version v1.0.0). At the bottom, a call-to-action button says "View Catalog", and a section titled "You're in Good Company" with the subtext "Developers from all industries use our platform to build integrations and custom applications." followed by six placeholder company names.

Enable It !

The screenshot shows the 'Dev Portal Overview' page within the InterSystems API Management interface. The left sidebar lists 'Workspaces', 'Dev Portals', 'Vitals', 'Teams', and 'Service Map'. The main content area displays a message: 'Dev Portal is disabled' with a blue 'Enable Developer Portal' button. A note below states: 'When turning on the portal for first time, it can take up to a few minutes to populate the files.' The bottom of the page includes a footer with 'I wish this page would...' and a feedback link.

```
curl -X PATCH  
http://localhost:8001/workspaces/default --data  
"config.portal=true"
```

Add your first spec

The screenshot shows two parts of a developer portal interface. The top part is the 'Dev Portal Overview' page from Kong Enterprise, which displays a message 'Dev Portal is enabled!' and provides a link to access it at <https://portal-reference.kong-cloud.com/default>. The bottom part is the 'Editor Mode' of the InterSystems IRIS Management Portal, showing the file `portal.conf.yaml` with its content displayed in a code editor.

Kong Enterprise Workspaces Dev Portals Vitals Teams Service Map Docs / Support kong_admin ▾

Change Workspace API Gateway Services Routes Consumers Plugins Upstreams Certificates SNI

Vitals Status Codes Alerts

Dev Portal Overview Dev Portal Docs

Dev Portal is enabled!

Access your Dev Portal at the link below

<https://portal-reference.kong-cloud.com/default>

Authentication is Enabled

Learn more about supported authentication plugins

* I wish this page would...

Editor

InterSystems IRIS Management Workspaces Dev Portals Vitals Teams Service Map Docs / Support

Editor Mode

All Changes Saved! Save Changes

New File +

`portal.conf.yaml`

CONTENT

- > about
- > account
- > applications
- > documentation
- > guides
- `dashboard.txt`
- `index.txt`
- `login.txt`
- `logout.txt`
- `oauth2-redirect.txt`
- `register.txt`
- `reset-password.txt`
- `settings.txt`
- `unauthorized.txt`

SPECS

- `httpbin.json`
- `petstore.yaml`

THEMES

- > base

EMAILS

- `account-verification-a...`
- `account-verification-p...`

View: Code Split Page

```
1  "info": {
2    "title": "InterSystems IRIS REST CRUD demo",
3    "description": "",
4    "version": "0.1"
5  },
6  "basePath": "/",
7  "paths": {
8    "/persons/all": {
9      "get": {
10        "description": " Retrieve all the records of Sample.Person",
11        "summary": " Get all records of Person class",
12        "operationId": "GetAllPersons",
13        "x-ISC_ServiceMethod": "GetAllPersons",
14        "responses": {
15          "default": {
16            "description": "(Unexpected Error)"
17          },
18          "200": {
19            "description": "(Expected Result)"
20          }
21        }
22      }
23    },
24    "/": {
25      "get": {
26        "description": " PersonsREST general information",
27        "summary": " Server Info",
28        "operationId": "GetInfo",
29        "x-ISC_CORS": true,
30        "x-ISC_ServiceMethod": "GetInfo",
31        "responses": {
32          "default": {
33            "description": "(Unexpected Error)"
34          },
35          "200": {
36            "description": "(Expected Result)"
37          }
38        }
39      }
40    }
41  }
42}
```

Select a file from the sidebar to get started.

```
curl -X POST
http://localhost:8001/default
/files -F "path=specs/iam-
training.yml" -F
"contents=@misc/spec.yml"
```

Test It !

<http://127.0.0.1:8003/default/documentation/iam-training>

What happen ? How-to solve it ?



Exercices

1. Add CORS plugin on route



Exercises

afefe836 ›

Create new cors plugin

This plugin is Enabled

Global
All Services & Routes in this workspace

Scoped
Specific Services and/or Routes in this workspace

Service ?

Route ?

Tags ?

e.g. tag1, tag2, tag3

Config.Credentials

Config.Exposed Headers

Enable CORS

```
curl -i -X POST  
http://localhost:8001/routes/cr  
ud-route/plugins \  
--data "name=cors"
```

