API Management

API Gateway Full Healthcheck Policies

Contents

1. Healthcheck APIGW 7.5.3 Architecture

This topic describes how to perform a full components healthcheck plus a zero downtime policy deployment to API Gateway in a multi-node API Gateway environment with a load balancer.

Note: The section Create Cassandra HC Policy is an improvement of the Healthcheck LB Service, and the reason is to validate, at the same time, API Gateway, API Manager and Cassandra. It is highly recommended.

Note2: This specification is designed for Axway API Gateway 7.5.3 or newer

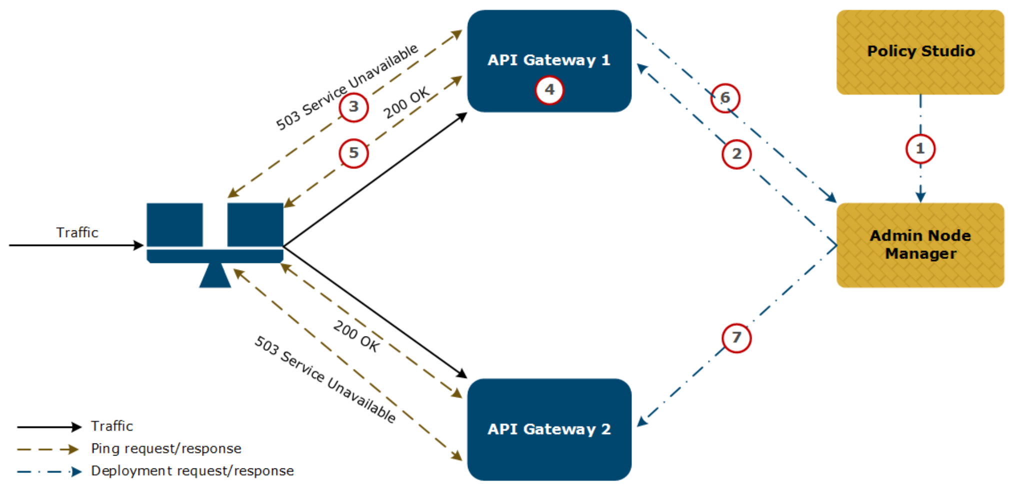
Source:
https://docs.axway.com/bundle/APIGateway_753_AdministratorGuide_allOS_en_HTML5/page/Content/AdminGuideTopics/admin_zdd.htm

2. Introduction to Zero Downtime Deploy

When you deploy configuration (for example, a .fed file) to a group of API Gateways, the configuration is deployed sequentially to each API Gateway in the group. While the configuration is being deployed to a given API Gateway there is a service interruption during which that API Gateway cannot process traffic.

Zero downtime deployment enables you to orchestrate deployment to a load-balanced set of API Gateways, ensuring that a subset can always process traffic.

The following diagram illustrates the process:



1. A user initiates deployment of new configuration from Policy Studio.
2. The Admin Node Manager starts deployment in `API Gateway 1`.
3. `API Gateway 1` starts responding to a ping from the load balancer with `503 Service Unavailable`. The load balancer stops routing traffic to `API Gateway 1`.
4. `API Gateway 1` performs the deployment.
5. `API Gateway 1` starts responding to the load balancer ping with `200 OK`. The load balancer starts routing traffic to `API Gateway 1` again.
6. `API Gateway 1` informs the Admin Node Manager that deployment is complete.
7. The Admin Node Manager repeats step 2 through step 6 for `API Gateway 2`.

3. Introduction to Virtualized Healthcheck

The purpose is to provide a full healthcheck solution to verify in one way, the API Gateway service (8443), API Manager Traffic Service (8065) and Cassandra Health.

The method is to expose a policy by a relative path and virtualize it on API Manager. The Load Balancer must use this virtualized API to healthcheck.

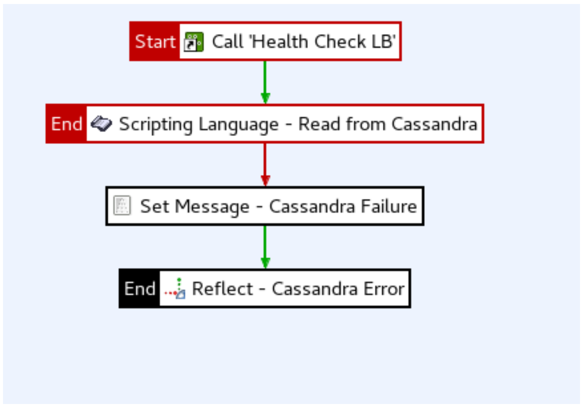
The policy does an explicit “SELECT” to Cassandra by reading the Community organization from API Portal collection.

- If return True, follow by the default HealthcheckLB validation
- If it can’t be reached there is an issue with Cassandra, and the healthcheck fail.

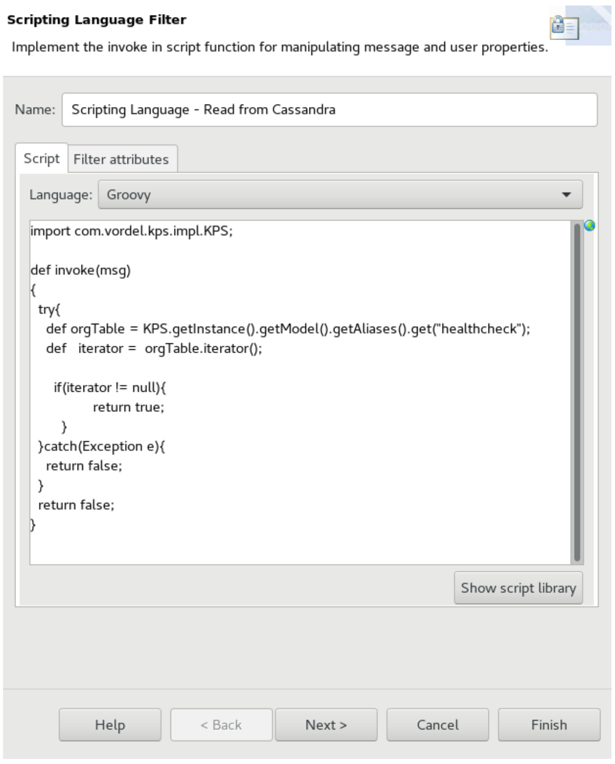
4. Create the CassandraHC Policy

Create a policy to enhance the HealthcheckLB policy to validate Cassandra health.

Create a new policy CassandraHC and organize the filters as the image below.


Start
Call 'Health Check LB'
End
Scripting Language - Read from Cassandra
Set Message - Cassandra Failure
End
Reflect - Cassandra Error

Edit the “Scripting Language – Read from Cassandra” filter to retrieve data from Cassandra table.


Scripting Language Filter

Implement the invoke in script function for manipulating message and user properties.

Name: Scripting Language - Read from Cassandra

Language: Groovy

```
import com.vordel.kps.impl.KPS;

def invoke(msg)
{
  try{
    def orgTable = KPS.getInstance().getModel().getAliases().get("healthcheck");
    def   iterator =  orgTable.iterator();

      if(iterator != null){
              return true;
        }
  }catch(Exception e){
    return false;
  }
  return false;
}
```

Full Groovy Code

```
import com.vordel.kps.impl.KPS;

def invoke(msg)
{
  try{
    def orgTable =
KPS.getInstance().getModel().getAliases().get("healthcheck");
    def   iterator =  orgTable.iterator();

      if(iterator != null){
            return true;
      }
  }catch(Exception e){
    return false;
  }
  return false;
}
```

Create the KPS Table "Healthcheck" to be used by the script

Create this table on any custom collection, the only requirement is to set the alias as "**healthcheck**"

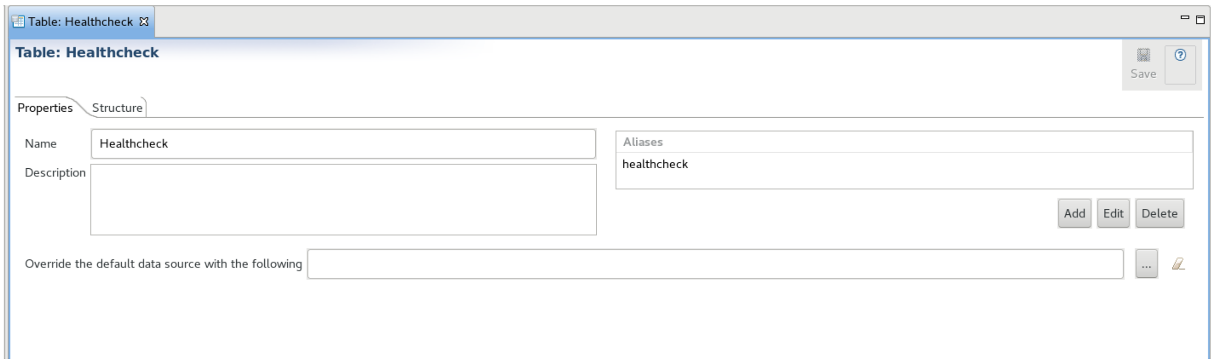


Table: Healthcheck

Name: Healthcheck

Description:

Aliases: healthcheck

Override the default data source with the following:

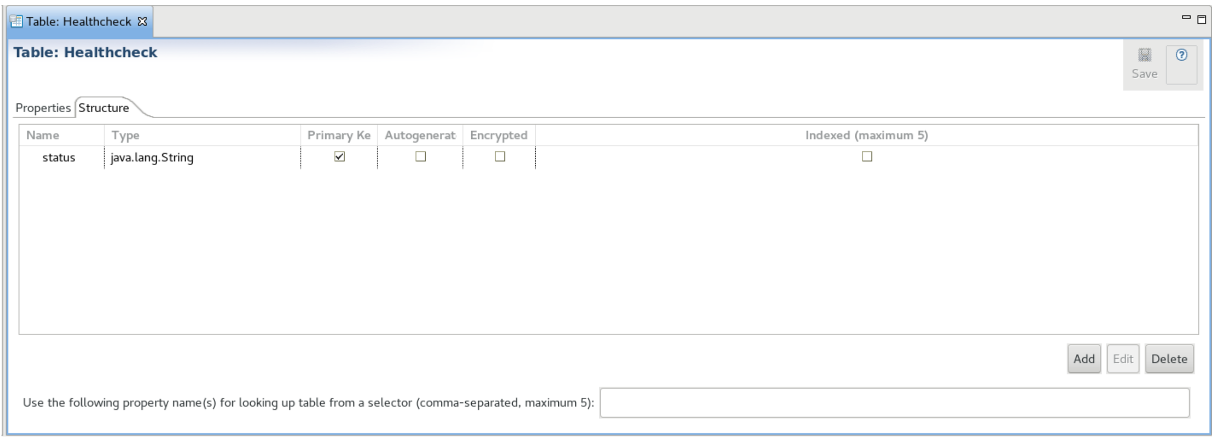


Table: Healthcheck

Properties | Structure

Name	Type	Primary Ke	Autogenerat	Encrypted	Indexed (maximum 5)
status	java.lang.String	☑	☐	☐	☐

Use the following property name(s) for looking up table from a selector (comma-separated, maximum 5):

Create one registry on the Healthcheck table

Use the API Gateway Manager to manage KPS Data

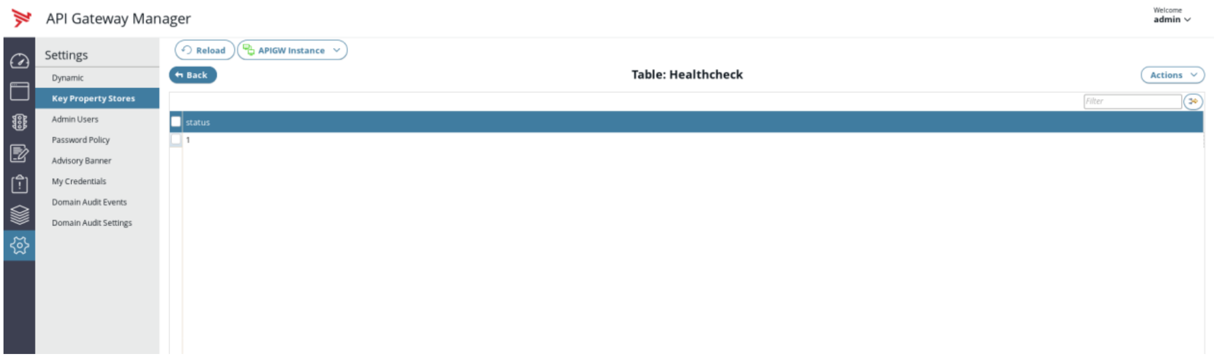
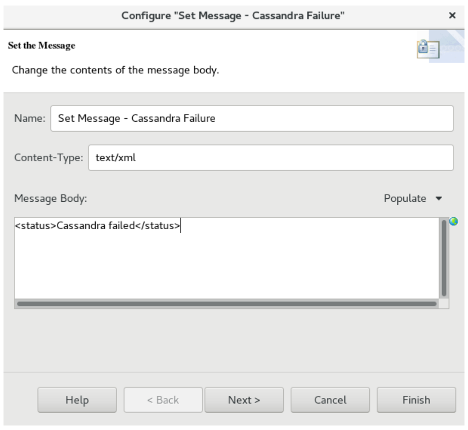


Table: Healthcheck

status
1

Edit the “Set Message” filter to create the Cassandra error message.


Configure "Set Message - Cassandra Failure"
Set the Message
Change the contents of the message body.
Populate
Help
< Back
Next >
Cancel
Finish

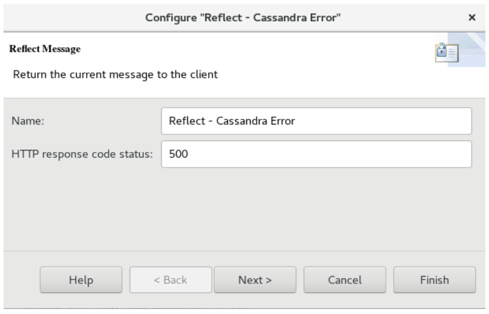
Name: Set Message - Cassandra Failure

Content-Type: text/xml

Message Body:

```
<status>Cassandra failed</status>
```

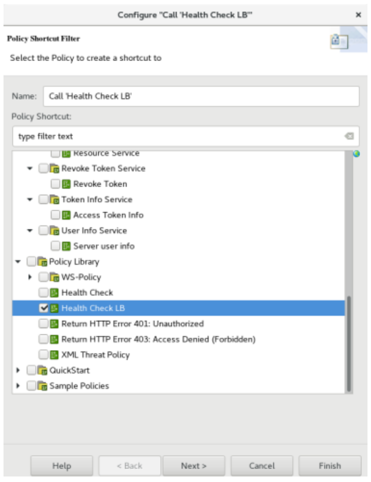
Edit the “Reflect Message” filter


Configure "Reflect - Cassandra Error"
Reflect Message
Return the current message to the client
Help
< Back
Next >
Cancel
Finish

Name: Reflect - Cassandra Error

HTTP response code status: 500

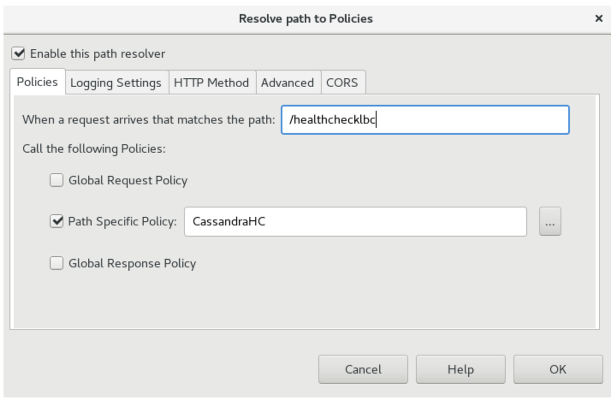
Create a Policy shortcut to HealthcheckLB.



Name: Call 'Health Check LB'

Policy Shortcut: ✓ Health Check LB

Create a relative path to expose the policy.


Resolve path to Policies

☑ Enable this path resolver

Policies | Logging Settings | HTTP Method | Advanced | CORS

When a request arrives that matches the path: /healthchecklbc

Call the following Policies:

☐ Global Request Policy

☑ Path Specific Policy: CassandraHC

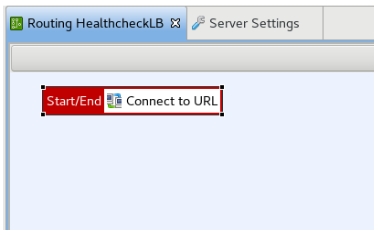
☐ Global Response Policy

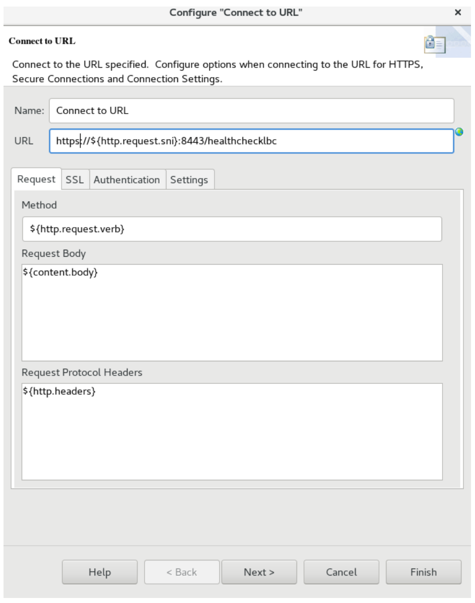
Deploy the configuration

5. Create the Routing Policy

The healthcheck must be fully executed into the same host. This policy ensures that each APIMGR always connect to its own APIGW. The variable `${http.request.sni}` has the hostname of the APIGW server processing the current transaction.

On Policy Studio, create the policy "Routing HealthcheckLB"




Configure "Connect to URL"

Connect to URL

Connect to the URL specified. Configure options when connecting to the URL for HTTPS, Secure Connections and Connection Settings.

Name: Connect to URL

URL: https://${http.request.sni}:8443/healthchecklbc

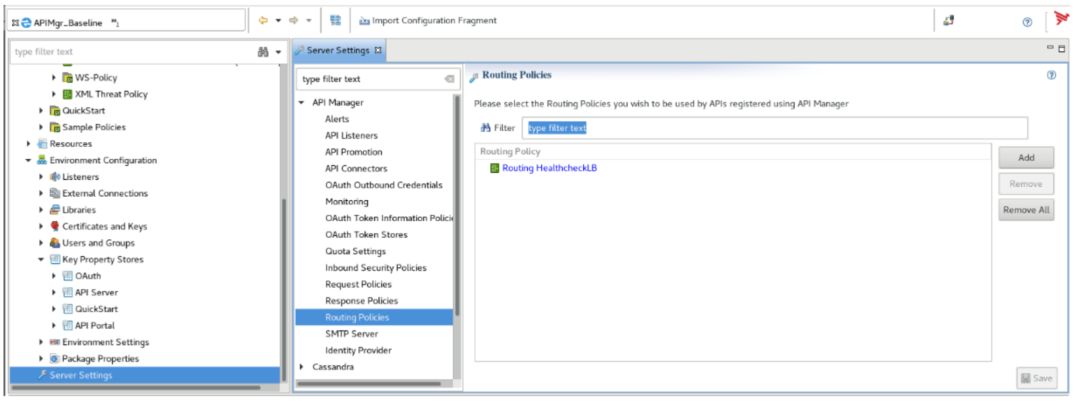
Method: ${http.request.verb}

Request Body: ${content.body}

Request Protocol Headers: ${http.headers}

Set the policy to API Manager Routing policies

Policy Studio > Server Settings > API Manager > Routing Policies



Routing Policies

Please select the Routing Policies you wish to be used by APIs registered using API Manager

Filter: type filter text

Routing Policy: Routing HealthcheckLB

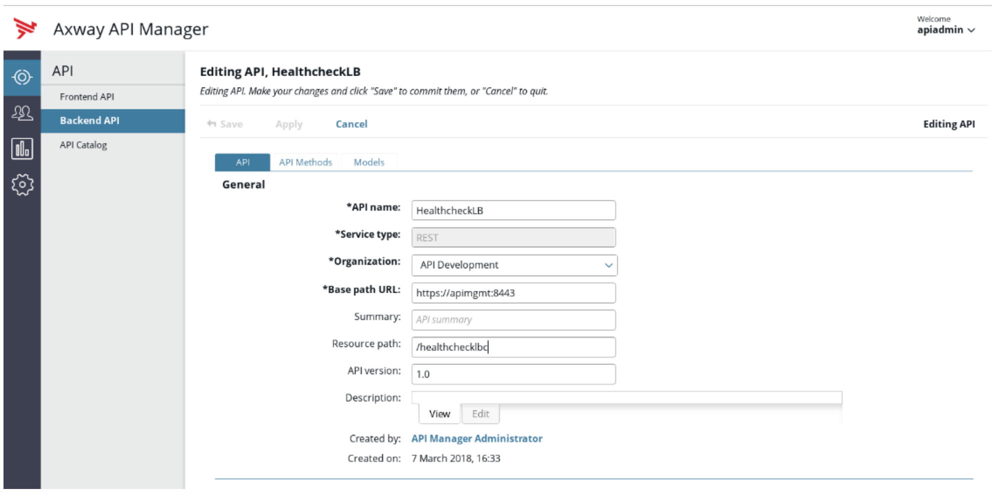
Deploy the configuration

6. Virtualize the API

Register Backend API

Create the API definitions and its Method, following the configurations below.

Note: In a HA architecture, can be used any host as Basepath URL. A policy will be used to route the traffic and ensure that the incoming API MGR request reach the same APIGW Server. (See 5 - Create the Routing Policy)

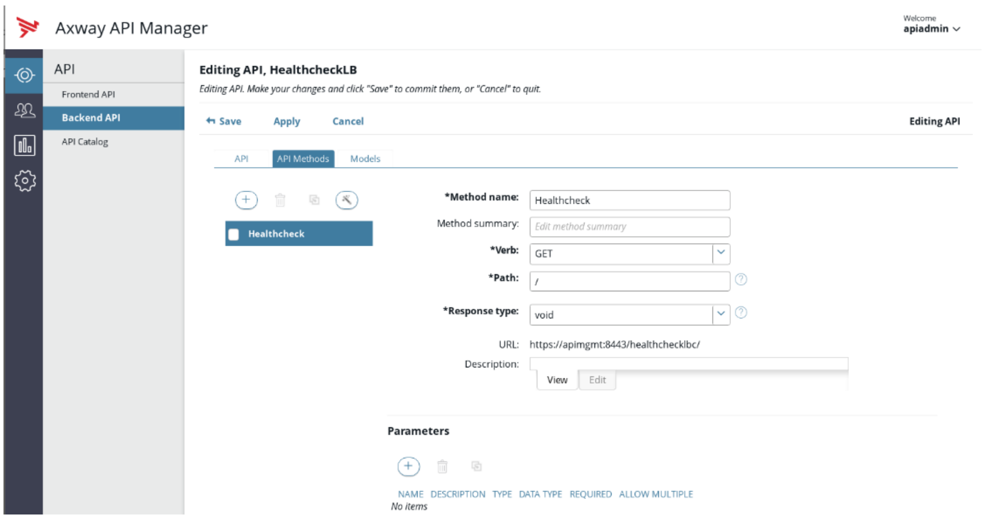


Editing API, HealthcheckLB

Editing API. Make your changes and click "Save" to commit them, or "Cancel" to quit.

General

*API name:	HealthcheckLB
*Service type:	REST
*Organization:	API Development
*Base path URL:	https://apimgmt:8443
Summary:	API summary
Resource path:	/healthcheckIbc
API version:	1.0
Description:	
Created by:	API Manager Administrator
Created on:	7 March 2018, 16:33



Editing API, HealthcheckLB

Editing API. Make your changes and click "Save" to commit them, or "Cancel" to quit.

*Method name:	Healthcheck
Method summary:	Edit method summary
*Verb:	GET
*Path:	/
*Response type:	void
URL:	https://apimgmt:8443/healthcheckIbc/
Description:	

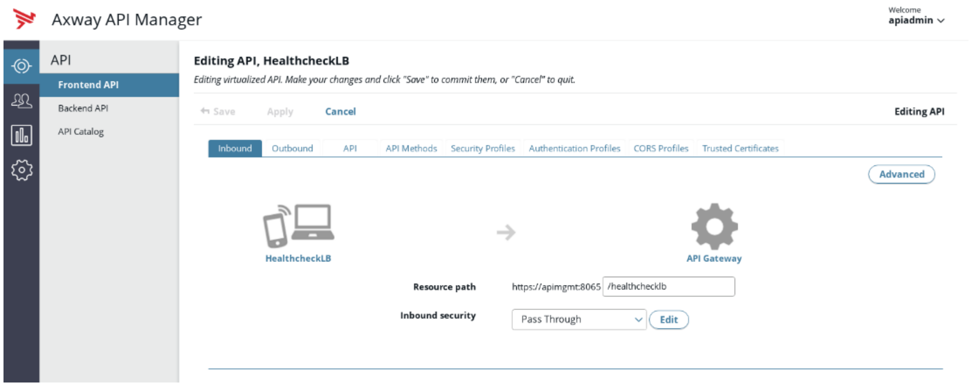
Parameters

NAME	DESCRIPTION	TYPE	DATA TYPE	REQUIRED	ALLOW MULTIPLE

No items

Virtualize on Frontend

Set the Inbound Security



Editing API, HealthcheckLB

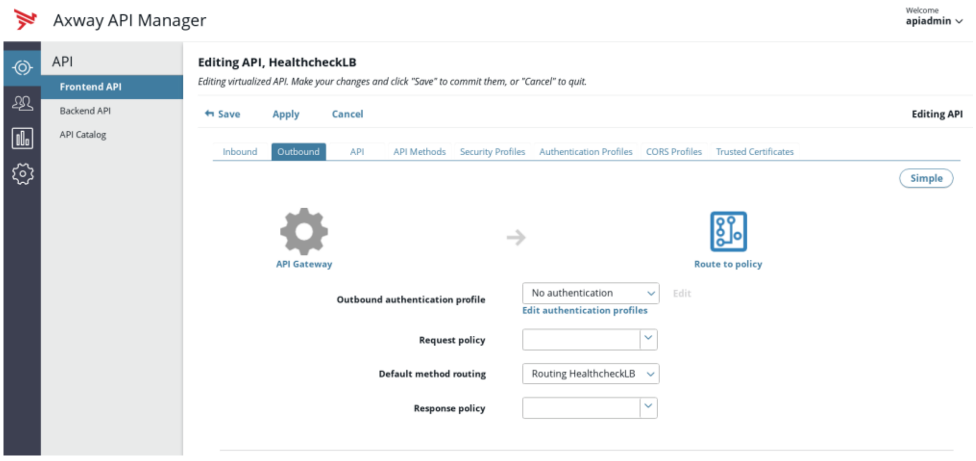
Editing virtualized API. Make your changes and click "Save" to commit them, or "Cancel" to quit.

Resource path: https://apimgmt:8065 /healthchecklb

Inbound security: Pass Through

Set the outbound Configuration

Click on Advanced button, top-right screen. Select the policy "Routing HealthcheckLB" as default method Routing.



Editing API, HealthcheckLB

Editing virtualized API. Make your changes and click "Save" to commit them, or "Cancel" to quit.

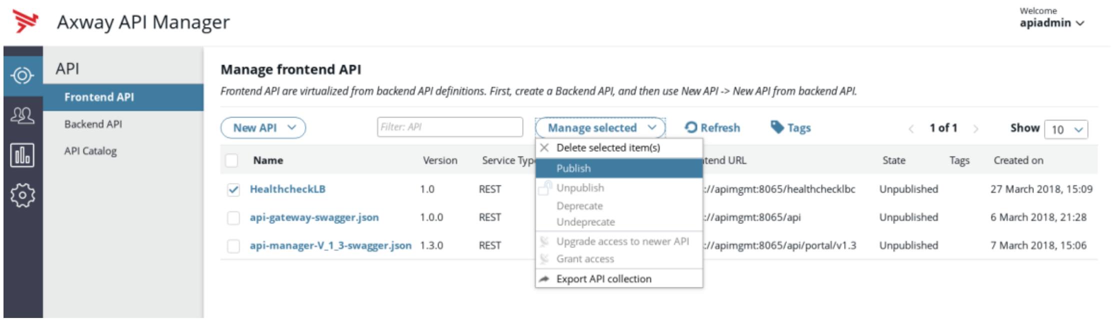
Outbound authentication profile: No authentication

Request policy:

Default method routing: Routing HealthcheckLB

Response policy:

Save the API and set the API as Published



Manage frontend API

Frontend API are virtualized from backend API definitions. First, create a Backend API, and then use New API -> New API from backend API.

	Name	Version	Service Typ	tend URL	State	Tags	Created on
✓	HealthcheckLB	1.0	REST	://apimgmt:8065/healthchecklbc	Unpublished		27 March 2018, 15:09
	api-gateway-swagger.json	1.0.0	REST	://apimgmt:8065/api	Unpublished		6 March 2018, 21:28
	api-manager-V_1_3-swagger.json	1.3.0	REST	://apimgmt:8065/api/portal/v1.3	Unpublished		7 March 2018, 15:06

Test the API

https://apimgmt:8065/healthchecklbc

7. Deploy Zero Downtime Configuration

To perform a zero downtime policy deployment, follow these steps:

1. Enable zero downtime deployment in Policy Studio, and set the delays before and after deployment. For more information, see Zero downtime settings.

2. Configure your load balancer to ping the Health Check LB policy periodically to determine if each API Gateway is healthy. This is available on the following default URL:

   https://APIGAMANAGER_HOST:8065/healthchecklbc

3. Initiate deployment to a group of API Gateways using API Gateway Manager, Policy Studio, or managedomain. For more information, see Deploy API Gateway configuration. The configuration is deployed sequentially to each API Gateway in the group.

4. When deployment is initiated on each API Gateway:
   a. The Health Check LB policy returns a 503 Service Unavailable response. This indicates to the load balancer that this API Gateway is not available for traffic and the load balancer stops routing to it.
   b. After the specified delay before deployment (for example, 10 seconds), the configuration is deployed to the API Gateway.
   c. When the deployment is complete, the Health Check LB policy returns a 200 OK response. This indicates to the load balancer that this API Gateway is available for traffic again.
   d. After the specified delay after deployment (for example, 10 seconds), a response is sent to the deployment request. Deployment can now be initiated to the next API Gateway in the group.