Hybrid Deployment : Edge Microgateway

Duration : 60 mins
Persona : API Team

Use cases

There are three primary use cases for implementing a microgateway pattern:

- Keep API traffic within the enterprise network boundaries for security or compliance purposes
- Incorporate lean API Management with your microservices architecture
- Add distributed security and traffic management with centralized visibility to your existing applications.

How does Edge Microgateway help?

Apigee Edge Microgateway provides lean API Management while benefiting from a full-fledged Apigee Edge API Management platform in the public or private cloud. Its main job is to process requests and responses to and from backend services securely while asynchronously pushing valuable API execution data to Apigee Edge where it is consumed by the Edge Analytics system. Edge Microgateway is easy to install and deploy -- you can have an instance up and running within minutes.

Apigee Edge Microgateway provides the following features:

- Analytics data is automatically collected and analyzed by the Apigee Edge platform
- Quota policies enforced in Microgateway based on settings on Apigee Edge platform
- OAuth 2 and API Keys are verified by Edge Microgateway. Spike Arrest policies are enforced.
- Protect inbound and outbound traffic via TLS 1.2
- Create your custom policies in Node.js and enforce by Microgateway

****This lab will only work on ‘paid’ orgs****

Overview

This topic shows you how to setup and configure a working Edge Microgateway instance. In general, you'll need to follow these same steps each time you set up a new instance of Edge

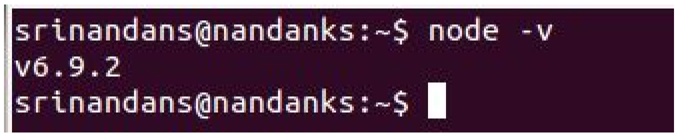
Microgateway.

After completing the steps in this topic, you'll have a fully configured, working Edge Microgateway installation capable of processing API requests. You'll test the setup by making secure API calls through Edge Microgateway to a backend target. At the end of this topic, you will learn how to add a spike arrest, api key and oauth plugins to the Microgateway.

Pre-requisites

- You must have Node.js version 4.x LTS or later installed on your system. You can check by executing:

```
node -v
```



```
srinandans@nandanks:~$ node -v
v6.9.2
srinandans@nandanks:~$
```

- Windows requires OpenSSL to be installed and added to the PATH
- cURL or a chrome extension like Postman.

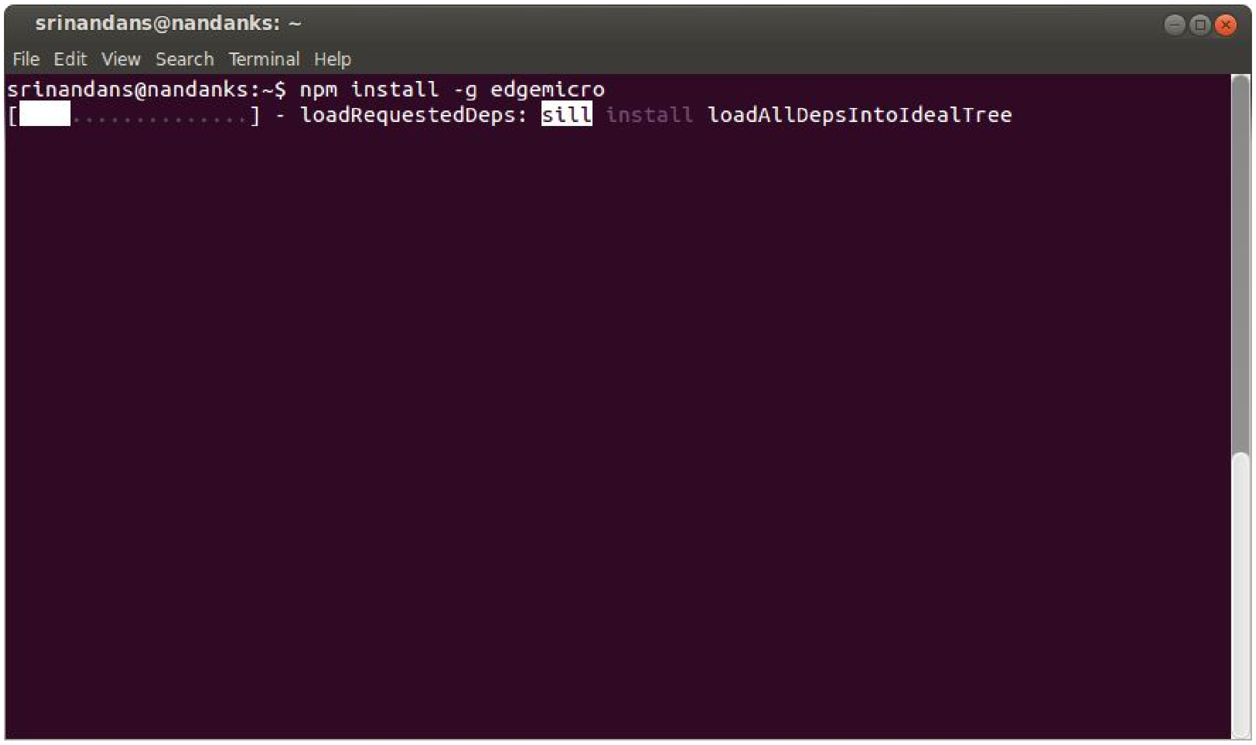
Instructions

Installing Edge Microgateway

This section explains how to install Edge Microgateway and initialize a default configuration.

1. Install Edge Microgateway with npm using the global install option. This command installs the software and puts the edgemicro executable in your path

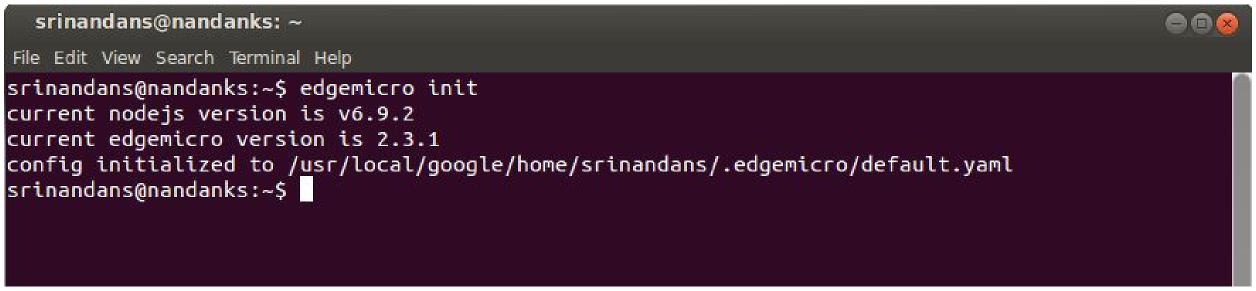
```
npm install -g edgemicro
```



```
srinandans@nandanks:~$ npm install -g edgemicro
[    ..............] - loadRequestedDeps: sill install loadAllDepsIntoIdealTree
```

2. Initialize Edge microgateway and create default configuration

```
edgemicro init
```



```
srinandans@nandanks:~$ edgemicro init
current nodejs version is v6.9.2
current edgemicro version is 2.3.1
config initialized to /usr/local/google/home/srinandans/.edgemicro/default.yaml
srinandans@nandanks:~$
```

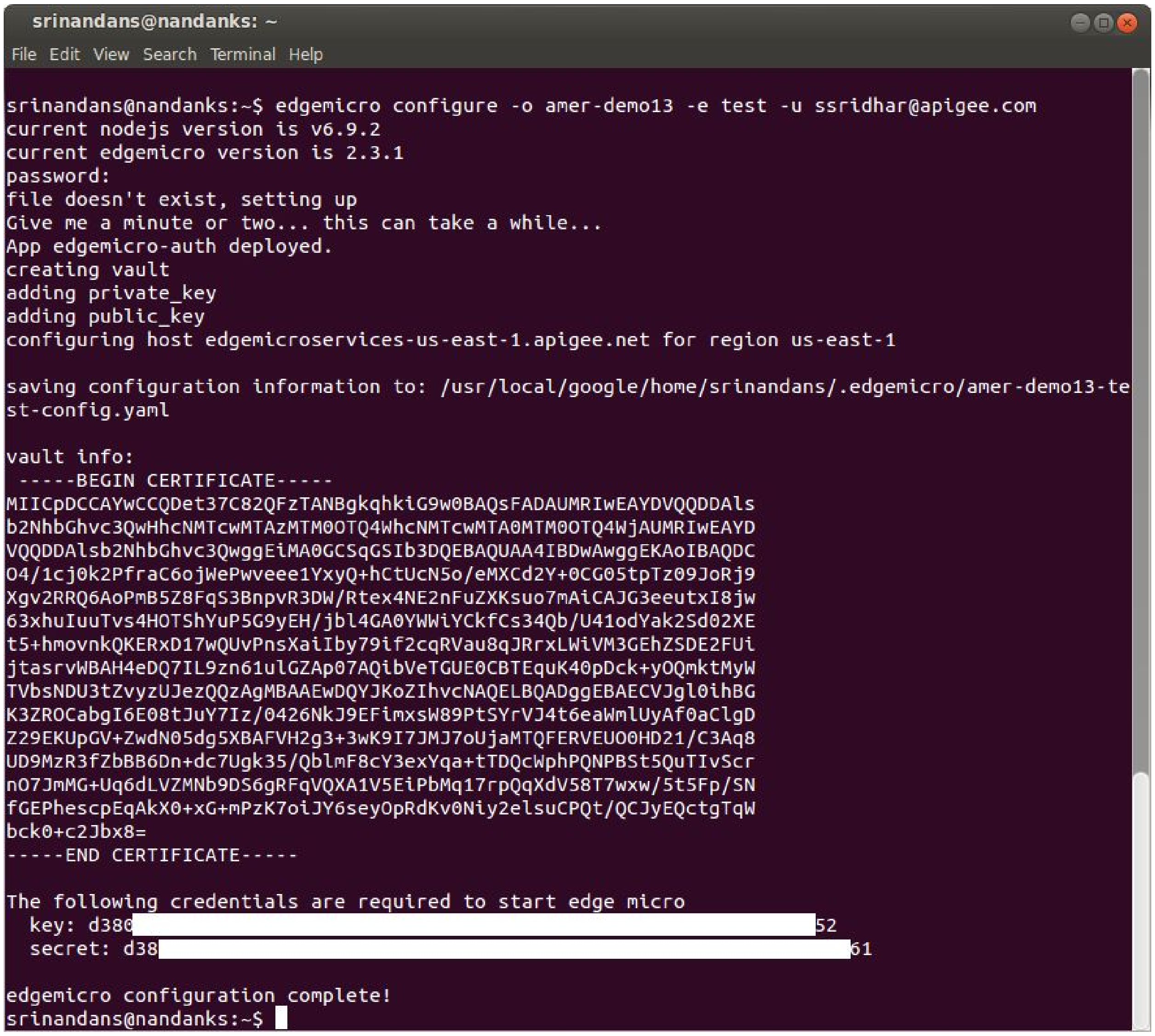
NOTE: This command creates a default configuration file in the home directory of the user.

Configure Edge Microgateway

This step will configure your Edge Microgateway instance with the Apigee Edge instance. In this lab, you will connect to a public cloud instance of Apigee Edge. The step will differ slightly when connecting to a private cloud instance. Please refer to the docs for more details.

This command requires standard information about your Apigee Edge account: organization name, environment name, username (email address), and password. **You must be an Edge organization administrator** to use this command:

```
edgemicro configure -o {org name} -e {env name} -u {username}
```

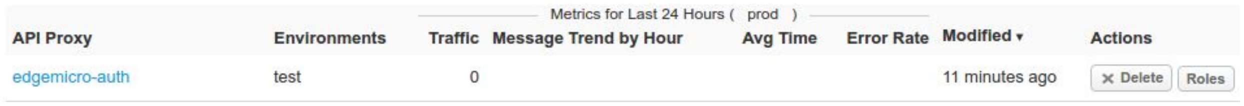


```
srinandans@nandanks:~$ edgemicro configure -o amer-demo13 -e test -u ssridhar@apigee.com
current nodejs version is v6.9.2
current edgemicro version is 2.3.1
password:
file doesn't exist, setting up
Give me a minute or two... this can take a while...
App edgemicro-auth deployed.
creating vault
adding private_key
adding public_key
configuring host edgemicroservices-us-east-1.apigee.net for region us-east-1

saving configuration information to: /usr/local/google/home/srinandans/.edgemicro/amer-demo13-te
st-config.yaml

vault info:
 -----BEGIN CERTIFICATE-----
MIICpDCCAYwCCQDet37C82QFzTANBgkqhkiG9w0BAQsFADAUMRIwEAYDVQQDDAls
b2NhbGhvc3QwHhcNMTcwMTAzMTM0OTQ4WhcNMTcwMTA0MTM0OTQ4WjAUMRIwEAYD
VQQDDAlsb2NhbGhvc3QwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDC
O4/1cj0k2PfraC6ojWePwveee1YxyQ+hCtUcN5o/eMXCd2Y+0CG05tpTz09JoRj9
Xgv2RRQ6AoPmB5Z8FqS3BnpvR3DW/Rtex4NE2nFuZXKsuo7mAiCAJG3eeutxI8jw
63xhuIuuTvs4HOTShYuP5G9yEH/jbl4GA0YWWiYCkfCs34Qb/U41odYak2Sd02XE
t5+hmovnkQKERxD17wQUvPnsXaiIby79if2cqRVau8qJRrxLWiVM3GEhZSDE2FUi
jtasrvWBAH4eDQ7IL9zn61ulGZAp07AQibVeTGUE0CBTEquK40pDck+yOQmktMyW
TVbsNDU3tZvyzUJezQQzAgMBAAEwDQYJKoZIhvcNAQELBQADggEBAECVJgl0ihBG
K3ZROCabgI6E08tJuY7Iz/0426NkJ9EFimxsW89PtSYrVJ4t6eaWmlUyAf0aClgD
Z29EKUpGV+ZwdN05dg5XBAFVH2g3+3wK9I7JMJ7oUjaMTQFERVEUO0HD21/C3Aq8
UD9MzR3fZbBB6Dn+dc7Ugk35/QblmF8cY3exYqa+tTDQcWphPQNPBSt5QuTIvScr
nO7JmMG+Uq6dLVZMNb9DS6gRFqVQXA1V5EiPbMq17rpQqXdV58T7wxw/5t5Fp/SN
fGEPhescpEqAkX0+xG+mPzK7oiJY6seyOpRdKv0Niy2elsuCPQt/QCJyEQctgTqW
bck0+c2Jbx8=
-----END CERTIFICATE-----

The following credentials are required to start edge micro
  key: d380[redacted]52
  secret: d38[redacted]61

edgemicro configuration complete!
srinandans@nandanks:~$
```

All of the configuration done so far allows Edge Microgateway to bootstrap itself to Apigee Edge. After the bootstrapping succeeds, Edge Microgateway retrieves a payload of additional configuration information from Apigee Edge.

NOTE: Please **save the key and secret in a secure location**. You will need it to start or reload the Edge microgateway instance.

The configuration command deploys a proxy “edgemicro-auth” in the org+env



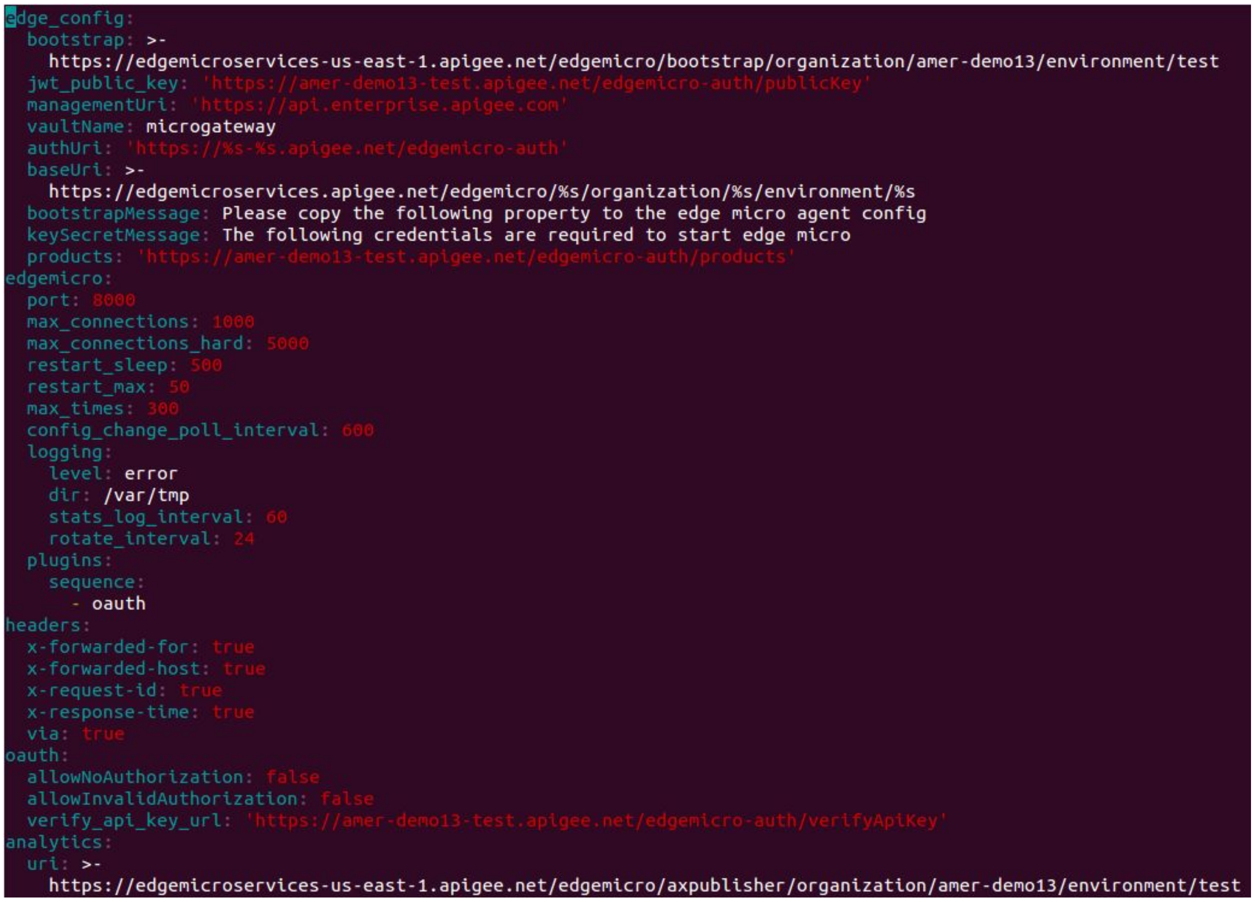
API Proxy	Environments	Traffic	Message Trend by Hour	Avg Time	Error Rate	Modified ▾	Actions
edgemicro-auth	test	0				11 minutes ago	✕ Delete Roles

This proxy is responsible for :

1. Providing Edge Microgateway with a list of all products in the org
2. Verifying API Keys
3. Generating JWS based Oauth Access Tokens.

The command also created a configuration file in the following location:

```
~/.edgemicro/{org name}-{env name}-config.yaml
```



```
edge_config:
  bootstrap: >-
    https://edgemicroservices-us-east-1.apigee.net/edgemicro/bootstrap/organization/amer-demo13/environment/test
  jwt_public_key: 'https://amer-demo13-test.apigee.net/edgemicro-auth/publicKey'
  managementUri: 'https://api.enterprise.apigee.com'
  vaultName: microgateway
  authUri: 'https://%s-%s.apigee.net/edgemicro-auth'
  baseUri: >-
    https://edgemicroservices.apigee.net/edgemicro/%s/organization/%s/environment/%s
  bootstrapMessage: Please copy the following property to the edge micro agent config
  keySecretMessage: The following credentials are required to start edge micro
  products: 'https://amer-demo13-test.apigee.net/edgemicro-auth/products'
edgemicro:
  port: 8000
  max_connections: 1000
  max_connections_hard: 5000
  restart_sleep: 500
  restart_max: 50
  max_times: 300
  config_change_poll_interval: 600
  logging:
    level: error
    dir: /var/tmp
    stats_log_interval: 60
    rotate_interval: 24
  plugins:
    sequence:
      - oauth
headers:
  x-forwarded-for: true
  x-forwarded-host: true
  x-request-id: true
  x-response-time: true
  via: true
oauth:
  allowNoAuthorization: false
  allowInvalidAuthorization: false
  verify_api_key_url: 'https://amer-demo13-test.apigee.net/edgemicro-auth/verifyApiKey'
analytics:
  uri: >-
    https://edgemicroservices-us-east-1.apigee.net/edgemicro/axpublisher/organization/amer-demo13/environment/test
```

Create Entities on Apigee Edge

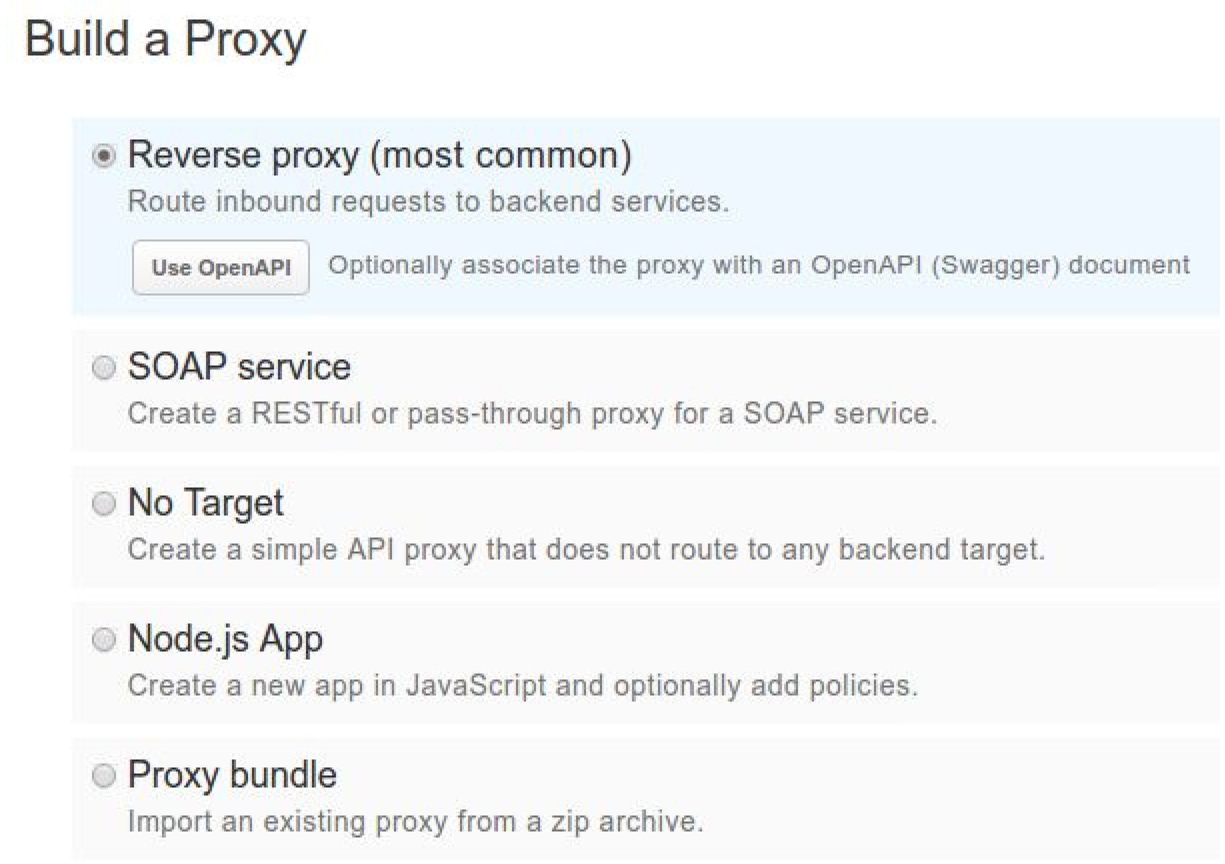
Edge Microgateway follows the philosophy of “*Centrally administer and author policies, federate policy enforcement.*”

In this section you will,

1. Define a proxy on Apigee Edge. When Edge Microgateway starts (and during reload), it downloads the configuration from Edge and exposes the API locally on Microgateway.
2. Define a product (that includes the proxy created in Step 1). Edge Microgateway downloads the product information and uses it to enforce API Keys or OAuth.
3. Finally, create a developer (optional) and a developer app (for client id and secret).

Create a Proxy

1. Log in to your organization on Apigee Edge.
2. Select **APIs > API Proxies** from the top menu.
3. In the API Proxies page, click **+ API Proxy**.
4. In the Build a Proxy wizard, select Reverse proxy (most common).



Build a Proxy

[✓ Reverse proxy (most common)]
Route inbound requests to backend services.
Optionally associate the proxy with an OpenAPI (Swagger) document

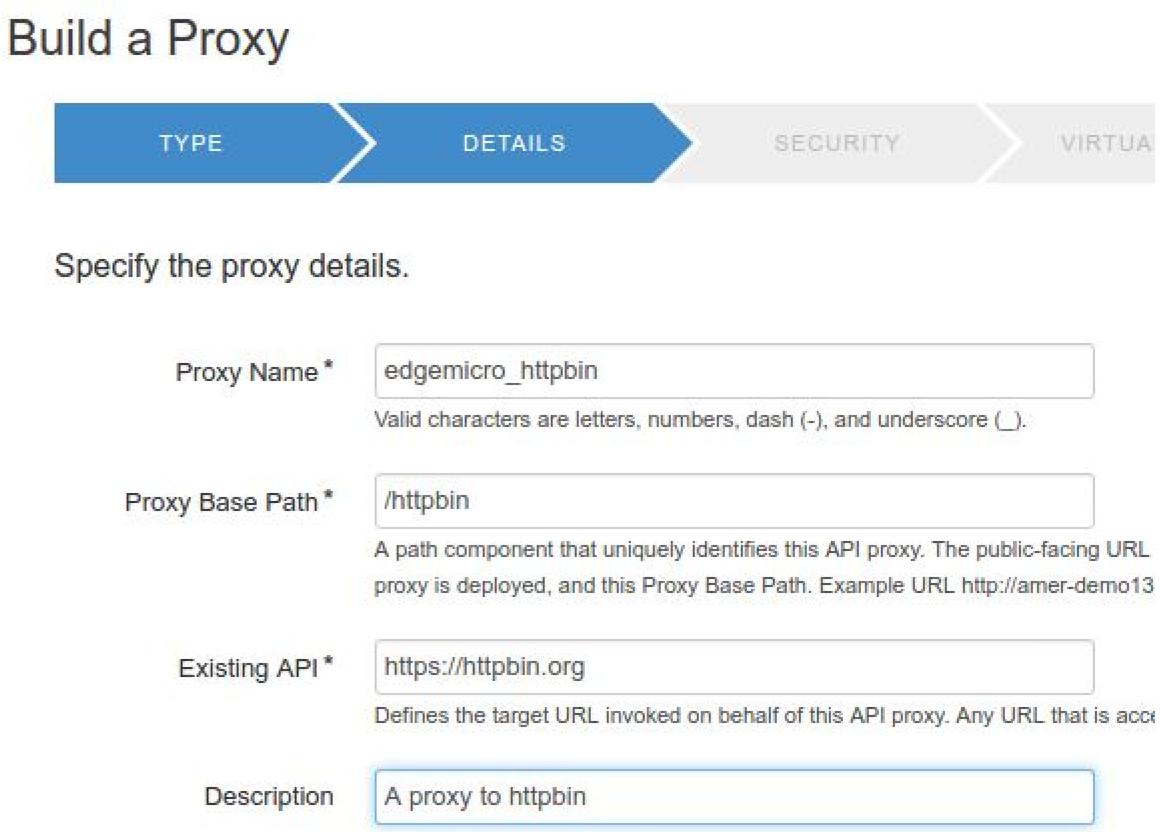
SOAP service
Create a RESTful or pass-through proxy for a SOAP service.

No Target
Create a simple API proxy that does not route to any backend target.

Node.js App
Create a new app in JavaScript and optionally add policies.

Proxy bundle
Import an existing proxy from a zip archive.

5. In the Details page of the wizard, configure as follows, where we point to an example target API that we can use for testing purposes:
   Proxy Name: edgemicro_httpbin
   Proxy Base Path: /httpbin
   Existing API: https://httpbin.org


Build a Proxy

Specify the proxy details.

Proxy Name * edgemicro_httpbin

Valid characters are letters, numbers, dash (-), and underscore (_).

Proxy Base Path * /httpbin

A path component that uniquely identifies this API proxy. The public-facing URL proxy is deployed, and this Proxy Base Path. Example URL http://amer-demo13

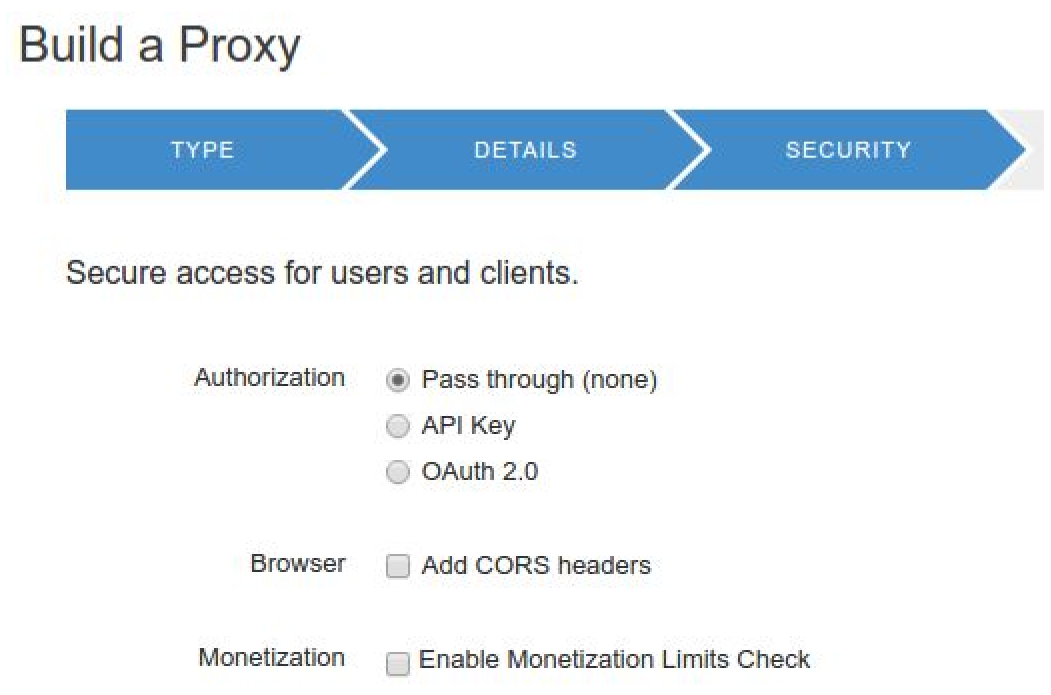
Existing API * https://httpbin.org

Defines the target URL invoked on behalf of this API proxy. Any URL that is acc

Description A proxy to httpbin

6. Click **Next**.

   Important: Edge Microgateway-aware proxy names **<u>must</u>** always begin with the prefix **edgemicro_**.
7. In the Security page of the wizard, select **Pass through (none)**



Secure access for users and clients.

Authorization: [● Pass through (none)] ○ API Key ○ OAuth 2.0

Browser: ☐ Add CORS headers

Monetization: ☐ Enable Monetization Limits Check

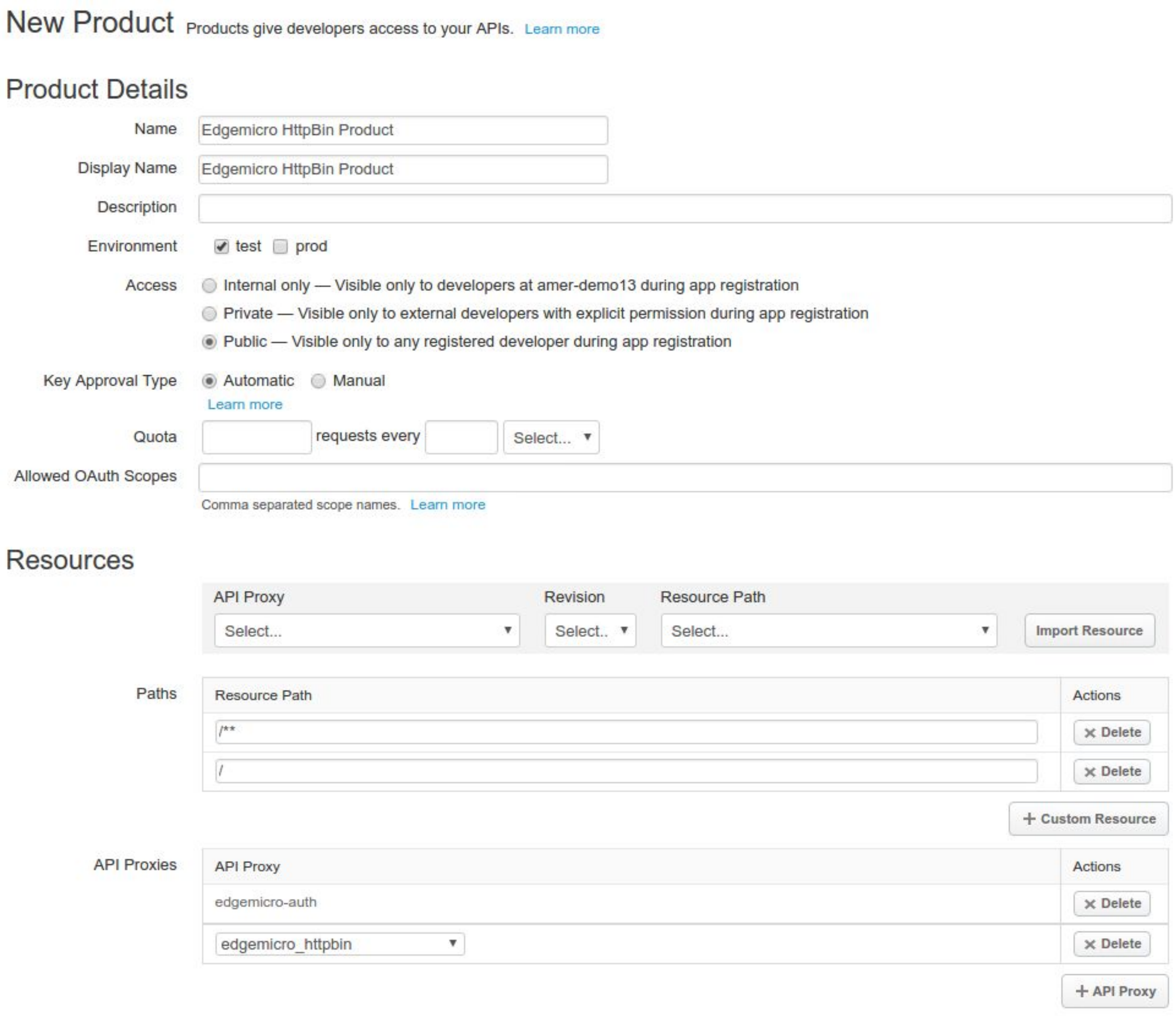
8. Click **Next**.
9. In the Virtual Hosts page of the wizard, accept the defaults.
10. Click **Next**.
11. In the Build page of the wizard, review your proxy settings. Make sure the test environment is selected.
12. Click **Build and Deploy**.

Create a Product

Create a product that contains your Edge Microgateway-aware proxy(s):

1. Log in to the Edge management UI
2. Go to **Publish > API Products**.
3. In the Products page, click **+ API Product**.
   Fill out the Product Details dialog as follows:
   a. **Name**: Edgemicro HttpBin Product
   b. **Display Name**: Edgemicro HttpBin Product
   c. **Environment**: test
   d. **Access**: Public
   e. **Key Approval Type**: Automatic
      Resources:
      API Proxy: Add these two proxies: edgemicro_httpbin and edgemicro-auth
      Revision: 1
      Resource Path: /**

f. Click **Import Resource**.
g. Click **Save**


New Product Products give developers access to your APIs. Learn more

Product Details

Name	Edgemicro HttpBin Product
Display Name	Edgemicro HttpBin Product
Description	
Environment	[✓ test] prod
Access	Internal only — Visible only to developers at amer-demo13 during app registration; Private — Visible only to external developers with explicit permission during app registration; [✓ Public — Visible only to any registered developer during app registration]
Key Approval Type	[✓ Automatic] Manual
Quota	requests every Select...
Allowed OAuth Scopes	Comma separated scope names.

Resources

API Proxy	Revision	Resource Path
Select...	Select..	Select...

Paths

Resource Path	Actions
/**	Delete
/	Delete

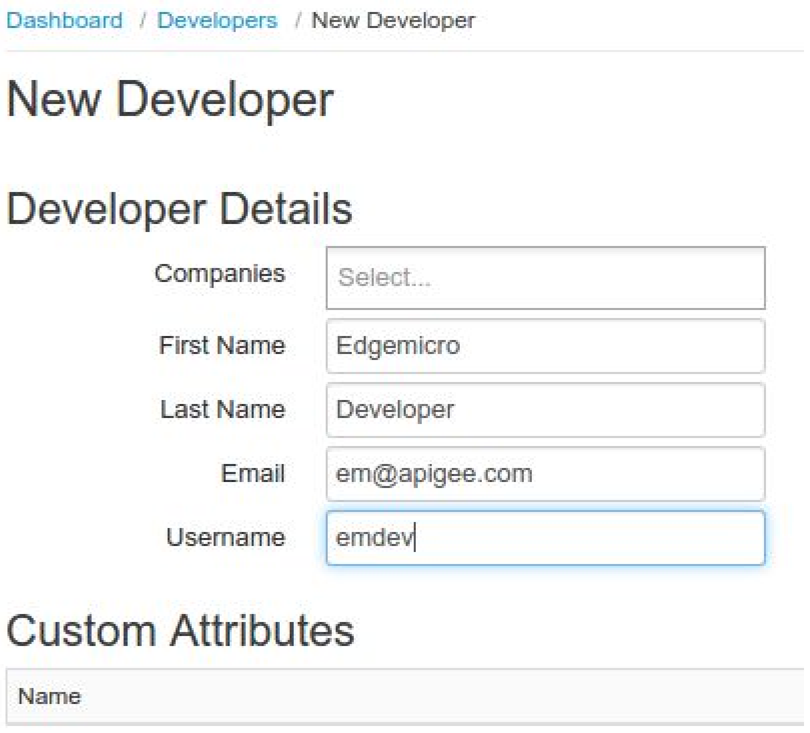
API Proxies

API Proxy	Actions
edgemicro-auth	Delete
edgemicro_httpbin	Delete

Create a Developer (optional)

For the purpose of this tutorial, you can use any existing developer for the next step, creating a developer app. But if you wish, create a test developer now:

1. Go to **Publish > Developers**.
2. In the Products page, click **+ Developer**.
3. Fill out the dialog to create a test developer.


Dashboard / Developers / New Developer

New Developer

Developer Details

Companies	Select...
First Name	Edgemicro
Last Name	Developer
Email	em@apigee.com
Username	emdev

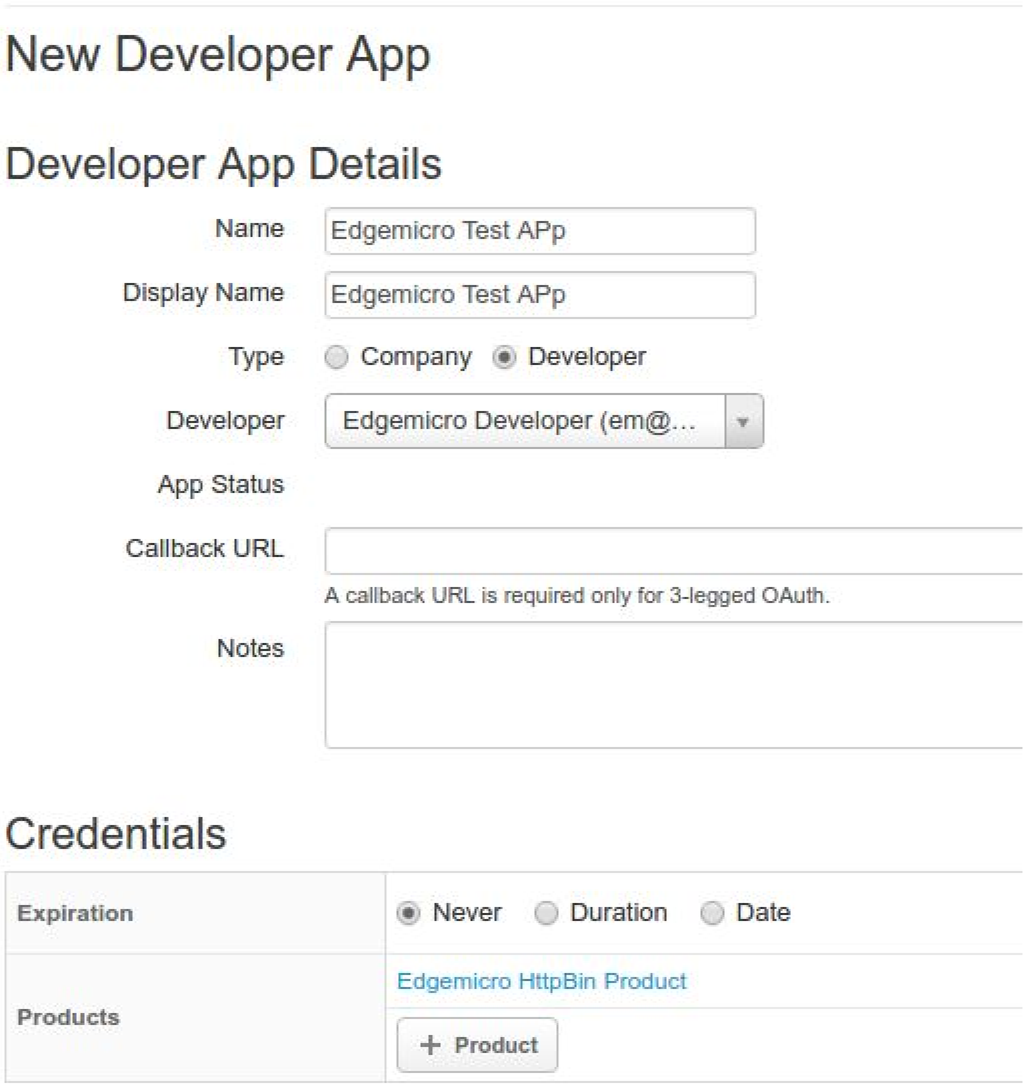
Custom Attributes

Name

Create a Developer App

You are going to use the client credentials from this app to make secure API calls through Edge Microgateway:

1. Go to **Publish > Apps**.
2. In the Developer Apps page, click **+ App**.
3. Fill out the Developer App dialog as follows:
   Name: Edgemicro Test APp
   Display Name: Edgemicro Test APp
   Developer: If you created a test developer, select it. Or, you can use any existing developer for the purpose of this tutorial.
   Products: Select Edgemicro HttpBin Product (the product you just created)



New Developer App

Developer App Details

Name	Edgemicro Test APp
Display Name	Edgemicro Test APp
Type	Company [✓ Developer]
Developer	Edgemicro Developer (em@…
App Status	
Callback URL	
	A callback URL is required only for 3-legged OAuth.
Notes	

Credentials

Expiration	[✓ Never] Duration Date
Products	Edgemicro HttpBin Product
	+ Product

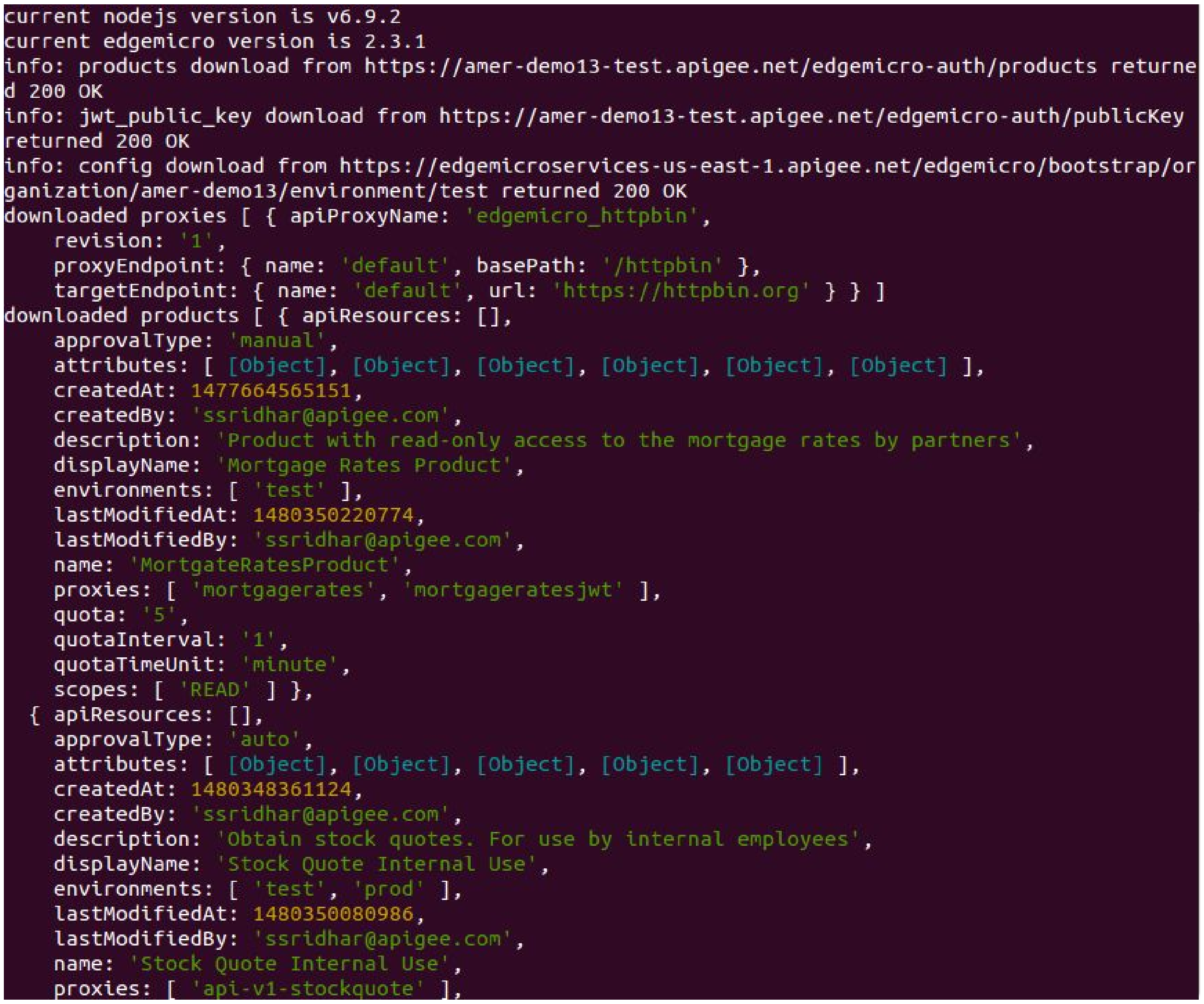
4. Click the checkmark button next to the Products field to add the product.
5. Click **Save**

Test Edge Microgateway Setup

Start Edge Microgateway

To start Edge Microgateway, run the following command:

```
edgemicro start -o {org-name} -e {env-name} -k {key} -s {secret}
```



```
current nodejs version is v6.9.2
current edgemicro version is 2.3.1
info: products download from https://amer-demo13-test.apigee.net/edgemicro-auth/products returne
d 200 OK
info: jwt_public_key download from https://amer-demo13-test.apigee.net/edgemicro-auth/publicKey
returned 200 OK
info: config download from https://edgemicroservices-us-east-1.apigee.net/edgemicro/bootstrap/or
ganization/amer-demo13/environment/test returned 200 OK
downloaded proxies [ { apiProxyName: 'edgemicro_httpbin',
    revision: '1',
    proxyEndpoint: { name: 'default', basePath: '/httpbin' },
    targetEndpoint: { name: 'default', url: 'https://httpbin.org' } } ]
downloaded products [ { apiResources: [],
    approvalType: 'manual',
    attributes: [ [Object], [Object], [Object], [Object], [Object], [Object] ],
    createdAt: 1477664565151,
    createdBy: 'ssridhar@apigee.com',
    description: 'Product with read-only access to the mortgage rates by partners',
    displayName: 'Mortgage Rates Product',
    environments: [ 'test' ],
    lastModifiedAt: 1480350220774,
    lastModifiedBy: 'ssridhar@apigee.com',
    name: 'MortgateRatesProduct',
    proxies: [ 'mortgagerates', 'mortgageratesjwt' ],
    quota: '5',
    quotaInterval: '1',
    quotaTimeUnit: 'minute',
    scopes: [ 'READ' ] },
  { apiResources: [],
    approvalType: 'auto',
    attributes: [ [Object], [Object], [Object], [Object], [Object] ],
    createdAt: 1480348361124,
    createdBy: 'ssridhar@apigee.com',
    description: 'Obtain stock quotes. For use by internal employees',
    displayName: 'Stock Quote Internal Use',
    environments: [ 'test', 'prod' ],
    lastModifiedAt: 1480350080986,
    lastModifiedBy: 'ssridhar@apigee.com',
    name: 'Stock Quote Internal Use',
    proxies: [ 'api-v1-stockquote' ],
```

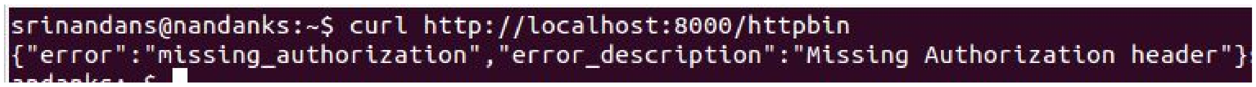
NOTES:

1. Edge microgateway starts on port 8000 by default
2. Edge microgateway starts in “cluster” mode by default. You will find one worker process per CPU.

Test API Proxy

Make a call to the API Proxy using cURL, Postman etc.

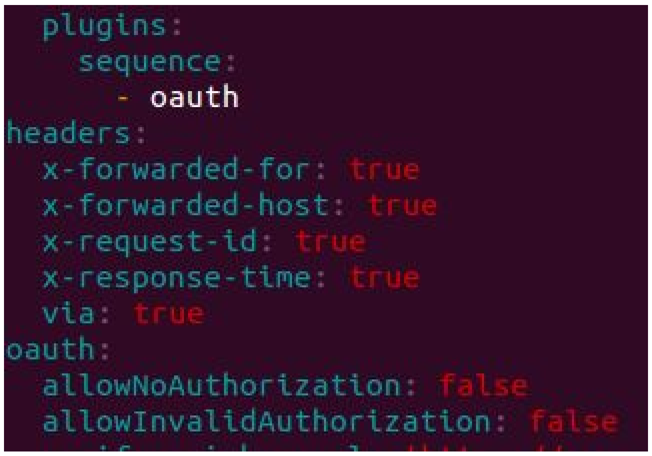
2



```
srinandans@nandanks:~$ curl http://localhost:8000/httpbin
{"error":"missing_authorization","error_description":"Missing Authorization header"}
```

NOTE: It is expected that the API call will fail due to “Missing Authorization Header”.

Explanation
The Edge microgateway configuration controls the security for the API proxies hosted in EM. The default configuration is to enable the OAuth plugin and not allow missing or invalid authorization headers.


```
  plugins:
    sequence:
      - oauth
headers:
  x-forwarded-for: true
  x-forwarded-host: true
  x-request-id: true
  x-response-time: true
  via: true
oauth:
  allowNoAuthorization: false
  allowInvalidAuthorization: false
```

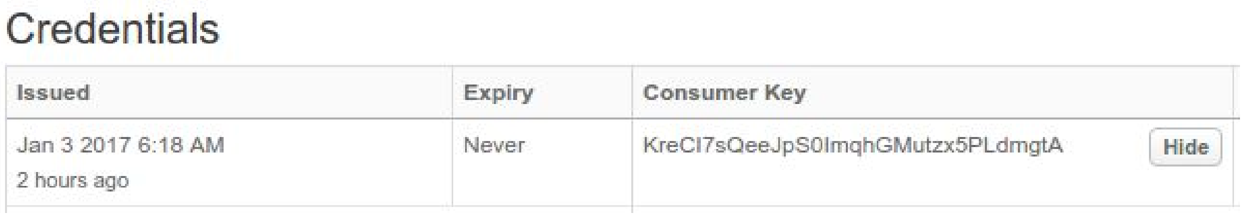
Notice, the Oauth plugin is enabled (because it is a part of the plugin sequence). Also the flags `allowNoAuthorization` and `allInvalidAuthorization` is set to `false`.

Enable API Key Verification

In this section you will access an Edge Microgateway proxy via API Keys. Since we have already created the proxy, product and app, we can go ahead and test the API access.

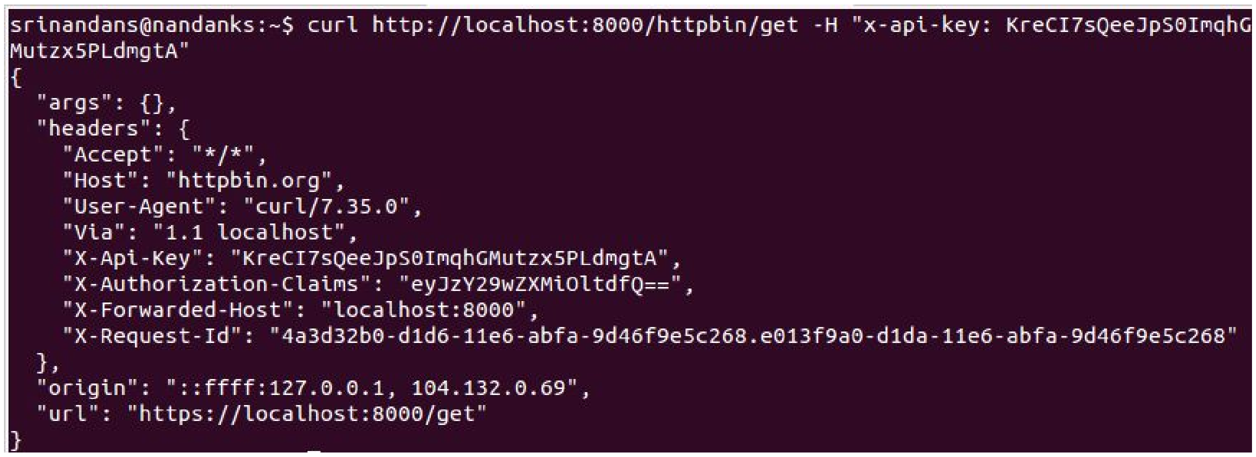
First, grab the API Key (consumer key) from the Edge UI.

1. In the Developer Apps list page.
2. Select the app you just created, Edgemicro Test APp
3. Click **Show** next to the Consumer Key and Consumer Secret


Credentials

Issued	Expiry	Consumer Key
Jan 3 2017 6:18 AM 2 hours ago	Never	KreCl7sQeeJpS0ImqhGMutzx5PLdmgtA

```
curl http://localhost:8000/httpbin/get -H "x-api-key:
KreCI7sQeeJpS0ImqhGMutzx5PLdmgtA"
```



```
srinandans@nandanks:~$ curl http://localhost:8000/httpbin/get -H "x-api-key: KreCI7sQeeJpS0ImqhG
Mutzx5PLdmgtA"
{
  "args": {},
  "headers": {
    "Accept": "*/*",
    "Host": "httpbin.org",
    "User-Agent": "curl/7.35.0",
    "Via": "1.1 localhost",
    "X-Api-Key": "KreCI7sQeeJpS0ImqhGMutzx5PLdmgtA",
    "X-Authorization-Claims": "eyJzY29wZXMiOltdfQ==",
    "X-Forwarded-Host": "localhost:8000",
    "X-Request-Id": "4a3d32b0-d1d6-11e6-abfa-9d46f9e5c268.e013f9a0-d1da-11e6-abfa-9d46f9e5c268"
  },
  "origin": "::ffff:127.0.0.1, 104.132.0.69",
  "url": "https://localhost:8000/get"
}
```

Enable Spike Arrest

In this topic, you will enable the spike arrest plugin to the microgateway instance.

1. Open the configuration file located here:

```
vi ~/.edgemicro/{org-name}-{env-name}-config.yaml
```

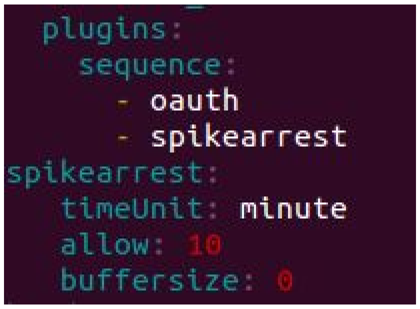
2. Add the following lines in the plugins stanza.

```
  plugins:
    sequence:
      - oauth
      - spikearrest
```

NOTE: **Order is important**! Plugins are always executed in the order of appearance in the configuration file. In this case, first the API Key (or OAuth token) is verified then the spike arrest plugin is executed.

3. Add the following spike arrest configuration anywhere in the config file:

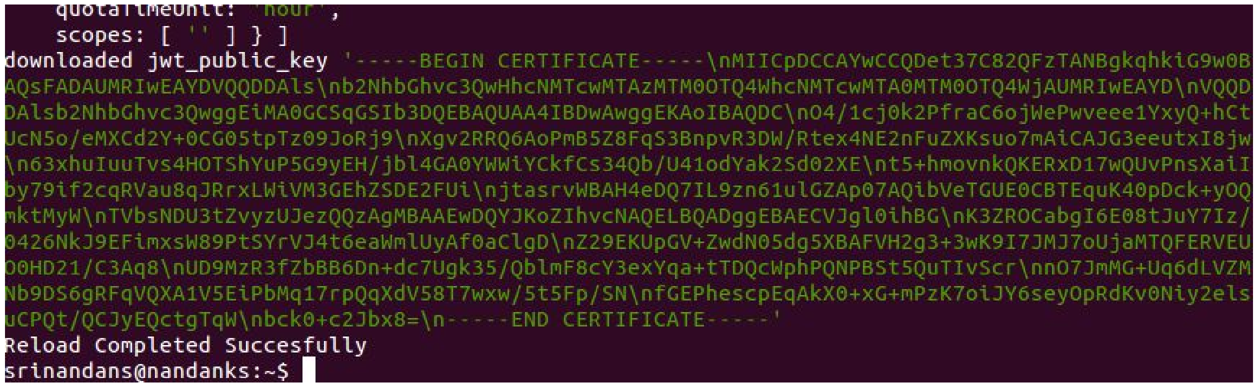
```
spikearrest:
      timeUnit: minute
      allow: 10
      buffersize: 0
```



```
  plugins:
    sequence:
      - oauth
      - spikearrest
spikearrest:
   timeUnit: minute
   allow: 10
   buffersize: 0
```

4. Reload the Edge microgateway instance (if running) in a new terminal window.

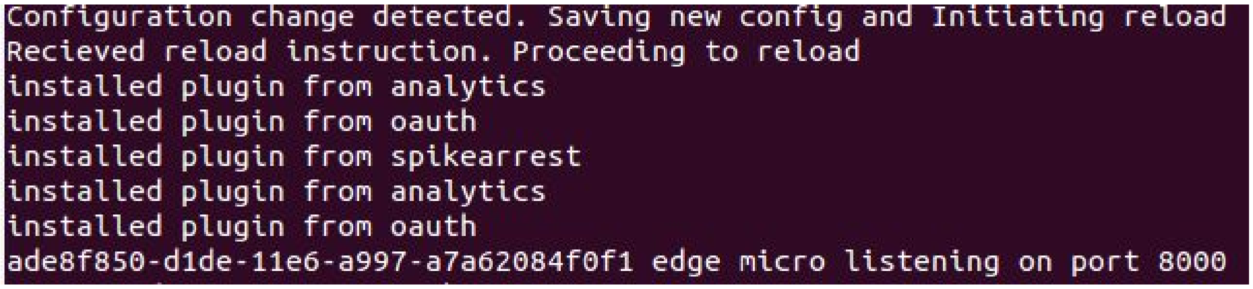
```
edgemicro reload -o {org-name} -e {env-name} -k {key} -s {secret}
```



```
    quotaTimeUnit: 'hour',
    scopes: [ '' ] } ]
downloaded jwt_public_key '-----BEGIN CERTIFICATE-----\nMIICpDCCAYwCCQDet37C82QFzTANBgkqhkiG9w0B
AQsFADAUMRIwEAYDVQQDDAls\nb2NhbGhvc3QwHhcNMTcwMTAzMTM0OTQ4WhcNMTcwMTA0MTM0OTQ4WjAUMRIwEAYD\nVQQD
DAlsb2NhbGhvc3QwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDC\nO4/1cj0k2PfraC6ojWePwveee1YxyQ+hCt
UcN5o/eMXCd2Y+0CG05tpTz09JoRj9\nXgv2RRQ6AoPmB5Z8FqS3BnpvR3DW/Rtex4NE2nFuZXKsuo7mAiCAJG3eeutxI8jw
\n63xhuIuuTvs4HOTShYuP5G9yEH/jbl4GA0YWWiYCkfCs34Qb/U41odYak2Sd02XE\nt5+hmovnkQKERxD17wQUvPnsXaiI
by79if2cqRVau8qJRrxLWiVM3GEhZSDE2FUi\njtasrvWBAH4eDQ7IL9zn61ulGZAp07AQibVeTGUE0CBTEquK40pDck+yOQ
mktMyW\nTVbsNDU3tZvyzUJezQQzAgMBAAEwDQYJKoZIhvcNAQELBQADggEBAECVJgl0ihBG\nK3ZROCabgI6E08tJuY7Iz/
0426NkJ9EFimxsW89PtSYrVJ4t6eaWmlUyAf0aClgD\nZ29EKUpGV+ZwdN05dg5XBAFVH2g3+3wK9I7JMJ7oUjaMTQFERVEU
O0HD21/C3Aq8\nUD9MzR3fZbBB6Dn+dc7Ugk35/QblmF8cY3exYqa+tTDQcWphPQNPBSt5QuTIvScr\nnO7JmMG+Uq6dLVZM
Nb9DS6gRFqVQXA1V5EiPbMq17rpQqXdV58T7wxw/5t5Fp/SN\nfGEPhescpEqAkX0+xG+mPzK7oiJY6seyOpRdKv0Niy2els
uCPQt/QCJyEQctgTqW\nbck0+c2Jbx8=\n-----END CERTIFICATE-----'
Reload Completed Succesfully
srinandans@nandanks:~$
```

NOTE: You **MUST** run the reload command from the **same directory** as where you ran the start command.

You will see the the microgateway instance has now picked up the spike arrest plugin

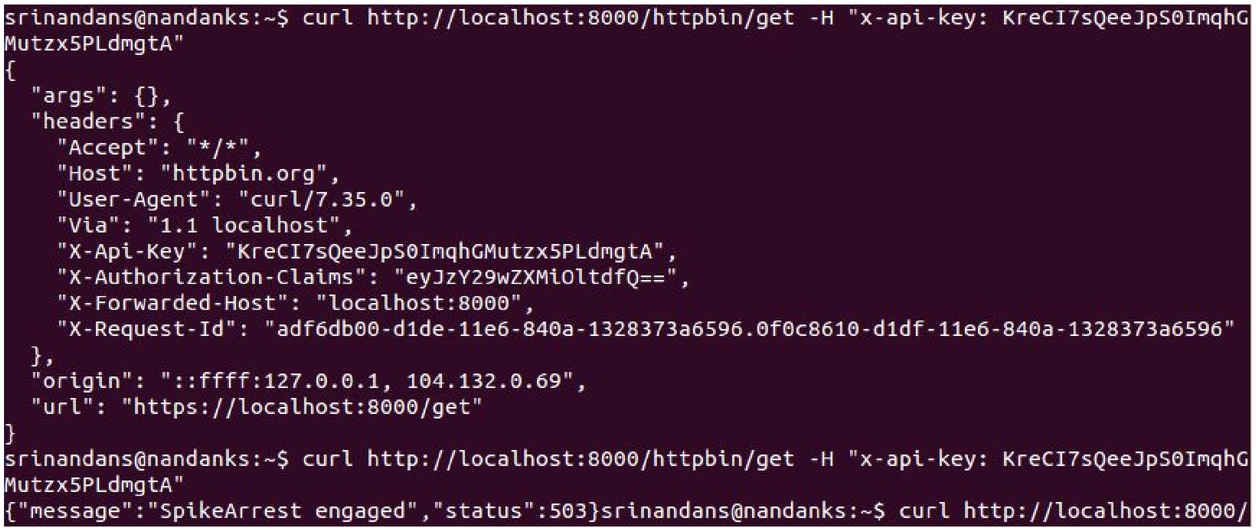


```
Configuration change detected. Saving new config and Initiating reload
Recieved reload instruction. Proceeding to reload
installed plugin from analytics
installed plugin from oauth
installed plugin from spikearrest
installed plugin from analytics
installed plugin from oauth
ade8f850-d1de-11e6-a997-a7a62084f0f1 edge micro listening on port 8000
```

NOTE: You will see an “installed plugin” message for each worker process.

5. Test Configuration: Keep running the following command till you see the failure.

```
curl http://localhost:8000/httpbin/get -H "x-api-key:
KreCI7sQeeJpS0ImqhGMutzx5PLdmgtA"
```



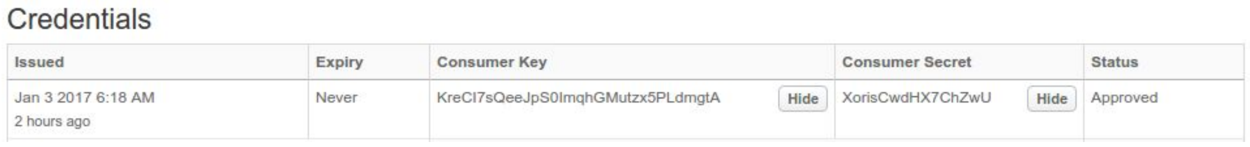
```
srinandans@nandanks:~$ curl http://localhost:8000/httpbin/get -H "x-api-key: KreCI7sQeeJpS0ImqhG
Mutzx5PLdmgtA"
{
  "args": {},
  "headers": {
    "Accept": "*/*",
    "Host": "httpbin.org",
    "User-Agent": "curl/7.35.0",
    "Via": "1.1 localhost",
    "X-Api-Key": "KreCI7sQeeJpS0ImqhGMutzx5PLdmgtA",
    "X-Authorization-Claims": "eyJzY29wZXMiOltdfQ==",
    "X-Forwarded-Host": "localhost:8000",
    "X-Request-Id": "adf6db00-d1de-11e6-840a-1328373a6596.0f0c8610-d1df-11e6-840a-1328373a6596"
  },
  "origin": "::ffff:127.0.0.1, 104.132.0.69",
  "url": "https://localhost:8000/get"
}
srinandans@nandanks:~$ curl http://localhost:8000/httpbin/get -H "x-api-key: KreCI7sQeeJpS0ImqhG
Mutzx5PLdmgtA"
{"message":"SpikeArrest engaged","status":503}srinandans@nandanks:~$ curl http://localhost:8000/
```

NOTE: Spike Arrest configuration is applied for **each worker process**.

Enable OAuth (Access Token Verification)

In this topic, you will be able to access the API proxies hosted by Edge Microgateway using OAuth access tokens. The OAuth plugin serves dual purpose (for validating Access Tokens and API Keys). Since the OAuth plugin is already loaded by microgateway, we can go directly to testing the configuration.

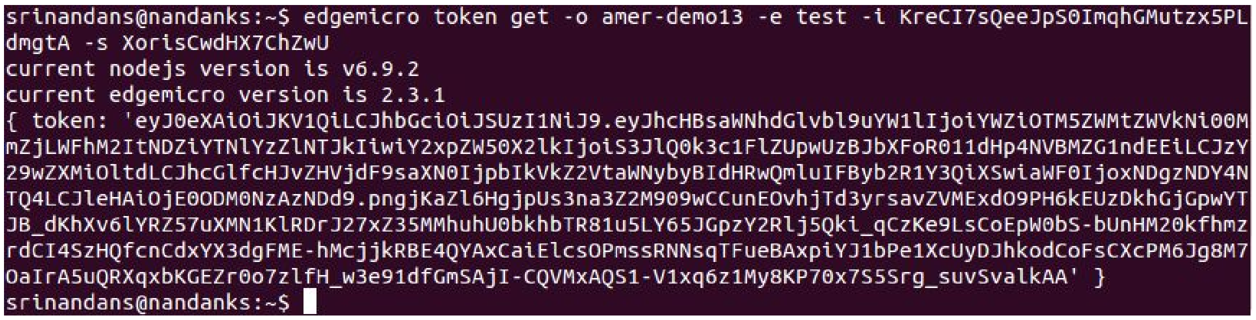
1. Obtain an Access Token: Get the new access token from CLI and by calling an API. Pass the org name, env and consumer key and secret (obtained from the Edge Management UI)



Credentials

Issued	Expiry	Consumer Key	Consumer Secret	Status
Jan 3 2017 6:18 AM 2 hours ago	Never	KreCI7sQeeJpS0ImqhGMutzx5PLdmgtA Hide	XorisCwdHX7ChZwU Hide	Approved

```
edgemicro token get -o {org-name} -e {env-name} -i {consumer key}
-s {consumer secret}
```

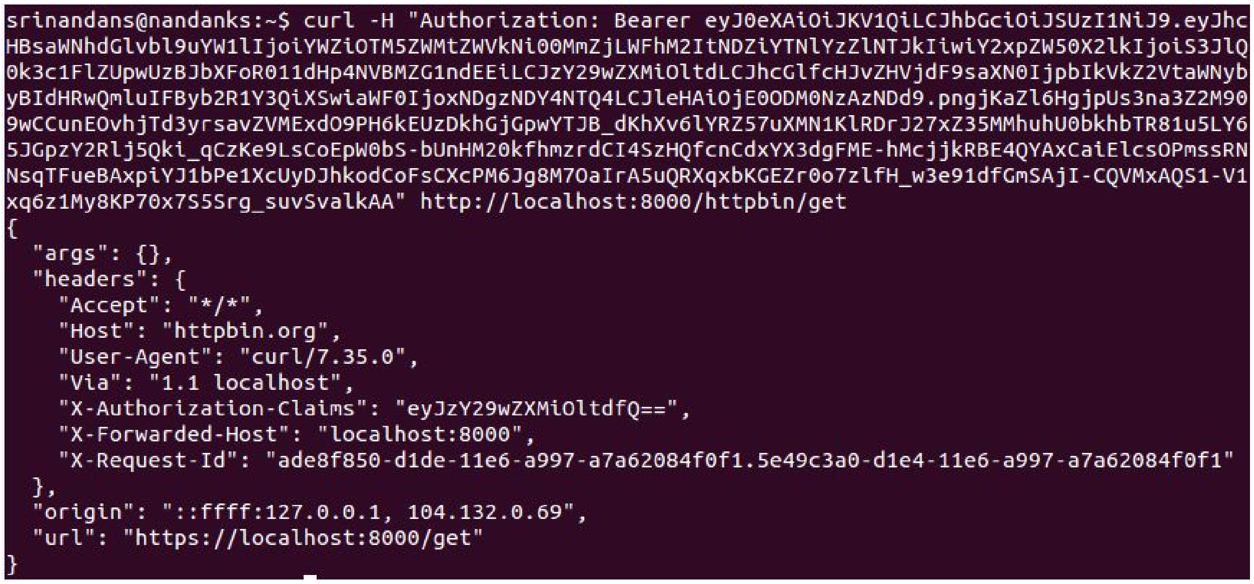


```
srinandans@nandanks:~$ edgemicro token get -o amer-demo13 -e test -i KreCI7sQeeJpS0ImqhGMutzx5PL
dmgtA -s XorisCwdHX7ChZwU
current nodejs version is v6.9.2
current edgemicro version is 2.3.1
{ token: 'eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJhcHBsaWNhdGlvbl9uYW1lIjoiYWZiOTM5ZWMtZWVkNi00M
mZjLWFhM2ItNDZiYTNlYzZlNTJkIiwiY2xpZW50X2lkIjoiS3JlQ0k3c1FlZUpwUzBJbXFoR011dHp4NVBMZG1ndEEiLCJzY
29wZXMiOltdLCJhcGlfcHJvZHVjdF9saXN0IjpbIkVkZ2VtaWNybyBIdHRwQmluIFByb2R1Y3QiXSwiaWF0IjoxNDgzNDY4N
TQ4LCJleHAiOjE0ODM0NzAzNDd9.pngjKaZl6HgjpUs3na3Z2M909wCCunEOvhjTd3yrsavZVMExdO9PH6kEUzDkhGjGpwYT
JB_dKhXv6lYRZ57uXMN1KlRDrJ27xZ35MMhuhU0bkhbTR81u5LY65JGpzY2Rlj5Qki_qCzKe9LsCoEpW0bS-bUnHM20kfhmz
rdCI4SzHQfcnCdxYX3dgFME-hMcjjkRBE4QYAxCaiElcsOPmssRNNsqTFueBAxpiYJ1bPe1XcUyDJhkodCoFsCXcPM6Jg8M7
OaIrA5uQRXqxbKGEZr0o7zlfH_w3e91dfGmSAjI-CQVMxAQS1-V1xq6z1My8KP70x7S5Srg_suvSvalkAA' }
srinandans@nandanks:~$
```

The resulting access token is a JWS (JSON Web Token - Signed). Use this token to make the API calls.

2. Invoke the API

```
curl -H “Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJhcHBsaWNhdGlvbl9uYW1lIjoiYW
ZiOTM5ZWMtZWVkNi00MmZjLWFhM2ItNDZiYTNlYzZlNTJkIiwiY2xpZW50X2lkIjoiS
3JlQ0k3c1FlZUpwUzBJbXFoR011dHp4NVBMZG1ndEEiLCJzY29wZXMiOltdLCJhcGlf
cHJvZHVjdF9saXN0IjpbIkVkZ2VtaWNybyBIdHRwQmluIFByb2R1Y3QiXSwiaWF0Ijo
xNDgzNDY4NTQ4LCJleHAiOjE0ODM0NzAzNDd9.pngjKaZl6HgjpUs3na3Z2M909wCCu
nEOvhjTd3yrsavZVMExdO9PH6kEUzDkhGjGpwYTJB_dKhXv6lYRZ57uXMN1KlRDrJ27
xZ35MMhuhU0bkhbTR81u5LY65JGpzY2Rlj5Qki_qCzKe9LsCoEpW0bS-bUnHM20kfhm
zrdCI4SzHQfcnCdxYX3dgFME-hMcjjkRBE4QYAxCaiElcsOPmssRNNsqTFueBAxpiYJ
1bPe1XcUyDJhkodCoFsCXcPM6Jg8M7OaIrA5uQRXqxbKGEZr0o7zlfH_w3e91dfGmSA
jI-CQVMxAQS1-V1xq6z1My8KP70x7S5Srg_suvSvalkAA”
http://localhost:8000/httpbin/get
```

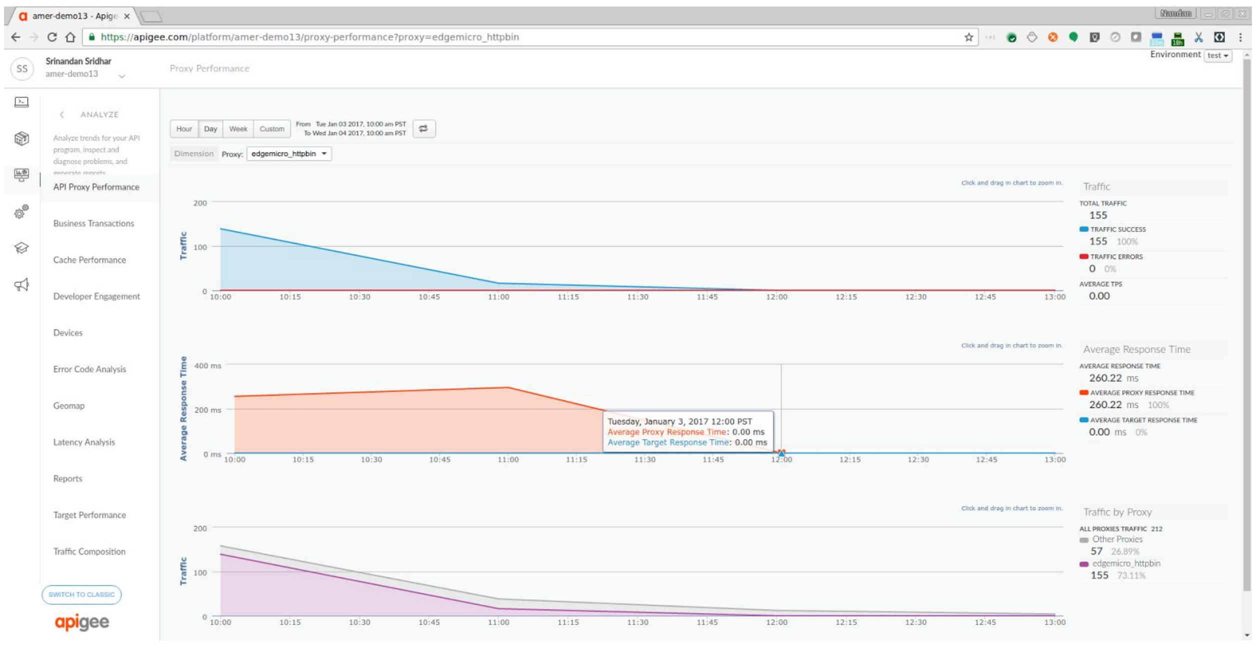


```
srinandans@nandanks:~$ curl -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJhc
HBsaWNhdGlvbl9uYW1lIjoiYWZiOTM5ZWMtZWVkNi00MmZjLWFhM2ItNDZiYTNlYzZlNTJkIiwiY2xpZW50X2lkIjoiS3JlQ
0k3c1FlZUpwUzBJbXFoR011dHp4NVBMZG1ndEEiLCJzY29wZXMiOltdLCJhcGlfcHJvZHVjdF9saXN0IjpbIkVkZ2VtaWNyb
yBIdHRwQmluIFByb2R1Y3QiXSwiaWF0IjoxNDgzNDY4NTQ4LCJleHAiOjE0ODM0NzAzNDd9.pngjKaZl6HgjpUs3na3Z2M90
9wCCunEOvhjTd3yrsavZVMExdO9PH6kEUzDkhGjGpwYTJB_dKhXv6lYRZ57uXMN1KlRDrJ27xZ35MMhuhU0bkhbTR81u5LY6
5JGpzY2Rlj5Qki_qCzKe9LsCoEpW0bS-bUnHM20kfhmzrdCI4SzHQfcnCdxYX3dgFME-hMcjjkRBE4QYAxCaiElcsOPmssRN
NsqTFueBAxpiYJ1bPe1XcUyDJhkodCoFsCXcPM6Jg8M7OaIrA5uQRXqxbKGEZr0o7zlfH_w3e91dfGmSAjI-CQVMxAQS1-V1
xq6z1My8KP70x7S5Srg_suvSvalkAA" http://localhost:8000/httpbin/get
{
  "args": {},
  "headers": {
    "Accept": "*/*",
    "Host": "httpbin.org",
    "User-Agent": "curl/7.35.0",
    "Via": "1.1 localhost",
    "X-Authorization-Claims": "eyJzY29wZXMiOltdfQ==",
    "X-Forwarded-Host": "localhost:8000",
    "X-Request-Id": "ade8f850-d1de-11e6-a997-a7a62084f0f1.5e49c3a0-d1e4-11e6-a997-a7a62084f0f1"
  },
  "origin": "::ffff:127.0.0.1, 104.132.0.69",
  "url": "https://localhost:8000/get"
}
```

Centralized Analytics/Visibility

We now have a fully functioning Edge Microgateway instance, let's see what's it's been up to! By default, the analytics plugin module is added to Edge Micro. This module silently (and asynchronously) pushes analytics data from Edge Micro to Apigee edge, where it is consumed by the Edge Analytics system.

1. Log in to your organization on Apigee Edge.
2. Select **Analyze > API Proxy Performance**.
3. In the **Proxy Performance** dashboard, select the edgemicro_httpbin proxy.
   The graph shows you information about the the proxy's traffic patterns, such as total traffic, average response time, average target response time, and more



Lab Video

If you are lazy and don't want to implement this use case, it's OK. You can watch this short video to see how to enforce throttling on the APIs to protect your backend.
https://youtu.be/XIB3S5hJyWY

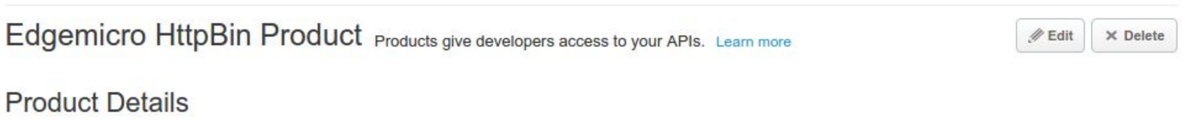
<<Record yourself doing this lab and share the link>>

Earn Extra-points

Instructions

Add or enforce quota policy in Edge microgateway.

1. Log in to the Edge management UI
2. Go to **Publish > API Products**.
3. In the Products page, select "**Edgemicro HttpBin Product**"
4. Edit the product

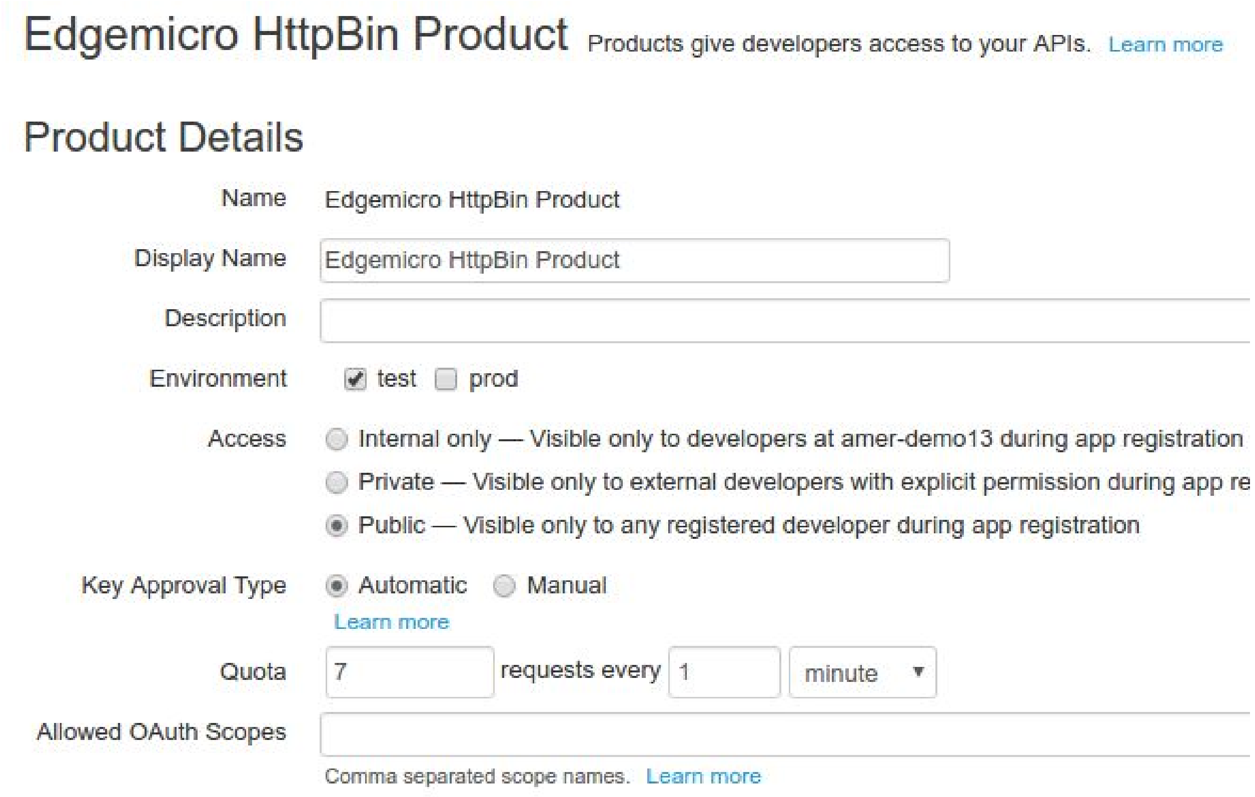

Edgemicro HttpBin Product Products give developers access to your APIs. Learn more

Edit Delete

Product Details

5. Add quota information to the Product in the Edge Management UI

a. Set **Quota**: 7
b. **Requests Every**: 1
c. **Time Unit**: Minute


Edgemicro HttpBin Product

Product Details

Name: Edgemicro HttpBin Product

Display Name: Edgemicro HttpBin Product

Description:

Environment: [✓ test] prod

Access:
Internal only — Visible only to developers at amer-demo13 during app registration
Private — Visible only to external developers with explicit permission during app re
[✓ Public — Visible only to any registered developer during app registration]

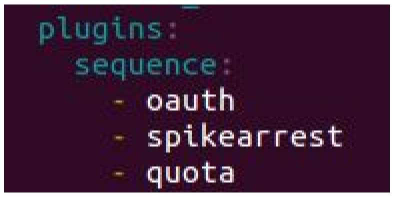
Key Approval Type: [✓ Automatic] Manual

Quota: 7 requests every 1 minute

Allowed OAuth Scopes:

Comma separated scope names.

6. Click **Save**.
7. Edit the configuration file (`~/.edgemicro/{org-name}-{env-name}-config.yaml`) and enable the Quota plugin

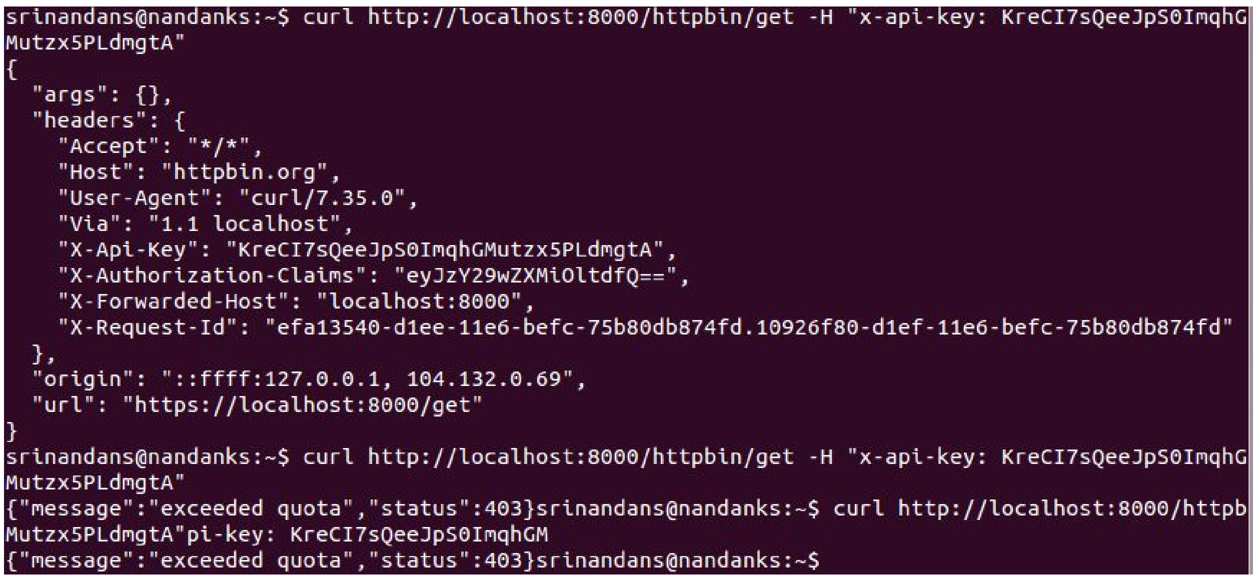


```
plugins:
  sequence:
    - oauth
    - spikearrest
    - quota
```

Test the Configuration

Keep running the following command till you see the failure.

```
curl http://localhost:8000/httpbin/get -H "x-api-key:
KreCI7sQeeJpS0ImqhGMutzx5PLdmgtA"
```



```
srinandans@nandanks:~$ curl http://localhost:8000/httpbin/get -H "x-api-key: KreCI7sQeeJpS0ImqhG
MutzxSPLdmgtA"
{
  "args": {},
  "headers": {
    "Accept": "*/*",
    "Host": "httpbin.org",
    "User-Agent": "curl/7.35.0",
    "Via": "1.1 localhost",
    "X-Api-Key": "KreCI7sQeeJpS0ImqhGMutzx5PLdmgtA",
    "X-Authorization-Claims": "eyJzY29wZXMiOltdfQ==",
    "X-Forwarded-Host": "localhost:8000",
    "X-Request-Id": "efa13540-d1ee-11e6-befc-75b80db874fd.10926f80-d1ef-11e6-befc-75b80db874fd"
  },
  "origin": "::ffff:127.0.0.1, 104.132.0.69",
  "url": "https://localhost:8000/get"
}
srinandans@nandanks:~$ curl http://localhost:8000/httpbin/get -H "x-api-key: KreCI7sQeeJpS0ImqhG
Mutzx5PLdmgtA"
{"message":"exceeded quota","status":403}srinandans@nandanks:~$ curl http://localhost:8000/httpb
Mutzx5PLdmgtA"pi-key: KreCI7sQeeJpS0ImqhGM
{"message":"exceeded quota","status":403}srinandans@nandanks:~$
```

Quiz

1. Do Apigee Edge policies work on Apigee Edge microgateway?
2. Can microgateway expose only some proxies?
3. Can microgateway route based on content or header?
4. How does microgateway sync with Edge?

Summary

In this lab you learned how to install and configure Apigee Edge Microgateway. You were able to proxy APIs on microgateway and enable policies like spike arrest, OAuth, API Key verification and Quotas. Finally, you were able to get a centralized view (analytics) of the APIs hosted on the microgateway.

References

- Link to Apigee docs page
  - Edge Microgateway Setup and Configure: http://docs.apigee.com/microgateway/v21x/edge-microgateway-tutorial-v21x
  - FAQ: http://docs.apigee.com/microgateway/content/edge-microgateway-faq
- Link to Community posts and articles with "Edge Micro"
- Writing Custom Plugins: http://docs.apigee.com/microgateway/latest/develop-custom-plugins

- Docker Apigee Edge Microgateway: https://apigee.com/about/blog/developer/running-apigee-edge-microgateway-docker-container
- Sample docker file: https://github.com/kevinswiber/apigee-edgemicro-docker

Rate this lab

How did you link this lab? Rate here.