Apigee Health APIx Deployment

Prepared by	Persistent Systems
Version	1.2
Date	16th December 2015

Revision History

Version	Date	Modified By	Comments
1.0	18th November 2015	Persistent Team	Deployment guidance of FHIR API proxies with OAuth proxy on Apigee Edge using MAVEN
1.1	26th November 2015	Persistent Team	Updated process for Deployment guidance of All FHIR API proxies with OAuth proxy on Apigee Edge using shell script as Section 2.2.1 Included deployment of identity solution as Section 3 Testing with Postman is Section 4 in this version. Updated section - Steps for Deployment of Single API proxy
1.2	16th December 2015	Persistent Team	Minor Document changes - Sec 3 - Added a step of giving deployment script executable permission - Sec 3 – details of proxies deployed - Sec 4 – Updated Fig 9 for product name.

Table of Contents

1. Introduction

Apigee HealthAPIx software solution assists healthcare providers in accelerating the development and delivery of digital services based on FHIR (Fast Healthcare Interoperability Resources) APIs.

Apigee HealthAPIx is built on the Apigee Edge API management platform, and features FHIR APIs and a healthcare developer portal to help hospitals meet the demand for data interoperability, deliver patient-centric healthcare, and move faster to the digital world.

The Apigee Health APIx solution provides App (both mobile and web) developers and digital teams with access to:

- A purpose-built developer portal with FHIR-ready APIs, including claims, patient, medication, and condition APIs. Explore documentation and test your APIs.
- An open source project with preconfigured API proxies for currently 16 FHIR APIs.
- Pre-integrated OAuth security framework and other key security functions.
- A reference implementation with a gallery of FHIR-enabled apps for inspiration.

This document, in particular, describes API Proxies high level overview and deployment details. The collection of pre-configured API proxies, currently 16 in number, is an open source project at https://github.com/apigee/flame/.

This document also describes OAuth security framework along with other key security functions.

This document only introduces Developer portal and does not scope in the details.

2. Deployment of API Proxies

This section explains the how to deploy those proxies on edge account using shell script.

Each API consists of two or more proxies, One External/Published proxy and at least One Connector/Internal proxy. In addition, oauth_clientcred proxy to generate token is deployed.

2.1 Pre-requisites

One has to have Apigee Edge account and Organizations details to deploy the proxies.
Softwares: Maven, Cygwin(in case of Windows OS).
NOTE: While installing Cygwin, make sure packages for zip, tar, curl, sed are installed

2.2 Deployment of FHIR APIs Proxies using shell script

Get the API proxy bundle from GIT repository

https://github.com/apigee/flame/tree/master/src/gateway/fhir.

Use command: git clone <path of git repository>

On unavailability of git repo, download code from

https://drive.google.com/folderview?id=0B-y5CAUxv0zuU2o5ZFNtOUdEMFE&usp=sharing

Pre-requisites:

1. You need to have access to deployed Apigee Edge Services with organization details. If you don't have this – please sign-up at Apigee Edge now. Maven is used for managing the dependencies and for build automation. Get it installed beforehand.

2. The script is tested on Linux and Windows (using Cygwin). The script should also run on a Mac however it is not tested. In case you would run on Mac, and an error is sees, please let us know.

If you are installing from a Windows machine, you need Cygwin package installed on Windows machine. The deployment would be comparatively slower than Linux server.

Download Cygwin DLL according to the Windows machine type - setup-x86.exe (32-bit installation) or setup-x86_64.exe (64-bit installation).

Run the exe as follows. (Change the exe name for 32 bit installation)

Run `setup-x86.exe -q -P curl`

```
This would install Cygwin with curl package. Follow instructions ahead
```

2.2.1 Steps for Deployment of All FHIR API proxies

This deployment process deploys all proxies by executing a parent install script. Please see the steps below.

1. Go to folder at location : \flame\src\gateway\fhir\setup-apis
2. For Linux machine, check permission of setup.sh script. Script should have execution permission. To change permission, use following command
   chmod +x setup.sh
3. Run setup.sh script using following command
   For Linux/Cygwin, run bash ./setup.sh
4. Following manual inputs required
   - Enter Apigee Enterprise Organization, followed by [ENTER]:
     Here user need to give apigee edge organization name.
   - Enter Organization's Environment, followed by [ENTER]:
     Here user need to enter environment on which deployment is required
     E.g. free account either test or prod. In case of paid organization one of value from test/dev/prod can be entered.
   - Enter Apigee Enterprise LOGIN EMAIL, followed by [ENTER]:
     Here enter registered Email Id
   - Enter Apigee Enterprise PASSWORD, followed by [ENTER]:
     Here enter valid password

Screen shows some execution messages related developer, product and developer app creation. After above step it will ask following inputs:

- To enable connector proxy security, press Y else press N [ENTER]:
  Here you should enter Y, if you want to connector proxies to be accessed from restricted IPs.
  Press Y only if you have IPs from APIGEE for message-processors.
  Press N if you do not have the IPs right now.
  If N is pressed script, starts deployment of proxies directly.
- If Y pressed, It will ask for number of IPs you have. You can give max 10 IPs at a time :
  Enter the number of IP address (max 10), followed by [ENTER]:
- On the based on number given, it will ask IPs. Enter IPs.
  After successful execution of script. It show following message
  Finally, this setup is complete. Have fun by visiting:
  https://enterprise.apigee.com/platform/#/{OrganaizationName}/apis

Once you see this message, your edge account is set to explore APIs.

2.2.2 Steps for Deployment of Single API proxy

The process is similar to above, the only difference is to run the deployment script of a single proxy instead of running a parent install script. Please see the steps below.

- If a user wants to deploy single proxy, go to corresponding proxy directory.
  E.g. for FHIR API Go to \flame\src\gateway\fhir\{Resource}-FHIR-API folder
  For HAPI Connector API go to \flame\src\gateway\fhir\Connector-API-HAPI-DSTU2\{Resource}-Connector-API-HAPI-DSTU2
  For Spark Connector API go to \flame\src\gateway\fhir\Connector-API-SPARK-DSTU2\{Resource}-Connector-API-SPARK-DSTU2
- For Linux machine, check permission of setup.sh script. Script should have execution permission. To change permission, use following command
  chmod +x setup.sh
- Run setup.sh script using following command
  For Linux/Cygwin, run bash ./setup.sh
- Following manual inputs required
  - Enter Apigee Enterprise Organization, followed by [ENTER]:
  - Enter Organization's Environment, followed by [ENTER]:
  - Enter Apigee Enterprise LOGIN EMAIL, followed by [ENTER]:
  - Enter Apigee Enterprise PASSWORD, followed by [ENTER]:
- If user is installing connector proxy, further inputs are asked :
  - To enable connector proxy security, press Y else press N [ENTER]:
    Here you should enter Y, if you want to connector proxies to be accessed from restricted IPs.
    Press Y only if you have IPs from APIGEE for message-processors.
    Press N if you do not have the IPs right now.

If N is pressed script, starts deployment of proxies directly.

- If Y pressed, It will ask for number of IPs you have. You can give max 10 IPs at a time :
  Enter the count of IP addresses (max 10), followed by [ENTER]:
- Based on the count given, it will ask those number of IPs. Enter valid IPs. After successful execution of script. It shows following message "Finally, this setup is complete. Have fun by visiting: https://enterprise.apigee.com/platform/#/{OrganaizationName}/apis"

As said in start of this section, Each API consists of two or more proxies, One External/Published proxy and at least One Connector/Internal proxy. In addition, oauth_clientcred proxy to generate token is deployed.

Currently, Total 36 total proxies must have been deployed. However this is subject to change with addition of more proxies. Please treat this count as a guideline only.

*For Resources: 16*2 (internal+external)*

For Basepath: 3 (1 ext+ 2 internal)

For oauth_clientcred: 1

NOTE: To explore single resource, Its FHIR and Connector proxy need to be deployed on apigee edge. Please check "oauth_clientcred" proxy is deployed. Also Product, developer and developer app is created. If not please use script (Location: flame\src\gateway\fhir\setup_apis\ resources.sh) to create resources. Use command to execute this script bash ./resources.sh (Check permissions, if error occurs).

After executing, following message appears:

"Resource creation process is completed. You can find testuser@apigee.com(as Developer),testFHIRproduct (as product),testFHIRApp(as developer app) are created on given organization."

3. Identity Solution (includes Oauth) Deployment

Deployment of identity solution is similar process as explained in previous section.

Pre-requisite: For Windows OS, need Cygwin package installed on Windows machine. The deployment would be comparatively slower than Linux server.

Download Cygwin DLL according to the Windows machine type - setup-x86.exe (32-bit installation) or setup-x86_64.exe (64-bit installation).

Run the exe as follows. (Change the exe name for 32 bit installation)

Run `setup-x86.exe -q -P curl`

```
This would install Cygwin with curl package. Follow instructions ahead
```

1. Git clone “grass” repo
2. Go to /flame/src/gateway/identityandoauth/setup-identity
3. For Linux machine, check permission of setup.sh script. Script should have execution permission. To change permission, use following command

   chmod +x setup.sh

4. Run bash setup.sh, if you are on Linux/Cygwin
5. When you run the setup.sh script it will ask for your organization name on Apigee Edge, the environment to setup the Identity solution and the Apigee Edge credentials.
6. It creates API service resources (4 cache resources named as consent-session-cache, nonce-cache, auth-req-param-cache, session-cookie-cache), a developer (Identity User named as user@identity.com), product (Identity App product named as identityproduct) and an app (Identity App named as IdentityApp) for the created developer.
7. Then it will ask for the name of the App Services organization i.e. BAAS organization (An app services organization will be created by default when you create an organization on Apigee Edge. So the same organization can be used.) And the name for the App to be created on App services.
8. Post this, it deploys the Identity API Proxies (7 proxies) to your specified organization and deploys to the environment you specified.

Please Note:

The setup.sh needs to be executed from setup-identity folder. It would fail otherwise since relative paths are used from the setup-identity folder.

4. Testing with POSTMAN- REST client

FHIR APIs deployed can be tested through POSTMAN REST client.

POSTMAN collection for all APIs:

flame\src\config\postman\Postman_Script_for_all_resources

A reference environment configuration file for POSTMAN to manage environment variables is available at:

flame\src\config\postman\Apigee_fhirsandbox.postman_environment

User can use the postman environment configuration and replace values as per existing environment.

Insert following values at corresponding places and hit Send button.

URL:

https://{OrgName}-{Environment}.apigee.net/oauth/v1/accesstoken?grant_type=client_credentials&scope={scope}

Scope: For any resource API, it should be “patient/{resource}.read”; for e.g.

For patient resource, it will be patient/Patient.read

For encounter resource, it will be patient/Encounter.read

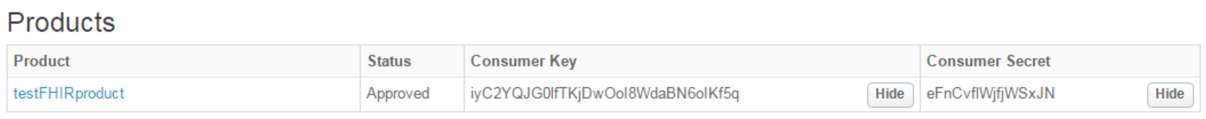
Verb: POST

Headers: Authorization Basic {base64 conversion of consumer key and consumer secret}

Use tool for conversion (https://www.base64encode.org/)

Consumer key and consumer secret will be generated in app creation step (Refer Steps to create product, developer, app)

E.g. In screen short show inside testFHIRApp, Consumer key and Consumer secret is show in front of testFHIRproduct. User should use the generated values for corresponding App. Screenshot shows sample value. Check actual values on your edge instance



Products

Product	Status	Consumer Key	Consumer Secret
testFHIRproduct	Approved	iyC2YQJG0lfTKjDwOol8WdaBN6olKf5q Hide	eFnCvflWjfjWSxJN Hide

Figure 9 Consumer key and consumer secret reference