IOT HUB IMPLEMENTATION (USING APACHE MODULES)

TABLE OF CONTENTS

INTRORDUCTION

Your life and your house went from one computer to an Internet of things (IoT). From tablets to light bulbs, from sensors to media boxes, everybody gets their own Internet. Apple thinks our iPhone or iPad should be the central hub of the Internet of Things also they recently released HomeKit[1], and HealthKit[2]. So that developers can connect their apps and devices more easily to the iPhone. Likewise Google's Nest countered with its own developer kit for smart things in the home and Android Wear [3] was announced recently.

The Internet of Things actually comprises of many layers, each with its own medium and protocols. Most of the applications may require considerable local processing in the hubs.

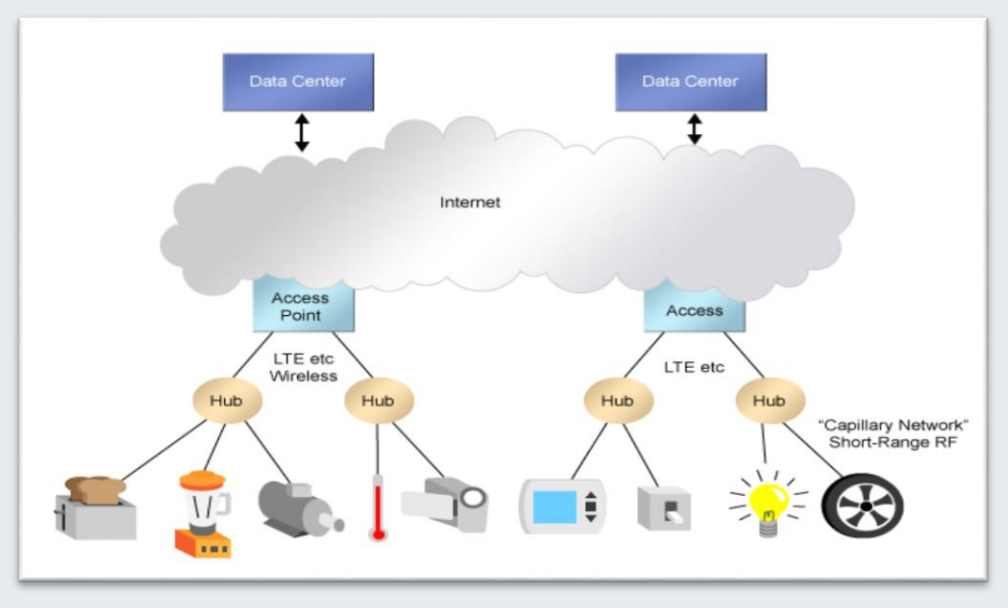


Figure 1: How IoT hub concept apply existing system

Does everyone make anything remotely smart wants to be at the center of the Internet of Thing? Can everybody agree on how the Internet of things should be connected or controlled? Does this IoT mean different things to different people so how they should be connected each other. How about centralize IoT hub concept?

PURPOSE

For Nest [4], it is home automation. For GE [5], it is industrial automation. For Android Wear [3], it is a bevy of personal fitness trackers on your body. So this centralize hub architecture can be used to connect those things together as one package.

SCOPE

Main scope is project is to centralize existing services or add new service as end uses want. Here services can be really complex ones such as what Google Nest does or simple service like home automation things. This proposed system main task is to give a secure service to clients. So every end uses who requested this service first needs to be registered on WSO2 API Manager through this and get their Access token before acquiring any services. There are several benefits can be gain acquiring this service. What will happens accessing service as mention above directly?

The IoT elements will have to cope with many protocols and formats, and data complexity greater than in other environments. Use cases will change rapidly. The network will be always-on, capable of remote management, provisioned with local intelligence. And it must be secure. Security may be

the proverbial elephant in the corner as the IoT evolves from remote-control unit to front-end for big data. A hacker exploiting the IoT to maliciously turn off my home thermostat would be an annoyance. But if that hacker could manipulate my furnace to trigger an explosion, or intercept sensor data to identify when my child is home along, security would take on an entirely different imperative.

Can us able to provide all those requirement to each services separately. Answer is no. That is why hub architecture is required to tackle this problems.

Application overview

API management is about service oriented architecture for the consumers of an API, it should not matter what the underlying functionality is. The API should just meet the contract without having to reveal the underlying implementation, and should guarantee delivery. Also it should be provided Security, usability, throughput and availability as well as governance is also important to provide enterprise sensibilities in documentation, version control and delivery of business processes for the management of APIs. If we have small devices which intended to provide some services how we are going to connect those devices to API Manager That is where IoT hub is involved. Centralize hub can be used to communicate with that small devices as well as API Manager

Application Characteristics

- *to operate in real-time*
- *a small number of concurrent users*
- *to be highly resilient or fault tolerant*
- *to provide security features to protect data*
- *to be scaleable and easily maintainable in the future*
- *to deploy in small devices so processing power is really low*
- *in memory database is used so can not do IPC(Inter Process Comunication)*

Application Architecture

This API will be based upon an n-tier, client-server architecture. This architecture will have secure managed reverse proxy is used between end-user and IoT hub to isolate REST API from illegal access. So there are no ways to communicate between clients. Because that is not the purpose of this. So client are secure through number of ways. There should be a HTTPS connection between end users and IoT hub. OAuth 2.0 is used as access authentication method.

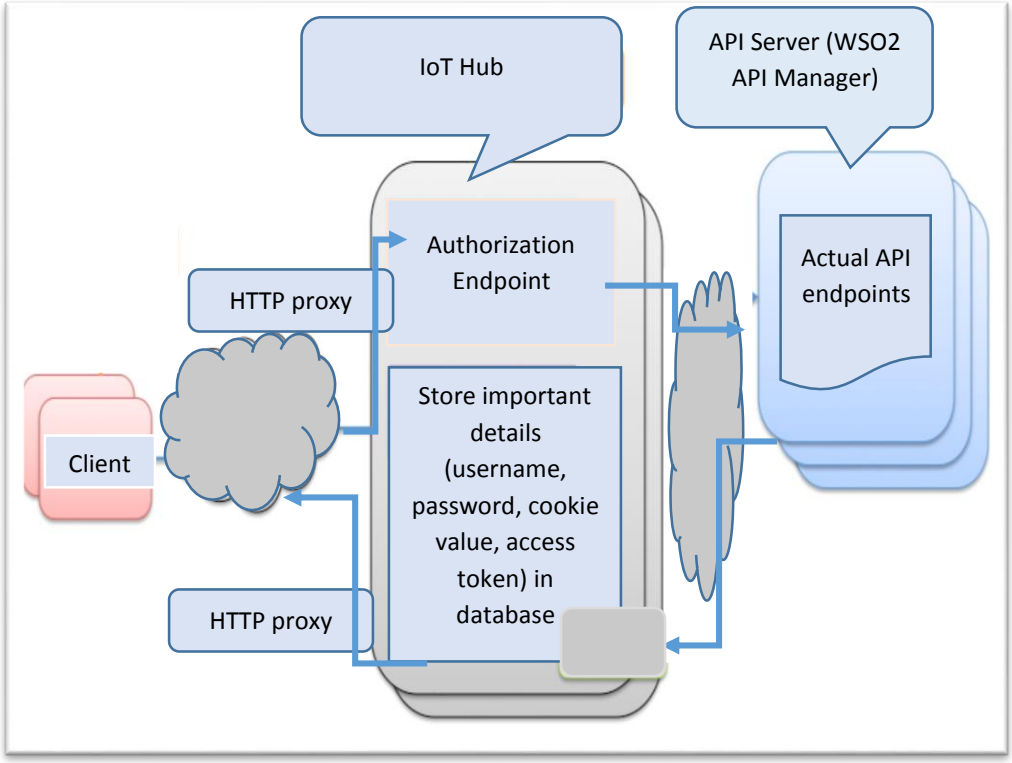


Figure 02: Application architecture

REST API OVERVIEW

This section describes the main features and characteristics of this API. So main task of this project is to create a REST API (Representational State Transfer) follows a RESTful architecture. In this API there are two type endpoints available. In particular, for administrative tasks such as creating a new API and publishing that API and normal end user endpoints such as user login, user add applications and so on.

RREST API CHARACTERISTICS

HTTP request header

Header	Supported Values	Description of Use	Required
Accept	Comma-delimited list of media types or media type patterns.	Indicates to the server what media type(s) this client is prepared to accept.	Recommended
Authorization	Used https, so server public key is requested	Identifies the authorized user making this request.	Yes , It must required
Content-Length	Body Length (in bytes)	Describes the size of the message body.	No, content-length is calculated by server

Header	Supported Values	Description of Use	Required
Content-Type	application/x-www-form-urlencoded	Describes the representation and syntax of the request message body.	Yes

Table 01

JSON Responses

The REST API returns many responses as JavaScript Object Notation (JSON). On that note, you may find it convenient to read responses in a JSON reader. Some responses may have an empty body, but indicate the response with standard HTTP codes.

Types of resources

There are number of different resources within the API and these resources require a different URI/RESTful-endpoint in order to perform an operations. Examples are given what are the request message and how it performs.

Note: Use your domain name and port where your IoT hub is deployed in each request.

Example: domain: www.example.com, Port: 443

https:// www.example.com:443/apimanager/user/signup

Sign Up

Description	Add a new API Consumer
URI	https://localhost/apimanager/user/sign-up
URI Parameters	action=addUser&username=user1&password=test123&allFieldsValues=firstname|lastname|email
HTTP Methods	POST
Example	Curl -X POST -c cookies https://localhost/apimanager/user/sign-up 'action=addUser&username=user1&password=test123&allFieldsValues=firstname|lastname|email '

Log in

Description	Log in to API Store. Through IoT hub
URI	https://localhost/apimanager/user/login
URI Parameters	action=login&username=xxx&password=xxx&tag=normal

HTTP Methods	POST
Example	Cur -X POST -c cookies https://localhost/apimanager/user/login -d 'action=login&username=admin&password=admin&tag=normal'

Description	Log in to server as administrative user.
URI	https://localhost/apimanager/apiuser/login/
URI Parameters	action=login&username=xxx&password=xxx&tag=apiuser
HTTP Methods	POST
Example	curl -X POST -c cookies https://localhost/apimanager/apiuser/login/ -d 'action=login&username=admin&password=admin&tag=apiuser'

Add application

Description	To subscribe any API which were published In WSO2 API Manager, You need to have an application. So create an application.
URI	https://localhost/apimanager/user/application-add
URI Parameters	action=addApplication&application=xxx&tier=xxx&description=xxx&callbackUrl&username=xxxx
HTTP Methods	POST
Example	curl -X POST -b cookies https://localhost/apimanager/user/application-add -d 'action=addApplication&application=NewApp1&tier=unlimited&description=&callbackUrl=&username=xxxxx'

Add subscription

Description	WSO2 API Manager is used Auth2.0 as authentication mechanism. To get your access token you have to subscribe your application.
URI	https://localhost/apimanager/user/subscription-add
URI Parameters	action=**addAPISubscription**&name=xxx&version=xxx&provider=xxx&tier=xxx&**applicationName**=xxx&username=xxxx
HTTP Methods	POST
Example	curl -X POST -b cookies https://localhost/apimanager/user/subscription-add -d 'action=addAPISubscription&name=TestAPI&version=1.0.0&provider=admin&tier=Gold&applicationName=DefaultApplication&username=xxxx'

Generate application key

Description	Ones you subscribed an application you can generate random key and it will be used to future communication.
URI	https://localhost/apimanager/user/application-key-generate

URI Parameters	action=generateApplicationKey&application=xxxxxxx&keytype=PRODUCTION&callbackUrl=xxxxxx&username=xxxxx
HTTP Methods	POST
Example	curl -X POST -b cookies https://localhost/apimanager/user/application-key-generate -d action=generateApplicationKey&application=xxxxxxx&keytype=PRODUCTION&callbackUrl=xxxxxx&username=xxxxx '

Add API

Description	If somebody made an API and that can be added to WSO2 API Manager as a service
URI	https://localhost/apimanager/apiuser/add-api/
URI Parameters	"action=addAPI&name=xxx&context=/xxx&version=x.x.x&visibility=<public|private|restricted>&thumbUrl<URL>&description=xxx&tags=x,y,z&resourceCount=0&resourceMethod-0=GET&resourceMethodAuthType-0=Application&resourceMethodThrottlingTier-0=Unlimited&endpointType=<secured|nonsecured>&wsdl=xxx&tier=<Gold,Silver,Bronze,Unlimited>&tiersCollection=Gold&http_checked=http&https_checked=https&" -d'endpoint_config={"production_endpoints":{"url":"<URL>","config":null},"endpoint_type":"http"}';

HTTP Methods	POST
Example	curl -X POST -b cookies https://localhost/apimanager/apiuser/add-api/ -d "action=addAPI&name=PhoneVerification&context=/phoneverify&version=1.0.0&visibility=public&thumbUrl=&description=Verify a phone number&tags=phone,mobile,multimedia&endpointType=nonsecured&wsdl=&wadl=&tier=Silver&tiersCollection=Gold,Bronze&http_checked=http&https_checked=https&resourceCount=0&resourceMethod-0=GET&resourceMethodAuthType-0=Application&resourceMethodThrottlingTier-0=Unlimited&uriTemplate-0=/*" -d'endpoint_config={"production_endpoints":{"url":"http://ws.cdyne.com/phoneverify/phoneverify.asmx","config":null},"endpoint_type":"http"}';

Publish an API

Description	Ones you added an API it can be published
URI	https://localhost/apimanager/apiuser/publish-api

URI Parameters	action=updateStatus&name=xxx&version=1.0.0&provider=apiCreateName&status=PUBLISHED&publishToGateway=true&requireResubscription=true&usrname=xxxxx
HTTP Methods	POST
Example	curl -X POST -b cookies 'https://localhost/apimanager/apiuser/publish-api' -d 'action=updateStatus&name=PhoneVerification&version=1.0.0&provider=admin&status=PUBLISHED&publishToGateway=true&requireResubscription&username=xxxxxx

REST API installation process

Prerequisites

1).Install Apache HTTP server

```
sudo apt-get install apache2
```

1).Install sqlite3 library

```
sudo apt-get install libsqlite3-dev
```

3).Install curl library

```
sudo apt-get libcurl4-openssl-dev
```

4).Install fcgi library

```
sudo apt-get install libfcgi-dev
```

Change these properties in endpointConfig.txt in conf directory

CGI_BIN

Change this to your Apache HTTP server CGI_BIN location

Example -: /usr/local/apache2/cgi-bin/

DB_PATH

Provide any location where you would like to store your database

ENDPOINT URLS

Here you have to change the domain name of server where WSO2 API Manager is currently running.

Add REST API endpoint urls in serverConfig.json in conf directory

Finally to deploy this module in Apache HTTP server just run in bin directory

sudo ./configure.sh

Testing

4.1 Purpose

This test plan describes the testing approach and overall framework that will drive the testing of the REST API. This introduces:

- Test Strategy: rules the test will be based on
- Execution Strategy: describes how the test will be performed and process

4.2Test Strategy

4.2.1 Test Objectives

The objective of the test is to verify that the functionality of REST API works according to the specifications as well as performing load tests on Apache HTTP server.
The test will execute and verify how it works and will make performance testing comparing with CGI over FCGI to verify how this behave when increasing load
The final product of the test is twofold: A production-ready software and a set of stable test scripts that can be reused for Functional test execution.

4.2.2 Test Assumptions

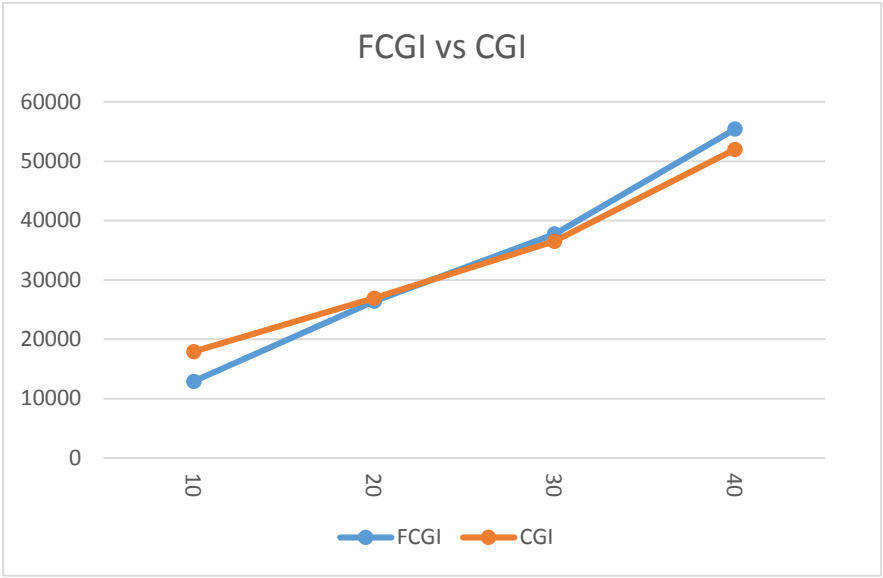
- Make sure that the operating system doesn't impose artificial limits on server's performance.
- Restricted use of the system during testing

4.3 Execution Strategy

Make random users who can perform all REST API call (sign up, login, add application, add subscription and generate application key) and start the test. Then total elapsed time is calculated for whole process. This will be continued increasing number of random users.

Number of random users	10	20	30	40
CGI	12886.5	26405.8	37724.7	55422.1
FCGI	17910.6	26908.1	36525.7	51998.2

Table 02 .Total elapsed time (ms) vs. Number of random users



Graph 01 Number of random users vs. total elapsed time (ms)

According to figure graph 01 .It is easy to see from this graph when the load is higher executing time of FCGI is less than CGI executing time. Even long running fastcgi is known to have memory leaks problems when we compare performance FCGI is better that CGI.

REFERENCE

1. https://developer.apple.com/homekit/
2. https://developer.apple.com/healthkit/
3. http://www.android.com/wear/
4. https://workswithnest.google.com/
5. http://www.geautomation.com/products/industrial-automation