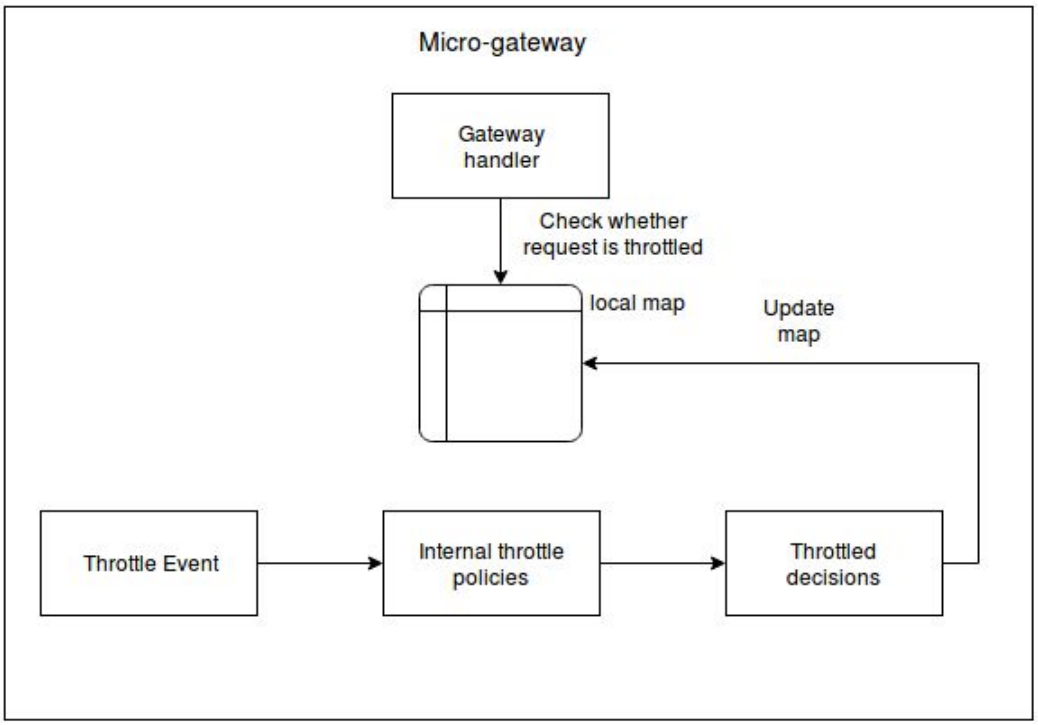
http://wso2.com/

Distributed Throttling for API Micro-gateway

Distributed Throttling In Micro-gateways

Problem

- Micro-gateway does not have access to central traffic manager system with the current architecture.
- Micro-gateway manages throttling in node level by maintaining node local counters. Existing implementation of throttling in micro-gateway is as follows.



- In scalable gateway environment this can be a problem as gateways scale dynamically. When it comes to application level throttling and backend throttling, node level would not be sufficient as number of instances can multiply allowed limits.

Solution

- Existing traffic manager solution which is connected to the API gateway executes the throttle policies against data coming with every published event and takes decisions based on the applicability of each throttle policy available in the system. If a particular request is throttled, then the Traffic Manager sends those details to a JMS topic. Every gateway node is subscribed to this JMS topic; hence the gateway nodes get notified of throttle decisions through JMS messages.
- This traffic manager solution can be used and need to be connected to micro-gateway.

- In case of a lock down environment or offline mode which do not have connection with Central Traffic management solution, node level throttling is also required.

Approach

- Enabling or disabling the connection with central traffic management solution is done through configuration file. In micro-gw.conf file throttling property is defined.
  [throttlingConfig]
  enabledGlobalTMEventPublishing=true

  When **enabledGlobalTMEventPublishing** is 'true' then the connection to central traffic management solution or distributed throttling is enabled.

- Throttle events are generated in the microgateway. That throttle event need to be published to traffic manager. To publish the data from micro-gateway to traffic manager http protocol is used.
- Json payload of the throttle event that is sent is in the format as follows.

```
json sendEvent = {
event: {
metaData: {},
correlationData: {},
payloadData: {
    messageID: throttleEvent.messageID,
    appKey: throttleEvent.appKey,
    appTier: throttleEvent.appTier,
    apiKey: throttleEvent.apiKey,
    apiTier: throttleEvent.apiTier,
    subscriptionKey: throttleEvent.subscriptionKey,
    subscriptionTier: throttleEvent.subscriptionTier,
    resourceKey: throttleEvent.resourceKey,
    resourceTier: throttleEvent.resourceTier,
    userId: throttleEvent.userId,
    apiContext: throttleEvent.apiContext,
    apiVersion: throttleEvent.apiVersion,
    appTenant: throttleEvent.appTenant,
    apiTenant: throttleEvent.apiTenant,
    appId: throttleEvent.appId,
    apiName: throttleEvent.apiName,
    properties: throttleEvent.properties

}
}
};
```

- This json message is published to traffic manager using http protocol.

```
endpoint http:Client throttleEndpoint {
        url: "https://localhost:9443/endpoints"
};

http:Request clientRequest = new;
clientRequest.setPayload(sendEvent);
var response = throttleEndpoint -> post("/throttleEventReceiver",
clientRequest);
```

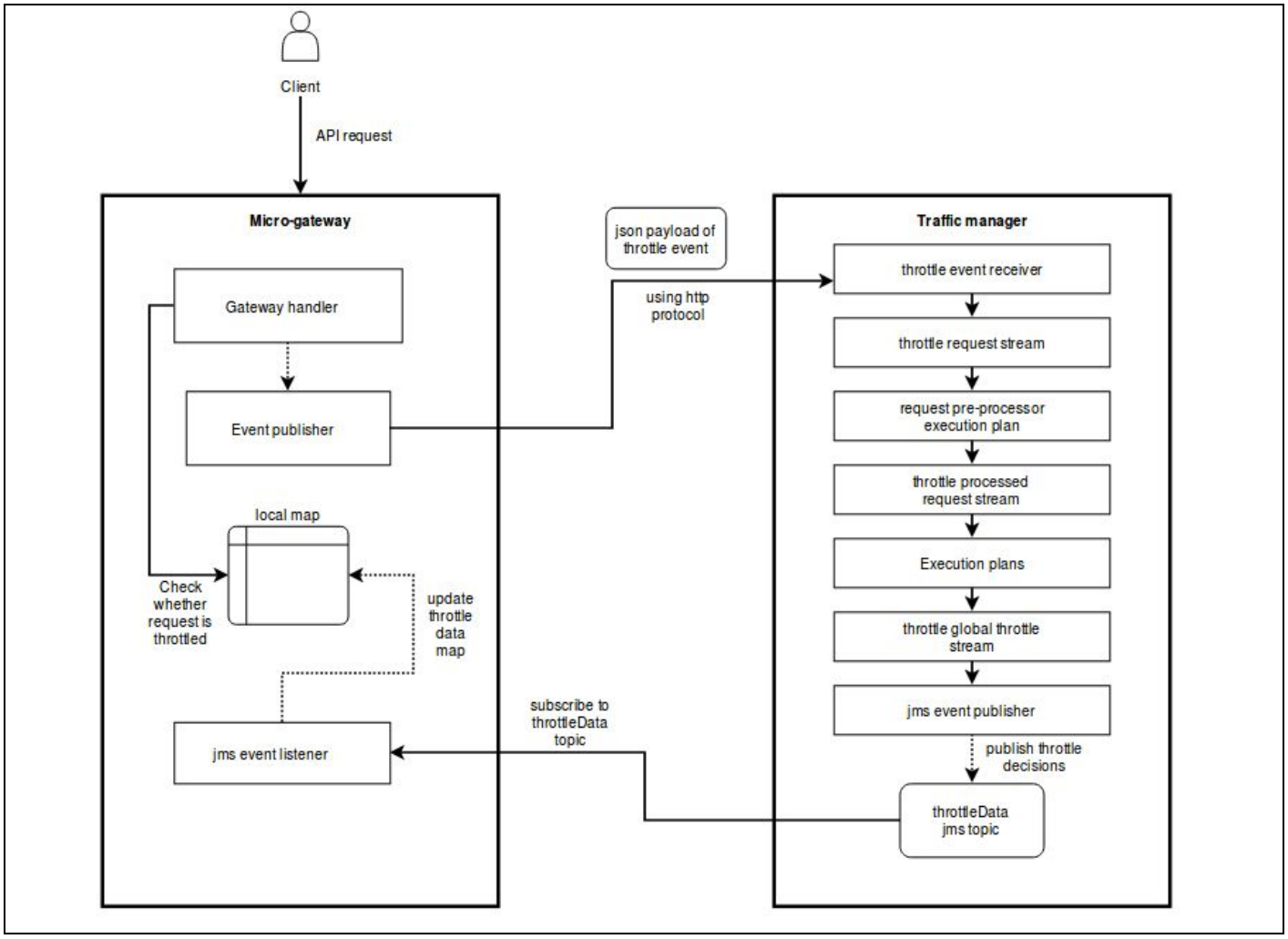
- In the central traffic manager *'throttleEventReceiver'* receives that throttle data and sends those data to *'org.wso2.throttle.request.stream'*. From that stream, data is sent to *'requestPreProcessorExecutionPlan'* and through *'org.wso2.throttle.processed.request.stream'* the data is published to execution plans. The throttled decisions are published to *'org.wso2.throttle.globalThrottle.stream'* and those throttled decisions are sent to *'jmsEventPublisher'*.
- Then in micro-gateway a jms event listener is created and that jms event listener is subscribed to the 'throttleData' topic.

```
jms:Connection jmsConnection = new ({
        initialContextFactory: "bmbInitialContextFactory",
        providerUrl:
"amqp://admin:admin@carbon/carbon?brokerlist='tcp://localhost:5672'"
        });

jms:Session jmsSession = new(jmsConnection, {
        acknowledgementMode: "AUTO_ACKNOWLEDGE"
        });
endpoint jms:TopicSubscriber subscriberEndpoint {
        session: jmsSession,
        topicPattern: "throttleData"
};
```

- Then through that subscriber endpoint the throttled decisions which are in jms message format are received by the micro-gateway jms event listener. When throttled decision is received, the local maps in the micro-gateway are updated according to that.

  The approach is represented in the following diagram.



How to enable the feature

1. Open “micro-gw.conf” file and locate “enabledGlobalTMEventPublishing” property.

```
[throttlingConfig]
enabledGlobalTMEventPublishing=false
```

When this property is false throttle event is published to internal policies.

2. Change “enabledGlobalTMEventPublishing” property as true.
When this property is true throttle event is published to central traffic manager.

Scenarios

Scenario	Use either distributed throttling or node level throttling (using internal throttle policies) in micro-gateway.
Background	All users can decide whether they want to use distributed throttling or node level throttling in micro-gateway.

User Story 1

User Story	Micro-gateway that is using node level throttling(using internal throttle policies).
Overview	In case of a lock down environment or offline mode which do not have connection with Central Traffic management solution, node level throttling is required.
Person	API creator, API Publisher
Prerequisites	None
Narrative	1. User setup micro-gateway distribution for a project. 2. Then user build micro-gateway distribution for the project and no additional steps are needed.
Acceptance Criteria	1. When node level throttling is enabled micro-gateway publishes the throttle event to internal policies and throttling is done.

User Story 2

User Story	Micro-gateway that is using distributed throttling.
Overview	In case of a scalable gateway environment when gateways scale dynamically, distributed throttling is required.
Person	API creator, API Publisher
Prerequisites	None
Narrative	1. User setup micro-gateway distribution for a project. 2. User build the micro-gateway distribution for the project.

	3. User change the micro-gw.conf file “enabledGlobalTMEventPublishing” property as ‘true’.
Acceptance Criteria	1. When distributed throttling is enabled, micro-gateway publishes throttle event to central traffic management solution and in traffic manager throttling is done and throttled decisions are received by the micro-gateway.