Google Cloud

Edge Advanced

Fault Rules and Error Response

Fault Handling

Faults in Edge are similar to exceptions in programming languages.

When a fault is raised, current policy processing is aborted and switches to error processing called **<FaultRules>.** This is supported in **Proxy Endpoint** and **Target Endpoint** configurations only.

```
<Step>
    <Name>SpikeArrest</Name>
</Step>
<Step>
    <Name>SetConfigurationVariables</Name>
</Step>
<Step>
    <Name>VerifyApiKey</Name>
</Step>
<Step>
    <Name>QuotaPolicy</Name>
</Step>
```

```
1  <DefaultFaultRule>
2      <Step>
3          <Name>SyslogPolicy</Name>
4      </Step>
5      <AlwaysEnforce>true</AlwaysEnforce>
6  </DefaultFaultRule>
7  <FaultRules>
8      <FaultRule name ="Fault.InvalidKey">
9          <Step>
10             <Name>Add-WWW-Authenticate-Header</Name>
11         </Step>
12     <Condition>(fault.name =="invalid_consumer_key")</Condition>
13     </FaultRule>
14 </FaultRules>
15 <HTTPProxyConnection>
16         <BasePath>/certification/v1/weather</BasePath>
17         <VirtualHost>default</VirtualHost>
18         <VirtualHost>secure</VirtualHost>
19 </HTTPProxyConnection>
```

Fault Handling (cont'd)

<FaultRule>

- Support for multiple fault rules, executed conditionally.
- Raised manually or automatically upon policy failure.
- If multiple conditional FaultRules defined, their conditions are evaluated in reverse order (bottom up).

<DefaultFaultRule>

- A catch-all / post processing fault rule is available using the flow
- If no FaultRule is matched, this flow will execute.
- Use <AlwaysEnforce>true</AlwaysEnforce> to use this flow for post processing in error flows. This will ensure the flow executes after a matched fault rule has completed its processing.

3 scenarios where processing will switch to fault processing:

- Using RaiseFault policy.
- Any policy failure when contineOnError=false (default setting).
- Non-success response received from service callout or backend request (4XX, 5XX status codes).

```
1  <DefaultFaultRule>
       <Step>
           <Name>SyslogPolicy</Name>
       </Step>
       <AlwaysEnforce>true</AlwaysEnforce>
   </DefaultFaultRule>
   <FaultRules>
       <FaultRule name ="Fault.InvalidKey">
           <Step>
               <Name>Add-WWW-Authenticate-Header</Name>
           </Step>
       <Condition>(fault.name =="invalid_consumer_key")</Condition>
       </FaultRule>
14 </FaultRules>
15 <HTTPProxyConnection>
16         <BasePath>/certification/v1/weather</BasePath>
17         <VirtualHost>default</VirtualHost>
18         <VirtualHost>secure</VirtualHost>
19 </HTTPProxyConnection>
```

Rewriting Backend Error Responses

Edge has a pre-defined fault response format:

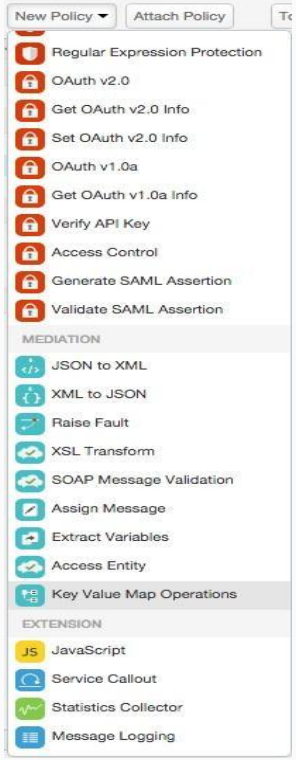
You can either:

1. Rewrite backend error responses to match Edge's format
2. Rewrite Edge fault responses to match the backend error response format

```
{
   "fault":{
      "faultstring":"%errorMessage#",
      "detail":{
         "errorcode":"%errocode#"
      }
   }
}
```

Raising a Fault (error)

Add a Raise Fault Policy



Name and Select the Proxy Endpoint (Ratings (Post)) from the flows,

Modify the content to return an appropriate fault message

Code: raise_fault_invalid_post

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RaiseFault async="false" continueOnError="false" enabled="true" name="raise_fault_invalid_post">
    <DisplayName>raise_fault_invalid_post</DisplayName>
    <FaultRules/>
    <Properties/>
    <FaultResponse>
        <Set>
            <Headers/>
            <Payload contentType="application/json">\{"error":"Invalid Post Data"}</Payload>
            <StatusCode>400</StatusCode>
            <ReasonPhrase>Bad Request</ReasonPhrase>
        </Set>
    </FaultResponse>
    <IgnoreUnresolvedVariables>true</IgnoreUnresolvedVariables>
</RaiseFault>
```

Raise Fault Policy Details

Fault Response Tag Format

- Set – Allows you to build an inline response to fault
  - StatusCode – HTTP Status Code
  - ReasonPhrase – HTTP Reason Phrase
  - Payload – Message Contents for your fault (follows same spec as assign message)
- Copy – Allows you to copy the request or the response into the fault response
  - Attr:source – request or response object

Note : Raise Fault can also be used as a mechanism to stop the request flow before getting to the target in a success scenario.

Raise Faults work with conditions

Select Post Ratings Resource from navigation pane

Add a condition to the Step (policy)

Code: post_ratings

```
<Flow name="post_ratings">
    <Description/>
    <Request>
        <Step>
            <FaultRules/>
            <Name>extract_post_data_ratings</Name>
        </Step>
        <Step>
            <FaultRules/>
            <Name>raise_fault_invalid_post</Name>
            <Condition>((truck = NULL) or (comment = NULL) or (commenter = NULL) or (score = NULL))</Condition>
        </Step>
    </Request>
    <Response/>
    <Condition>(proxy.pathsuf  latchesPath &quo  tings&quot;) an  quest.verb = &q  OST&quot;)</Condition>
</Flow>
```

THANK YOU