Azure API Management troubleshooting documentation

Welcome to Azure API Management troubleshooting. These articles explain how to determine, diagnose, and fix various issues that you might encounter when you use Azure API Management. In the navigation pane on the left, browse through the article list or use the search box to find issues and solutions.

Azure API Management

HOW-TO GUIDE

Azure API is returning blank response

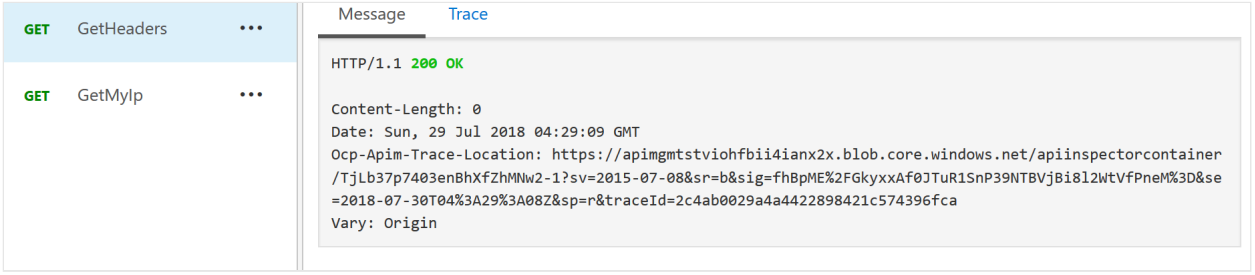
Article • 04/12/2024

Referring to the article on Azure API Management Troubleshooting Series, this is the first scenario of the lab. Make sure you have followed the lab setup instructions as per this, to recreate the problem.

Original product version: API Management Service
Original KB number: 4464936

Symptoms

The API **Blank API** consists of two operations **GetHeaders** and **GetMyIp**. **GetMyIp** returns the value of X-FORWARDED-FOR header value and **GetHeaders** returns all the request header values. **GetMyIp** returns expected output but suddenly **GetHeaders** started returning a blank response (no response body).



Expected output of **GetHeaders** API should be something like below:

HTML

```
{
  "headers": {
    "Accept":
"text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8",
    "Accept-Encoding": "gzip, deflate",
    "Accept-Language": "en-US,en;q=0.5",
    "Connection": "close",
    "Cookie": "_gauges_unique_day=1; _gauges_unique_month=1;
_gauges_unique_year=1; _gauges_unique=1",
    "Host": "eu.httpbin.org",
    "Upgrade-Insecure-Requests": "1",
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:59.0)
Gecko/20100101 Firefox/59.0"
  }
}
```

Troubleshooting Steps

To debug this kind of issues, the best approach is collect APIM inspector trace to inspect request processing inside APIM pipeline.

- If you look into the trace, you would notice that forward-request policy is missing.
- The forward-request policy forwards the incoming request to the backend service specified in the request context.
- Removing this policy results in the request not being forwarded to the backend service and the policies in the outbound section are evaluated immediately upon the successful completion of the policies in the inbound section.
- Hence if you check the <backend> section of the **GetHeaders** operation under **Blank-API** you would notice that forward-request policy is removed.
- Add the forward-request policy in the backend section or add **<base />** element so that it inherits forward-request policy from the parent level (i.e. from the API level), which should resolve the problem.

Read about the forward-request policy to know more about it.

Contact us for help

If you have questions or need help, create a support request, or ask Azure community support. You can also submit product feedback to Azure feedback community.

Feedback

Request throttling problems and HTTP 403 - Forbidden issues

Article • 04/12/2024

Referring to the article on Azure API Management Troubleshooting Series, this is the fifth scenario of the lab. Make sure you have followed the lab setup instructions as per this, to recreate the problem.

Original product version: API Management Service
Original KB number: 4464928

Symptoms

The **Resources** API fetches user's personal details, social media posts, comments, and photos and utilizes the response returned for a machine learning project. Strangely after few days of using it, **GetPosts** operation started throwing **HTTP 403 - Forbidden** error whereas the other operations are working fine as expected.

```
{
"statusCode": 403,
"message": "Forbidden"
}
```

Apart from the above, we are also encountering **HTTP 429 - Too many requests** error while invoking **GetComments** operation for every second request. The issue automatically gets resolve after 10 secs, however it reoccurs once the first call to API is made again. The behavior is not observed for the other operations.

```
{
"statusCode": 429,
"message": "Rate limit is exceeded. Try again in 5 seconds."
}
```

Troubleshooting steps

- **HTTP 403 - Forbidden** error can be thrown when there is any access restriction policy implemented.

- Check the APIM inspector trace and you should notice the existence of a 'ip-filter' policy that filters (allows/denies) calls from specific IP addresses and/or address ranges.

- To check the scope of the 'ip-filter' policy, select the **Calculate effective policy** button. If you don't see any access restriction policy implemented at any scopes, next validation step should be done at product level, by navigating to the associated product and then click on Policies option.

  XML

  ```
  <inbound>
      <base />
      <choose>
          <when condition="@(context.Operation.Name.Equals("GetPosts"))">
              <ip-filter action="forbid">
                  <address-range from="0.0.0.0" to="255.255.255.255" />
              </ip-filter>
          </when>
      </choose>
  </inbound>
  ```

- For the second issue (**HTTP 429 - Too many requests**) we will follow the same procedure by checking the APIM inspector trace and check if there is any 'rate-limit' or 'rate-limit-by-key' policy implemented at any scope.

- If you calculate the effective policy, you should notice an access restriction policy (rate-limit-by-key) implemented at Global scope, i.e under 'Inbound processing' in 'All APIs' option.

  XML

  ```
  <inbound>
      <choose>
          <when
  condition="@(context.Operation.Name.Equals("GetComments"))">
              <rate-limit-by-key calls="1" renewal-period="10" increment-
  condition="@(context.Response.StatusCode == 200)" counter-
  key="@(context.Request.IpAddress)" />
          </when>
      </choose>
  </inbound>
  ```

Read more about ip-filter and rate-limit-by-key policies in APIM.

Contact us for help

If you have questions or need help, create a support request, or ask Azure community support. You can also submit product feedback to Azure feedback community.

Feedback

Was this page helpful? Yes No

SOAP-based API is returning 404 and 500 HTTP status codes

Article • 04/12/2024

Referring to the article on Azure API Management Troubleshooting Series, this is the second scenario of the lab. Make sure you have followed the lab setup instructions as per this, to recreate the problem.

Original product version: API Management Service
Original KB number: 4464934

Symptoms

The Calculator API can perform four operations - **Add**, **Subtract**, **Multiply**, and **Divide** based upon two input parameters **intA** and **intB**. Name of the operations is self-explanatory to what function they perform. It's an ASMX service (http://www.dneonline.com/calculator.asmx) following SOAP 1.1 protocol so the input parameters are passed in the soap envelope body section. Add and Subtract operations are working fine as expected, but you are encountering **HTTP 404** while invoking Multiply operation and **HTTP 500** while invoking Divide operation.

Expected output of Multiply operation should be something like below:

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soap:Body>
      <MultiplyResponse xmlns="http://tempuri.org/">
         <MultiplyResult>int</MultiplyResult>
      </MultiplyResponse>
   </soap:Body>
</soap:Envelope>
```

Expected output of Divide operation should be something like below:

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
```

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soap:Body>
      <DivideResponse xmlns="http://tempuri.org/">
         <DivideResult>int</DivideResult>
      </DivideResponse>
   </soap:Body>
</soap:Envelope>
```

Troubleshooting steps

You need to understand who is throwing these HTTP 404 and 500 responses, APIM, or the backend SOAP API. The best way to get that answer is to collect APIM inspector trace to inspect request and response.

- Multiply operation throwing HTTP - 404 (Not Found) status code indicates that the origin server did not find a current representation for the target resource or is not willing to disclose that one exists.

- If you examine the backend section of APIM inspector trace, the same observation is evident from the message as well:

```
{
  "backend": [
    {
      "source": "configuration",
      "timestamp": "2018-07-29T12:30:08.3500317Z",
      "elapsed": "00:00:00.7276962",
      "data": {
        "message": "Unable to identify Api or Operation for this
request. Responding to the caller with 404 Resource Not Found."
      }
    }
  ]
}
```

- Hence you should first examine the request url and headers sent from APIM to the backend API from the **Test** tab and compare it with the sample of SOAP request for Multiply operation - http://www.dneonline.com/calculator.asmx .

  Request headers from APIM inspector trace look something like below:

```
{
  "data": {
```

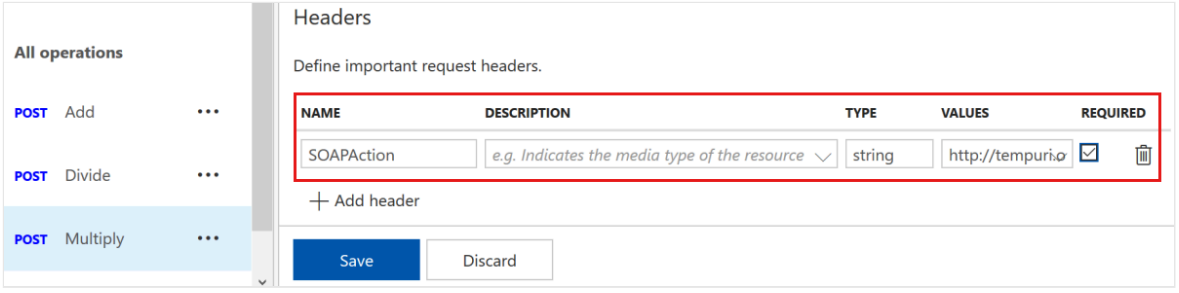
```
    "request": {
      "method": "POST",
      "url": "https://pratyay.azure-api.net/calc",
      "headers": [
        {
          "name": "Ocp-Apim-Subscription-Key",
          "value": "34ae22db7f2c4c5da7b74a55adf03223"
        },
        {
          "name": "X-Forwarded-For",
          "value": "223.226.79.35"
        },
        {
          "name": "Cache-Control",
          "value": "no-cache"
        },
        {
          "name": "Connection",
          "value": "Keep-Alive"
        },
        {
          "name": "Content-Length",
          "value": "292"
        },
        {
          "name": "Content-Type",
          "value": "application/soap+xml;
action=http://tempuri.org/Multiply"
        },
        {
          "name": "Accept",
          "value": "*/*"
        },
        {
          "name": "Accept-Encoding",
          "value": "gzip,deflate,br"
        },
        {
          "name": "Accept-Language",
          "value": "en-US,en;q=0.5"
        },
        {
          "name": "Host",
          "value": "pratyay.azure-api.net"
        },
        {
          "name": "Referer",
          "value":
"https://apimanagement.hosting.portal.azure.net/apimanagement/Content/1
.0.385.3/apimap/apimap-apis/index.html?
locale=en&trustedAuthority=https://ms.portal.azure.com"
        }
      ]
    }
```

```
  }
}
```

- As per the backend ASMX service definition you would notice SOAP 1.1 request needs a request header **SOAPAction that is missing in the request sent from APIM.

```
XML

Host: www.dneonline.com
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/Multiply"
```

- Adding **SOAPAction** header with the value http://tempuri.org/Multiply will resolve the problem. You can add the request header under the **Frontend** definition of the Multiply operation and set the value as a default one under **Headers** tab so that you don't have to send that header every time on each request.



- Divide operation throwing HTTP 500 (Internal Server Error) status code indicates that the server encountered an unexpected condition that prevented it from fulfilling the request.

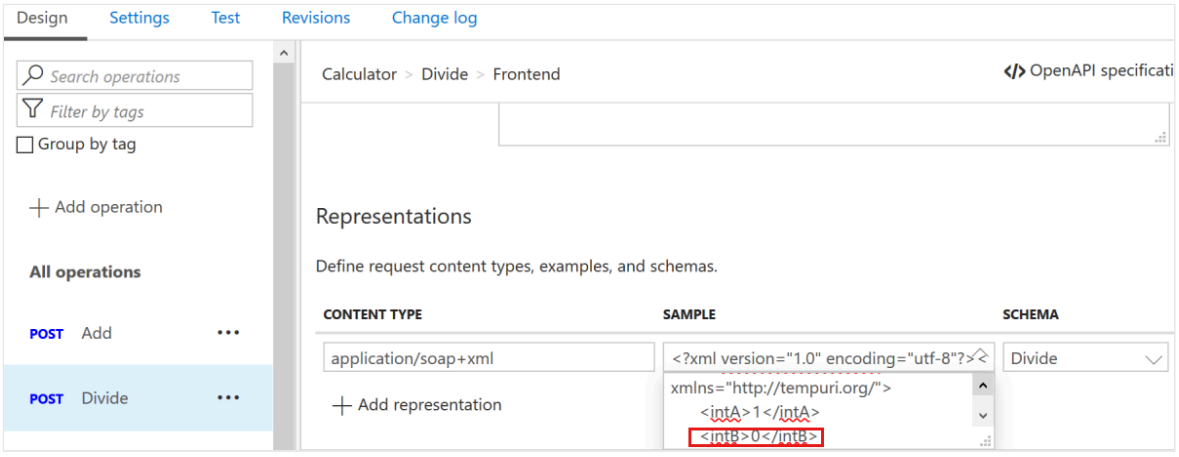
- In other words, backend service is not able to process your request body sent from APIM. You can examine the request body sent from APIM.

- Upon checking the SOAP body, you would notice that denominator (**intB**) is set as zero, leading to an unhandled exception, hence causing HTTP 500 (Internal Server Error).

```
XML

POST calc HTTP/1.1

Host: pratyay.azure-api.net
SOAPAction: http://tempuri.org/Divide
Cache-Control: no-cache
Ocp-Apim-Trace: true
```

```
Content-Type: application/soap+xml; action=http://tempuri.org/Divide
Ocp-Apim-Subscription-Key: ********************************

<?xml version="1.0" encoding="utf-8"?>
<Envelope xmlns="http://www.w3.org/2003/05/soap-envelope">
  <Body>
    <Divide xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://tempuri.org/">
      <intA>1</intA>
      <intB>0</intB>
    </Divide>
  </Body>
</Envelope>
```

- If you check the request content representation from the **Request** tab present in **Frontend** definition of the Divide operation, you would notice that **intB** value is set to zero. You need to change the value of **intB** to a non-zero value and it should resolve the issue.



Contact us for help

If you have questions or need help, create a support request, or ask Azure community support. You can also submit product feedback to Azure feedback community.

Feedback

Was this page helpful? Yes No

Unauthorized errors (401) while invoking APIs

Article • 04/12/2024

Referring to the article on Azure API Management Troubleshooting Series, this is the third scenario of the lab. Make sure you have followed the lab setup instructions as per this, to recreate the problem.

Original product version: API Management Service
Original KB number: 4464930

> **Note**
>
> Was this article helpful? Your input is important to us. Please use the **Feedback** button on this page to let us know how well this article worked for you or how we can improve it.

Symptoms

The **Echo API** suddenly started throwing diverse types of **HTTP 401 - Unauthorized** errors while invoking the operations under it. **Create resource** and **Retrieve resource** operations are showing this error message:

```
{
"statusCode": 401,
"message": "Access denied due to invalid subscription key. Make sure to provide a valid key for an active subscription."
}
```

Whereas rest of the operations are showing

```
{
"statusCode": 401,
"message": "Access denied due to missing subscription key. Make sure to include subscription key when making requests to an API."
}
```

The expected HTTP response code for all the operations is 200, however the response body will vary as the backend API always echoes whatever you send as a request body in addition to headers.

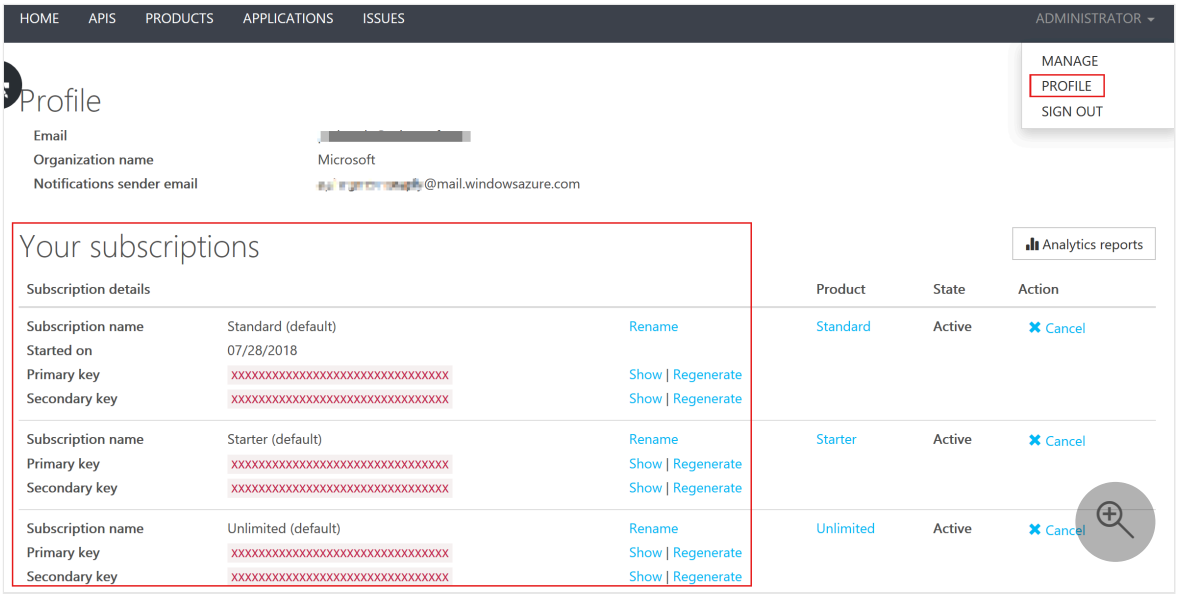
Troubleshooting steps

- To get access to the API, developers must first subscribe to a product. When they subscribe, they get a subscription key that is sent as a part of request header that is good for any API in that product. **Ocp-Apim-Subscription-Key** is the request header sent for the subscription key of the product that is associated with this API. The key is filled in automatically.

- Regarding error **Access denied due to invalid subscription key. Make sure to provide a valid key for an active subscription**, it's clear that you are sending a wrong value of **Ocp-Apim-Subscription-Key** request header while invoking **Create resource** and **Retrieve resource** operations.

- You can check your subscription key for a particular product from APIM Developer portal by navigating to **Profile** page after sign-in as shown below.

- Select the **Show** button to see the subscription keys for respective products you have subscribed to.



- If you check the headers being sent from **Test** tab, you notice that the value of **Ocp-Apim-Subscription-Key** request header is wrong. You might be wondering how come that is possible, because APIM automatically fills this request header with the right subscription key.

- Let's check the Frontend definition of **Create resource** and **Retrieve resource** operations under **Design** tab. Upon careful inspection, you would notice that these

operations got a wrong hard-coded value of **Ocp-Apim-Subscription-Key** request header added under **Headers** tab.

- You can remove it, this should resolve the invalid subscription key problem, but still you would get missing subscription key error.

  You may get the following error message:

  > HTTP/1.1 401 Unauthorized
  >
  > Content-Length: 152
  > Content-Type: application/json
  > Date: Sun, 29 Jul 2018 14:29:50 GMT
  > Vary: Origin WWW-Authenticate: AzureApiManagementKey realm="`https://pratyay.azure-api.net/echo`",name="Ocp-Apim-Subscription-Key",type="header" {
  > "statusCode": 401,
  > "message": "Access denied due to missing subscription key. Make sure to include subscription key when making requests to an API." }

- Go to the Echo APIsettings and check if it is associated with any of the available products. If not, then you must associate this API with a product so that you get a subscription key.

  Developers must first subscribe to a product to get access to the API. When they subscribe, they get a subscription key that is good for any API in that product. If you created the APIM instance, you are an administrator already, so you are subscribed to every product by default.

Contact us for help

If you have questions or need help, create a support request, or ask Azure community support. You can also submit product feedback to Azure feedback community.

Feedback

Troubleshoot performance issues in API calls

Article • 04/12/2024

Referring to the blog on Azure API Management Troubleshooting Series, this is the fourth scenario of the lab. Make sure you have followed the lab setup instructions as per this, to recreate the problem.

Original product version: API Management Service
Original KB number: 4464929

Symptoms

The API **ProductStore** in APIM communicates with the backend endpoint (`https://productstoreapp.azurewebsites.net`) to easily create, read, update, and delete records as and when required. However, you may face some performance issues and exceptions while invoking the API operations listed below. For ease in testing, keep only three products having IDs ranging from 1 to 3.

- One of the API functions **Products_GetAllProducts** is taking 5 seconds to return the results, whereas the expected response time is less than a second.
- While deleting a product having any of the above mentioned IDs (1 to 3), you are getting **HTTP 500 - Internal Server Error** with the below message by calling **Products_DeleteProduct** operation.

  ```
  {
  "Message": "An error has occurred."
  }
  ```

- **Products_PutProduct** operation that updates a product is getting throttled unexpectedly, throwing **HTTP 429 - Too many requests** with the below error message irrespective of product ID and request body, which you send in the request. For example, if the customer updates the product price of "Tomato Soup" having product ID = 1 with the below Json body he gets HTTP 429 status code.

  ```
  Template parameter ID : 1
    Request Body: {"Name": "Tomato soup","Category": "Groceries","Price": 2.45}
    Response Body:
  {
  ```

> Rate limit is exceeded. Try again after some time.
> }

Troubleshooting steps

- While troubleshooting performance issues, the best way fault isolation technique is capturing [APIM inspector trace that shows time taken in each section (Inbound / Backend / Outbound).

- If you analyze the API Inspector trace for the first issue you would notice that Backend section is taking most of the time (approx. 5 seconds), which means there is some slowness or long running operation is taking place at the backend.

> "source": "forward-request",
> "timestamp": "2018-07-29T16:16:46.6615081Z",
> **"elapsed": "00:00:05.5844430",**"data": {
> "response": {
> "status": {
> "code": 200,
> "reason": "OK"
> }

- Once you have isolated that the slowness is at the backend, you need to investigate the backend application code of the Web API application. For scenarios where you don't have access to the backend, you can implement caching at APIM level like below. Read about how you can implement caching policies to improve performance in Azure API Management.

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<policies>
   <inbound>
      <base />
      <cache-lookup vary-by-developer="true" vary-by-developer-
groups="true" must-revalidate="true" downstream-caching-type="public"
/>
   </inbound>
   <backend>
      <base />
   </backend>
   <outbound>
      <base />
      <cache-store duration="60" />
   </outbound>
```

```
    <on-error>
        <base />
    </on-error>
</policies>
```

- For the second issue (**HTTP 500 - Internal Server Error**), follow the same procedure of analyzing the APIM inspector trace and we should see HTTP 500 status code under 'forward-request' response attribute.

- This means backend API returned HTTP 500 due to some unhandled exception occurred at the backend code, there is no issue at APIM level.

> forward-request (841.060 ms)
> {
> "response": {
> "status": {
> "code": 500,
> "reason": "Internal Server Error"
> }

- For the third issue (**HTTP 429 - Too many requests**) it looks like you are hitting API call rate limit. Probably you can check if there is any 'rate-limit' or 'rate-limit-by-key' policy implemented at operation level.

- If you cannot find any such policies at operation level, click on the **Calculate effective policy** button, which will show all the inherited policies from various levels, like you may have some policies at product level that can cause this issue.

- Here you should notice that some policies are implemented at API level which not really limiting the API call rate but mimics its action by returning a customized response back to the client using 'return-response' and 'set-status' policies at outbound section.

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<outbound>
    <!--base: Begin Api scope-->
    <return-response>
        <set-status code="429" reason="Too many requests" />
        <set-body><![CDATA[{

Rate limit is exceeded. Try again after some time.

}]]></set-body>
    </return-response>
```

```
    <!--base: End Api scope-->
</outbound>
```

Contact us for help

If you have questions or need help, create a support request, or ask Azure community support. You can also submit product feedback to Azure feedback community.

Feedback

Was this page helpful? Yes No

Azure API Management Troubleshooting Series

Article • 04/12/2024

Original product version: Cloud and Online Services
Original KB number: 4466541

Summary

These troubleshooting series articles provide some of the scenarios, which would be helpful for you having intermediate experience in Azure API Management (APIM). To troubleshoot these issues, you need to have some basic understanding of how APIM works along with the key concepts and terminology associated with it. You can refer this link if you are at beginner or Rookie level with regards to APIM.

Here are the prerequisites or instructions that you need to follow to set up the lab.

This troubleshooting series will not only help you to get accustomed with various APIM policies but also give you an idea how to leverage APIM inspector trace to debug issues related to failed API calls, exceptions thrown from policies, performance problems, etc.

Troubleshooting scenarios:

- Scenario 1 - API is returning blank response.
- Scenario 2 - SOAP based API is returning 404 and 500 HTTP status codes.
- Scenario 3 - Unauthorized errors (401) while invoking APIs.
- Scenario 4 - Performance hit in API calls.
- Scenario 5 - Request throttling problems and HTTP 403 - Forbidden issues.

Contact us for help

If you have questions or need help, create a support request, or ask Azure community support. You can also submit product feedback to Azure feedback community.

Feedback