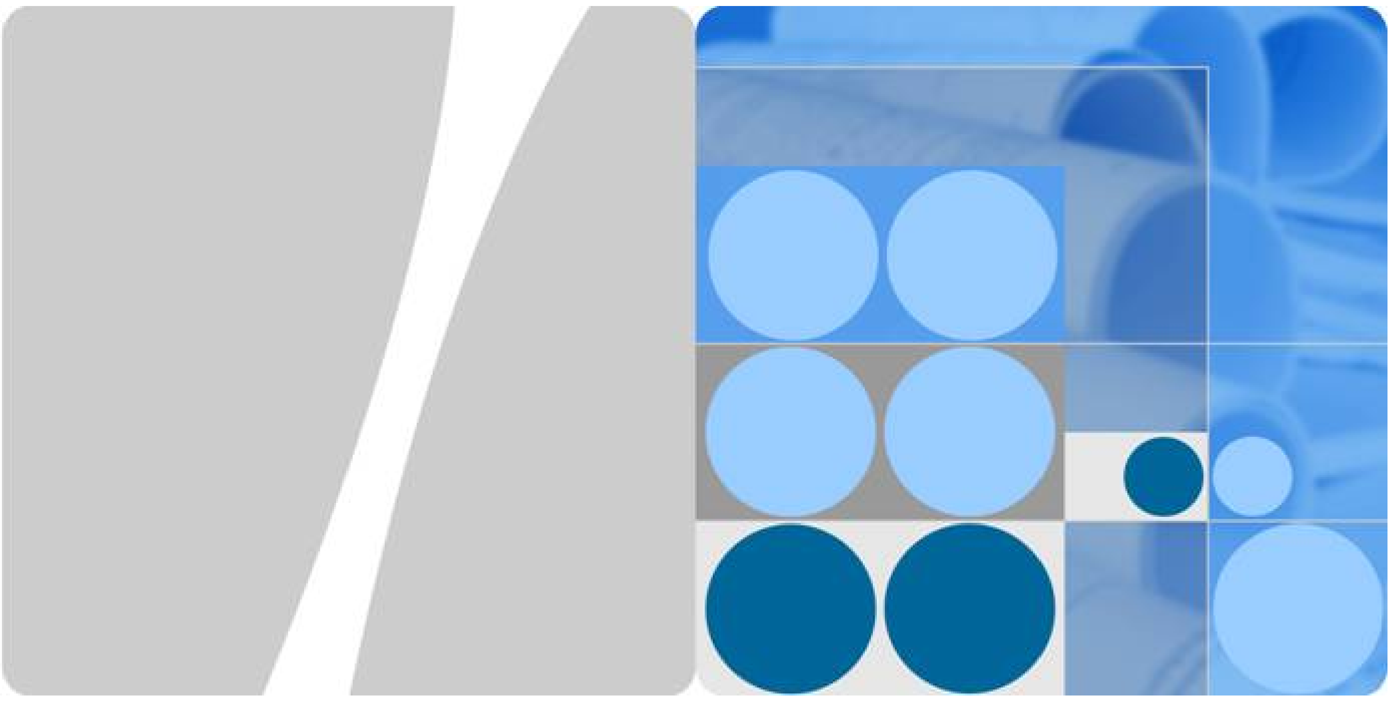


Application Performance Management

API Reference

Date 2021-10-11

Contents

1 Before You Start

1.1 Overview

Welcome to use Application Performance Management (APM). APM monitors and manages the performance of cloud applications in real time. APM provides performance analysis of distributed applications, helping O&M personnel quickly locate and resolve faults and performance bottlenecks.

This document describes how to use APIs to perform operations on APM. For details, see **1.1 Overview**.

1.2 API Calling

Application Performance Management (APM) supports Representational State Transfer (REST) APIs, allowing you to call APIs using HTTPS. For details about API calling, see **3 Calling APIs**.

1.3 Endpoints

An endpoint is the request address for calling an API. Endpoints vary depending on services and regions. For the endpoints of all services, see **Regions and Endpoints**.

1.4 Concepts

- Account

  An account is created upon successful registration with the cloud. The account has full access permissions for all of its cloud services and resources. It can be used to reset user passwords and grant user permissions. The account is a payment entity and should not be used directly to perform routine management. For security purposes, create Identity and Access Management (IAM) users and grant them permissions for routine management.

- IAM user

  An IAM user is created using an account to use cloud services. Each IAM user has its own identity credentials (password and access keys).

  An IAM user can view the account ID and user ID on the **My Credentials** page of the console. The account name, username, and password will be required for API authentication.

- Region

  Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.

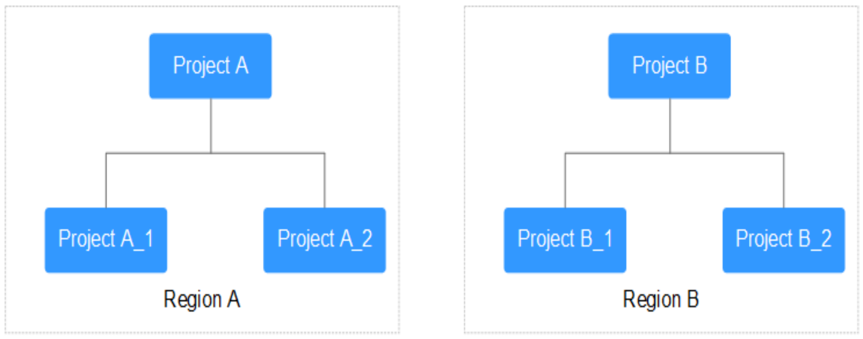
- AZ

  An AZ contains one or more physical data centers. Each AZ has independent cooling, fire extinguishing, moisture-proof, and electricity facilities. Within an AZ, computing, network, storage, and other resources are logically divided into multiple clusters. AZs within a region are interconnected using high-speed optical fibers to support cross-AZ high-availability systems.

- Project

  Projects group and isolate resources (including compute, storage, and network resources) across physical regions. A default project is provided for each region, and subprojects can be created under each default project. Users can be granted permissions to access all resources in a specific project. For more refined access control, create subprojects under a project and purchase resources in the subprojects. Users can then be assigned permissions to access only specific resources in the subprojects.

Figure 1-1 Project isolating model



2 API Overview

Application Performance Management (APM) provides open APIs, helping you quickly and cost-effectively implement application O&M.

API	Description
4.1 Querying an Application List	Query an application list.
4.2 Querying a Service List	Query a service list.
4.3 Querying a Service Instance List	Query an instance list of a specified service.
4.4 Querying a Service Transaction List	Query a transaction list of a specified service.
4.5 Querying Tracing Data	Query tracing data based on filter criteria.
4.6 Query Tracing Details	Query tracing details based on trace IDs.

3 Calling APIs

3.1 Making an API Request

This section describes the structure of a Representational State Transfer (REST) API request, and uses the Identity and Access Management (IAM) API for **obtaining a user token** as an example to demonstrate how to call an API. The obtained token can then be used to authenticate the calling of other APIs.

Request URI

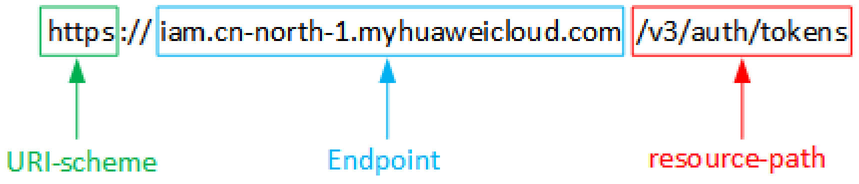
A request URI is in the following format:

{URI-scheme} :// {Endpoint} / {resource-path} ? {query-string}

Although a request URI is included in the request header, most programming languages or frameworks require the request URI to be transmitted separately.

- **URI-scheme**: Protocol used to transmit requests. All APIs use HTTPS.
- **Endpoint**: Domain name or IP address of the server bearing the REST service endpoint. Obtain the value from **Regions and Endpoints**.
- **resource-path**: Access path of an API for performing a specified operation. Obtain the path from the URI of an API. For example, the **resource-path** of the API used to obtain a user token is **/v3/auth/tokens**.
- **query-string**: Query parameter, which is optional. Ensure that a question mark (?) is included before each query parameter that is in the format of "Parameter name=Parameter value". For example, **? limit=10** indicates that a maximum of 10 data records will be displayed.

Figure 3-1 Example URI



NOTE

To simplify the URI display in this document, each API is provided only with a **resource-path** and a request method. The **URI-scheme** of all APIs is **HTTPS**, and the endpoints of all APIs in the same region are identical.

Request Methods

The HTTP protocol defines the following request methods that can be used to send a request to the server:

- **GET**: requests the server to return specified resources.
- **PUT**: requests the server to update specified resources.
- **POST**: requests the server to add resources or perform special operations.
- **DELETE**: requests the server to delete specified resources, for example, an object.
- **HEAD**: requests a server resource header.
- **PATCH**: requests the server to update partial content of a specified resource. If the resource does not exist, a new resource will be created.

For example, in the case of the API used to **obtain a user token**, the request method is POST. The request is as follows:

```
POST https://{Endpoint}/v3/auth/tokens
```

Request Header

You can also add additional header fields to a request, such as the fields required by a specified URI or HTTP method. For example, to request for the authentication information, add **Content-Type**, which specifies the request body type.

Common request header fields are as follows:

- **Content-Type**: specifies the request body type or format. This field is mandatory and its default value is **application/json**. Other values of this field will be provided for specific APIs if any.
- **X-Auth-Token**: specifies a user token only for token-based API authentication. The user token is a response to the API used to **obtain a user token**. This API is the only one that does not require authentication.

NOTE

In addition to supporting token-based authentication, APIs also support authentication using access key ID/secret access key (AK/SK). During AK/SK-based authentication, an SDK is used to sign the request, and the **Authorization** (signature information) and **X-Sdk-Date** (time when the request is sent) header fields are automatically added to the request.

For more details, see **AK/SK-based Authentication**.

The API used to **obtain a user token** does not require authentication. Therefore, only the **Content-Type** field needs to be added to requests for calling the API. An example of such requests is as follows:

```
POST https://{Endpoint}/v3/auth/tokens
Content-Type: application/json
```

Request Body

The body of a request is often sent in a structured format as specified in the **Content-Type** header field. The request body transfers content except the request header.

The request body varies between APIs. Some APIs do not require the request body, such as the APIs requested using the GET and DELETE methods.

In the case of the API used to **obtain a user token**, the request parameters and parameter description can be obtained from the API request. The following provides an example request with a body included. Replace *username*, *domainname*, ******** (login password), and xxxxxxxxxxxxxxxxxx (project ID) with the actual values. To learn how to obtain a project ID, see **6.3 Obtaining a Project ID**.

NOTE

The **scope** parameter specifies where a token takes effect. You can set **scope** to an account or a project under an account. In the following example, the token takes effect only for the resources in a specified project. For more information about this API, see **Obtaining a User Token**.

```
POST https://{Endpoint}/v3/auth/tokens
Content-Type: application/json
{
    "auth": {
        "identity": {
            "methods": [
                "password"
            ],
            "password": {
                "user": {
                    "name": "username",
                    "password": "********",
                    "domain": {
                        "name": "domainname"
                    }
                }
            }
        },
        "scope": {
            "project": {
                "id": "xxxxxxxxxxxxxxxxxx"
            }
        }
    }
}
```

If all data required for the API request is available, you can send the request to call the API through **curl**, **Postman**, or coding. In the response to the API used to **obtain a user token**, **x-subject-token** is the desired user token. This token can then be used to authenticate the calling of other APIs.

3.2 Authentication

Requests for calling an API can be authenticated using either of the following methods:

- Token-based authentication: Requests are authenticated using a token.
- AK/SK-based authentication: Requests are authenticated by encrypting the request body using an AK/SK pair.

Token-based Authentication

NOTE

The validity period of a token is 24 hours. When using a token for authentication, cache it to prevent frequently calling the Identity and Access Management (IAM) API used to obtain a user token.

A token specifies temporary permissions in a computer system. During API authentication using a token, the token is added to requests to get permissions for calling the API.

In **3.1 Making an API Request**, the process of calling the API used to **obtain a user token** is described. After a token is obtained, the **X-Auth-Token** header field must be added to requests to specify the token when calling other APIs. For example, if the token is **ABCDEFJ....**, **X-Auth-Token: ABCDEFJ....** can be added to a request as follows:

```
GET https://{Endpoint}/v3/auth/projects
Content-Type: application/json
X-Auth-Token: ABCDEFJ....
```

AK/SK-based Authentication

NOTE

AK/SK-based authentication supports API requests with a body not larger than 12 MB. For API requests with a larger body, token-based authentication is recommended.

In AK/SK-based authentication, AK/SK is used to sign requests and the signature is then added to the requests for authentication.

- AK: access key ID, which is a unique identifier used in conjunction with a secret access key to sign requests cryptographically.
- SK: secret access key used in conjunction with an AK to sign requests cryptographically. It identifies a request sender and prevents the request from being modified.

In AK/SK-based authentication, you can use an AK/SK to sign requests based on the signature algorithm or use the signing SDK to sign requests.

3.3 Response

Status Code

After sending a request, you will receive a response, including the status code, response header, and response body.

A status code is a group of digits, ranging from 1xx to 5xx. It indicates the status of a request. For more information, see **6.1 Status Code**.

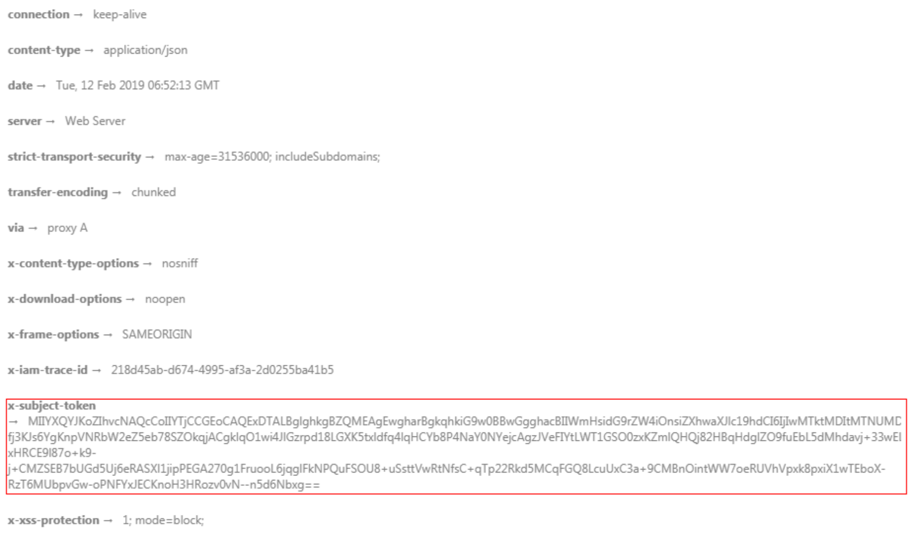
For example, if status code **201** is returned for calling the API used to **obtain a user token**, the request is successful.

Response Header

Similar to a request, a response also has a header, for example, **Content-Type**.

Figure 3-2 shows the response header for the API used to **obtain a user token**. The **x-subject-token** header field is the desired user token. This token can then be used to authenticate the calling of other APIs.

Figure 3-2 Header fields of the response to the request for obtaining a user token



```
connection → keep-alive

content-type → application/json

date → Tue, 12 Feb 2019 06:52:13 GMT

server → Web Server

strict-transport-security → max-age=31536000; includeSubdomains;

transfer-encoding → chunked

via → proxy A

x-content-type-options → nosniff

x-download-options → noopen

x-frame-options → SAMEORIGIN

x-iam-trace-id → 218d45ab-d674-4995-af3a-2d0255ba41b5

x-subject-token
→ MIIYXQYJKoZIhvcNAQcCoIIYTjCCGEoCAQExDTALBglghkgBZQMEAgEwgharBgkqhkiG9w0BBwGgghacBIIWmHsidG9rZW4iOnsiZXhwaXJlc19hdCI6IjIwMTktMDItMTNUMD
fj3KJs6YgKnpVNRbW2eZ5eb78SZOkqjACgklqO1wi4JIGzrpd18LGXK5txldfq4lqHCYb8P4NaY0NYejcAgzJVeFIYtLWT1GSO0zxKZmIQHQj82HBqHdgIZO9fuEbL5dMhdavj+33wEl
xHRCE9I87o+k9-
j+CMZSEB7bUGd5Uj6eRASXl1jipPEGA270g1FruooL6jqglFkNPQuFSOU8+uSsttVwRtNfsC+qTp22Rkd5MCqFGQ8LcuUxC3a+9CMBnOintWW7oeRUVhVpxk8pxiX1wTEboX-
RzT6MUbpvGw-oPNFYxJECKnoH3HRozv0vN--n5d6Nbxg==

x-xss-protection → 1; mode=block;
```

Response Body

The body of a response is often returned in structured format as specified in the **Content-Type** header field. The response body transfers content except the response header.

The following is part of the response body for the API used to **obtain a user token**.

```
{
    "token": {
        "expires_at": "2019-02-13T06:52:13.855000Z",
        "methods": [
            "password"
        ],
        "catalog": [
            {
                "endpoints": [
                    {
                        "region_id": "xxx",
......
```

If an error occurs during API calling, an error code and a message will be displayed. The following shows an error response body.

```
{
    "error_msg": "The format of message is error",
    "error_code": "AS.0001"
}
```

In the response body, **error_code** is an error code, and **error_msg** provides information about the error.

4 APIs

4.1 Querying an Application List

Function

This API is used to query an application list.

URI

GET /v1/{projectId}/atps/monitorgroups

Request

Path parameters

Table 4-1 describes the path parameter.

Table 4-1 Path parameter

Parameter	Type	Description
projectId	String	Project ID.

Example request

```
/v1/0/atps/monitorgroups
```

Response

Response parameters

Table 4-2 describes the response parameters.

Table 4-2 Response parameters

Parameter	Type	Description
errorCode	String	Error code. SVCSTG.ATPS.2000: Query succeeded.
errorMessage	String	Error message.
responseInfo	List (string)	Application ID list.

Example response

```
{
  "errorCode": "SVCSTG.ATPS.2000",
  "errorMessage":null,
  "responseInfo": ["11d5c9b83c1b2e04579fa5a34d191bb5"]
}
```

Status Code

- Success response

  Table 4-3 describes the status code.

Table 4-3 Status code

Status Code	Description
200	The request has succeeded.

4.2 Querying a Service List

Function

This API is used to query a service list.

URI

GET /v1/{projectId}/ats/applications

Request

Path parameters

Table 4-4 describes the path parameter.

Table 4-4 Path parameter

Parameter	Type	Description
projectId	String	Project ID.

Request parameters

Table 4-5 describes the request parameter.

Table 4-5 Request parameter

Parameter	Mandatory	Type	Description
monitorGroup	Yes	String	Application ID. See the **responseInfo** field in the response of **4.1 Querying an Application List**.

Example request

```
/v1/0/ats/applications?monitorGroup=11d5c9b83c1b2e04579fa5a34d191bb5
```

Response

Response parameters

Table 4-6 describes the response parameters.

Table 4-6 Response parameters

Parameter	Type	Description
errorCode	String	Error code. SVCSTG.ATS.2000: Query succeeded. SVCSTG.ATS.400101: Parameter verification failed. SVCSTG.ATS.200103: No service information found.
errorMessage	String	Error message.
responseInfo	List (string)	Service name list (including port information). See the following example response.

Example response

```
{
  "errorCode": "SVCSTG.ATS.2000",
  "errorMessage":null,
  "responseInfo": ["ams-calc:8080","ams-metric:8080"]
}
```

Status Code

- Success response

  Table 4-7 describes the status code.

Table 4-7 Status code

Status Code	Description
200	The request has succeeded.

4.3 Querying a Service Instance List

Function

This API is used to query an instance list of a specified service.

URI

GET /v1/{projectId}/ats/applications/{application}/instances

Request

Path parameters

Table 4-8 describes the path parameters.

Table 4-8 Path parameters

Parameter	Type	Description
projectId	String	Project ID.
application	String	Service name (including port information). See the **responseInfo** field in the response of 4.2 Querying a Service List.

Request parameters

Table 4-9 describes the request parameter.

Table 4-9 Request parameter

Parameter	Mandatory	Type	Description
monitorGroup	Yes	String	Application ID. See the **responseInfo** field in the response of **4.1 Querying an Application List**.

Example request

```
/v1/0/ats/applications/ams-metric:8080/instances?monitorGoup=11d5c9b83c1b2e04579fa5a34d191bb5
```

Response

Response parameters

Table 4-10 describes the response parameters.

Table 4-10 Response parameters

Parameter	Type	Description
errorCode	String	Error code. SVCSTG.ATS.2000: Query succeeded. SVCSTG.ATS.400101: Parameter verification failed. SVCSTG.ATS.200103: No instance information found.
errorMessage	String	Error message.
responseInfo	List (string)	Instance ID list of a specified service.

Example response

```
{
  "errorCode": "SVCSTG.ATS.2000",
  "errorMessage":null,
  "responseInfo": [ "d056db8ebf2350c118ea7ace383ac5dd"]
}
```

Status Code

- Success response

  Table 4-11 describes the status code.

Table 4-11 Status code

Status Code	Description
200	The request has succeeded.

4.4 Querying a Service Transaction List

Function

This API is used to query a transaction list of a specified service.

URI

GET /v1/{projectId}/ats/applications/{application}/transactions

Request

Path parameters

Table 4-12 describes the path parameters.

Table 4-12 Path parameters

Parameter	Type	Description
projectId	String	Project ID.
application	String	Service name (including port information). See the **responseInfo** field in the response of **4.2 Querying a Service List**.

Request parameters

Table 4-13 describes the request parameter.

Table 4-13 Request parameter

Parameter	Mandatory	Type	Description
monitorGroup	Yes	String	Application ID. See the **responseInfo** field in the response of **4.1 Querying an Application List**.

Example request

```
/v1/0/ats/applications/ams-metric:8080/transactions?monitorGroup=11d5c9b83c1b2e04579fa5a34d191bb5
```

Response

Response parameters

Table 4-14 describes the response parameters.

Table 4-14 Response parameters

Parameter	Type	Description
errorCode	String	Error code. SVCSTG.ATS.2000: Query succeeded. SVCSTG.ATS.400101: Parameter verification failed. SVCSTG.ATS.200103: No transaction information found.
errorMessage	String	Error message.
responseInfo	List (string)	Transaction list of a specified service.

Example response

```
{
  "errorCode": "SVCSTG.ATS.2000",
  "errorMessage":null,
  "responseInfo": [
    "/amsalarm/v1/alarm/{projectId}"
  ]
}
```

Status Code

- Success response

  Table 4-15 describes the status code.

  Table 4-15 Status code

Status Code	Description
200	The request has succeeded.

4.5 Querying Tracing Data

Function

This API is used to query tracing data by filter criteria.

URI

GET /v1/{projectId}/ats/traces

Request

Path parameters

Table 4-16 describes the path parameter.

Table 4-16 Path parameter

Parameter	Type	Description
projectId	String	Project ID.

Request parameters

Table 4-17 describes the request parameters.

Table 4-17 Request parameters

Parameter	Mandatory	Type	Value Range	Description
startTime	Yes	Integer	<endTime	Start time for querying tracing data (unit: ms).
endTime	Yes	Integer	>startTime	End time for querying tracing data (unit: ms).
application	Yes	String	See **4.2 Querying a Service List**.	Service name, which must consist of lowercase letters. Example: test-service.
monitorGroup	No	String	See **4.1 Querying an Application List**.	Application name.

Parameter	Mandatory	Type	Value Range	Description
instance	No	String	See **4.3 Querying a Service Instance List**.	Instance name, which must consist of lowercase letters. Example: test-service-4195149926-0fvhn.
transaction	No	String	See **4.4 Querying a Service Transaction List**.	Transaction name. Example: GET_/rest/healthz/*.
limit	No	Integer	(0, 1000]	Number of data records returned each time. Default value: 20. Maximum value: 1000.
duration	No	Integer	Integer (≥ 0)	Minimum call duration (unit: ms). Default value: 0.
status	No	Integer	1: Only the transactions that fail to be executed are queried.	Transaction status. By default, all transactions are queried. If the value is **1**, only the transactions that fail to be executed are queried.

Example request

```
/v1/0/ats/traces?
startTime=1506214200000&endTime=1506214428000&application=datamgmtservice&monitorGroup=apm&l
imit=1
```

Response

Response parameters

Table 4-18 describes the response parameters.

Table 4-18 Response parameters

Parameter	Type	Description
errorCode	String	Error code. SVCSTG.ATS.2000: Query succeeded. SVCSTG.ATS.400101: Parameter verification failed. SVCSTG.ATS.200103: No tracing data found.
errorMessage	String	Error message.
responseInfo	Result	Tracing query result.

Table 4-19 result parameters

Parameter	Type	Description
count	Integer	Tracing quantity.
traceChains	List<TraceChainBase>	Tracing data set.

Table 4-20 TraceChainBase parameters

Parameter	Type	Description
traceId	String	Trace ID, which is globally unique.
type	String	Service type.
status	Integer	Call response status.
duration	Integer	Service call duration (unit: µs).
application	String	Service name.
instance	String	Instance name.
transaction	String	Service call API/name.
startTime	Integer	Start time for calling a service (unit: µs).
endTime	Integer	End time for calling a service (unit: µs).
address	String	IPv4 address of the client.

Example response

```
{
  "errorCode": "SVCSTG.ATS.2000",
  "errorMessage":null,
  "responseInfo": {
    "count": 1,
    "traceChains": [
      {
        "traceId": "000000004fa102d1",
        "type": "TOMCAT_METHOD",
        "status": 0,
        "duration": 10000,
        "application": "datamgmtservice",
        "instance": "datamgmtservice-4267750592-2ngmz",
        "transaction": "/rest/plat/sysmgr/v1/sysagent/alarm/report",
        "startTime": 1506214214095000,
        "endTime": 1506214214105000,
        "address": "192.168.0.1"
      }
    ]
  }
}
```

Status Code

- Success response

  Table 4-21 describes the status code.

Table 4-21 Status code

Status Code	Description
200	The request has succeeded.

4.6 Query Tracing Details

Function

This API is used to query tracing details based on trace IDs.

URI

GET /v1/{projectId}/ats/spans

Request

Path parameters

Table 4-22 describes the path parameter.

Table 4-22 Path parameter

Parameter	Type	Description
projectId	String	Project ID.

Request parameters

Table 4-23 describes the request parameter.

Table 4-23 Request parameter

Parameter	Mandatory	Type	Value Range	Description
traceId	Yes	String	Obtained from tracing data.	Trace ID.

Example request

```
/v1/0/ats/spans?traceId=0000000027046b00
```

Response

Response parameters

Table 4-24 describes the response parameters.

Table 4-24 Response parameters

Parameter	Type	Description
errorCode	String	Error code. SVCSTG.ATS.2000: Query succeeded. SVCSTG.ATS.400101: Parameter verification failed. SVCSTG.ATS.200103: No tracing data found.
errorMessage	String	Error message.
responseInfo	List (string). See **Table 4-25**.	Query result.

Table 4-25 spans parameters

Parameter	Type	Description
traceId	String	Trace ID, which is globally unique.
name	String	Service name: Instance name: Transaction name
id	String	Span ID
parentId	String	Upper-level span ID.
timestamp	Integer	Call start time (unit: μs).
duration	Integer	Span call duration (unit: μs).
annotations	List (string). See **Table 4-26**.	Service information about the client or server.
binaryAnnotations	List (string). See **Table 4-27**.	Extended information.

Table 4-26 Annotation parameters

Parameter	Type	Description
timestamp	Integer	Current system time when an event occurs (unit: μs).
endpoint	See **Table 4-28**.	(Optional) Service information about the client.
value	String	Event type. Value: CS, SR, SS, or CR. CS: The client sends an event. CR: The client receives an event. SR: The server receives an event. SS: The server sends an event.

Table 4-27 BinarryAnnotation parameters

Parameter	Type	Description
key	String	Name of the extended information.

Parameter	Type	Description
endpoint	See **Table 4-28**.	(Optional) Service information about the client.
value	String	Value of the extended information.

Table 4-28 endpoint parameters

Parameter	Type	Description
serviceName	String	(Optional) Service name of the client.
ipv4	String	(Optional) IP address of the client.
port	String	(Optional) Port of the client.

Example response

```
{
  "errorCode": "SVCSTG.ATS.2000",
  "errorMessage":null,
  "responseInfo": [
    "{\"traceId\":\"0000000027046b00\",\"id\":\"b42460f5cf86cab4\",\"name\":\"aos-apiserver:aos-
apiserver-1005774711-ll63p:/api/v1/namespaces/manage/pods\",\"timestamp\":
1506260836597000,\"duration\":67000,\"annotations\":[{\"timestamp\":1506260836597000,\"value\":\"cs\",
\"endpoint\":{\"serviceName\":\"aos-apiserver\",\"ipv4\":\"10.186.60.43\",\"port\":6443}},{\"timestamp\":
1506260836664000,\"value\":\"cr\",\"endpoint\":{\"serviceName\":\"aos-apiserver\",\"ipv4\":\"10.186.60.43\",
\"port\":6443}}],\"binaryAnnotations\":[{\"key\":\"append\",\"value\":\"GET\"},{\"key\":\"async\",\"value\":
\"0\"},{\"key\":\"goid\",\"value\":\"58\"},{\"key\":\"result\",\"value\":\"0\"},{\"key\":\"resultCode\",\"value\":
\"200\"},{\"key\":\"seqno\",\"value\":\"1506260836597048618\"},{\"key\":\"type\",\"value\":\"1\"}]}"
  ]
}
```

Status Code

- Success response

  Table 4-29 describes the status code.

  Table 4-29 Status code

Status Code	Description
200	The request has succeeded.

5 Permissions Policies and Supported Actions

5.1 Introduction

This chapter describes fine-grained permissions management for your APM. If your account does not need individual IAM users, then you may skip over this chapter.

By default, new IAM users do not have any permissions assigned. You need to add a user to one or more groups, and assign permissions policies or roles to these groups. The user then inherits permissions from the groups it is a member of. This process is called authorization. After authorization, the user can perform specified operations on APM.

You can grant users permissions by using roles and policies. Roles are a type of coarse-grained authorization mechanism that defines permissions related to user responsibilities. Policies define API-based permissions for operations on specific resources under certain conditions, allowing for more fine-grained, secure access control of cloud resources.

NOTE

Policy-based authorization is useful if you want to allow or deny the access to an API.

An account has all the permissions required to call all APIs, but users must be assigned the required permissions. The permissions required for calling an API are determined by the actions supported by the API. Only users who have been granted permissions allowing the actions can call the API successfully. For example, if an IAM user queries metrics using an API, the user must have been granted permissions that allow the **apm:metric:get** action.

Supported Actions

IAM provides system-defined policies that can be directly used. You can also create custom policies and use them to supplement system-defined policies, implementing more refined access control. Operations supported by policies are specific to APIs. The following lists common concepts related to policies:

- Permissions: Defined by actions in custom policies.
- APIs: REST APIs that can be called by a user who has been granted specific permissions.
- Actions: Specific operations that are allowed or denied.
- Dependent actions: Actions on which a specific action depends to take effect. When assigning permissions for the action to a user, you also need to assign permissions for the dependent actions.
- IAM and enterprise projects: Type of projects for which an action will take effect. Policies that contain actions for both IAM and enterprise projects can be used and take effect for both IAM and Enterprise Management. Policies that only contain actions for IAM projects can be used and only take effect for IAM.

APM supports the following actions that can be defined in custom policies:

- **5.2 Actions**: includes the actions supported by APM APIs, such as the APIs for querying the application list, service list, service instance list, service transaction list, tracing data, and tracing details.

5.2 Actions

NOTE

√: supported; x: not supported

Table 5-1 API actions

Permissions	API	Action	IAM Project	Enterprise Project
Querying the application list	GET /v1/{project_id}/atps/monitorgroups	apm:inventory:get	√	√
Querying the service list	GET /v1/{project_id}/ats/applications	apm:ats:get	√	√
Querying the service instance list	GET /v1/{project_id}/ats/applications/{application}/instances	apm:ats:get	√	√

Permissions	API	Action	IAM Project	Enterprise Project
Querying the service transaction list	GET /v1/{project_id}/ats/applications/{application}/transactions	apm:ats:get	√	√
Querying tracing data	GET /v1/{project_id}/ats/traces	apm:ats:get	√	√
Querying tracing details	GET /v1/{project_id}/ats/spans	apm:ats:get	√	√

6 Appendix

6.1 Status Code

Status codes are listed in **Table 1 Status codes**.

Table 6-1 Status codes

Status Code	Code	Description
100	Continue	The server has received the initial part of the request and the client should continue to send the remaining part. It is issued on a provisional basis while request processing continues. It alerts the client to wait for a final response.

Status Code	Code	Description
101	Switching Protocols	The requester has asked the server to switch protocols and the server has agreed to do so. The protocol can only be switched to a newer protocol. For example, the current HTTP protocol is switched to a later version of HTTP.
200	OK	The request has succeeded.
201	Created	The request has been fulfilled and resulted in a new resource being created.
202	Accepted	The request has been accepted for processing, but the processing has not been completed.
203	Non-Authoritative Information	The server successfully processed the request, but is returning information that may be from another source.
204	NoContent	The server has successfully processed the request, but does not return any content. The status code is returned in response to an HTTP OPTIONS request.
205	Reset Content	The server has successfully processed the request, but does not return any content. Unlike a 204 response, this response requires that the requester reset the content.
206	Partial Content	The server has successfully processed a part of the GET request.

Status Code	Code	Description
300	Multiple Choices	There are multiple options for the requested resource. For example, this code could be used to present a list of resource characteristics and addresses from which the client such as a browser may choose.
301	Moved Permanently	This and all future requests should be permanently directed to the given URI indicated in this response.
302	Found	The requested resource was temporarily moved.
303	See Other	View other addresses by using GET and POSTrequests
304	Not Modified	The requested resource has not been modified. In such case, there is no need to retransmit the resource since the client still has a previously-downloaded copy.
305	Use Proxy	The requested resource is available only through a proxy.
306	Unused	This HTTP status code is no longer used.
400	BadRequest	The request is invalid. The client should modify the request instead of re-initiating it.
401	Unauthorized	The authorization information provided by the client is incorrect or invalid.
402	Payment Required	This status code is reserved for future use.

Status Code	Code	Description
403	Forbidden	The server has received the request and understood it, but the server is refusing to respond to it. The client should modify the request instead of re-initiating it.
404	NotFound	The requested resource could not be found. The client should modify the request instead of re-initiating it.
405	MethodNotAllowed	A request method is not supported for the requested resource. The client should not repeat the request without modifications.
406	Not Acceptable	The server could not fulfill the request according to the content characteristics of the request.
407	Proxy Authentication Required	This code is similar to 401, but indicates that the client must first authenticate itself with the proxy.
408	Request Time-out	The server timed out waiting for the request. The client may re-initiate the request without modifications at any later time.
409	Conflict	The request could not be processed due to a conflict in the request, such as an edit conflict between multiple simultaneous updates or the resource that the client attempts to create exists.

Status Code	Code	Description
410	Gone	The requested resource has been deleted permanently and will not be available again.
411	Length Required	The server refused to process the request because the request does not specify the length of its content.
412	Precondition Failed	The server does not meet one of the preconditions that the requester puts on the request.
413	Request Entity Too Large	The request is larger than the server is willing or able to process. The server may close the connection to prevent the client from continuing the request. If the server temporarily cannot process the request, the response will contain a Retry-After header field.
414	Request-URI Too Large	The URI provided was too long for the server to process.
415	Unsupported Media Type	The server does not support the media type in the request.
416	Requested range not satisfiable	The requested range is invalid.
417	Expectation Failed	The server fails to meet the requirements of the Expect request-header field.
422	UnprocessableEntity	The request was well-formed but was unable to be followed due to semantic errors.

Status Code	Code	Description
429	TooManyRequests	The client has sent more requests than its rate limit is allowed within a given amount of time, or the server has received more requests than it is able to process within a given amount of time. In this case, the client should repeat requests after the time specified in the Retry-After header of the response expires. In this case, it is advisable for the client to re-initiate requests after the time specified in the Retry-After header of the response expires.
500	InternalServerError	The server is able to receive the request but it could not understand the request.
501	Not Implemented	The server does not support the requested function.
502	Bad Gateway	The server was acting as a gateway or proxy and received an invalid request from a remote server.
503	ServiceUnavailable	The requested service is invalid. It is advisable for the client to modify the request instead of re-initiating the request.
504	ServerTimeout	The server could not return a timely response. The response will reach the client only if the request carries a timeout parameter.

Status Code	Code	Description
505	HTTP Version not supported	The server does not support the HTTP protocol version used in the request.

6.2 Error Codes

If an error occurs in API calling, no result is returned. Identify the cause based on the error code of each API. If an error occurs in API calling, HTTP status code 4xx or 5xx is returned. The response body contains the specific error code and information. If you are unable to identify the cause of an error, contact the administrator and provide the error code so that we can help you solve the problem as soon as possible.

Format of an Error Response Body

If an error occurs during API calling, an error code and a message will be displayed. The following shows an error response body.

```
{
   "errorCode": "SVCSTG_AMS_4000001",
   "errorMessage": "Request param invalid"
}
```

In the preceding information, **errorCode** is an error code, and **errorMessage** describes the error.

Error Code Description

Status Code	Error Code	Error Message	Solution
200	SVCSTG.ATPS.2000	Information queried successfully.	-
200	SVCSTG.ATS.2000	Information queried successfully.	-
400	SVCSTG.ATS.400101	Verification fails.	Check whether the parameter meets requirements.
400	SVCSTG.ATS.200103	No service information found.	Check whether the parameter meets requirements.
400	SVCSTG.ATS.200103	No instance information found.	Check whether the parameter meets requirements.

Status Code	Error Code	Error Message	Solution
400	SVCSTG.ATS.200103	No transaction information found.	Check whether the parameter meets requirements.
400	SVCSTG.ATS.200103	No tracing data found.	Check whether the parameter meets requirements.
400	SVCSTG.ATS.200103	No call data found.	Check whether the parameter meets requirements.

6.3 Obtaining a Project ID

Obtaining a Project ID from the Console

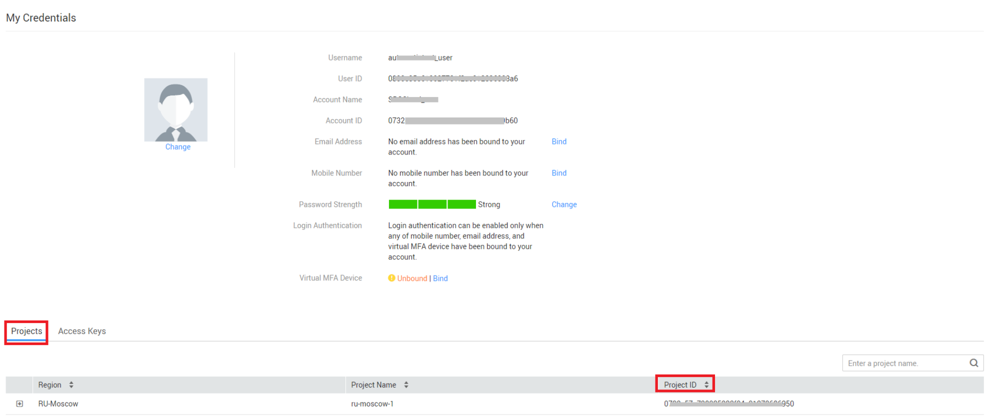
A project ID is required for some URLs when an API is called. To obtain a project ID, perform the following operations:

Step 1 Log in to the management console.

Step 2 Click the username and select **My Credentials** from the drop-down list.

On the **My Credentials** page, view a project ID in the project list.

Figure 6-1 Viewing a project ID



My Credentials

Username	au[redacted]_user
User ID	08[redacted]3a6
Account Name	S[redacted]
Account ID	0732[redacted]b60
Email Address	No email address has been bound to your account. Bind
Mobile Number	No mobile number has been bound to your account. Bind
Password Strength	Strong Change
Login Authentication	Login authentication can be enabled only when any of mobile number, email address, and virtual MFA device have been bound to your account.
Virtual MFA Device	Unbound | Bind

Region	Project Name	Project ID
RU-Moscow	ru-moscow-1	07[redacted]950

If there are multiple projects in one region, expand **Region** and view subproject IDs in the **Project ID** column.

----End

6.4 Obtaining an Account ID

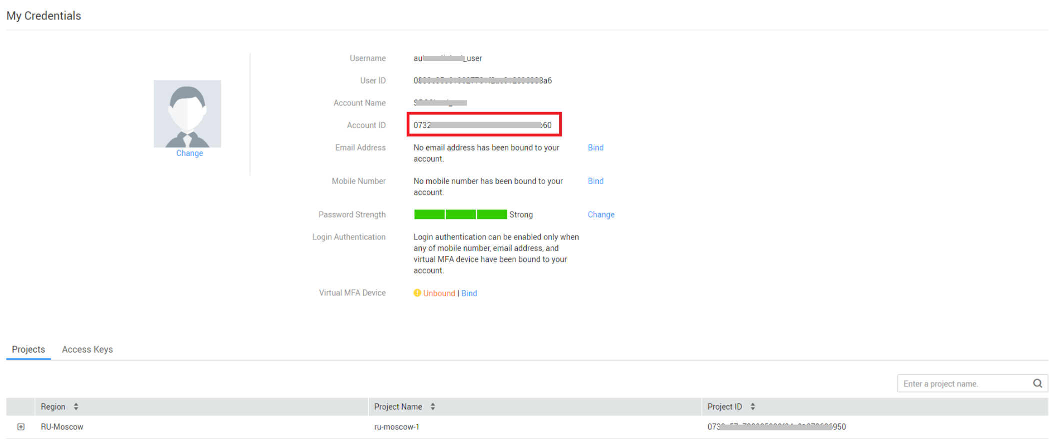
An account ID is required for some URLs when an API is called. To obtain an account ID, perform the following operations:

Step 1 Log in to the management console.

Step 2 Click the username and select **My Credentials** from the drop-down list.

On the **My Credentials** page, view an account ID.

Figure 6-2 Obtaining an account ID



My Credentials

Username	au[redacted]_user
User ID	08[redacted]3a6
Account Name	[redacted]
Account ID	0732[redacted]60
Email Address	No email address has been bound to your account.
Mobile Number	No mobile number has been bound to your account.
Password Strength	🟩🟩🟩 Strong
Login Authentication	Login authentication can be enabled only when any of mobile number, email address, and virtual MFA device have been bound to your account.
Virtual MFA Device	Unbound | Bind

Region	Project Name	Project ID
RU-Moscow	ru-moscow-1	073[redacted]950

----End

6.5 Common Request Headers

Table 6-2 Common request headers

Header	Description	Mandatory	Example
X-Auth-Token	User token.	Mandatory if token authentication is in use.	-
Content-Type	Type of content in a request. The value of this field is **application/ json;charset=utf8**.	Mandatory	application/ json;charset=utf8

Header	Description	Mandatory	Example
x-sdk-date	Time to send a request. The format is (YYYYMMDD'T'HHMMSS'Z'). The value is the Greenwich Mean Time (GMT) of the system.	Mandatory if AK/SK authentication is in use.	20160629T101459Z
Authorization	Authentication information, which is the result of request signing.	Mandatory if AK/SK authentication is in use.	-
Host	Requested server information, which is obtained from the URL in an API request. The value is hostname[:port]. If no port number is specified, the default port number will be selected. For HTTPS, the default port number is 443.	Mandatory if AK/SK authentication is in use.	-

6.6 Common Response Headers

A response message usually contains the following header fields:

Table 6-3 Common response headers

Header	Description	Example
Date	Time to send a response. The time format is defined by RFC822.	Mon, 12 Nov 2007 15:55:01 GMT
Server	Software information used by the server to process a request.	Apache

Header	Description	Example
Content-Length	The decimal number of bytes contained in a response message body.	xxx
Content-Type	MIME type of a response message body.	application/json

7 Change History

Table 7-1 Change history

Released On	Description
2020-02-26	This issue is the first release.