

# LAB: Z/OSMF - CHOOSE YOUR OWN TOPIC

## Getting started with z/OS Data Gatherer REST Services

Alexander Giemsa  
IBM Corporation

[alexander.giemsa@ibm.com](mailto:alexander.giemsa@ibm.com)

## Content:

Introduction .....	3
Lesson 1 – Getting started with the Swagger interface .....	5
Lesson 2 – Authenticate to REST API and use simple REST requests .....	9
Lesson 3 – Use REST requests with filter .....	22
Next steps .....	30
Use IBM® z/OS® Data Gatherer SMF REST APIs .....	30
Learn more about z/OSMF Data Gatherer REST services.....	30

## Figures:

Figure 1 Overview of z/OS Data Gatherer SMF REST services .....	3
Figure 2 Swagger UI link to OpenAPI document .....	5
Figure 3 Generated server url .....	6
Figure 4 SMF REST API documentation in schema section .....	7
Figure 5 SMF REST API documentation.....	8

# Introduction

The SMF REST services of the z/OS Data Gatherer provide access to SMF data through REST calls. As shown in the following picture, SMF REST services run inside a z/OSMF managed Liberty server instance. It provides several REST request paths (endpoints) which are to be called by client applications. The IBM® z/OS® Data Gatherer GRBSMFR service is called to retrieve SMF record data and/or metadata on the SMF dataset, depending on the request input. The application then processes the data into a JSON representation of the requested SMF data.

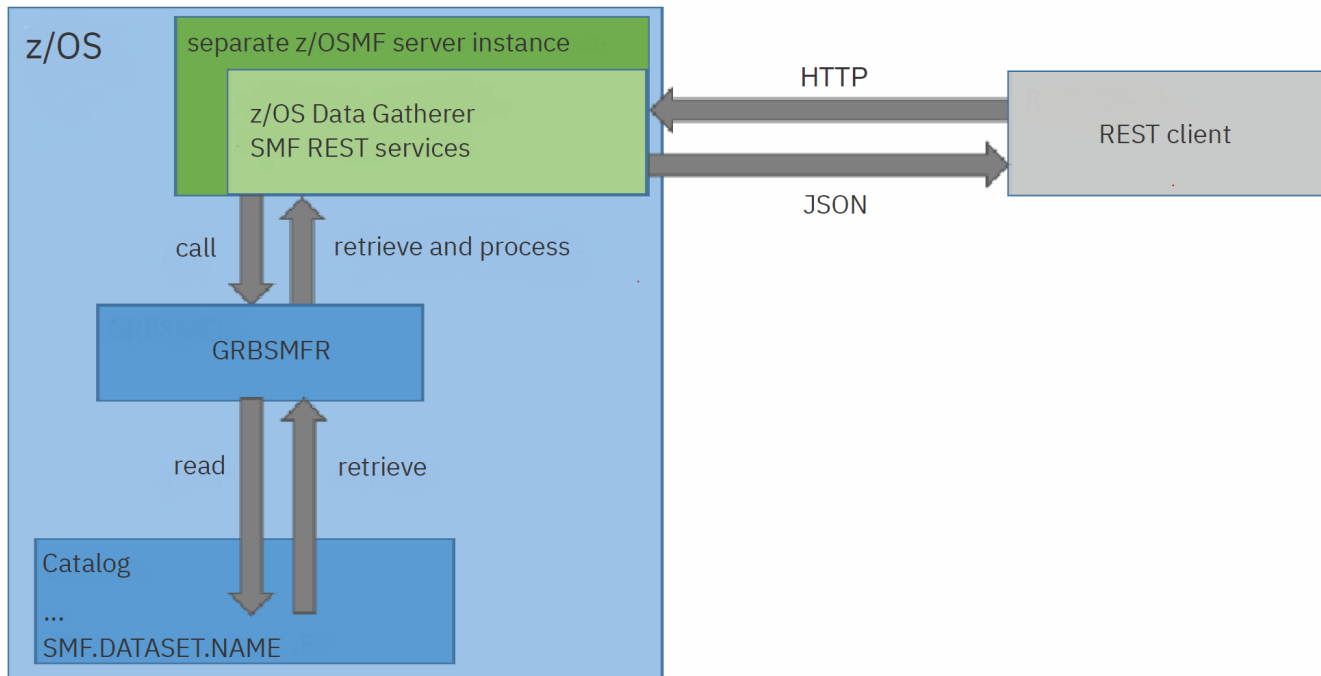


Figure 1 Overview of z/OS Data Gatherer SMF REST services

Swagger is an open source software framework that allows developers to design, build, document, and use RESTful APIs (application programming interfaces). It provides a set of tools and specifications for creating interactive API documentation, making it simpler to understand and use APIs.

At its core, Swagger uses the standard specification format OpenAPI to define the structure and behavior of APIs. The RESTful API of the z/OS Data Gatherer includes an OpenAPI document as part of the application.

In the following exercises, you are introduced to the Swagger UI, which is a convenient way to interact with the z/OS Data Gatherer REST API server. You learn how to:

- Use the Swagger interface.
- Authenticate and make REST requests.
- Use filters within REST requests.

No previous knowledge about Swagger UI or REST services is needed.

Allow at least 30 minutes to complete the exercises in this scenario.

# Lesson 1 – Getting started with the Swagger interface

Use the Swagger interface and IBM® z/OS® Data Gatherer REST API documentation.

Swagger™ provides a simple interface that is called Swagger UI, which automatically generates interactive API documentation based on the Open API specification. This documentation includes details about each endpoint, such as the HTTP methods it supports, input parameters, expected responses, and even sample requests that can be run directly from the documentation.

You must start the browser to load Swagger interface.

1. To start working with Swagger UI, open your browser, preferably **Firefox** or **Chrome** with JSON

Enter the following URL in the address bar:

<https://share.centers.ihost.com:444/zosmf/zosdg/smf/swagger-ui/index.html#/> .

2. At the top of your browser screen, you notice general information about z/OS Data Gatherer REST Services and the link to the OpenAPI specification document: </zosmf/zosdg/smf/v3/api-docs>. If you click the link with the right click and then open in the new tab, you can find the full technical description of the API.



Figure 2 Swagger UI link to OpenAPI document

The Swagger UI interface presents the API documentation in a visually appealing and intuitive manner. Developer and API consumers are able to explore the API's capabilities, understand its inputs and outputs, and even test it without leaving the documentation.

### 3. Continue looking at Swagger UI.

Scrolling down, you can see the generated server url of REST services:

<https://share.centers.ihost.com:444/zosmf/zosdg/smf> and the **Authorize** action button.

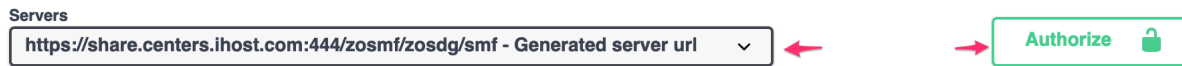


Figure 3 Generated server url

You will use **Authorize** in the next lesson to authenticate against z/OS Data Gatherer REST services.

### 4. Following the generated server URL you will find the list of Endpoints.

The Endpoints section is divided by the categories, such as:

- Discovery.
- SMF 113.
- SMF 70.
- SMF 71.

Each category has an arrow up or down to the right. When the arrow shows up, the category is expanded, and individual endpoints are shown. When the arrow shows down, the category is collapsed, and the endpoints are hidden. Clicking anywhere on the category line results in expanding or collapsing the category.

Discovery methods or endpoints are used to discover the content of the SMF data source, while all other methods are reading and retrieving SMF data.

When you expand the Discovery category, you can see 3 methods or endpoints inside:

- GET /v1/smf/discover/dataset/exists/{datasetName}.
- GET /v1/smf/discover/describe/{datasetName}.
- POST /v1/smf/discover/describe/{datasetName}.

When you expand any of the endpoints, you see the information about that endpoint, the required or optional parameters and possible return codes.

Each of the following categories represents the supporting SMF record and contains one endpoint per SMF record subtype. For example section SMF 70 contains two endpoints:

- GET /v1/smf/type/70/subtype/1.
- GET /v1/smf/type/70/subtype/2.

5. If you scroll down, after Endpoints section, you can see the documentation in place inside Schema section:

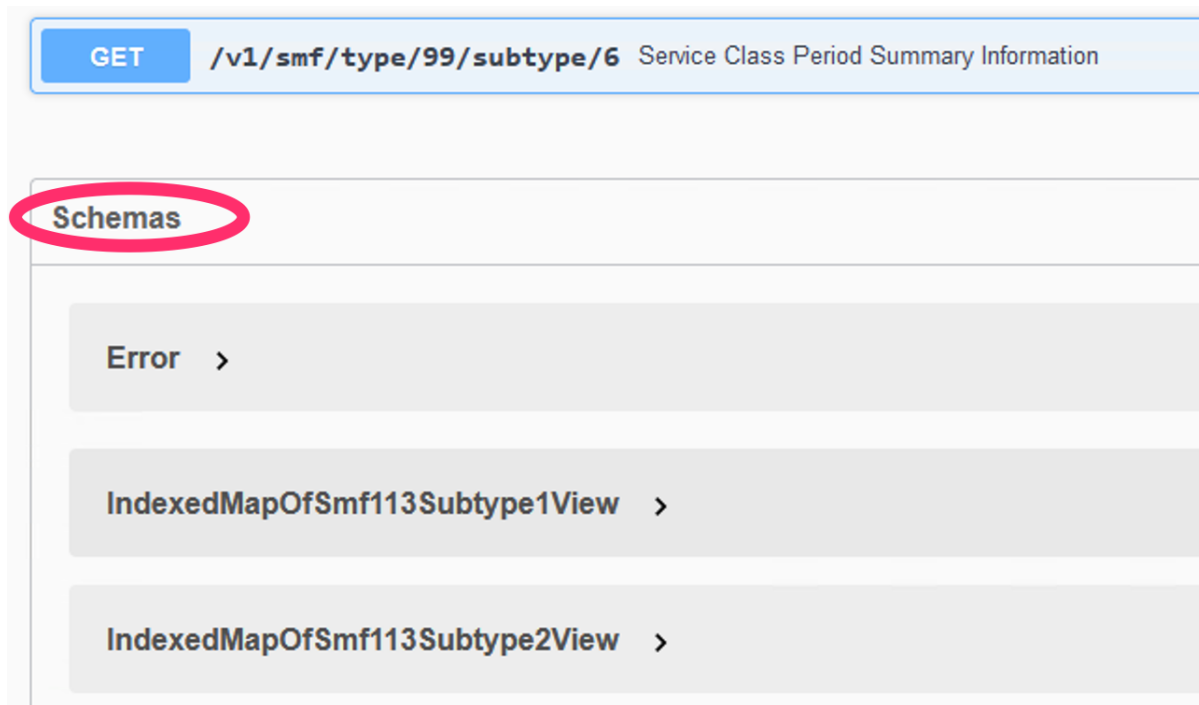


Figure 4 SMF REST API documentation in schema section

By clicking any item, for example SMF70\_SUBTYPE1, you can expand it and investigate its structure.

At the top, the expanded description lists all fields of a structure, e.g. the individual SMF70 fields. By clicking the "> [...]" button next to a field information like its datatype and field description can be displayed. Likewise a "> {...}" button will expand a more complex structure.

Below the fields you can find information on the substructures, e.g. the individual data sections for SMF70 Subtype 1.

Any displayed substructure (e.g. SMF70\_SUBTYPE1\_PRODUCT\_SECTION) is listed somewhere in the schema section but can be also directly investigated by clicking on it.

The symbol "< \* >" indicates a 1-n relationship and for example describes that a SMF70.1 record has multiple CPU Data Sections.

SMF70_SUBTYPE1 ▾ {	
description:	
SMF70LEN	> [...]
SMF70SEG	> [...]
SMF70FLG	> [...]
SMF70RRF	> [...]
SMF70SUT	> [...]
SMF70V4	> [...]
SMF70ESA	> [...]
SMF70VXA	> [...]
● ● ●	
SMF70TNN	> [...]
smf70Subtype1ProductSection	SMF70_SUBTYPE1_PRODUCT_SECTION > {...}
smf70Subtype1AsidArea	SMF70_SUBTYPE1_ASID_AREA > {...}
smf70Subtype1CpuControl	SMF70_SUBTYPE1_CPU_CONTROL > {...}
smf70Subtype1CpuData	▾ {
	< * >: SMF70_SUBTYPE1_CPU_DATA > {...}
	}
smf70Subtype1PrSmPartitionData	> {...}
smf70Subtype1PrSmLogicalProcessorData	> {...}
smf70Subtype1CpuIdentifications	> {...}
smf70Subtype1LogicalCoreData	> {...}
smf70Subtype1TenantResourceGroupData	> {...}
x-znl-structure-name	"SMF70HDR"
x-zdg-max-size	100
x-znl-version	"797"
}	

Figure 5 SMF REST API documentation

In summary, Swagger provides a comprehensive interface for documenting, and testing RESTful APIs. It simplifies the process of API development by promoting consistency, providing interactive documentation, and enabling seamless collaboration between API providers and consumers. In this first task, you got familiar with Swagger UI. The next step is to authenticate and make the first requests by using Swagger interface.



## Lesson 2 – Authenticate to REST API and use simple REST requests

In this exercise, you use IBM® z/OS® Data Gatherer REST services to check whether the data set with the provided name exists and to retrieve SMF data.

You use the web client program from the previous exercise.

Before you can make REST request, you need to be authorized. You can authorize by clicking **Authorize**, or you are requested to authorize when you submit the first REST request.

1. Click **Authorize** right to the Generated server URL:

A green rectangular button with rounded corners. It contains the word "Authorize" in green text and a green padlock icon to its right.

2. Log in with Username and Password provided by the instructor and click **Authorize** after it.

### Available authorizations

A small black 'x' icon used to close the dialog.

#### Basic (http, Basic)

Base64 encoding of ID and password joined by a single colon (':')

Username:

Password:

A green rectangular button with rounded corners and the word "Authorize" in green text. A red arrow points to it from the left.A gray rectangular button with rounded corners and the word "Close" in black text.

#### LTPA2\_COOKIE (apiKey)

The LtpaToken2-Cookie with the Lightweight Third Party Authentication Token (LTPAv2) obtained from the Authority Provider

Name: LtpaToken2

In: cookie

Value:

A green rectangular button with rounded corners and the word "Authorize" in green text.A gray rectangular button with rounded corners and the word "Close" in black text.

To close Available authorizations screen, click Close next to Logout.

## Available authorizations x

### Basic (http, Basic)

Authorized

Base64 encoding of ID and password joined by a single colon (':')

Username:

Password: \*\*\*\*\*

Logout

Close

### LTPA2\_COOKIE (apiKey)

The LtpaToken2-Cookie with the Lightweight Third Party Authentication Token (LTPAv2) obtained from the Authority Provider

Name: LtpaToken2

In: cookie

Value:

Authorize

Close

- Next, expand **Discovery** section to list the methods to discover the content of the SMF data source, if it is not already expanded:

**Discovery**
Methods to discover the content of the SMF data source

GET	/v1/smf/discover/dataset/exists/{datasetName}	Checks if the dataset with the name provided exists	▼	🔒
GET	/v1/smf/discover/describe/{datasetName}	The summary about each SMF (sub-)type	▼	🔒
POST	/v1/smf/discover/describe/{datasetName}	The summary about each SMF (sub-)type	▼	🔒

- Now, expand the first method or the GET `/v1/smf/discover/dataset/exists/{datasetName}` endpoint, if not already expanded:

GET
/v1/smf/discover/dataset/exists/{datasetName}
Checks if the dataset with the name provided exists

This end-point performs a quick-check of the dataset existence

Parameters
Try it out

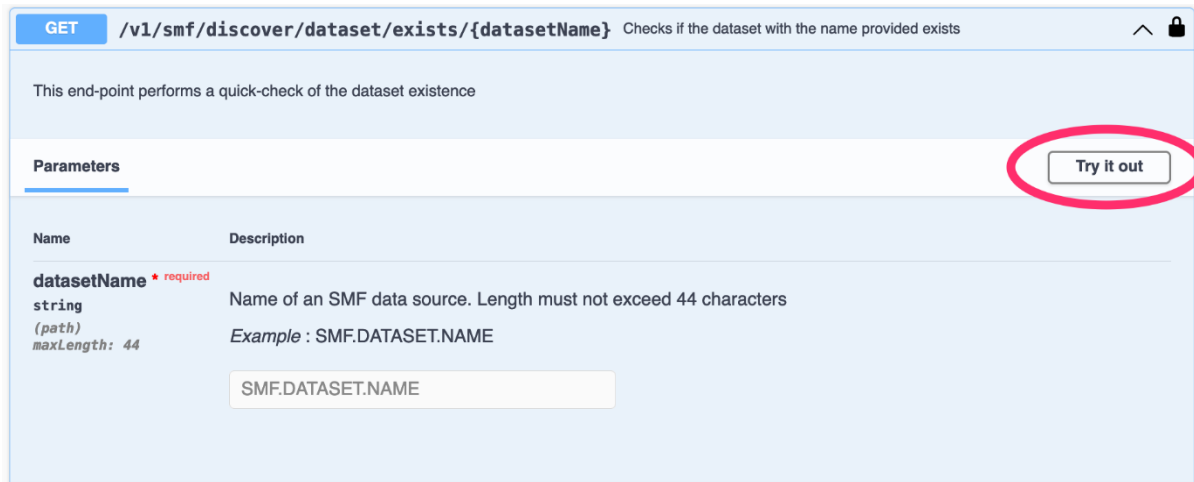
Name	Description
<b>datasetName</b> * required string (path) maxLength: 44	Name of an SMF data source. Length must not exceed 44 characters Example : SMF.DATASET.NAME <input type="text" value="SMF.DATASET.NAME"/>

Responses

Code	Description	Links
200	No error occurred and the response contains all the data requested  Media type <input type="text" value="*/*"/> Controls Accept header.  Example Value   Schema <div> true </div>	No links
400	Cannot process the request due to client error  Media type <input type="text" value="application/json"/>  Example Value   Schema <div> { "message": "Save 418 Movement", "status": "418 I'M A TEAPOT" } </div>	No links

You can read that this end point checks the data set existence. The parameter **datasetName** is the only parameter and it is required. The result of the request is **true**, if the provided data set exists on the system. In that case, the expected return code is 200.

5. Click **Try it out**.



GET /v1/smf/discover/dataset/exists/{datasetName} Checks if the dataset with the name provided exists

This end-point performs a quick-check of the dataset existence

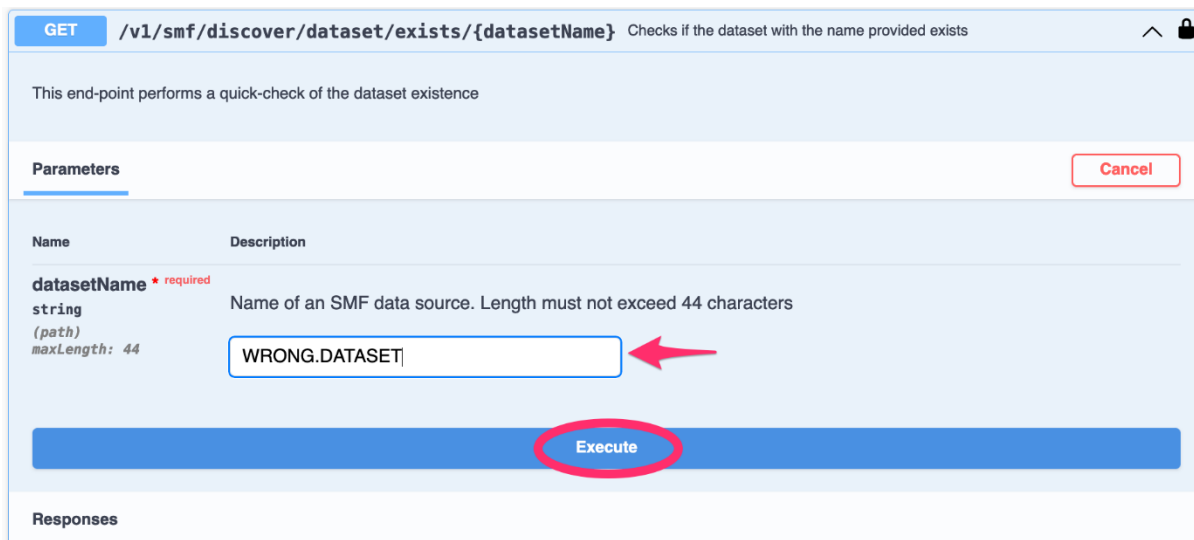
**Parameters**

Name	Description
<b>datasetName</b> <span style="color: red;">★ required</span> string (path) maxLength: 44	Name of an SMF data source. Length must not exceed 44 characters <i>Example : SMF.DATASET.NAME</i>

SMF.DATASET.NAME

Try it out

Provide **datasetName**: WRONG.DATASET and click Execute.



GET /v1/smf/discover/dataset/exists/{datasetName} Checks if the dataset with the name provided exists

This end-point performs a quick-check of the dataset existence

**Parameters**

Name	Description
<b>datasetName</b> <span style="color: red;">★ required</span> string (path) maxLength: 44	Name of an SMF data source. Length must not exceed 44 characters

WRONG.DATASET

Execute

Cancel

Responses

The result of the request is the JSON document with the message: Dataset 'WRONG.DATASET' not found and the Return code 404.

Request URL

`https://share.centers.ihost.com:444/zosmf/zosdg/smf/v1/smf/discover/dataset/exists/WRONG.DATASET`

Server response

Code	Details
404	<p>Error: Not Found</p> <p>Response body</p> <pre>{   "message": "Dataset 'WRONG.DATASET' not found",   "status": "Not Found" }</pre> <p>Download</p>

6. Now, provide `datasetName: PMUENCH.SMFDATA.SMF7X.RECORDS` and click **Execute**:

GET `/v1/smf/discover/describe/{datasetName}` The summary about each SMF (sub-)type

Provides the cumulative information about each (sub-)type present in the SMF data source. If the SMF resource is accessed for the first time, or if the cache-validation failed, the result is cached and used every next time the discovery data is needed. Cache is stored for the time specified by the application settings

Parameters

Name	Description
<b>datasetName</b> * required	Name of an SMF data source. Length must not exceed 44 characters
string (path) maxLength: 44	

PMUENCH.SMFDATA.SMF7X.RECORDS

Execute Clear

The result of the request is `true` with the Return code 200, which is good.

Server response

Code	Details
200	<p>Response body</p> <pre>true</pre> <p>Download</p>

7. Next, look at the next endpoint from the Discovery section to extract metadata on the SMF dataset. Expand the method or the endpoint `GET /v1/smf/describe/dataset/{datasetName}`, and click **Try it out**.

GET
/v1/smf/discover/describe/{datasetName}
The summary about each SMF (sub-)type

Provides the cumulative information about each (sub-)type present in the SMF data source. If the SMF resource is accessed for the first time, or if the cache-validation failed, the result is cached and used every next time the discovery data is needed. Cache is stored for the time specified by the application settings

Parameters
Try it out

Name	Description
<b>datasetName</b> <span>★ required</span> string (path) maxLength: 44	Name of an SMF data source. Length must not exceed 44 characters Example : SMF.DATASET.NAME <input type="text" value="SMF.DATASET.NAME"/>

Provide datasetName: PMUENCH.SMFDATA.SMF7X.RECORDS and click **Execute**. The result of the request is the summary about each SMF subtype, with the Return code of “200”.

- Copy the Request URL  
<https://share.centers.ihost.com:444/zosmf/zosdg/smf/v1/smf/discover/describe/PMUENCH.SMFDATA.SMF7X.RECORDS> into the clipboard.

Execute
Clear

Responses

Curl

```
curl -X 'GET' \ 'https://192.148.8.225:444/zosmf/zosdg/smf/v1/smf/discover/describe/PMUENCH.SMFDATA.SMF7X.RECORDS' \ -H 'accept: application/json'
```

Request URL
https://192.148.8.225:444/zosmf/zosdg/smf/v1/smf/discover/describe/PMUENCH.SMFDATA.SMF7X.RECORDS

Server response
Code
Details

200
Response body

```
{
  "datasetCreationDate": "2023-07-28",
  "discoveryTimestamp": "2023-08-03T14:49:59.472+02:00",
  "estimatedByteSize": 69354148,
  "totalNumberOfRecords": 4154,
  "smfIds": [
    "J80 ",
    "S2  "
  ],
  "earliestSmfRecordMoveTime": "2022-05-04T10:00:00.04",
  "latestSmfRecordMoveTime": "2023-07-28T10:45:41.93",
  "earliestStartOfSmfInterval": "2022-05-04T09:30:00",
  "latestStartOfSmfInterval": "2022-05-04T13:00:00",
  "earliestEndOfSmfInterval": "2022-05-04T09:59:59.943",
  "latestEndOfSmfInterval": "2022-05-04T13:30:00.001",
  "smfGroups": {
    "2": {
      "estimatedByteSize": 18,
      "totalNumberOfRecords": 1,
      "smfIds": [
        "S2  "
      ],
      "earliestSmfRecordMoveTime": "2023-07-28T10:45:37.5",
      "latestSmfRecordMoveTime": "2023-07-28T10:45:37.5"
    }
  }
}
```

Open the new tab in the browser, paste the Request URL in the address bar and press **ENTR**.

If Firefox is your browser of choice, or Chrome with JSON Viewer extension, you get a convenient format of the JSON document, with the possibility to expand and collapse the sections in the document.





## SMF 70 RMF Processor Activity

**GET** /v1/smf/type/70/subtype/1 CPU, PR/SM, and ICF Activity

**GET** /v1/smf/type/70/subtype/2 Cryptographic Hardware Activity

You can see the list of the endpoints for each subtype of SMF 70 record.

- Next, expand the first method or the endpoint **GET /v1/smf/type/70/subtype/1** and click **Try it out**:

**GET**
/v1/smf/type/70/subtype/1
 CPU, PR/SM, and ICF Activity

This end-point retrieves SMF 70 subtype 1 data from the specific SMF record using the dataset provided

Parameters

datasetName

string  
 (query)  
 maxLength: 44

Name of an SMF data source. Length must not exceed 44 characters  
 Example : SMF.DATASET.NAME

systemName

string  
 (query)  
 maxLength: 4

Four-character SMF system ID that identifies the z/OS system from which the data is gathered

startTime

string(\$date-time)  
 (query)

Data gathering begin time in ISO8601 format (YYYY-MM-DDThh:mm:ss.sss, e.g., '2020-08-14T13:01:12')

endTime

string(\$date-time)  
 (query)

Data gathering end time in ISO8601 format (YYYY-MM-DDThh:mm:ss.sss, e.g., '2020-08-14T13:01:12')

showMeta

boolean  
 (query)

If not set to true, all the fields marked with 'x-zdg-is-meta=true' are excluded from the response. These fields may be, however, present in the response, if are explicitly requested

SMF70\_SUBTYPE1\_PRODUCT\_SECTION

array[string]  
 (query)

A set of the fields that are to be returned for SMF70\_SUBTYPE1\_PRODUCT\_SECTION-structure. If used, only the listed fields are provided. '\*' encodes all fields.

You can see the list of input parameters and filters. The only required input parameter is **datasetName**, while optional parameters are **systemName**, **startTime** and **endTime**.

If the parameter **showMeta** is set to true, other fields that are marked as "meta" are included into the response.

11. Put the name of the data set in the datasetName field: PMUENCH.SMFDATA.SMF7X.RECORDS and click **Execute**:

SMF 70 RMF Processor Activity

GET

/v1/smf/type/70/subtype/1 CPU, PR/SM, and ICF Activity

This end-point retrieves SMF 70 subtype 1 data from the specific SMF record using the dataset provided

Parameters

Cancel

Name	Description
<b>datasetName</b> <span style="color: red;">* required</span> string (query) maxLength: 44	Name of an SMF data source. Length must not exceed 44 characters  <input type="text" value="PMUENCH.SMFDATA.SMF7X.RECORDS"/>
systemName string (query) maxLength: 4	Four-character SMF system ID that identifies the z/OS system from which the data is gathered  <input type="text" value="systemName"/>
startTime string(\$date-time) (query)	Data gathering begin time in ISO8601 format (YYYY-MM-DDThh:mm:ss.sss, e.g., '2020-08-14T13:01:12')  <input type="text" value="startTime"/>
endTime string(\$date-time) (query)	Data gathering end time in ISO8601 format (YYYY-MM-DDThh:mm:ss.sss, e.g., '2020-08-14T13:01:12')  <input type="text" value="endTime"/>

SMF70\_SUBTYPE1\_LOGICAL\_CORE\_DATA  
 array[string]  
 (query)

A set of the fields that are to be returned for SMF70\_SUBTYPE1\_LOGICAL\_CORE\_DATA-structure. If used, only the listed fields are provided. "" encodes all fields.

Execute

Now, the process could take longer the dataset records are read for the first time.

The result is the JSON document with the Return code of 200. The JSON document provides the array of the records of SMF type 70 subtype 1 from the provided data set.

Look at the result in the **Response body** after the **Execute**.

You can use the slider to scroll up and down:

The screenshot shows the SHARE interface with the following elements:

- Buttons:** "Execute" (highlighted in blue) and "Clear".
- Responses Section:**
  - Curl:** A text box containing the command: `curl -X 'GET' \ 'https://192.148.8.225:444/zosmf/zosdg/smf/v1/smf/type/70/subtype/1?datasetName=PMUENCH.SMFDATA.SMF7X.RECORDS' \ -H 'accept: application/json'`
  - Request URL:** A text box containing the URL: `https://192.148.8.225:444/zosmf/zosdg/smf/v1/smf/type/70/subtype/1?datasetName=PMUENCH.SMFDATA.SMF7X.RECORDS`. A red arrow points to this field.
  - Server response:**
    - Code:** A table with two columns: "Code" and "Details". The "Code" column shows "200", with a red arrow pointing to it.
    - Response body:** A large text area displaying a JSON document. A red arrow points to the "Response body" label, and another red arrow points to a scrollbar on the right side of the JSON text area.

The JSON document in the response body is as follows:

```
{
  "1": {
    "SMF70SEG": 0,
    "SMF70RRF": true,
    "SMF70SUT": true,
    "SMF70V4": true,
    "SMF70ESA": true,
    "SMF70VXA": true,
    "SMF70OS": true,
    "SMF70BFY": true,
    "SMF70RTY": 70,
    "SMF70TME": "10:00:00.04",
    "SMF70DTE": "2022-05-04",
    "SMF70SID": "J80 ",
    "SMF70SSI": "RMF ",
    "SMF70STY": 1,
    "SMF70TRN": 9,
    "smf70Subtype1ProductSection": {
      "SMF70PRD": "RMF ",
      "SMF70IST": "09:30:00",
      "SMF70DAT": "2022-05-04",
      "SMF70INT": "00:29:59.943",

```

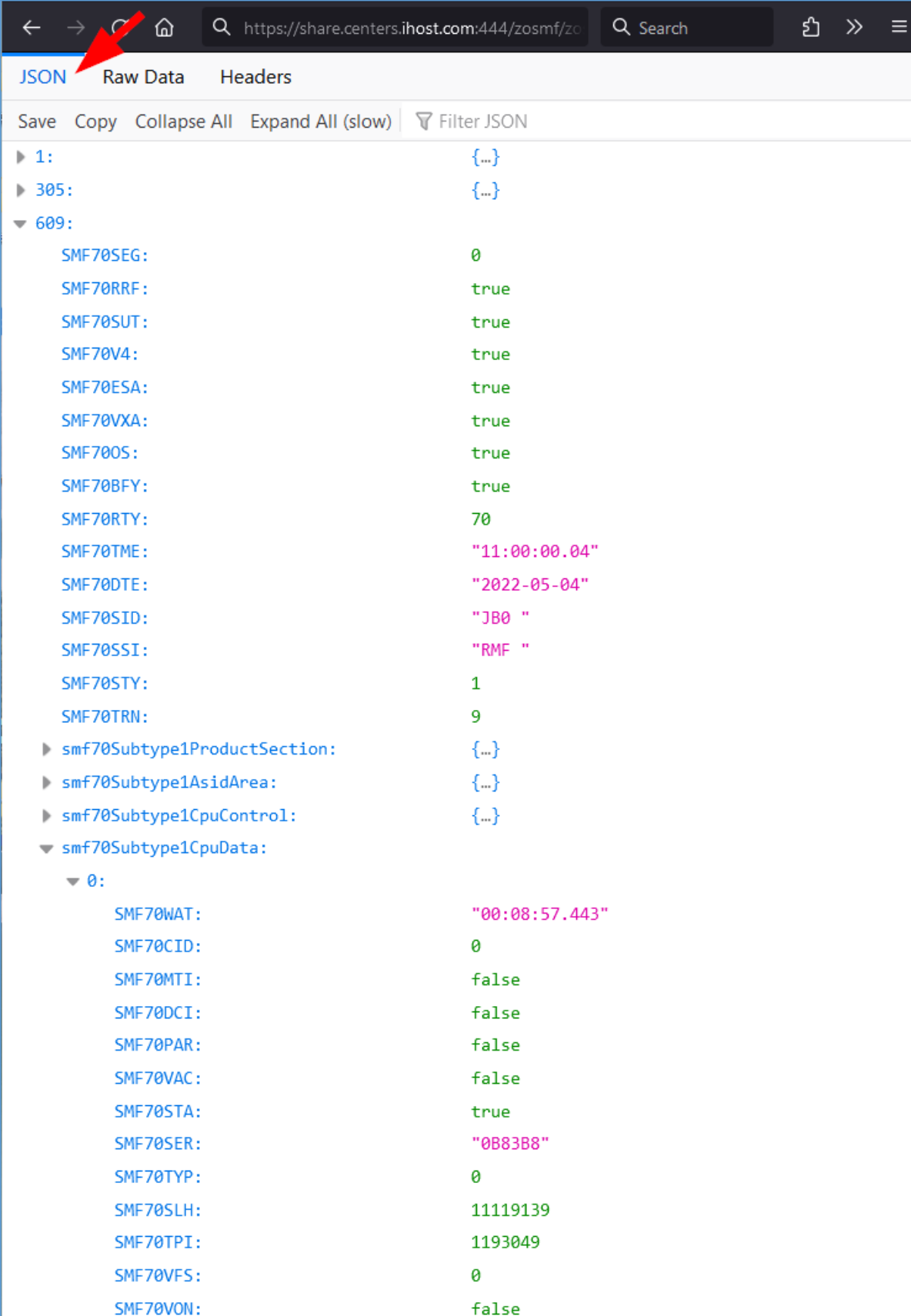
The Request URL contains the URL of the current request.

## 12. Copy the **Request URL**:

<https://share.centers.ihost.com:444/zosmf/zosdg/smf/v1/smf/type/70/subtype/1?datasetName=PMUENCH.SMFDATA.SMF7X.RECORDS> into the clipboard (right click and copy with the mouse).

Open the new tab in the browser, paste the content of the clipboard in the address bar and press **ENTER**.

You get a convenient format of the JSON document, with the possibility to expand and collapse the sections in the document.



The screenshot shows a web browser window with the address bar displaying `https://share.centers.ihost.com:444/zosmf/zosmf/`. The 'JSON' tab is selected, and the JSON data is displayed in a collapsible tree view. A red arrow points to the 'JSON' tab.

```

{
  "1": {},
  "305": {},
  "609": {
    "SMF70SEG": 0,
    "SMF70RRF": true,
    "SMF70SUT": true,
    "SMF70V4": true,
    "SMF70ESA": true,
    "SMF70VXA": true,
    "SMF70OS": true,
    "SMF70BFY": true,
    "SMF70RTY": 70,
    "SMF70TME": "11:00:00.04",
    "SMF70DTE": "2022-05-04",
    "SMF70SID": "JB0 ",
    "SMF70SSI": "RMF ",
    "SMF70STY": 1,
    "SMF70TRN": 9,
    "smf70Subtype1ProductSection": {},
    "smf70Subtype1AsidArea": {},
    "smf70Subtype1CpuControl": {},
    "smf70Subtype1CpuData": {
      "0": {
        "SMF70WAT": "00:08:57.443",
        "SMF70CID": 0,
        "SMF70MTI": false,
        "SMF70DCI": false,
        "SMF70PAR": false,
        "SMF70VAC": false,
        "SMF70STA": true,
        "SMF70SER": "0B83B8",
        "SMF70TYP": 0,
        "SMF70SLH": 11119139,
        "SMF70TPI": 1193049,
        "SMF70VFS": 0,
        "SMF70VON": false
      }
    }
  }
}

```

Review the content and close the tab when done

In this second task, you learned to authenticate and make first requests with minimal input parameters.

The next step is to make requests with more extensive use of other parameters to filter the data.

## Lesson 3 – Use REST requests with filter

In this exercise, you use IBM® z/OS® Data Gatherer REST services retrieve SMF data with more extensive use of other parameters to filter the data.

You use the web client program from the previous exercise.

1. If you closed the browser session from the previous exercise, open it again:

Enter the following URL in the address bar:

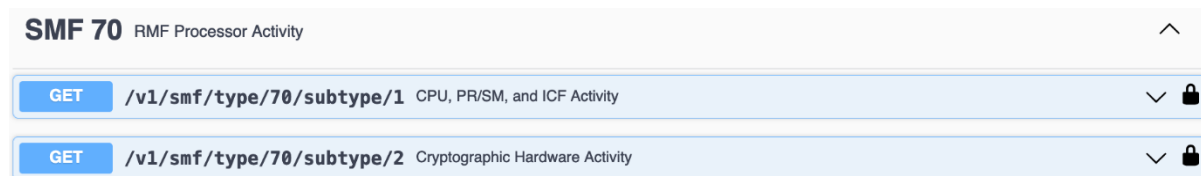
<https://share.centers.ihost.com:444/zosmf/zosdg/smf/swagger-ui/index.html#> .

2. If you are not already authorized, click **Authorize** right to the Generated server URL.

Put Username and Password under Basic section in **Available authorizations**, provided by the instructor and click **Authorize** after it.

To close Available authorizations screen, click **Close** next to **Logout**.

3. Expand **SMF 70** section to list the methods to read SMF type 70 records.



You can see the endpoints section for each subtype of SMF 70 record.

- Next, expand the first method or the endpoint GET /v1/smf/type/70/subtype/1 and click **Try it out** (if not already clicked during the previous exercise):

GET /v1/smf/type/70/subtype/1 CPU, PR/SM, and ICF Activity

This end-point retrieves SMF 70 subtype 1 data from the specific SMF record using the dataset provided

**Parameters**

Name	Description
<b>datasetName</b> * required string (query) maxLength: 44	Name of an SMF data source. Length must not exceed 44 characters <i>Example</i> : SMF.DATASET.NAME <input type="text" value="SMF.DATASET.NAME"/>
systemName string (query) maxLength: 4	Four-character SMF system ID that identifies the z/OS system from which the data is gathered <input type="text" value="systemName"/>
startTime string(\$date-time) (query)	Data gathering begin time in ISO8601 format (YYYY-MM-DDThh:mm:ss.sss, e.g., '2020-08-14T13:01:12') <input type="text" value="startTime"/>
endTime string(\$date-time) (query)	Data gathering end time in ISO8601 format (YYYY-MM-DDThh:mm:ss.sss, e.g., '2020-08-14T13:01:12') <input type="text" value="endTime"/>
showMeta boolean (query)	If not set to true, all the fields marked with 'x-zdg-is-meta=true' are excluded from the response. These fields may be, however, present in the response, if are explicitly requested <input type="text" value=""/>
SMF70_SUBTYPE1_PRODUCT_SECTION array[string] (query)	A set of the fields that are to be returned for SMF70_SUBTYPE1_PRODUCT_SECTION-structure. If used, only the listed fields are provided. '*' encodes all fields. <input type="text" value=""/>

You can see the list of input parameters and filters. The only mandatory input parameter is `datasetName`, while the parameters `systemName`, `startTime` and `endTime` are optional.

If the parameter `showMeta` is set to `true`, extra fields that are marked as "meta" are included into the response.

Put the name of the data set in the `datasetName` field: `PMUENCH.SMFDATA.SMF7X.RECORDS`.

- Now, you have a chance to provide other input parameters to filter the data.

You can filter the data by:

- `systemName`.
- `startTime`.
- `endTime`.

You can also choose to retrieve the data only from one or more specific sections of SMF records. Each SMF record subtype has their own set of sections. SMF 70 subtype 1 has the following sections:

- SMF70\_SUBTYPE1\_PRODUCT\_SECTION.
- SMF70\_SUBTYPE1\_PR\_SM\_LOGICAL\_PROCESSOR\_DATA.
- SMF70\_SUBTYPE1\_TENANT\_RESOURCE\_GROUP\_DATA.
- SMF70\_SUBTYPE1\_LOGICAL\_CORE\_DATA.
- SMF70\_SUBTYPE1\_CPU\_CONTROL.
- SMF70\_SUBTYPE1.
- SMF70\_SUBTYPE1\_CPU\_DATA.
- SMF70\_SUBTYPE1\_ASID\_AREA.
- SMF70\_SUBTYPE1\_PR\_SM\_PARTITION\_DATA.
- SMF70\_SUBTYPE1\_CPU\_IDENTIFICATION.

For each section you can select the fields to retrieve, by clicking and marking the fields. You can select “\*” to retrieve all the fields from the section or you can select “--” to ignore the section and doesn’t retrieve any field from it.

6. Specify some optional input parameters to filter the output data:

Specify the values for the following optional parameters:

- In the field `systemName`, specify JBO value.
- `startTime`, specify 2022-05-04T11:00:00.000 value.
- `endTime`, specify 2022-05-04T12:00:00.000 value.



SMF 70 RMF Processor Activity

GET

/v1/smf/type/70/subtype/1 CPU, PR/SM, and ICF Activity

This end-point retrieves SMF 70 subtype 1 data from the specific SMF record using the dataset provided

Parameters

Cancel

Name	Description
<b>datasetName</b> * required string (query) maxLength: 44	Name of an SMF data source. Length must not exceed 44 characters <input type="text" value="PMUENCH.SMFDATA.SMF7X.RECORDS"/>
<b>systemName</b> string (query) maxLength: 4	Four-character SMF system ID that identifies the z/OS system from which the data is gathered <input type="text" value="JB0"/>
<b>startTime</b> string(\$date-time) (query)	Data gathering begin time in ISO8601 format (YYYY-MM-DDThh:mm:ss.sss, e.g., '2020-08-14T13:01:12') <input type="text" value="2022-05-04T11:00:00.000"/>
<b>endTime</b> string(\$date-time) (query)	Data gathering end time in ISO8601 format (YYYY-MM-DDThh:mm:ss.sss, e.g., '2020-08-14T13:01:12') <input type="text" value="2022-05-04T12:00:00.000"/>
<b>showMeta</b> boolean	If not set to true, all the fields marked with 'x-zdg-is-meta=true' are excluded from the

- Click **Execute**, after the last SMF section of this endpoint.

Now you already limited the JSON output to system JB0 and the intervals between 11:00 and 12:00 on 4<sup>th</sup> of May.

You can copy the Request URL:

<https://share.centers.ihost.com:444/zosmf/zosdg/smf/v1/smf/type/70/subtype/1?datasetName=PMUENCH.SMFDATA.SMF7X.RECORDS&systemName=JB0&startTime=2022-05-04T11%3A00%3A00.000&endTime=2022-05-04T12%3A00%3A00.000> into the Clipboard and paste it into the new browser tab and review the resulting JSON Output.

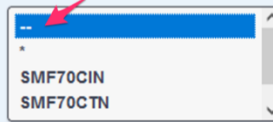
- The next step is to limit the JSON output to selected fields.

Now, going back to the same endpoint, where you filtered the data by `systemName`, `startTime`, and `endTime`, you can select or deselect the sections of the record.

Ignore all sections, but `SMF70_SUBTYPE1_LOGICAL_CORE_DATA` by selecting -- next to them, for example:

SMF70\_SUBTYPE1\_CPU\_IDENTIFICATION  
array[string]  
(query)

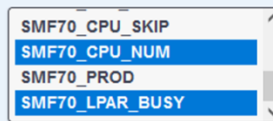
A set of the fields that are to be returned for SMF70\_SUBTYPE1\_CPU\_IDENTIFICATION-structure. If used, only the listed fields are provided. '\*' encodes all fields.



9. Select the following fields from SMF70\_SUBTYPE1\_LOGICAL\_CORE\_DATA section:

SMF70\_SUBTYPE1\_LOGICAL\_CORE\_DATA  
array[string]  
(query)

A set of the fields that are to be returned for SMF70\_SUBTYPE1\_LOGICAL\_CORE\_DATA-structure. If used, only the listed fields are provided. '\*' encodes all fields.



- SMF70\_CORE\_ID.
- SMF70\_CPU\_NUM.
- SMF70\_LPAR\_BUSY.

**Note:** When selecting multiple fields, you must hold the Control key on your keyboard.

10. Click **Execute**, after the last SMF section of this endpoint.

SMF70\_SUBTYPE1\_TENANT\_RESOURCE\_GROUP\_DATA  
array[string]  
(query)

A set of the fields that are to be returned for SMF70\_SUBTYPE1\_TENANT\_RESOURCE\_GROUP\_DATA-structure. If used, only the listed fields are provided. '\*' encodes all fields.

--  
\*  
SMF70\_TRG\_NAME  
SMF70\_TRG\_DESC

SMF70\_SUBTYPE1\_LOGICAL\_CORE\_DATA  
array[string]  
(query)

A set of the fields that are to be returned for SMF70\_SUBTYPE1\_LOGICAL\_CORE\_DATA-structure. If used, only the listed fields are provided. '\*' encodes all fields.

SMF70\_CPU\_SKIP  
SMF70\_CPU\_NUM  
SMF70\_PROD  
SMF70\_LPAR\_BUSY

Execute

Clear

Responses

The result is the JSON document with the Return code 200.

JSON provides the array of the records that contain only the specified fields from SMF 70 subtype 1 from the provided data set. The records are filtered by systemName, startTime, and startDate.

You can use the slider to scroll up and down:

## Responses

### Curl

```
curl -X 'GET' \ 'https://192.148.8.225:444/zosmf/zosdg/smf/v1/smf/type/70/subtype/1?datasetName=PMUENCH.SMFDATA.SMF7X.RECORDS&systemName=JB0&startTime=2022-05-04T11%3A00%3A00.000&endTime=2022-05-04T12%3A00%3A00.000&SMF70_SUBTYPE1_CPU_IDENTIFICATION=&SMF70_SUBTYPE1_CPU_CONTROL=&SMF70_SUBTYPE1_PR_SM_PARTITION_DATA=&SMF70_SUBTYPE1_CPU_DATA=&SMF70_SUBTYPE1_ASID_AREA=&SMF70_SUBTYPE1_PR_SM_LOGICAL_PROCESSOR_DATA=&SMF70_SUBTYPE1_PRODUCT_SECTION=&SMF70_SUBTYPE1_TENANT_RESOURCE_GROUP_DATA=&SMF70_SUBTYPE1_LOGICAL_CORE_DATA=SMF70_CORE_ID&SMF70_SUBTYPE1_LOGICAL_CORE_DATA=SMF70_CPU_NUM&SMF70_SUBTYPE1_LOGICAL_CORE_DATA=SMF70_LPAR_BUSY' \ -H 'accept: application/json'
```

### Request URL

```
https://192.148.8.225:444/zosmf/zosdg/smf/v1/smf/type/70/subtype/1?datasetName=PMUENCH.SMFDATA.SMF7X.RECORDS&systemName=JB0&startTime=2022-05-04T11%3A00%3A00.000&endTime=2022-05-04T12%3A00%3A00.000&SMF70_SUBTYPE1_CPU_IDENTIFICATION=&SMF70_SUBTYPE1_CPU_CONTROL=&SMF70_SUBTYPE1_PR_SM_PARTITION_DATA=&SMF70_SUBTYPE1_CPU_DATA=&SMF70_SUBTYPE1_ASID_AREA=&SMF70_SUBTYPE1_PR_SM_LOGICAL_PROCESSOR_DATA=&SMF70_SUBTYPE1_PRODUCT_SECTION=&SMF70_SUBTYPE1_TENANT_RESOURCE_GROUP_DATA=&SMF70_SUBTYPE1_LOGICAL_CORE_DATA=SMF70_CORE_ID&SMF70_SUBTYPE1_LOGICAL_CORE_DATA=SMF70_CPU_NUM&SMF70_SUBTYPE1_LOGICAL_CORE_DATA=SMF70_LPAR_BUSY
```

### Server response

#### Code

#### Details

200

#### Response body

```
{
  "20": {
    "SMF70_CORE_ID": 20,
    "SMF70_CPU_NUM": 2,
    "SMF70_LPAR_BUSY": 811447
  },
  "21": {
    "SMF70_CORE_ID": 21,
    "SMF70_CPU_NUM": 2,
    "SMF70_LPAR_BUSY": 160084
  },
  "22": {
    "SMF70_CORE_ID": 22,
    "SMF70_CPU_NUM": 2,
    "SMF70_LPAR_BUSY": 42958
  },
  "23": {
    "SMF70_CORE_ID": 23,
    "SMF70_CPU_NUM": 2,
    "SMF70_LPAR_BUSY": 12990
  },
  "24": {
    "SMF70_CORE_ID": 24,
    "SMF70_CPU_NUM": 2,
    "SMF70_LPAR_BUSY": 0
  }
}
```

Download

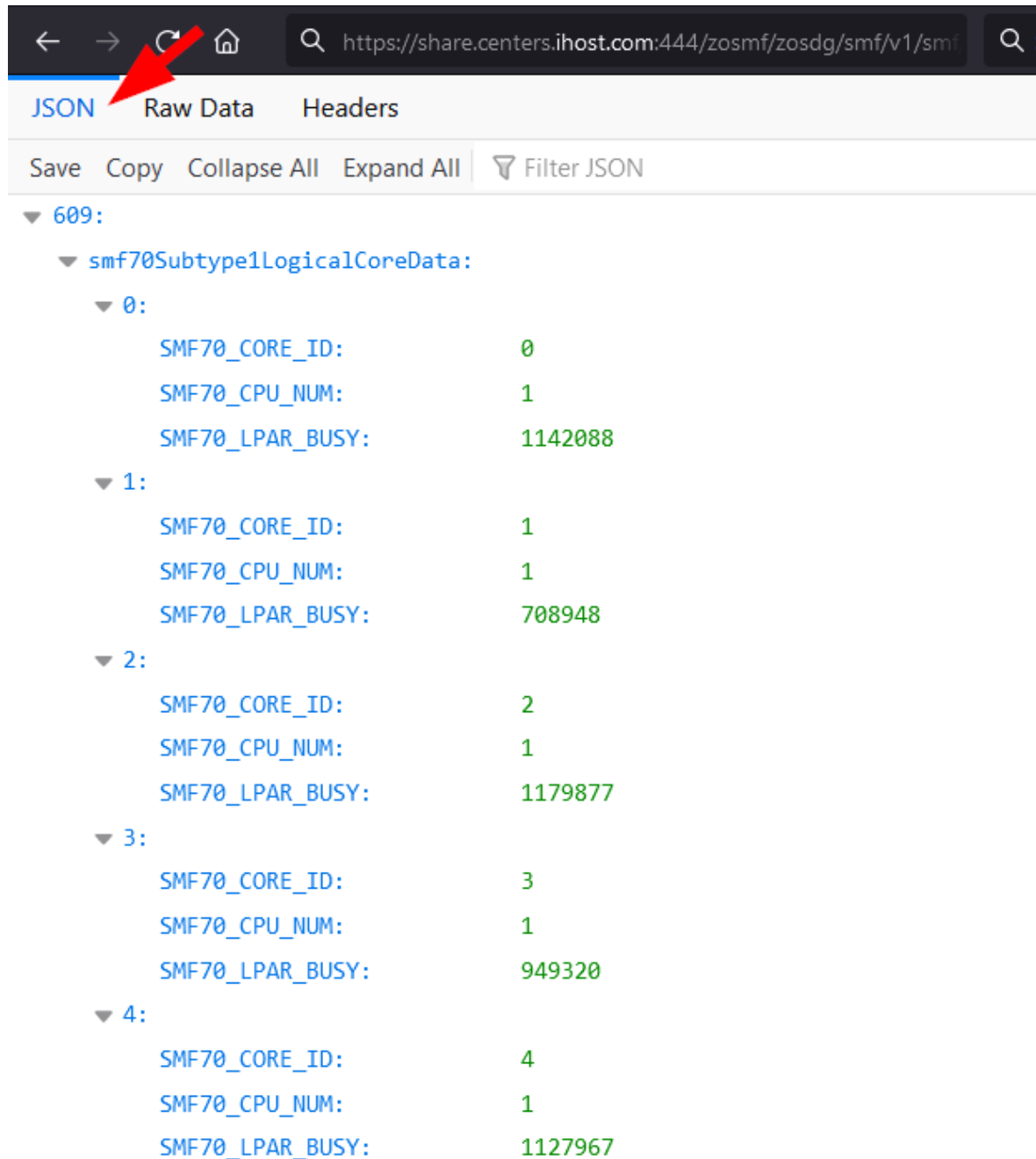
The Request URL contains the URL of the current request:

[https://share.centers.ihost.com:444/zosmf/zosdg/smf/v1/smf/type/70/subtype/1?datasetName=PMUENCH.SMFDATA.SMF7X.RECORDS&systemName=JB0&startTime=2022-05-04T11%3A00%3A00.000&endTime=2022-05-04T12%3A00%3A00.000&SMF70\\_SUBTYPE1\\_CPU\\_IDENTIFICATION=&SMF70\\_SUBTYPE1\\_CPU\\_CONTROL=&SMF70\\_SUBTYPE1\\_PR\\_SM\\_PARTITION\\_DATA=&SMF70\\_SUBTYPE1\\_CPU\\_DATA=&SMF70\\_SUBTYPE1\\_ASID\\_AREA=&SMF70\\_SUBTYPE1\\_PR\\_SM\\_LOGICAL\\_PROCESSOR\\_DATA=&SMF70\\_SUBTYPE1\\_PRODUCT\\_SECTION=&SMF70\\_SUBTYPE1\\_TENANT\\_RESOURCE\\_GROUP\\_DATA=&SMF70\\_SUBTYPE1\\_LOGICAL\\_CORE\\_DATA=SMF70\\_CORE\\_ID&SMF70\\_SUBTYPE1\\_LOGICAL\\_CORE\\_DATA=SMF70\\_CPU\\_NUM&SMF70\\_SUBTYPE1\\_LOGICAL\\_CORE\\_DATA=SMF70\\_LPAR\\_BUSY](https://share.centers.ihost.com:444/zosmf/zosdg/smf/v1/smf/type/70/subtype/1?datasetName=PMUENCH.SMFDATA.SMF7X.RECORDS&systemName=JB0&startTime=2022-05-04T11%3A00%3A00.000&endTime=2022-05-04T12%3A00%3A00.000&SMF70_SUBTYPE1_CPU_IDENTIFICATION=&SMF70_SUBTYPE1_CPU_CONTROL=&SMF70_SUBTYPE1_PR_SM_PARTITION_DATA=&SMF70_SUBTYPE1_CPU_DATA=&SMF70_SUBTYPE1_ASID_AREA=&SMF70_SUBTYPE1_PR_SM_LOGICAL_PROCESSOR_DATA=&SMF70_SUBTYPE1_PRODUCT_SECTION=&SMF70_SUBTYPE1_TENANT_RESOURCE_GROUP_DATA=&SMF70_SUBTYPE1_LOGICAL_CORE_DATA=SMF70_CORE_ID&SMF70_SUBTYPE1_LOGICAL_CORE_DATA=SMF70_CPU_NUM&SMF70_SUBTYPE1_LOGICAL_CORE_DATA=SMF70_LPAR_BUSY)

- Copy the **Request URL** content into the clipboard (right click and copy with the mouse).

Open the new tab in the browser, paste the content of the clipboard in the address bar and press **ENTER**.

You get a convenient format of the JSON document, with the possibility to expand and collapse the sections in the document.



← → ↻ 🏠 🔍 https://share.centers.ihost.com:444/zosmf/zosdg/smf/v1/smf 🔍 S

JSON Raw Data Headers

Save Copy Collapse All Expand All 🚫 Filter JSON

▼ 609:

▼ smf70Subtype1LogicalCoreData:

▼ 0:

SMF70_CORE_ID:	0
SMF70_CPU_NUM:	1
SMF70_LPAR_BUSY:	1142088

▼ 1:

SMF70_CORE_ID:	1
SMF70_CPU_NUM:	1
SMF70_LPAR_BUSY:	708948

▼ 2:

SMF70_CORE_ID:	2
SMF70_CPU_NUM:	1
SMF70_LPAR_BUSY:	1179877

▼ 3:

SMF70_CORE_ID:	3
SMF70_CPU_NUM:	1
SMF70_LPAR_BUSY:	949320

▼ 4:

SMF70_CORE_ID:	4
SMF70_CPU_NUM:	1
SMF70_LPAR_BUSY:	1127967

Review the content and close the tab when done.

Now you've learned how to make the REST request with more extensive use of other optional parameters to filter the data.

## Next steps

Did you find the scenarios useful? Consider some next steps.

### Use IBM® z/OS® Data Gatherer SMF REST APIs

Now, you are familiar with using z/OS Data Gatherer SMF REST services to retrieve SMF data and check the status of the data set. You can use this knowledge by trying these services in your own programs.

### Learn more about z/OSMF Data Gatherer REST services

For more information about z/OS Data Gatherer REST services, see the z/OS Data Gatherer publication in the IBM Documentation: <https://www.ibm.com/docs/en/zos/3.1.0?topic=zdpgg-accessing-data-using-zos-data-gatherer-rest-services>