



ABAP SDK

Implementation guide

for Azure Event hubs

<https://github.com/Microsoft/ABAP-SDK-for-Azure>

Author: Microsoft SAP Team

Version: 1.0

Contents

What is Azure Event hub?.....	3
Prerequisites.....	3
How to setup Event hub in Azure?.....	3
How to send data from SAP to Azure Event hub?.....	10
Creation of RFC destination to Azure Event hub	10
STRUST Setup.....	13
Configuration	15
ZREST_CONFIG	15
ZREST_CONF_MISC	15
ZADF_CONFIG	16
ZADF_EHUB_POLICY	17
DEMO Program	17
View sent data in Azure Eventhub	17
ABAP SDK Monitor	17
Auto re-processing of failed messages	19

What is Azure Event hub?

Stream millions of events per second

Azure Event Hubs is a hyper-scale telemetry ingestion service which collects, transforms and stores millions of events. As a distributed streaming platform, it gives you low latency and configurable time retention, which enables you to ingress massive amounts of telemetry into the cloud and read the data from multiple applications using publish-subscribe semantics.

For more details on Azure Event hubs, visit [Microsoft Azure Event hub](#)

Prerequisites

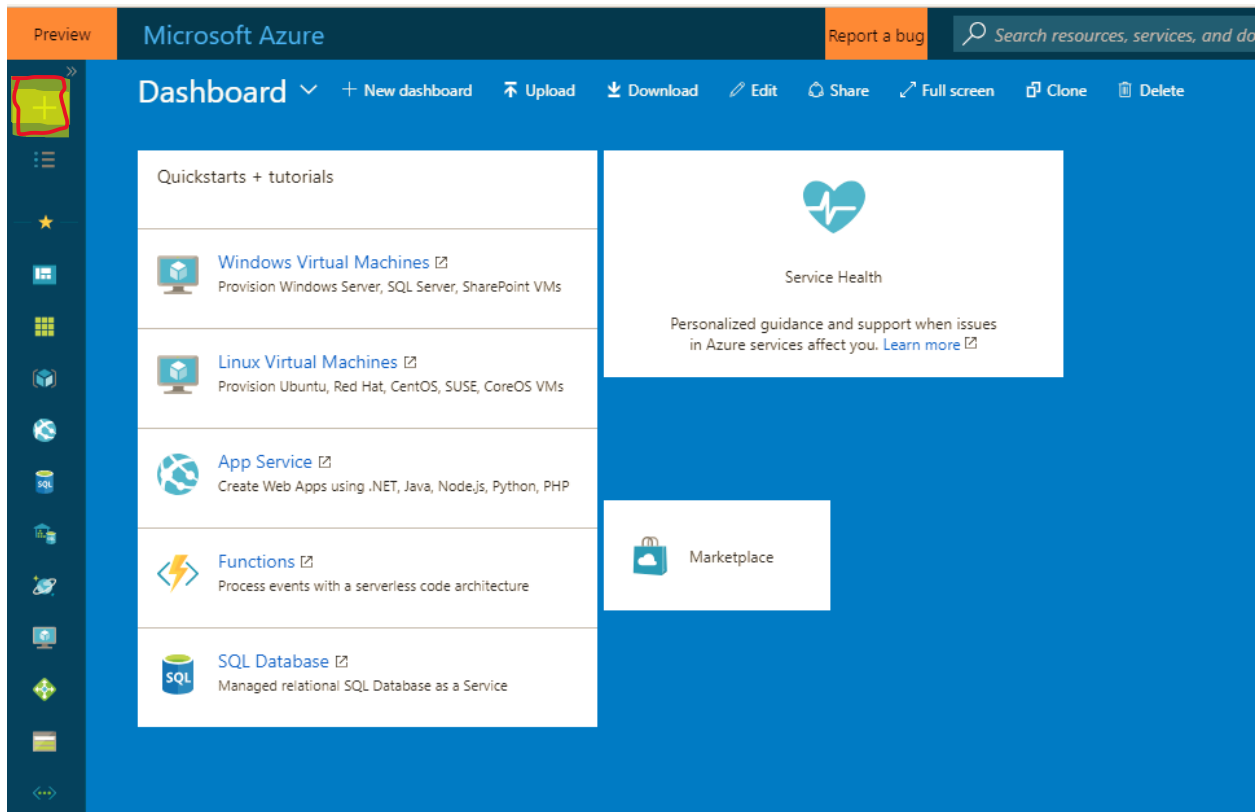
Make sure you have installed ABAP SDK for Azure in your SAP system. Refer document ‘ABAP SDK for Azure – Github’ for more details.

How to setup Event hub in Azure?

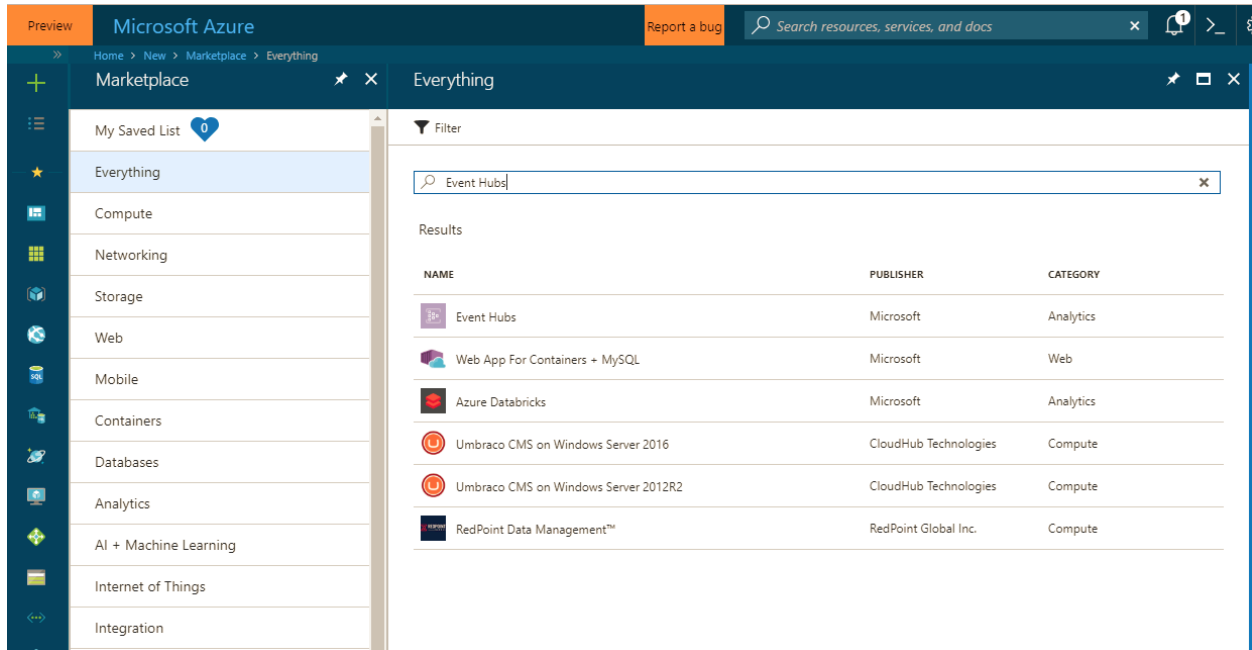
Login to [Microsoft Azure portal](#).

Note: If you do not have an account already. please create a new [Azure account](#). You can start free

Once you are logged into portal, chose push button ‘+’ on the left to create a new Azure service.



Search for Event hubs and Select “Event Hubs’ with Microsoft as publisher.



Now choose the Create push button to create new event hub instance in your subscription.

Event Hubs

Microsoft

Azure Event Hubs is a highly scalable publish-subscribe service that can ingest millions of events per second and stream them into multiple applications. This lets you process and analyze the massive amounts of data produced by your connected devices and applications.

Use Event Hubs to:

- Log millions of events per second in near real time.
- Connect devices using flexible authorization and throttling.
- Use time-based event buffering.
- Get a managed service with elastic scale.
- Reach a broad set of platforms using native client libraries.
- Pluggable adapters for other cloud services.

[Save for later](#)

SomeEHName

Event Hubs Namespace

Search (Ctrl+F)

- Overview
- Access control (IAM)
- Tags
- Diagnose and solve problems

SETTINGS

- Shared access policies
- Scale
- Geo-Recovery
- Event Grid
- Locks
- Automation script

ENTITIES

- Event Hubs

+ Event Hub Delete

Resource group (change)	Status	Location
Website	Active	West Central US
Subscription (change)	Subscription ID	Provisioning state
Visual Studio Enterprise	21ed19f1-af37-4389-af22-f77d86...	Succeeded
Created	Updated	Connection Strings
Tuesday, December 12, 2017, 11:3...	Tuesday, December 12, 2017, 11:3...	Connection Strings

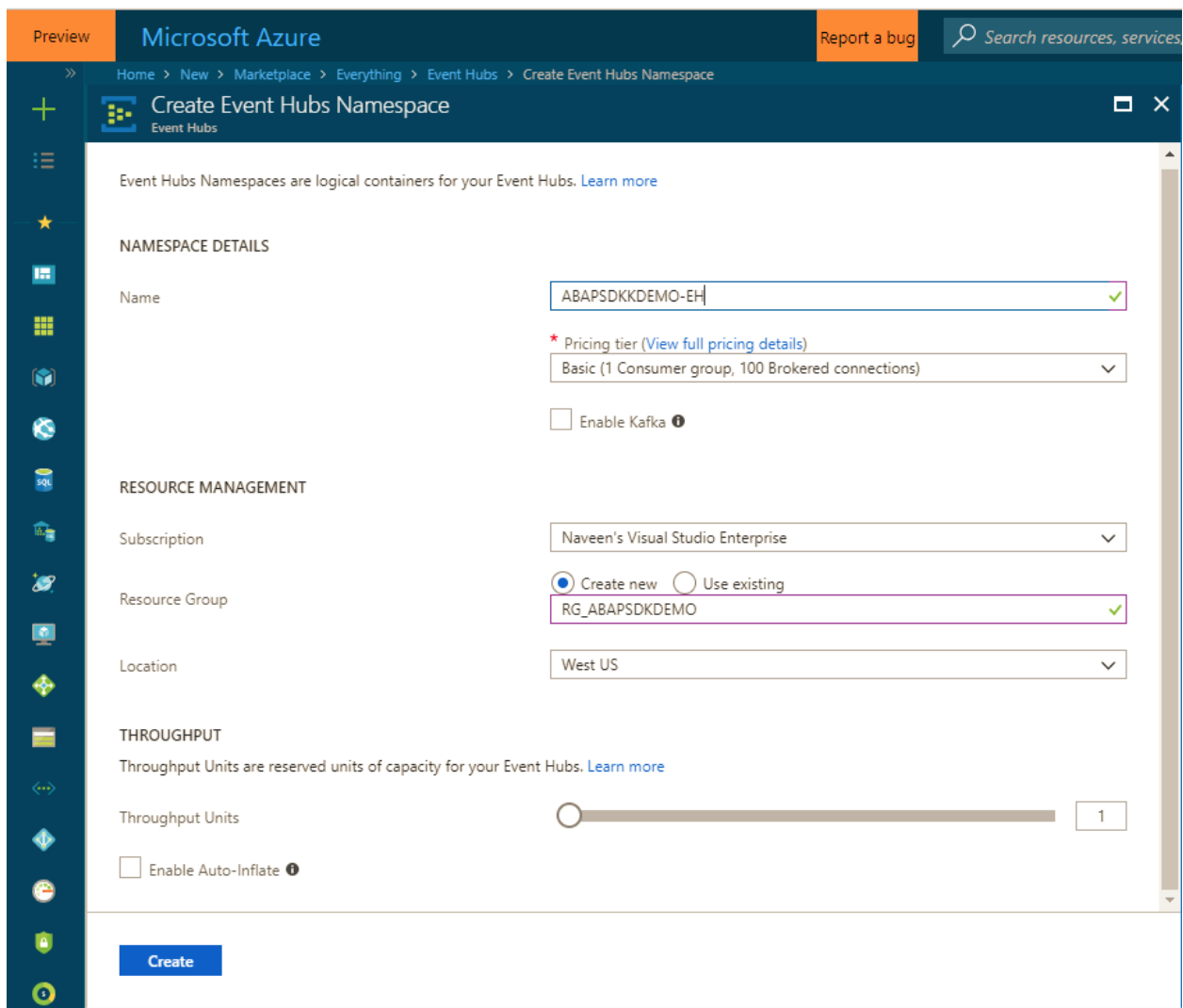
NAMESPACE CONTENTS PROVISIONING TIER: STANDBY

Show metrics data for the last: 1 hour 6 hours 12 hours 1 day 7 days 30 days [Click for additional metrics](#)

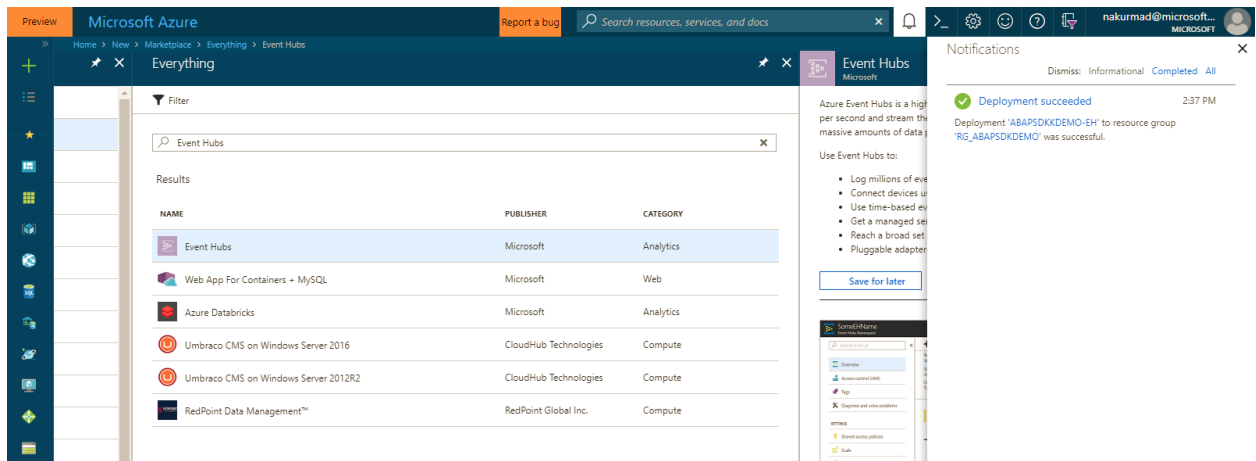
Requests Messages Throughput

[Create](#)

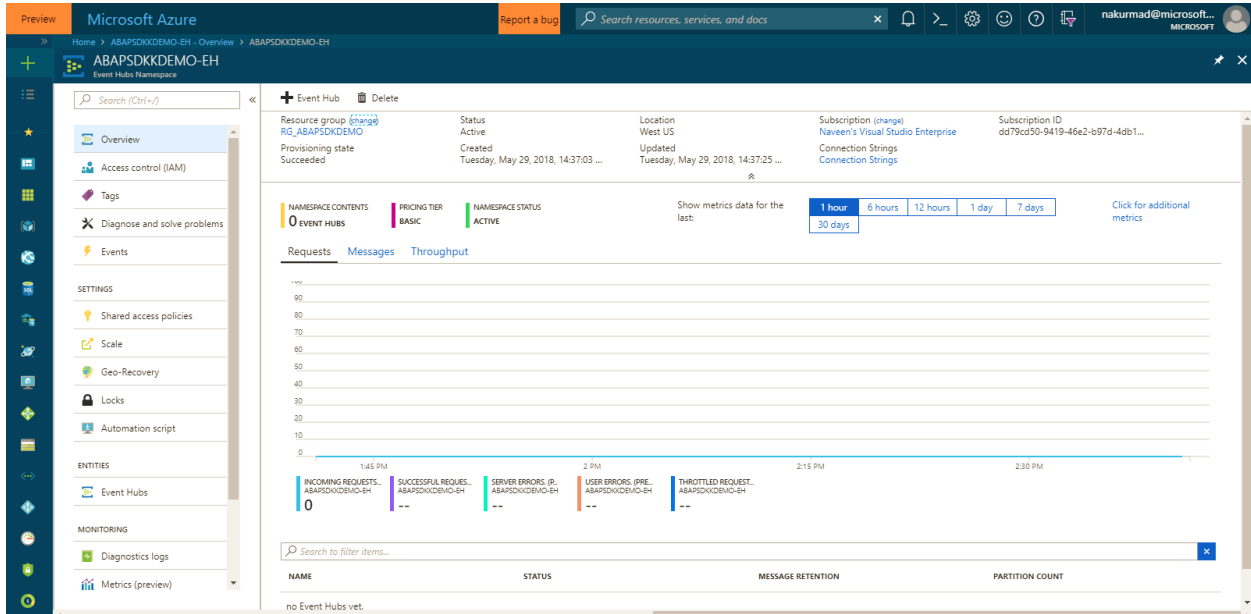
Provide appropriate values in the fields and hit push button **Create**



This should create a new Event Hub. This might take some time, so please wait until it is created successfully. You can check status in **Notifications**.

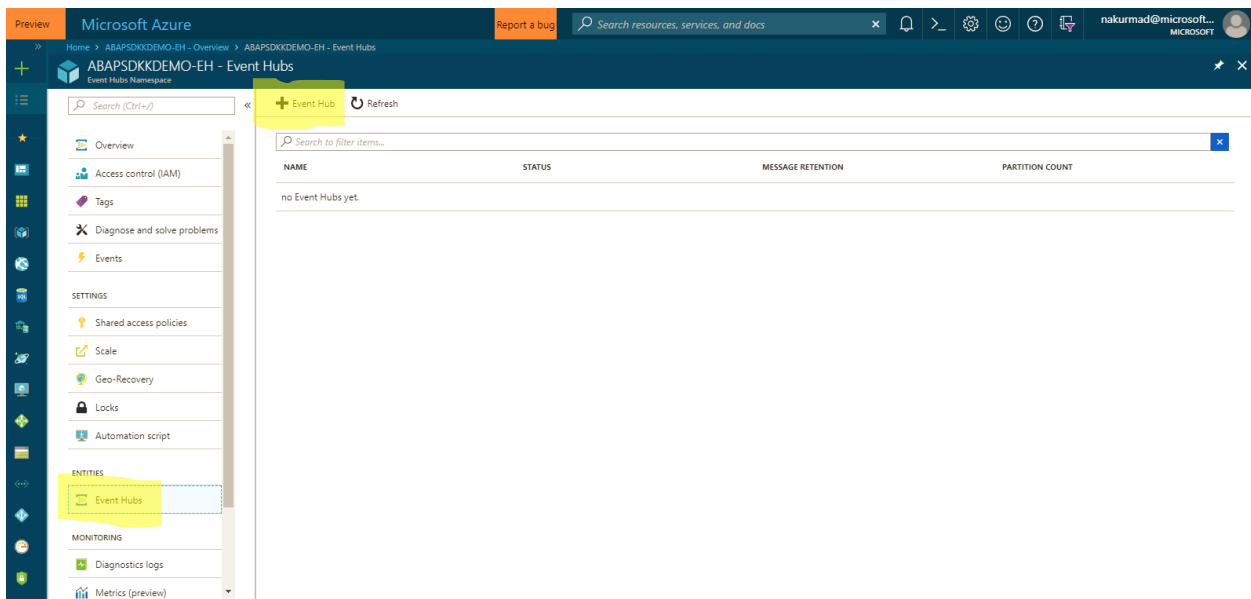


Once it is successfully created, navigate to your Event Hub either by selecting from Navigations pane or from Home screen.



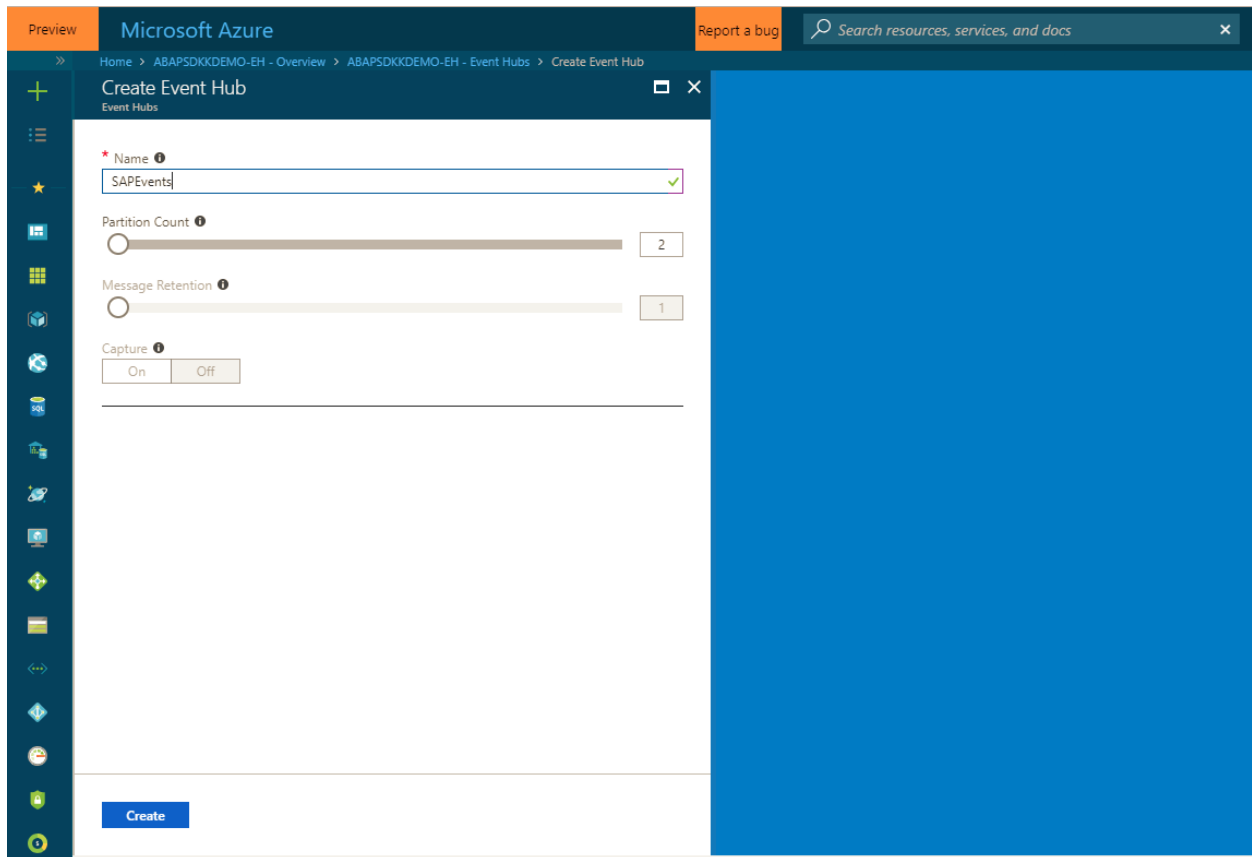
You can see multiple options in Azure Event Hubs. We are not discussing each of them in detail in this document. Please visit [Microsoft Azure Website](https://www.microsoft.com/azure/event-hubs/) for more details.

Once you have entered your Event Hubs Namespace. Now you must create a new Event Hub Entity by selecting **Event Hub** under entities.

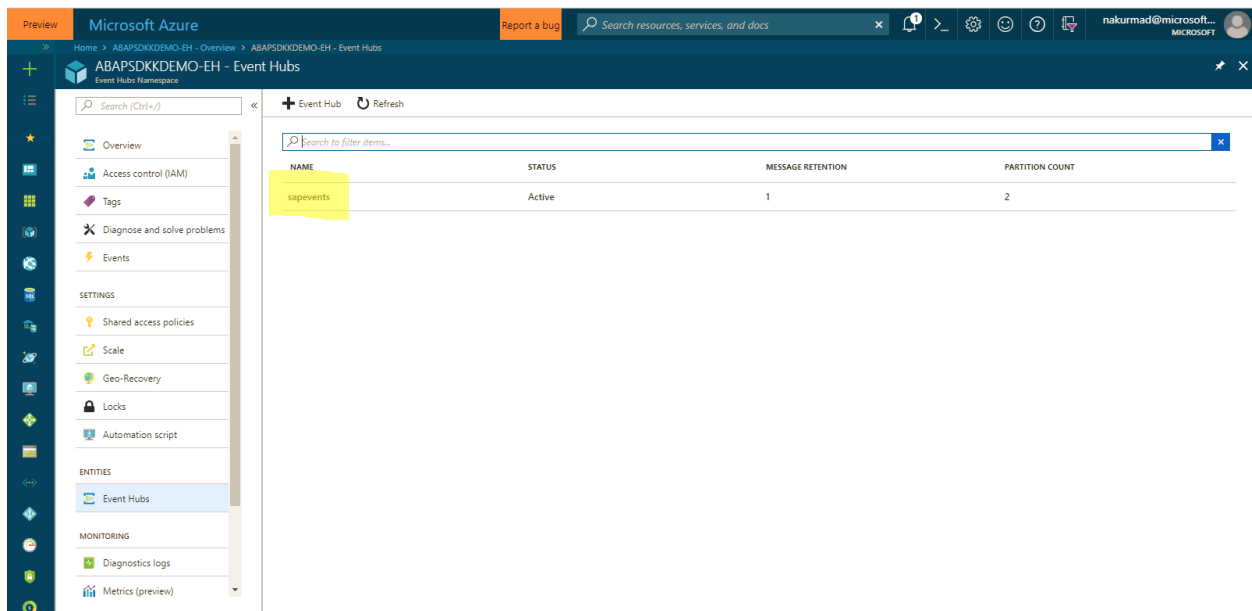


Create a new Event Hub with appropriate name like 'SAPEvents'. You can choose any name of your choice and **Create**.

We would post messages from our SAP system to this eventhub.

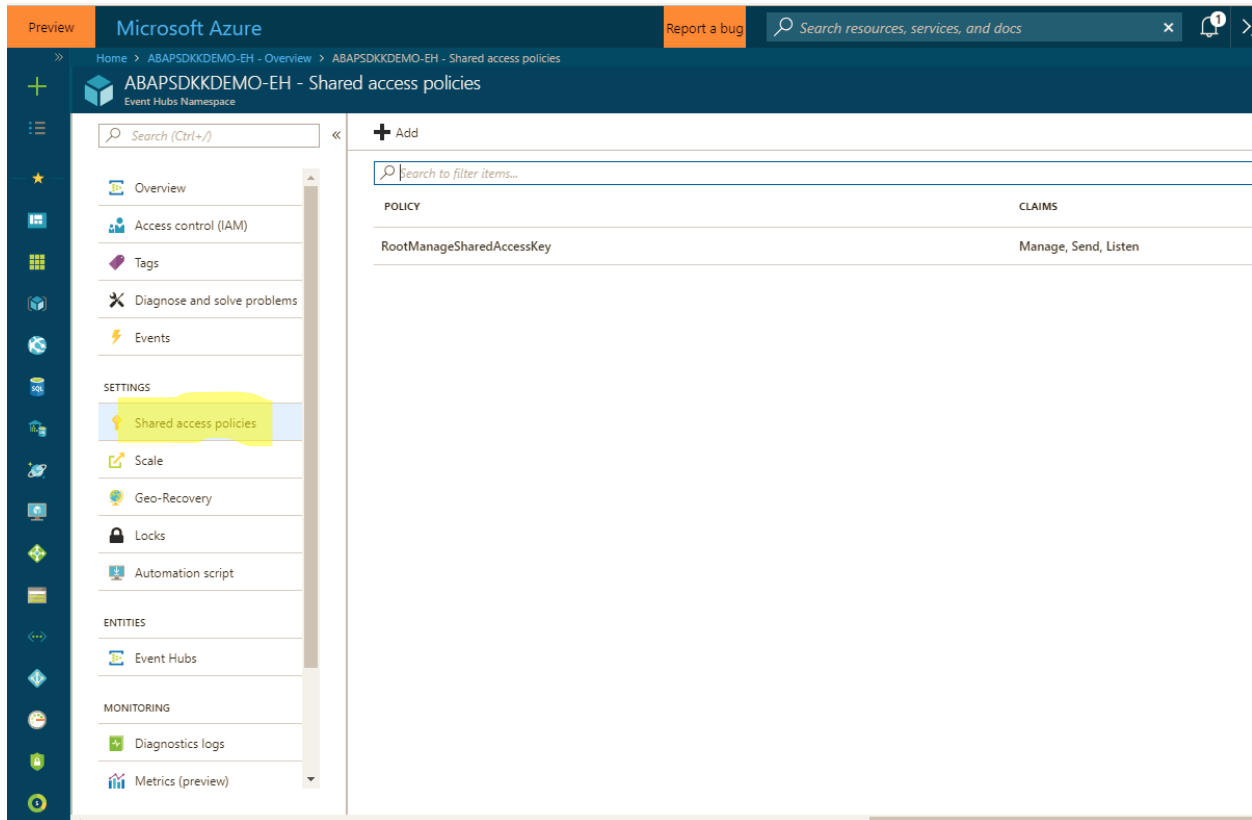


This will now create a new Event Hub in your Event Hubs namespace.

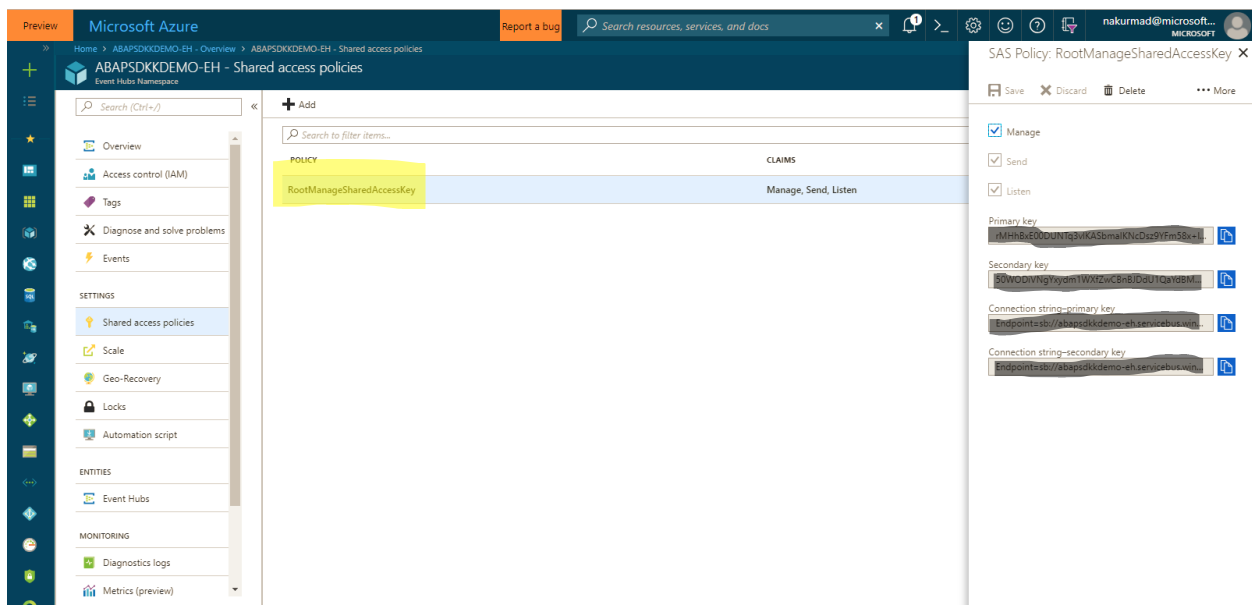


Once you are done with creating Event Hub. Now you must create Shared access Policy. By Default, you will have policy **'RootManageSharedAccessKey'** which will have complete access to Read, Write and Manage. In case you want to restrict users to only Read and Write, create your own policy accordingly.

Note: We will use the SAS key of this policy to access Azure Event Hub from SAP through HTTPS protocol.



Once you select the policy, this will display SAS keys that will be used to Authenticate Azure Event Hub. These are secret keys and not to be shared with anyone.



With this we are ready with the setup process in Azure portal. We will now configure and code in SAP system to send data from SAP system directly to Azure Event Hub.

How to send data from SAP to Azure Event hub?

Creation of RFC destination to Azure Event hub

Go to transaction SM59 in your SAP system and create new RFC destination of type 'G'. Maintain your Event hubs namespace endpoint in the Target host and Event Hub name in path prefix as shown below.

Target host: <Eventhub Namespace>.servicebus.windows.net

Port: 443

Path Prefix: /<Eventhub name>/messages

RFC Destination AZURE_ABAPSDKKDEMO-EH

Connection Test

RFC Destination: AZURE_ABAPSDKKDEMO-EH

Connection Type: HTTP Connection to External Serv Description

Description

Description 1: Connection to Azure Event Hub 'ABAPSDKKDEMO-EH'

Description 2:

Description 3:

Administration Technical Settings Logon & Security Special Options

Target System Settings

Target Host: ABAPSDKKDEMO-EH.servicebus.windows.net Service No.: 443

Path Prefix: /sapevents/messages

HTTP Proxy Options

Global Configuration

Proxy Host:

Proxy Service:

Proxy User:

Proxy PW Status: is initial

Proxy Password:

Now go to 'Logon & Security' tab and choose radio button SSL 'Active' and select SSL certificate 'DEFAULT SSL Client (Standard)'.

RFC Destination AZURE_ABAPSDKKDEMO-EH

Connection Test

RFC Destination: AZURE_ABAPSDKKDEMO-EH

Connection Type: G HTTP Connection to External Serv Description

Description

Description 1: Connection to Azure Event Hub 'ABAPSDKKDEMO-EH'

Description 2:

Description 3:

Administration | Technical Settings | **Logon & Security** | Special Options

Logon Procedure

Logon with User

Do Not Use a User

Basic Authentication

User:

PW Status: is initial

Logon with Ticket

Do Not Send Logon Ticket

Send Logon Ticket Without Target System Reference

Send Assertion Ticket for Dedicated Target System

System ID: Client:

Security Options

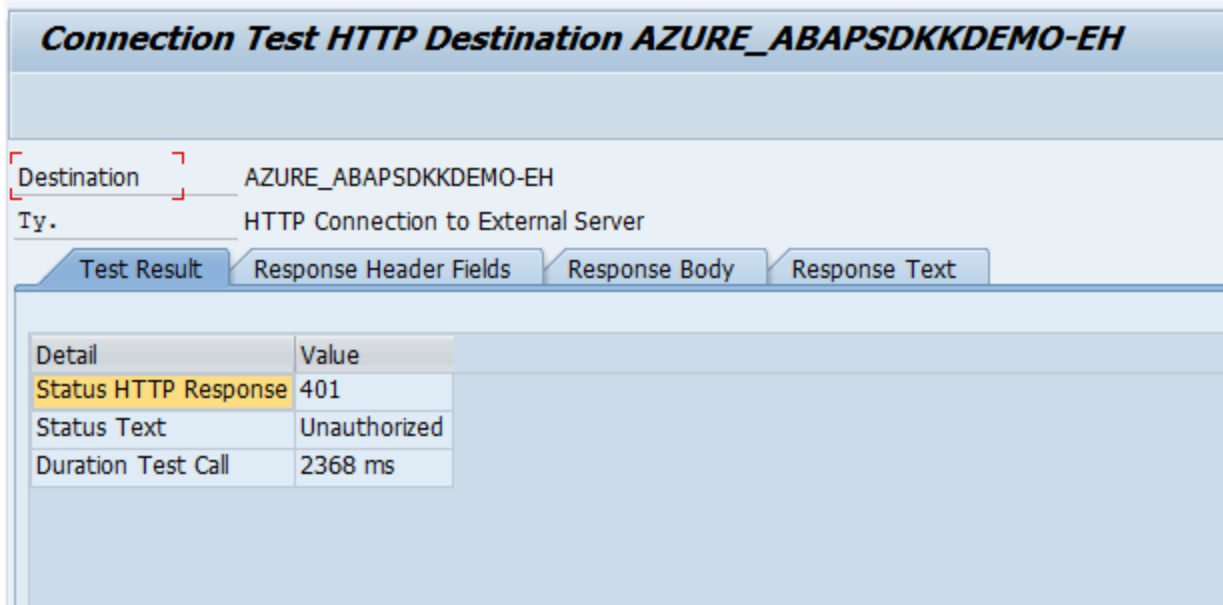
Status of Secure Protocol

SSL: Inactive Active

SSL Certificate: DEFAULT SSL Client (Standard) Cert. List

Authorization for Destination:

Do a connection test to make sure it is working. A popup might appear asking for user name and password, select Cancel button. Even if you get 401 Not authorized error, do not worry. you are still good and your RFC destination setup is complete.



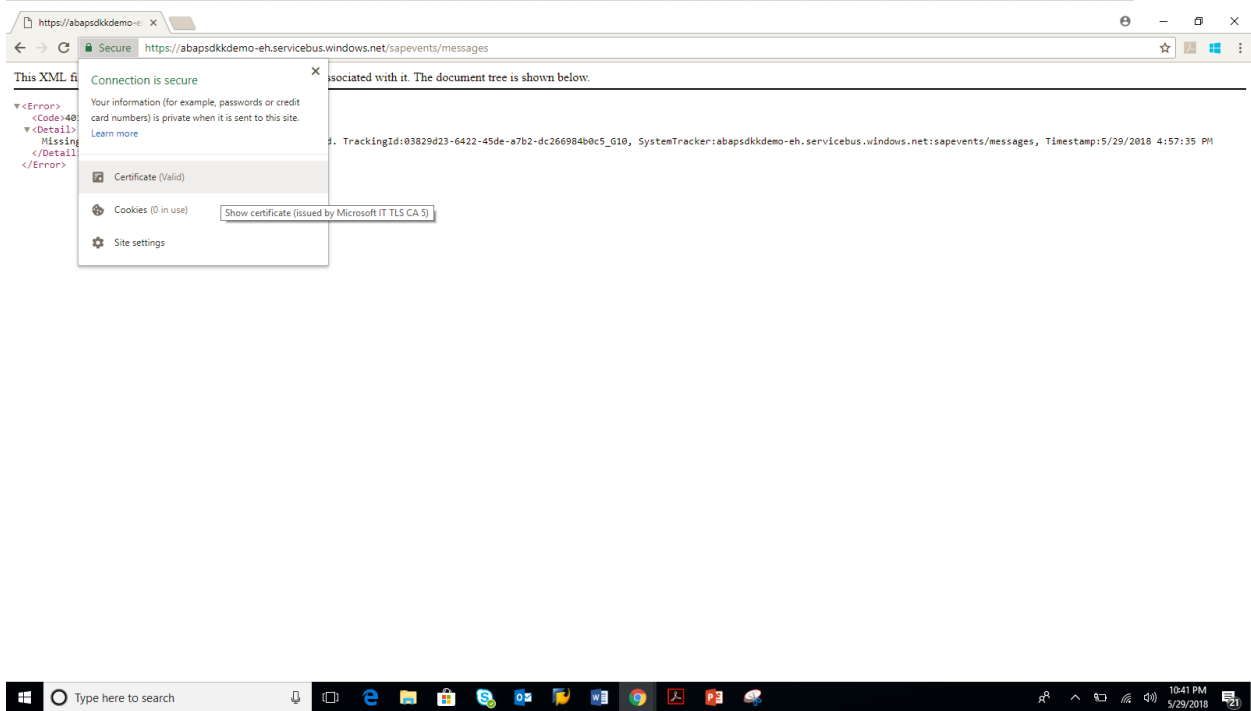
STRUST Setup

We need to import Microsoft’s Certificate and import in STRUST for SSL handshake between SAP system and Azure Eventhub over HTTPS protocol. To download the certificate, in your browser, go to URL with the hostname and path prefix you used for creating RFC destination.

<https://abapsdkkdemo-eh.servicebus.windows.net/sapevents/messages>

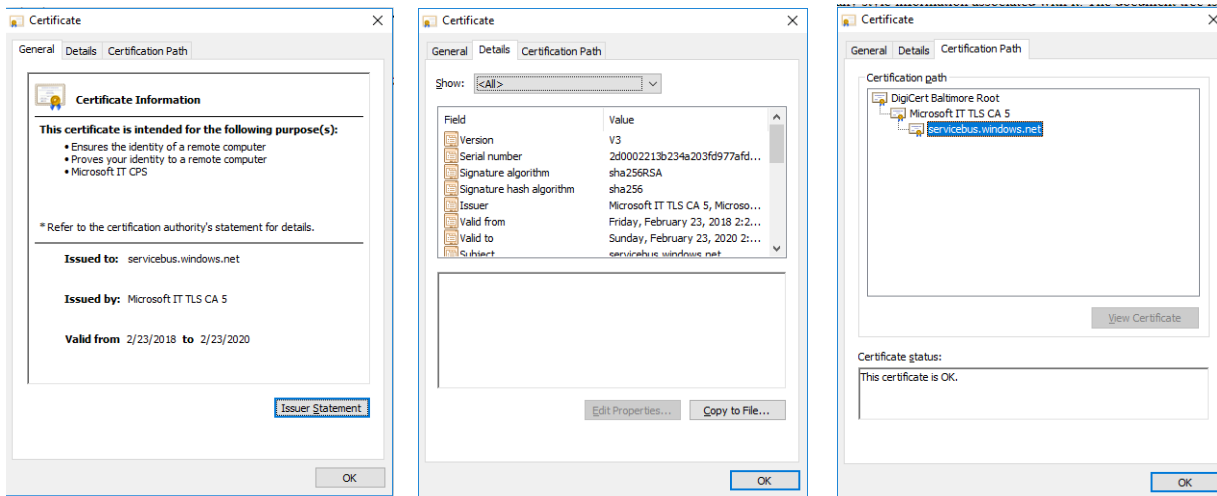


Click on the Lock symbol you find next to refresh button in Chrome browser and select **Certificate** to view the certificate used for communication



In the certificate, go to Details tab and Choose button ‘Copy to File’ to download the certificate to your local machine. Repeat the process and download all the certificate until root. In this case, you need to download three certificates.

1. Servicebus.windows.net
2. Microsoft IT TLS CA 5
3. DigiCert Baltimore Root



when all the certificates are downloaded, Go to STRUST transaction in your SAP system and Import all these certificates in DFAULT PSE.

Note: We are not going through the process of Importing certificates in STRUST in this document. It is straight forward, and your BASIS team can help you to do this activity.

Configuration

ABAP SDK has following main configuration tables and they need to be maintained. We will create a new Interface ID to establish connection between SAP system and target Azure Event Hub. A new Interface ID needs to be created for each Event Hub namespace.

Note: Currently all these Configurations tables must be maintained manually using SM30. We are developing a new Graphical Interface to simplify the configuration steps. This will be released with next versions. Keeping looking for Updates.

ZREST_CONFIG – Master Table for Interface ID Maintenance. You must define a new Interface name and maintain the RFC destination that was created for target Event hub.

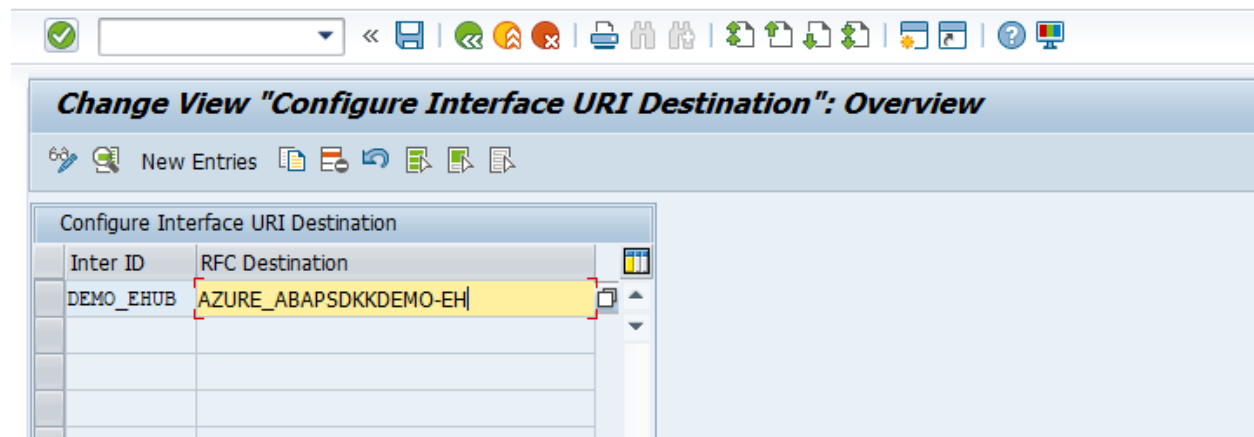
ZREST_CONF_MISC – This is an Interface Miscellaneous table which contains information on Alerts and re-processing of failed messages automatically.

ZADF_CONFIG – This is an Interface extension table. This stores data that is more specific to Azure Services like SAS keys, AAD secrets and processing Method.

ZADF_EHUB_POLICY – This is Event hub specific table that stores Azure Event hub policy details that shall be used during communication.

ZREST_CONFIG

Create a new Interface ID like 'DEMO_EHUB' and Maintain the RFC destination you created earlier.



ZREST_CONF_MISC

Create an entry in table 'ZREST_CONF_MISC' for the above interface Id 'DEMO_EHUB'.

Details of configuration:

- METHOD is 'POST'.
- MAX_RETRY is number of retry in case of service failure.
- EMAIL_ID is the email id for sending alerts.
- MAIL_BODY_TXT is Text Id to be maintained for the mail content.
- RETRY_METHOD is type of retrial (Regular '0' or exponential '1')

Table ZREST_CONF_MISC Display	
MANDT	300
INTERFACE ID	DEMO_EHUB
METHOD	POST
MAX RETRY	2
EMAIL ID	nakurmad@microsoft.com
MAIL BODY TXT	ERROR WHILE SENDING MESSAGE TO EVENTHUB
RETRY METHOD	0

ZADF_CONFIG

Create an entry in table 'ZADF_CONFIG' for the above interface Id 'DEMO_EHUB'.

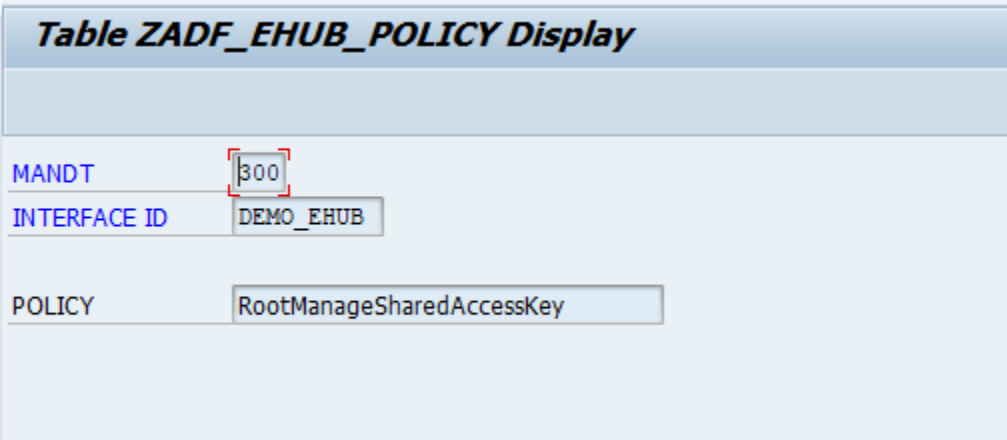
Details of configuration:

- INTERFACE_TYPE is 'EVENTHUB'.
- SAS_KEY is the shared access key. This you can retrieve from Azure portal. Check page 10 of this document. Primary key is the SAS key. Every time you generate a new SAS key in portal, you need to change in this config table.
- URI is left blank. This may be required for future versions.
- SERVICE_TYPE can be synchronous(S) or asynchronous(A)
- IS_TRY is a reprocessing flag, maintain as blank or 'X'. it can be configured for reprocessing in case of failure of services.

Table ZADF_CONFIG Display	
MANDT	300
INTERFACE ID	DEMO_EHUB
INTERFACE TYPE	EVENTHUB
SAS KEY	*****
URI	
SERVICE TYPE	S
IS TRY	X

ZADF_EHUB_POLICY

You need to maintain the policy name that you have created in Azure portal in this config table. This is required to generate token for authentication with Azure Event hubs. Refer Page 10 of this document to get the policy name. Azure creates default policy 'RootManageSharedAccessKey', but if you had created your own policy, then maintain that policy name here.



The screenshot shows the SAP Table ZADF_EHUB_POLICY Display. The table has three rows of data:

Table ZADF_EHUB_POLICY Display	
MANDT	300
INTERFACE ID	DEMO_EHUB
POLICY	RootManageSharedAccessKey

DEMO Program

Please refer to DEMO program 'ZADF_DEMO_AZURE_EVENTHUB' to send sample data from your SAP system to Azure Event hub.

View sent data in Azure Eventhub

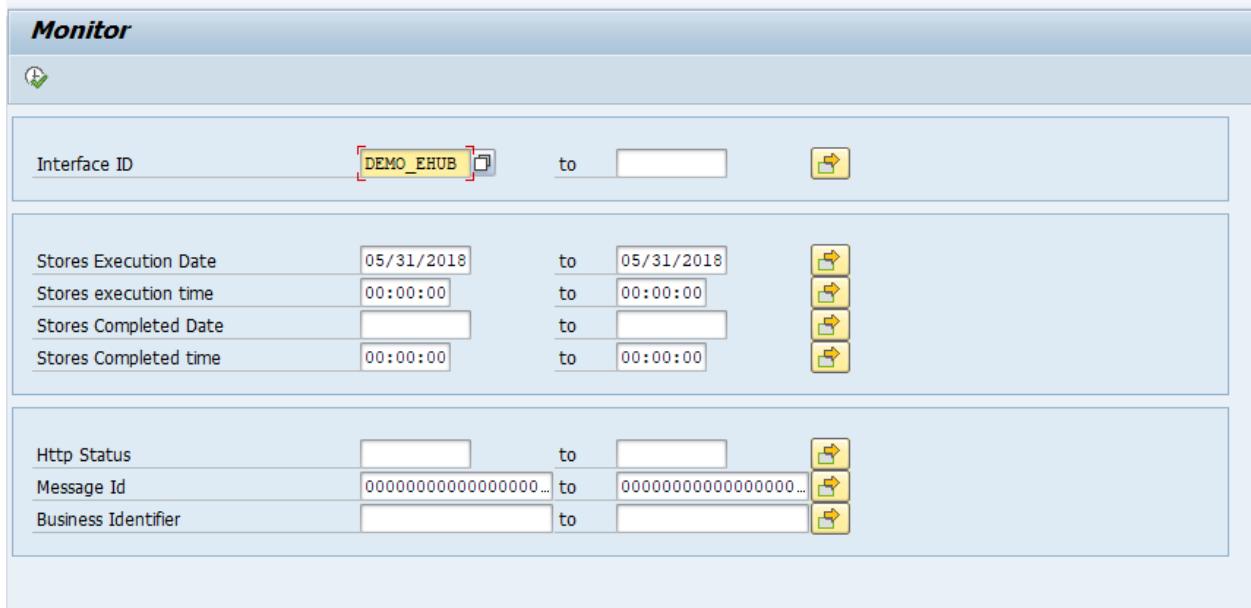
To view data in Azure Event hub, Install [Service bus explorer](#)

You can get more details on How to setup Service Bus explorer in [MSDN Blog](#)

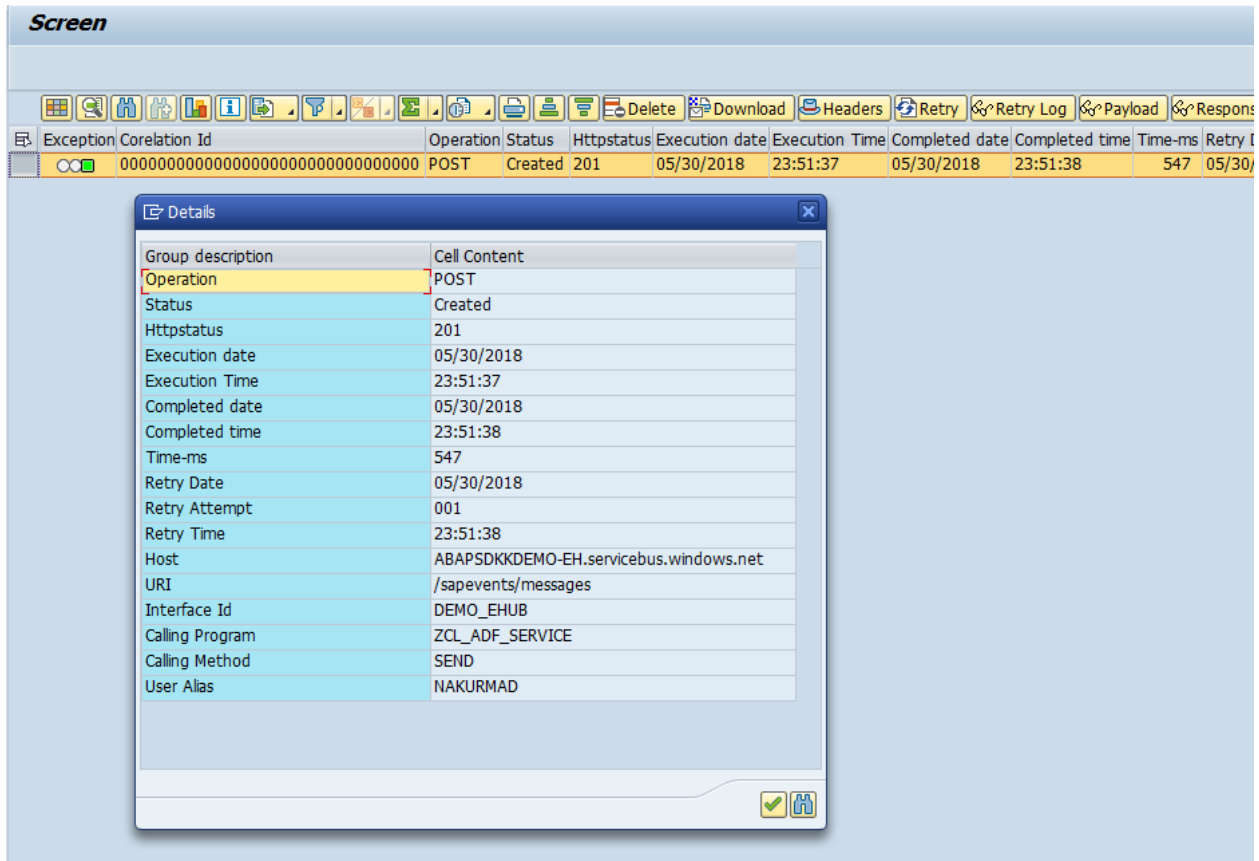
ABAP SDK Monitor

We have provided an Interface Monitor (Transaction ZREST_UTIL), using this monitor you can view history of all the messages that were posted to Azure Services. In case you have a scheduled a background job to post messages to Azure, you can view the statuses of the messages in this monitor. This Monitor can be used for troubleshooting and re-processing of the message as well.

Go to transaction ZREST_UTIL and provide your Interface ID in the selection screen and execute to view all the messages



In this monitor, you can view the status of the HTTPs message and its headers, response, payload and so on. In case of errors, you can also re-process the message from this tool.



Auto re-processing of failed messages

For auto-processing of messages in case of failures, you must schedule a background job for program 'ZREST_SCHEDULER' as a pre-requisite.