

Solace PubSub+ Distributed Tracing using self-managed OTEL Collector with Jaeger, Dynatrace, New Relic and DataDog

solace• PubSub+ Distributed Tracing

Demo 101 for dummies, experts
and everyone in between

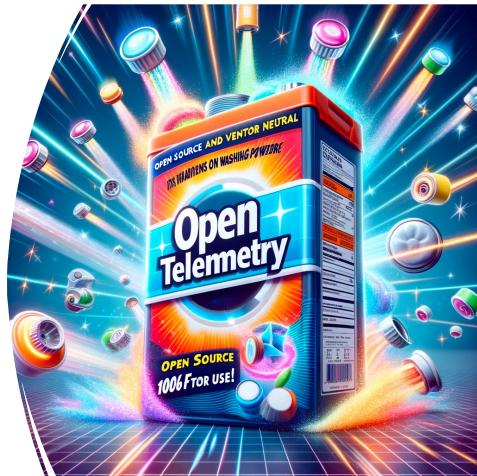


Table of Contents

1	Purpose	2
1.1	Application chain.....	2
1.2	Reference	2
2	Setup.....	2
2.1	Prerequisites	2
2.2	Configuration	2
2.3	Configure Broker	4
2.4	Jaeger.....	13
2.5	OTEL collector.....	14
3	Testing the application chain.....	16
3.1	Solace SDKPerf	16
3.2	Installation	16
4	Results	19
4.1	PMOTE.....	19
4.2	Jaeger.....	20
4.3	Dynatrace.....	22
4.4	New Relic	29
4.5	DataDog	34
4.6	Splunk	36
5	Resources	36

1 Purpose

Describe how to get OpenTelemetry (OTEL) traces from a simple application chain to Jaeger and/or other observability Cloud solutions using standard self-managed versions of OTEL collector and Jaeger.

1.1 Application chain

Solace SDKPerf **publisher** – Solace Broker **topic** – Solace Broker **queue1** and **queue2** – Solace SDKPerf **consumer**

1.2 Reference

Solace PubSub+ Distributed Tracing is an additional option for Solace PubSub+ brokers. This demo is based on work from my colleague Daniel Brunold (<https://github.com/dabgmx>, also well-known for his work on the Solace Prometheus Exporter, see <https://github.com/solacecommunity/solace-prometheus-exporter>)

For this demo Daniel got some good inspiration from the Solace Codelabs ‘Getting Started with Solace Distributed Tracing and Context Propagation’ at <https://codelabs.solace.dev/codelabs/dt-otel/>

2 Setup

2.1 Prerequisites

A Solace broker running in the PubSub+ Cloud platform or self-managed.

2.2 Configuration

For Solace PubSub+ Cloud brokers Distributed Tracing can be enabled on individual services by assigning a license to them. This license is available for multiple Connection Tiers.

The screenshot shows the 'Distributed Tracing' tab selected in the 'Account Details' section of the Solace PubSub+ Cloud interface. The 'Connection Tier' table shows usage statistics for different tracing tiers:

Connection Tier	In Use	Available	Limits
PubSub+ Distributed Tracing 250 Tier	6	44	50
PubSub+ Distributed Tracing 1K Tier	4	1	5
PubSub+ Distributed Tracing 10K Tier	0	1	1
PubSub+ Distributed Tracing 200K Tier	0	1	1

The 'Tracing Destination' section lists a single entry:

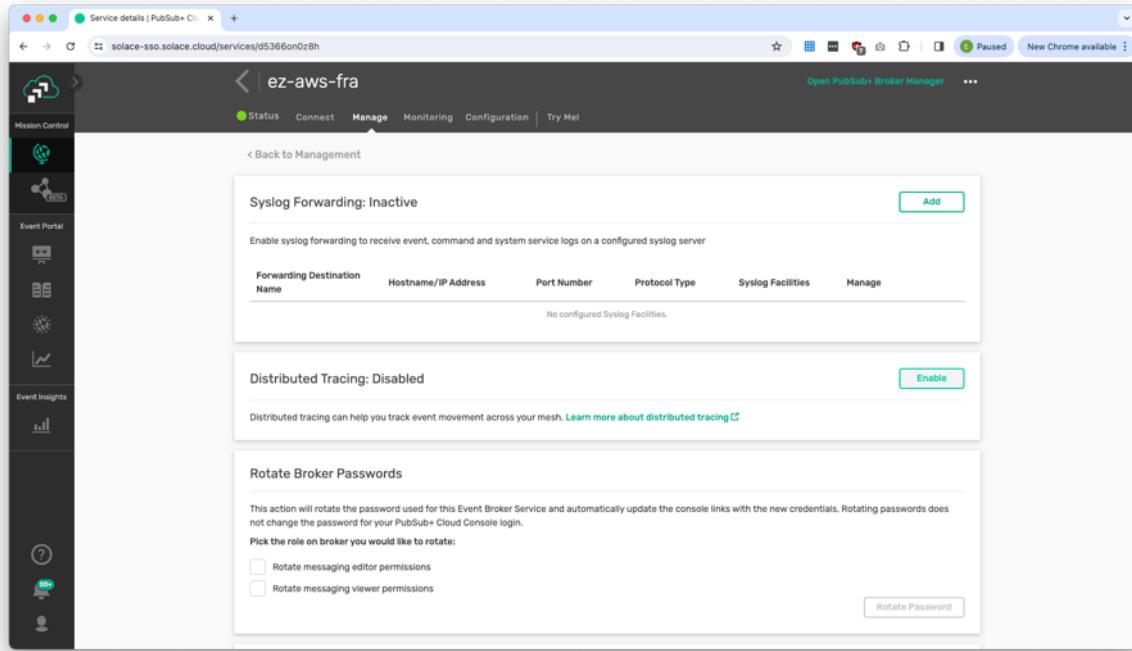
Name	Type	Associated Services
DataDog destination	Datadog	3 ***
	OTLP/gRPC	1 ***

The 'Where do you send your data?' section provides links to Datadog and another destination:

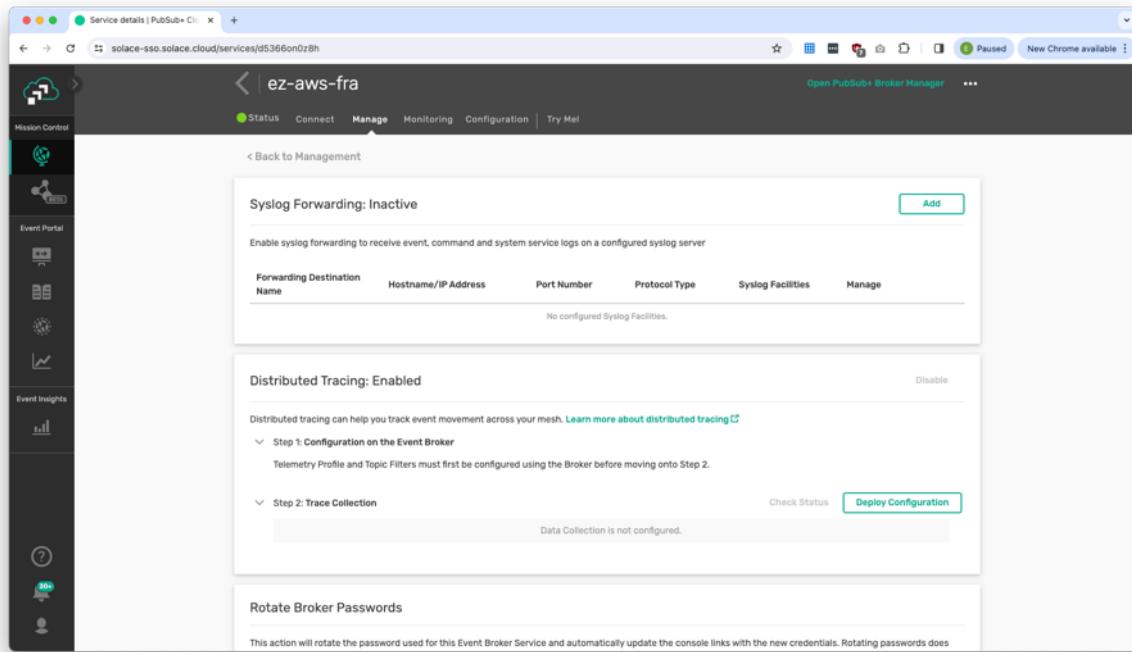
- Datadog**: Connect directly to a Datadog account or via your PubSub+ Insights subscription.
- Another destination?**: Send your tracing data to many other destinations using OpenTelemetry™ Protocols (OTLP).

In this demo you will enable Distributed Tracing but you will not use the [Deploy Configuration] option to deploy a managed OTEL collector as you will use a self managed OTEL collector with local Jaeger and optional other destinations.

In Mission Control go to Cluster Manager and select the service where you want to use Distributed Tracing. In the service overview click Manage then Advanced Options. In the Distributed Tracing section click [Enable]



Do not click [Deploy Configuration] as you will setup your own OTEL collector in this demo.



When enabled you'll find a section Distributed Tracing in the Status overview.

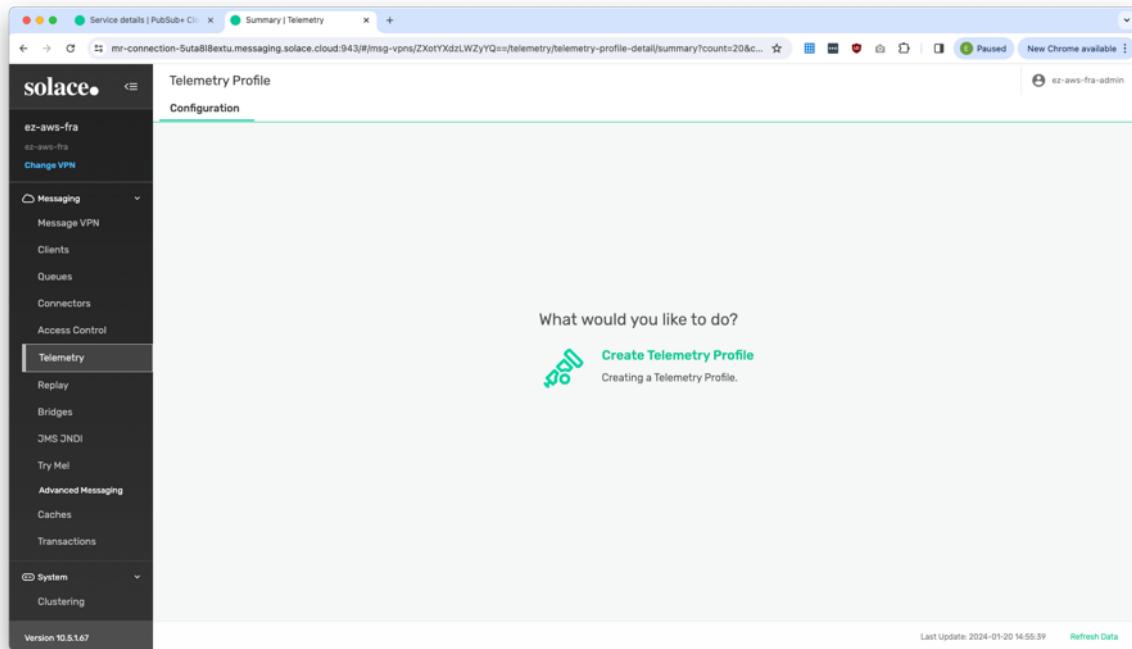
The screenshot shows the 'Status' tab in the Solace PubSub+ Broker Manager. On the left, there's a vertical sidebar with icons for Mission Control, Event Portal, Event Insights, and Scale Up. The main content area has tabs for Status, Connect, Manage, Monitoring, Configuration, and Try Me!. Under the Status tab, there's a 'Distributed Tracing' section with the sub-sub-section 'Distributed Tracing is enabled on this service.' and a 'View Tracing Details' button.

2.3 Configure Broker

Open PubSub+ Broker Manager and verify under Access Control that Basic Authentication is enabled and set to Type Internal database.

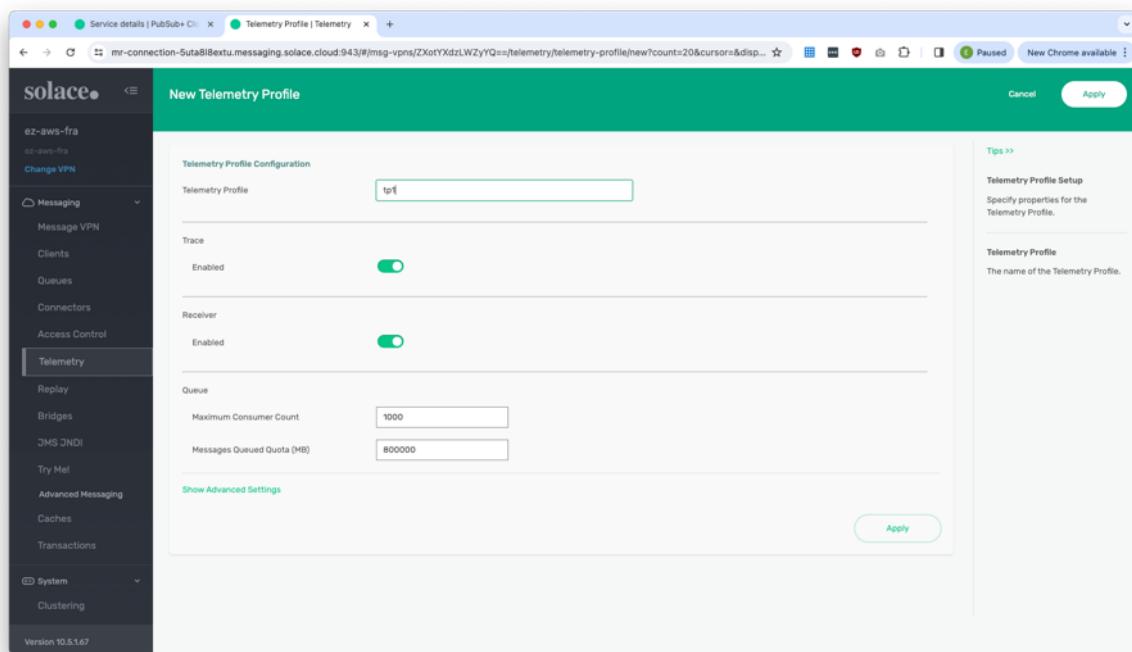
The screenshot shows the 'Client Authentication' settings page in the Solace PubSub+ Broker Manager. The left sidebar includes sections for Messaging (Message VPN, Clients, Queues, Connectors), Access Control, Telemetry, Replay, Bridges, JMS JNDI, Try Me!, Advanced Messaging (Caches, Transactions), and System (Clustering). The right pane shows the 'Client Authentication' tab selected, with sub-options for Settings, Authorization Groups, OAuth Profile, and Client Certificate Rules. Under 'Basic Authentication', the 'Type' dropdown is set to 'Internal database'. Other settings include RADIUS Domain (disabled), Client Certificate Authentication (disabled), Maximum Chain Depth (set to 8), Validate Certificate Dates (disabled), Allow API Provided Username (disabled), Username Source (Common Name), Enable Certificate Matching Rules (disabled), Revocation Check Mode (Allow Valid), Kerberos Authentication (disabled), Allow API Provided Username (disabled), OAuth Authentication (disabled), Default Profile Name (internal database), and Authorization Type (Internal database). A 'Show Advanced Settings' link is visible on the right, along with tips for interacting with the interface.

At Telemetry click Create a Telemetry Profile.



The screenshot shows the Solace Management Console interface. The left sidebar has a 'Telemetry' section selected. In the center, there is a message: 'What would you like to do?' followed by a green 'Create Telemetry Profile' button with the sub-instruction 'Creating a Telemetry Profile.' At the bottom right, it says 'Last Update: 2024-01-20 14:55:39' and 'Refresh Data'.

Use name "tp1" and check if Trace and Receiver are enabled then click [Apply]



The screenshot shows the 'New Telemetry Profile' configuration dialog. It includes fields for 'Telemetry Profile' (set to 'tp1'), 'Trace' (Enabled), 'Receiver' (Enabled), and 'Queue' settings ('Maximum Consumer Count' at 1000 and 'Messages Queued Quota (MB)' at 800000). There is also a 'Show Advanced Settings' link and an 'Apply' button. A 'Tips >>' section on the right provides information about Telemetry Profile Setup and the name of the profile.

Telemetry Profile is displayed with Client Profile Name and Queue Name #telemetry-tp1

The screenshot shows the Solace Management Console interface. On the left, there's a navigation sidebar with sections like 'ez-aws-fra', 'Change VPN', 'Messaging' (Message VPN, Clients, Queues, Connectors), 'Access Control' (Telemetry, Replay, Bridges, JMS JNDI, Try Mel, Advanced Messaging, Caches, Transactions), and 'System' (Clustering). The 'Telemetry' section is currently selected. The main content area is titled 'Telemetry Profile | tp1'. It has tabs for 'Summary', 'Settings', 'Receiver Connect ACLs', and 'Trace Filters'. Under 'Summary', it shows 'Trace Enabled: On', 'Trace Demo Mode: licensed', 'Receiver Enabled: On', 'Client Profile Name: #telemetry-tp1', and 'Queue Name: #telemetry-tp1'. There are buttons for 'Action' and 'Edit'. At the bottom, it shows 'Refresh Rate: 5s', 'Last Update: 2024-01-20 14:59:07', and a 'Refresh Data' button.

Click Edit at Receiver Connect ACLs to switch from Disallow to Allow and click [Apply]

The screenshot shows the 'Receiver Connect ACLs' configuration page for the 'tp1' telemetry profile. The left sidebar is identical to the previous screenshot. The main content area is titled 'Telemetry Profile | tp1' and has tabs for 'Summary', 'Settings', 'Receiver Connect ACLs', and 'Trace Filters'. Under 'Receiver Connect ACLs', it shows 'Client Connect Default Action: Allow'. Below this is a table for 'Exceptions' with a search bar 'Search by address' and a 'New' button. A tooltip for 'Client Connect Default Action' explains the difference between 'Allow' (allow client connection unless an exception is found) and 'Disallow' (disallow client connection unless an exception is found for it). At the bottom, it shows 'Last Update: 2024-01-20 14:59:54', 'Refresh Data', and a 'Show' dropdown set to 20.

The screenshot shows the Solace Management UI for a Telemetry Profile named 'tp1'. The left sidebar has 'Telemetry' selected. The main navigation bar includes 'Summary', 'Settings', 'Receiver Connect ACLs', and 'Trace Filters', with 'Trace Filters' highlighted and a red box around it. Below the navigation is a search bar and a table with one row labeled 'Trace Filter Name'. A green button at the top right labeled '+ Trace Filters' is also highlighted with a red box.

At Trace Filters click [+ Trace Filters] to add a filter with name "default" and set it to Enabled.

The screenshot shows the 'Edit Trace Filter Settings' dialog for a filter named 'default'. The left sidebar has 'Telemetry' selected. The dialog has tabs for 'Settings' and 'Actions'. Under 'Settings', the filter is named 'default' and is 'Enabled' (indicated by a green toggle switch). A 'Tips >' section provides information about enabling or disabling the trace filter. A 'Cancel' and 'Apply' button are at the top right.

Open this filter and create a subscription using the *greater than wildcard sign ">"*

The screenshot shows the 'Create Subscription' dialog for the 'default' trace filter. The left sidebar has 'Telemetry' selected. The dialog title is 'Create Subscription' and it shows '1Subscriptions'. A text input field contains '>'. A 'Create' button is at the bottom right. The status bar at the bottom indicates 'Last Update: 2024-01-20 10:02:19' and 'Refresh Data'.

Subscriptions | Telemetry

solace •

ez-aws-fra

ez-aws-fra

Change VPN

Messaging

- Message VPN
- Clients
- Queues
- Connectors

Access Control

Telemetry

- Replay
- Bridges
- JMS JNDI
- Try Me!
- Advanced Messaging
- Caches
- Transactions

System

- Clustering

Version 10.5.167

Trace Filters | default

Subscriptions

Search by subscription

Filter Subscription

Action + Subscription

Trace Filter Subscription Syntax

SMF

Last Update: 2024-01-20 15:02:50 Refresh Data Show 20

At Access Control go to Client Usernames and click [+ Client Username] to create a client username "trace" with password "trace123" and "#telemetry-tp1" profiles:

Client Usernames | Access C

solace •

ez-aws-fra

ez-aws-fra

Change VPN

Messaging

- Message VPN
- Clients
- Queues
- Connectors

Access Control

- Telemetry
- Replay
- Bridges
- JMS JNDI
- Try Me!
- Advanced Messaging
- Caches
- Transactions

System

- Clustering

Version 10.5.167

Client Authentication Client Profiles ACL Profiles Client Usernames

Search by name

Action + Client Username

Client Username	Client Profile	ACL Profile	Enable	Subscription Manager	Dynamic
client-username	#client-profile	#acl-profile	Yes	No	No
default	default	default	No	No	No
solace-cloud-client	default	default	Yes	No	No

Last Update: 2024-01-20 15:04:23 Refresh Data Show 20

Screenshot of the Solace Management Center web interface showing the creation of a new Client Username.

The left sidebar shows the navigation menu under the "Access Control" section, including "ez-aws-fra", "Change VPN", "Messaging" (Message VPN, Clients, Queues, Connectors), "Access Control" (selected), "Telemetry", "Replay", "Bridges", "JMS JNDI", "Try Me!", "Advanced Messaging", "Caches", "Transactions", "System" (Clustering), and "Version 10.5.167".

The main content area shows the "Client Usernames" tab selected. A modal dialog titled "Create Client Username" is open, showing a search bar and a list of existing client usernames: "client-username", "default", and "solace-cloud-client". The "Client Username" field contains "trace".

Enable	Subscription Manager	Dynamic
Yes	No	No
No	No	No
Yes	No	No

Buttons at the bottom of the modal include "Cancel" and "Create".

Screenshot of the Solace Management Center web interface showing the editing of a Client Username settings.

The left sidebar shows the navigation menu under the "Access Control" section, including "ez-aws-fra", "Change VPN", "Messaging" (Message VPN, Clients, Queues, Connectors), "Access Control" (selected), "Telemetry", "Replay", "Bridges", "JMS JNDI", "Try Me!", "Advanced Messaging", "Caches", "Transactions", "System" (Clustering), and "Version 10.5.167".

The main content area shows the "Edit Client Username Settings" dialog for "trace". The "Enable" switch is turned on. The "New Password" and "Confirm Password" fields both contain "*****". The "Client Profile" dropdown is set to "#telemetry-tpt". The "ACL Profile" dropdown is also set to "#telemetry-tpt". The "Guaranteed Endpoint Permission Override" switch is turned off. The "Subscription Manager" switch is turned off.

A "Tips >>" link and an "ACL Profile" note are visible on the right side of the dialog.

Create two or more queues (queue-trace1, queue-trace2, ...) with subscription "demo/trace". These are the messaging queues.

The screenshot shows the Solace Management Center interface. On the left, a sidebar menu is open under the 'Messaging' section, with 'Queues' selected. The main area displays a list of existing queues, including 'telemetry-tpt1'. A modal dialog titled 'Create Queue' is open in the center, prompting for a 'Queue Name' which is set to 'queue-trace1'. At the bottom right of the dialog is a green 'Create' button. The background shows a summary table with columns for 'Messages', 'Consumers', 'Reply State', and 'Durable', with values 800,000, 0, N/A, and Yes respectively. The top right corner of the screen shows the status 'Paused' and 'New Chrome available'.

The screenshot shows the 'Edit Queue Settings' dialog for the 'queue-trace1' queue. The dialog has a green header bar with 'Cancel' and 'Apply' buttons. The main form contains several configuration fields: 'Incoming' and 'Outgoing' toggles are both turned on; 'Access Type' is set to 'Exclusive'; 'Messages Queued Quota (MB)' is set to 5000; 'Owner' is listed as 'ez-aws-fra'; 'Non-Owner Permission' is set to 'Consume'; and 'Maximum Consumer Count' is set to 1000. On the right side of the dialog, there are 'Show Advanced Settings' and 'Owner' tips. The sidebar on the left remains the same as the previous screenshot, showing the 'Queues' section is still selected.

Screenshot of the Solace Management UI showing the Queues page.

The left sidebar shows the navigation menu with the 'Queues' option selected under 'Messaging'.

The main table displays the following data:

Queue Name	Incoming	Outgoing	Access Type	Partition Count	Messages Queued (%)	Messages Queued (msgs)	Messages Queued (MB)	Messages Queued Quota	Consumers	Replay State	Durable
telemetry-tp1	Off	On	Non-Exclusive	0	0	0	800.000	0	N/A	Yes	
queue-trace1	On	On	Exclusive	0	0	0	5.000	0	N/A	Yes	
queue-trace2	On	On	Exclusive	0	0	0	5.000	0	N/A	Yes	

Page navigation: Page First Next > Last Update: 2024-01-20 15:10:19 Refresh Data Show 20

Screenshot of the Solace Management UI showing the Create Subscription dialog.

The left sidebar shows the navigation menu with the 'Queues' option selected under 'Messaging'.

The dialog title is 'Create Subscription' and it shows 1 Subscriptions.

The subscription topic is listed as 'demo/trace'.

Instructions: Add new subscription by pressing Enter in topic field or clicking [Add New](#).

Buttons: Cancel, Create.

Right panel: Action, Subscription, Created by Management API.

Page navigation: Page First Last > Last Update: 2024-01-20 15:10:36 Refresh Data Show 20

The screenshot shows the Solace Management UI interface. The left sidebar contains navigation links for 'ez-aws-fra' (Change VPN, Messaging, Clients, Queues, Connectors, Access Control, Telemetry, Replay, Bridges, JMS JNDI, Try Me!, Advanced Messaging, Caches, Transactions, System, Clustering), and 'Version 10.5.167'. The main content area is titled 'Subscriptions | queue-trace1' and includes tabs for Summary, Settings, Subscriptions (selected), Consumers, Messages Queued, and Stats. A search bar 'Search by topic' is present. The 'Subscriptions' table lists one entry: 'demo/trace' (Topic, Action: Created by Management API, Yes). The bottom of the screen shows pagination (Page, First, Next >), last update time (Last Update: 2024-01-20 15:10:56), refresh data button, and a 'Show' dropdown set to 20.

Configuration of the broker is now done.

2.4 Jaeger

Download Jaeger from here: <https://www.jaegertracing.io/download/>

Example for MacOS: jaeger-1.53.0-darwin-amd64.tar.gz

Installation

```
mkdir -p ~/jaeger  
cd ~/jaeger  
tar xzvf <your download directory>/jaeger-1.53.0-darwin-amd64.tar.gz
```

Gives output like:

```
x jaeger-1.53.0-darwin-amd64/  
x jaeger-1.53.0-darwin-amd64/example-hotrod  
...  
x jaeger-1.53.0-darwin-amd64/jaeger-ingester  
x jaeger-1.53.0-darwin-amd64/jaeger-query
```

Remove extended attributes from extracted files to avoid MacOS popup warnings on downloaded (executable) files, assuming you are allowed to administer the Mac you are working on:

```
xattr -rc '~/jaeger'
```

If necessary, add sudo.

Start

```
cd ~/jaeger/jaeger-1.53.0-darwin-amd64/  
.jaeger-all-in-one  
# Or detached:  
#nohup ./jaeger-all-in-one > /dev/null 2>&1 &
```

To stop kill the process with Control-C.

2.5 OTEL collector

Download from here:

<https://github.com/open-telemetry/opentelemetry-collector-releases/releases/>

Example for MacOS ARM: otelcol-contrib_0.96.0_darwin_arm64_darwin_arm64.tar.gz

Installation

```
mkdir -p ~/otelcol/otelcol-contrib_0.96.0_darwin_arm64  
cd ~/otelcol/otelcol-contrib_0.96.0_darwin_arm64  
tar xzvf <your download directory>/otelcol-contrib_0.96.0_darwin_arm64_darwin_arm64.tar.gz
```

Gives output like:

```
x LICENSE  
x README.md  
x otelcol-contrib
```

Prepare config files here:

```
cd ~/otelcol
```

Ping for IP address (what is preferred method to obtain -static- IP address?):

```
ping mr-connection-5uta8l8extu.messaging.solace.cloud
```

Added a custom hostname ez-dt.messaging.solace.cloud

Example for one broker: otel-collector-config-single.yaml:

<TODO: add yaml>

When working with multiple brokers define additional receivers and include in service/receivers:

```
receivers:  
solace/broker1:  
  broker: [1.2.3.4:5671]  
  max_unacknowledged: 500  
  auth:  
    sasl_plain:  
      username: trace  
      password: trace123  
  queue: queue://#telemetry-tp1  
  tls:  
    insecure: false  
    insecure_skip_verify: true  
  
solace/broker2:  
  broker: [4.3.2.1:5671]  
  max_unacknowledged: 500  
  auth:  
    sasl_plain:  
      username: trace  
      password: trace123  
  queue: queue://#telemetry-tp1  
  tls:  
    insecure: false  
    insecure_skip_verify: true  
  
service:  
telemetry:  
  logs:  
    level: "debug"  
pipelines:  
  traces:  
    receivers: [solace/broker1, solace/broker2]  
    processors: [batch]  
    exporters: [logging, otlp/jaeger]
```

To get rid of other Apple messages like "can't be opened because Apple cannot check it for malicious software." , assuming you are allowed to administer the Mac you are working on:

```
sudo spctl --master-disable
```

Start

```
cd ~/otelcol/otelcol-contrib_0.96.0_darwin_arm64  
../otelcol-contrib --config=../otel-collector-config-single.yaml  
# Or detached:  
#nohup ./otelcol-contrib --config=../otel-collector-config-single.yaml > /dev/null 2>&1 &
```

To stop kill the process with Control-C.

In the Broker verify that the Telemetry queue #telemetry-tp1 has a (1) consumer at Consumers:

Queue Name	Incoming	Outgoing	Access Type	Partition Count	Messages Queued (%)	Messages Queued (msgs)	Messages Queued (MB)	Messages Overrun Quota	Consumers	Replay State	Durable
#telemetry-tp1	Off	On	Non-Exclusive	0	<div style="width: 0%;"></div>	0	0	800,000	1	N/A	Yes
queue-trace1	On	On	Exclusive	0	<div style="width: 0%;"></div>	0	0	5,000	0	N/A	Yes
queue-trace2	On	On	Exclusive	0	<div style="width: 0%;"></div>	0	0	5,000	0	N/A	Yes

3 Testing the application chain

3.1 Solace SDKPerf

In this demo you will use Solace SDKPerf, a general purpose testing tool with support for OpenTelemetry. You can find information about SDKPerf at

<https://docs.solace.com/API/SDKPerf/SDKPerf.htm> and downloads at

https://solace.com/downloads/?fwp_downloads_types=other

On MacOS you can for example use the Java version: sdkperf-jcsmp-8.4.14.10.zip or sdkperf-mqtt-8.4.15.5.zip

3.2 Installation

```
mkdir -p ~/sdkperf
cd ~/sdkperf
tar xzvf <your download directory>/sdkperf-jcsmp-8.4.14.10.zip
tar xzvf <your download directory>/sdkperf-mqtt-8.4.15.5.zip
```

Gives output like:

```
x sdkperf-jcsmp-8.4.14.10/
x sdkperf-jcsmp-8.4.14.10/lib/
...
x sdkperf-jcsmp-8.4.14.10/sdkperf_java.bat
x sdkperf-jcsmp-8.4.14.10/sdkperf_java.sh
```

And similar for the MQTT version

In Broker Manager > Messaging > Acces Control > Client Usernames create user user-demo with password default.

The screenshot shows the Solace Management Center interface. On the left, there's a navigation sidebar with various options like Messaging, Access Control, Telemetry, and System. In the center, under 'Access Control', a modal window titled 'Create Client Username' is open. It has a search bar at the top and a list of existing client usernames: 'client-username', 'default', 'solace-cloud-client', and 'trace'. Below this, a text input field is populated with 'user-demo'. At the bottom right of the modal are 'Cancel' and 'Create' buttons. To the right of the modal, a table lists client username settings for 'user-demo': 'Enable' (Yes), 'Subscription Manager' (No), and 'Dynamic' (No). The table has three rows, each with the same values. At the bottom of the page, there are pagination controls ('Page First Last') and a timestamp ('Last Update: 2024-01-20 10:40:05').

This screenshot shows the 'Edit Client Username Settings' dialog for 'user-demo'. The left side of the dialog contains fields for 'New Password' and 'Confirm Password', both set to '*****'. Below these are dropdowns for 'Client Profile' (set to 'default') and 'ACL Profile' (set to 'default'). There are also toggle switches for 'Guaranteed Endpoint Permission Override' and 'Subscription Manager', both of which are off. On the right side, there's a 'Password' section with a note: 'The password for the Client Username. It is valid for the client username to have no password.' At the top right of the dialog are 'Cancel' and 'Apply' buttons.

Publish a message and receive it from 2 queues:

```
cd ~/sdkperf/sdkperf-jcsmp-8.4.14.10
```

Using Distributed Tracing from/to SDKPerf

<https://solace.community/discussion/1633/distributed-tracing-context-propagation>

With default user solace-cloud-client and initial auto-generated hostname

Can add -md flag to dump message, or -tmd to dump trace message (sort of works does drop an error)

```
./sdkperf_java.sh -cip=tcp://mr-connection-5uta8l8extu.messaging.solace.cloud:55443 -cu=solace-cloud-client@ez-aws-fra -cp=deun1l905ashrflooldf1qhrfg -ptl='demo/trace' -sql='queue-trace1,queue-trace2' -mt=persistent -mn=1 -mr=1 -msa=32768 -q -tcc -tcrc -tcep="http://localhost:4317"
```

Run repeatedly every 10 seconds

```
while true; do ./sdkperf_java.sh -cip=tcps://mr-connection-5uta8l8extu.messaging.solace.cloud:55443 -cu=solace-cloud-client@ez-aws-fra -cp=deun1l905ashrflooldf1qhrfg -ptl='demo/trace' -sql='queue-trace1,queue-trace2' -mt=persistent -mn=1 -mr=1 -msa=32768 -q -tcc -tcrc -tecip="http://localhost:4317"; sleep 10; done
```

With default user solace-cloud-client and initial auto-generated hostname

```
./sdkperf_java.sh -cip=tcps://mr-connection-5uta8l8extu.messaging.solace.cloud:55443 -cu=solace-cloud-client@ez-aws-fra -cp=deun1l905ashrflooldf1qhrfg -ptl='demo/trace' -sql='queue-trace1,queue-trace2' -mt=persistent -mn=1 -mr=1 -msa=32768 -q
```

With default user solace-cloud-client and additional created hostname

```
./sdkperf_java.sh -cip=tcps://ez-dt.messaging.solace.cloud:55443 -cu=solace-cloud-client@ez-aws-fra -cp=deun1l905ashrflooldf1qhrfg -ptl='demo/trace' -sql='queue-trace1,queue-trace2' -mt=persistent -mn=1 -mr=1 -msa=32768 -q
```

With default user solace-cloud-client and IP address (Dynamic? Going round between 18.159.178.64, 18.153.239.155, ... How to find these, and/or create static?)

```
./sdkperf_java.sh -cip=tcps://18.159.178.64:55443 -cu=solace-cloud-client@ez-aws-fra -cp=deun1l905ashrflooldf1qhrfg -ptl='demo/trace' -sql='queue-trace1,queue-trace2' -mt=persistent -mn=1 -mr=1 -msa=32768 -q
```

With created user user-demo

```
./sdkperf_java.sh -cip=tcps://mr-connection-5uta8l8extu.messaging.solace.cloud:55443 -cu=user-demo@ez-aws-fra -cp=default -ptl='demo/trace' -sql='queue-trace1,queue-trace2' -mt=persistent -mn=1 -mr=1 -msa=32768 -q
```

For MQTT

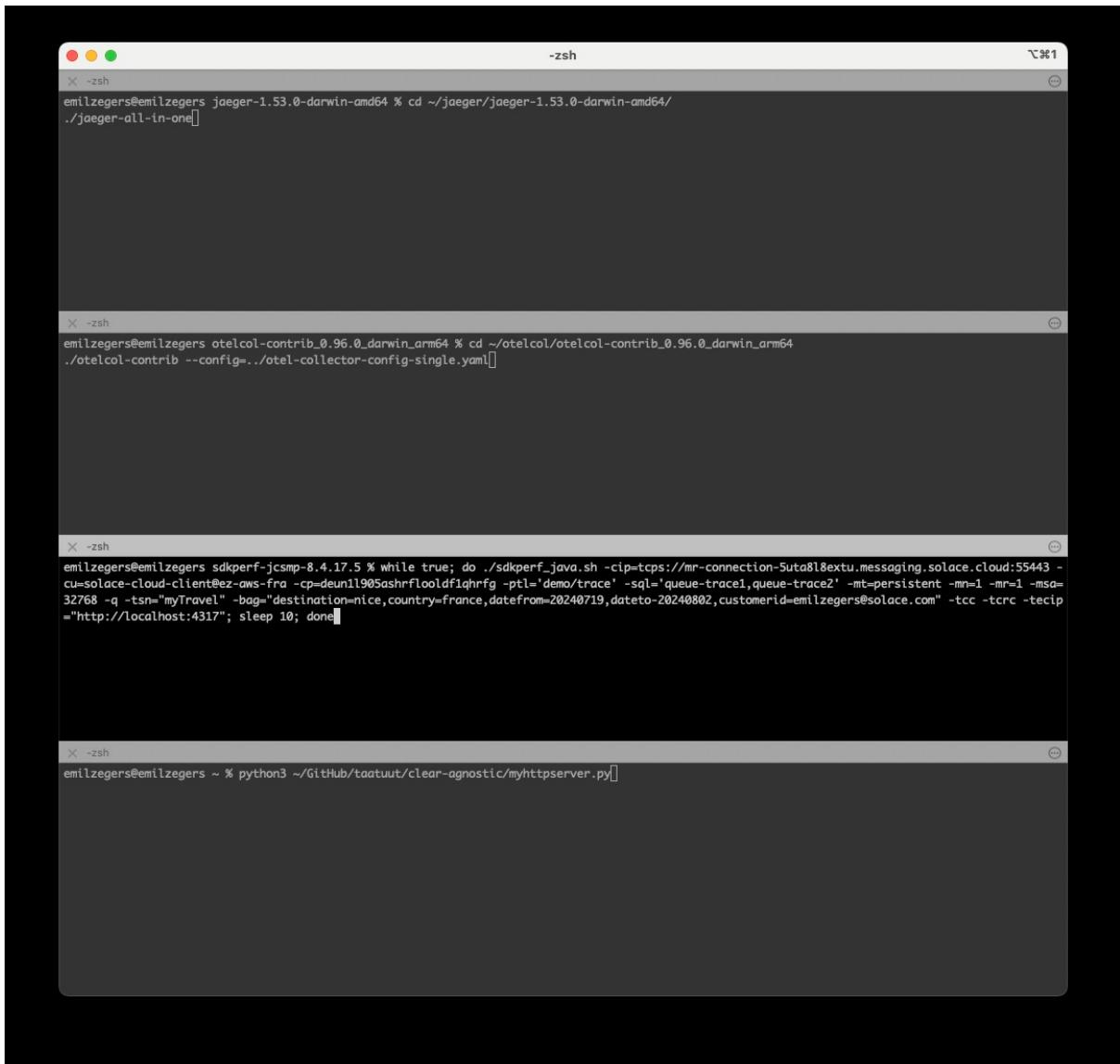
```
./sdkperf_mqtt.sh -cip=ssl://ez-dt.messaging.solace.cloud:8883 -cu=solace-cloud-client@ez-aws-fra -cp=deun1l905ashrflooldf1qhrfg -ptl='demo/trace' -sql='queue-trace1,queue-trace2' -mpq=1 -msq=1 -mn=1 -mr=1 -msa=32768 -q -tcc -tcrc -tecip="http://localhost:4317"
```

MQTT5

```
./sdkperf_mqtt5.sh -cip=ssl://ez-dt.messaging.solace.cloud:8883 -cu=solace-cloud-client@ez-aws-fra -cp=deun1l905ashrflooldf1qhrfg -ptl='demo/trace' -sql='queue-trace3' -mpq=1 -msq=1 -mn=1 -mr=1 -msa=32768 -q -tcc -tcrc -tecip="http://localhost:4317"
```

<https://docs.solace.com/API/SDKPerf/Command-Line-Options.htm>

<https://docs.solace.com/API/SDKPerf/Example-Commands.htm>



Can use something like iTerm to have all terminals together (jaeger, otelcollector, sdkperf and pmotel from top to bottom).

4 Results

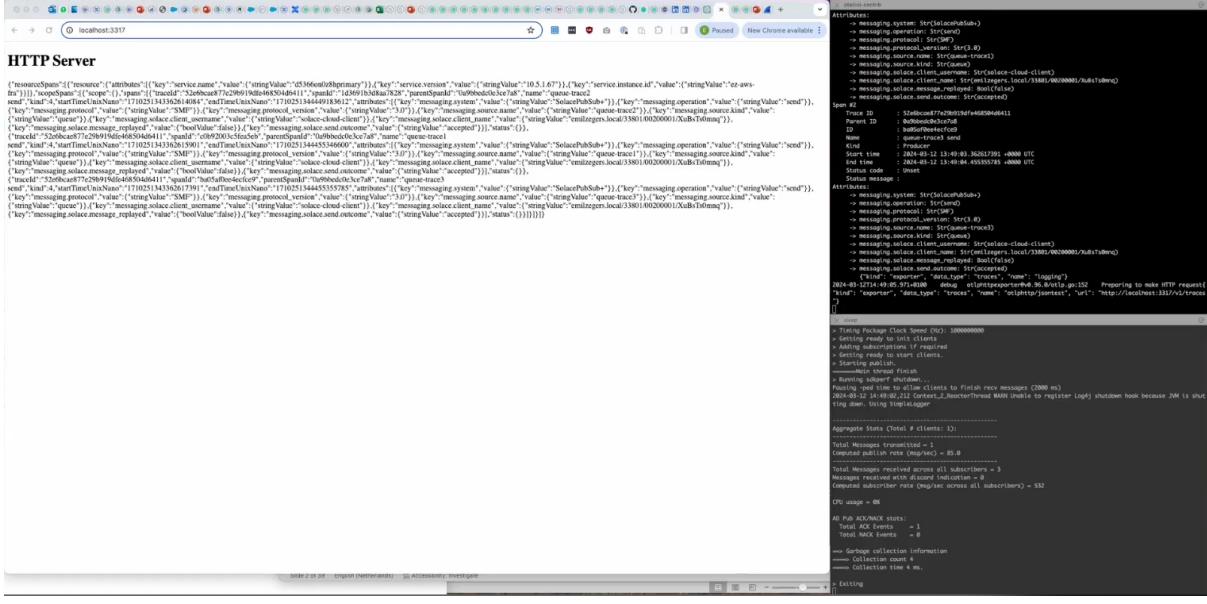
4.1 PMOTEL

Poor Man's OTEL endpoint, a Python script that processes POST requests from otel collector exporter with metrics in protobuf, (gzipped) JSON or something else and just displays the data received. See `myhttpserver.py` script, and relevant configuration in otel collector yaml file.

Example configuration for JSON:

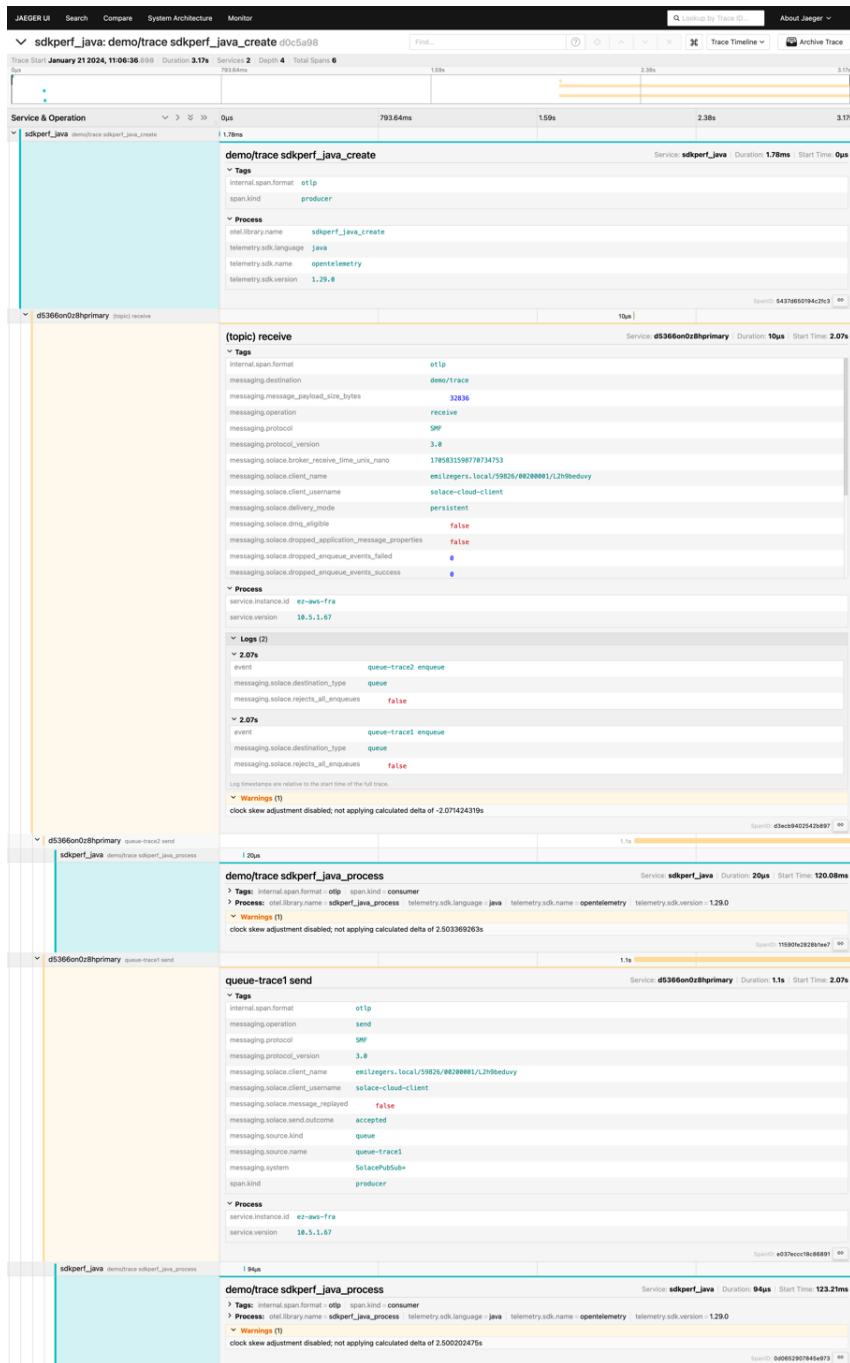
```
otlphttp/jsontest:  
  endpoint: "http://localhost:3317/"  
  compression: "none"  
  encoding: "json"
```

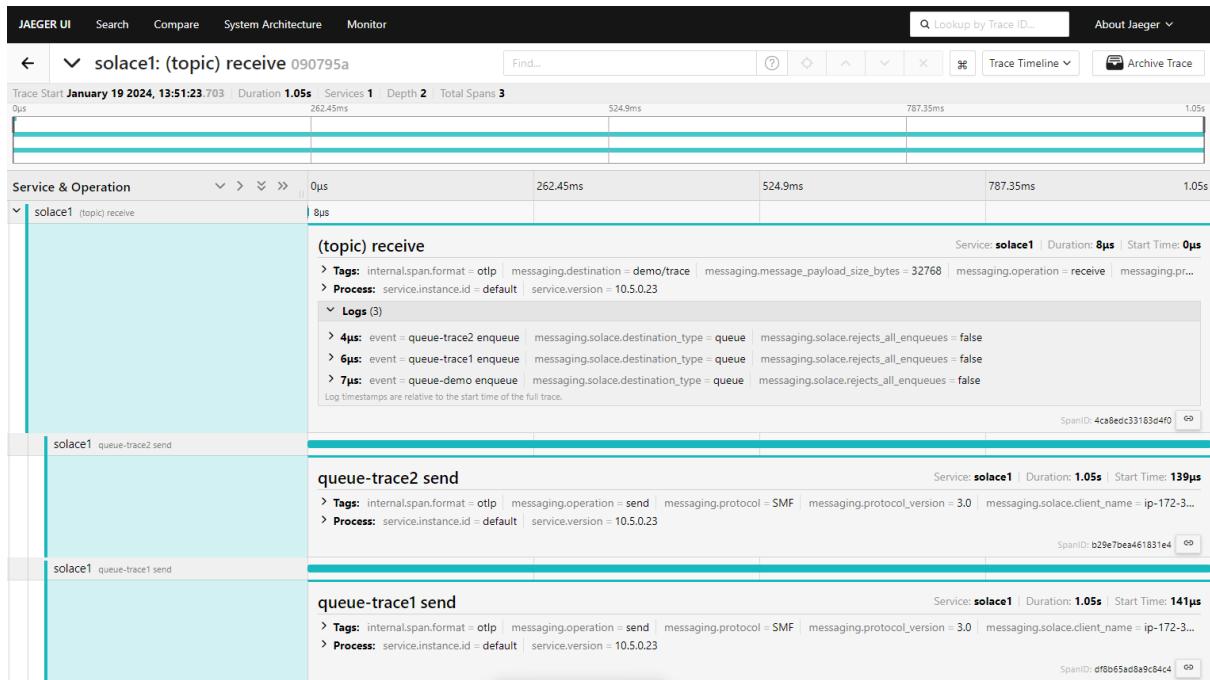
```
  tls:  
    insecure: true  
  headers:  
    Content-Type: "application/json"
```



4.2 Jaeger

Navigate to <http://localhost:16686> to access the Jaeger UI (server status info at <http://0.0.0.0:14269/>, see <https://www.jaegertracing.io/docs/1.53/deployment/> for more info).





4.3 Dynatrace

<https://docs.dynatrace.com/docs/extend-dynatrace/extend-metrics/ingestion-methods/opentelemetry>

Screenshots of the Dynatrace documentation page titled "Send OpenTelemetry metrics to Dynatrace".

The page includes a sidebar with navigation links for various metrics ingestion methods like OpenTelemetry, Micrometer, Prometheus, StatsD, Telegraf, and AWS Distro.

Send OpenTelemetry metrics to Dynatrace

Scenarios for OpenTelemetry metric ingestion

You can send metrics to Dynatrace via any of the following options:

- Via an OTLP (OpenTelemetry Protocol) metrics exporter directly to Dynatrace
- Via an OTLP OpenTelemetry Protocol metrics exporter to an OpenTelemetry Collector
- Via the Dynatrace OpenTelemetry metrics exporter directly to Dynatrace
- Via AWS Distro for OpenTelemetry

Related topics

[OpenTelemetry](#)

Learn how to integrate and ingest OpenTelemetry data (traces, metrics, and logs) into Dynatrace.

<https://docs.dynatrace.com/docs/extend-dynatrace/opentelemetry/getting-started/metrics/ingest/metrics-via-otel-collector>

Screenshots of the Dynatrace documentation page titled "Send metrics via the OpenTelemetry collector".

The page includes a sidebar with navigation links for various metrics ingestion methods like OpenTelemetry, Micrometer, Prometheus, StatsD, Telegraf, and AWS Distro.

Send metrics via the OpenTelemetry collector

The OpenTelemetry collector comes with the OTLP metrics exporter and many others, including the Dynatrace metrics exporter.

You can send metrics to Dynatrace via OTLP HTTP or the Dynatrace metrics exporter.

[Via the OTLP HTTP metrics exporter](#) [Via the Dynatrace metrics exporter](#)



For more information on how to deploy and configure it, see [OpenTelemetry collector](#).

Related topics

[Migrate from the Dynatrace OpenTelemetry metrics exporter to OTLP metrics exporter](#)

Migrate from the Dynatrace OpenTelemetry metrics exporter to OTLP metrics exporter.

<https://docs.dynatrace.com/docs/extend-dynatrace/opentelemetry/collector#example-configuration>

Screenshots illustrating the configuration of a Collector in Dynatrace:

Configuration example (Documentation)

The screenshot shows a code editor displaying a YAML configuration file for a Collector. The file defines processors, exporters, and services. It includes an OTELHTTP exporter configured to send data to a specific endpoint and headers, including an Authorization header with a placeholder API token.

```

4   grpc;
5   http;
6
7   processors:
8     cumulativeedelta;
9
10  exporters:
11    otlphttp;
12    endpoint: "https://{{your-environment-id}}.live.dynatrace.com/api/v2/otlp";
13    headers:
14      Authorization: "Api-Token ${API_TOKEN}";
15
16  service:
17    pipelines:
18      traces:
19        receivers: [otlp];
20        processors: [cumulativeedelta];
21        exporters: [otlphttp];
22      metrics:
23        receivers: [otlp];
24        processors: [cumulativeedelta];
25        exporters: [otlphttp];
26      logs:
27        receivers: [otlp];
28        processors: [];
29        exporters: [otlphttp];

```

In this YAML file, we configure the following components:

- An OTEL receiver (`otlp`) that can receive data via gRPC and HTTP
- A processor to convert any metrics with cumulative temporality to delta temporality (see [Delta metrics](#) for more details)

Screenshot of the Dynatrace UI showing the list of dashboards:

Dashboards

The dashboard lists 9 dashboards, each with a preview icon, name, popularity, and modification date. The dashboards include:

- Kubernetes cluster overview
- Kubernetes namespace resource quotas
- Real User Monitoring
- Kubernetes workload overview
- Synthetic Monitoring
- Davis® health self-monitoring
- DPS Usage Details DEMO
- Metric & Dimension Usage + Rejections
- OneAgent Traces - Adaptive traffic management (Classic License)

On the right side of the interface, there are various links and credits information:

- Host units credits: 0/1,000
- User session credits: 0/50,000
- Synthetic monitor credits: 0/30,000
- Davis data units credits: 0/200,000
- Buy now button
- Support Resources, Support Center, Release notes, Documentation, University, Community, Product ideas, Dynatrace API, Environment API v2, Environment API v1, Configuration API, Personal access tokens, Mobile apps, Receive alerts via mobile app, Sign out
- Dynatrace version 1.283.64

The screenshot shows the Dynatrace web interface with the URL <https://xgo98208.live.dynatrace.com/ui/personal-access-tokens?gtfs=2h&gf=all>. The left sidebar is dark-themed and includes sections like Favorites, Infrastructure Observability, Application Observability, Application Security, Digital Experience, Business Analytics, and Manage. The main content area has a teal header "Personal access tokens" and a sub-header "Personal access tokens for emil.zegers@solace.com". It displays a message: "Personal access tokens have been disabled. You can still manage your existing tokens but they won't work until the feature is re-enabled." Below this, a table lists "Personal access tokens (0/5)" with columns: Token name, Last used, Created, Number of scopes, Enabled, Delete, and Details. A note says "No access tokens available. Either adjust filtering to show more tokens or generate a new access token." At the top right, there are "Deploy Dynatrace" and "New Chrome available" buttons.

The screenshot shows the Dynatrace web interface with the URL <https://xgo98208.live.dynatrace.com/ui/dashboards?gtfs=2h&gf=mail>. The left sidebar is dark-themed and includes sections like Favorites, Infrastructure Observability, Application Observability, Application Security, Digital Experience, Business Analytics, and Manage. A search bar at the top is set to "access tokens". A modal window titled "Access tokens" is open, listing three categories: "Access tokens", "Personal access token", and "Access tokens and permissions". The "Access tokens" category is selected. The main content area shows a table titled "9 Dashboards" with columns: Favorite, Name, Popularity, Modified at, Owner, and Actions (three dots). The dashboards listed are: Kubernetes cluster overview, Kubernetes namespace resource quotas, Real User Monitoring, Kubernetes workload overview, Synthetic Monitoring, Davis® health self-monitoring, DPS Usage Details DEMO, Metric & Dimension Usage + Rejections, and OneAgent Traces - Adaptive traffic management (Classic License).

The screenshot shows the Dynatrace Settings interface. The left sidebar has sections like Favorites, Dashboards, Deploy Dynatrace, Problems, Observe and explore, Infrastructure Observability, Automations, Application Observability, Application Security, Digital Experience, Business Analytics, and Manage. The main content area is titled 'Access tokens' and contains the following text:
Configure Dynatrace API access token and personal access token generation. For details about tokens and authentication go to [Dynatrace API authentication documentation](#).
To create an access token go to [Access tokens](#).
Create Dynatrace API tokens in the new format.
Check out this [blog post](#) to find out more about the new Dynatrace API token format.
Enable personal access tokens.
Allow users of this environment to generate personal access tokens based on user permissions. Note that existing personal access tokens will become unusable while this setting is disabled.

The screenshot shows the Dynatrace Access tokens page. The left sidebar is identical to the previous screenshot. The main content area is titled 'Access tokens' and contains the following text:
Manage API access tokens, which provide access to this environment's monitoring and configuration data. To manage your personal access tokens for API access, go to [Personal access tokens](#) in the Dynatrace user menu.
Use the [Dynatrace API Explorer](#) or read the [API documentation](#) for use-cases and examples.

Below this, there is a table header for '0 access tokens' with columns: Token name, Last used, Created, Number of scopes, Owner, Enabled, Delete, Details. A message below the table says 'No access tokens available' and 'Either adjust filtering to show more tokens or generate a new access token.'

Token name: solace-otel

Search for OpenTelemetry and add Scopes

The screenshot shows the Dynatrace interface for generating an access token. The token name is set to 'solace-otel'. The 'Selected scopes' section includes 'Ingest logs', 'Ingest metrics', and 'Ingest OpenTelemetry traces'. Below the scopes, a curl command is provided for automation:

```
curl -X POST "https://xgo98208.live.dynatrace.com/api/v2/apiTokens" -H "accept: application/json; charset=utf-8" -H "Content-Type: application/json; charset=utf-8" -d "{\"name\":\"solace-otel\",\"scopes\":[\"logs.ingest\",\"metrics.ingest\",\"opentelemetryTrace.ingest\"]}" -H "Authorization: Api-Token XXXXXXXX"
```

Click [Generate token]

The screenshot shows the Dynatrace interface after generating the token. A success message states 'Token 'solace-otel' generated successfully with:'. The generated token is shown in a modal box:

```
dt0c01AACBTf2GOLYOVFSLV20ALCSZ76P4Hh
```

Below the token, a curl command for generating the token is displayed:

```
curl -X POST "https://xgo98208.live.dynatrace.com/api/v2/apiTokens" -H "accept: application/json; charset=utf-8" -H "Content-Type: application/json; charset=utf-8" -d "{\"name\":\"solace-otel\",\"scopes\":[\"logs.ingest\",\"metrics.ingest\",\"opentelemetryTrace.ingest\"]}" -H "Authorization: Api-Token XXXXXXXX"
```

Copy token and add to yaml.

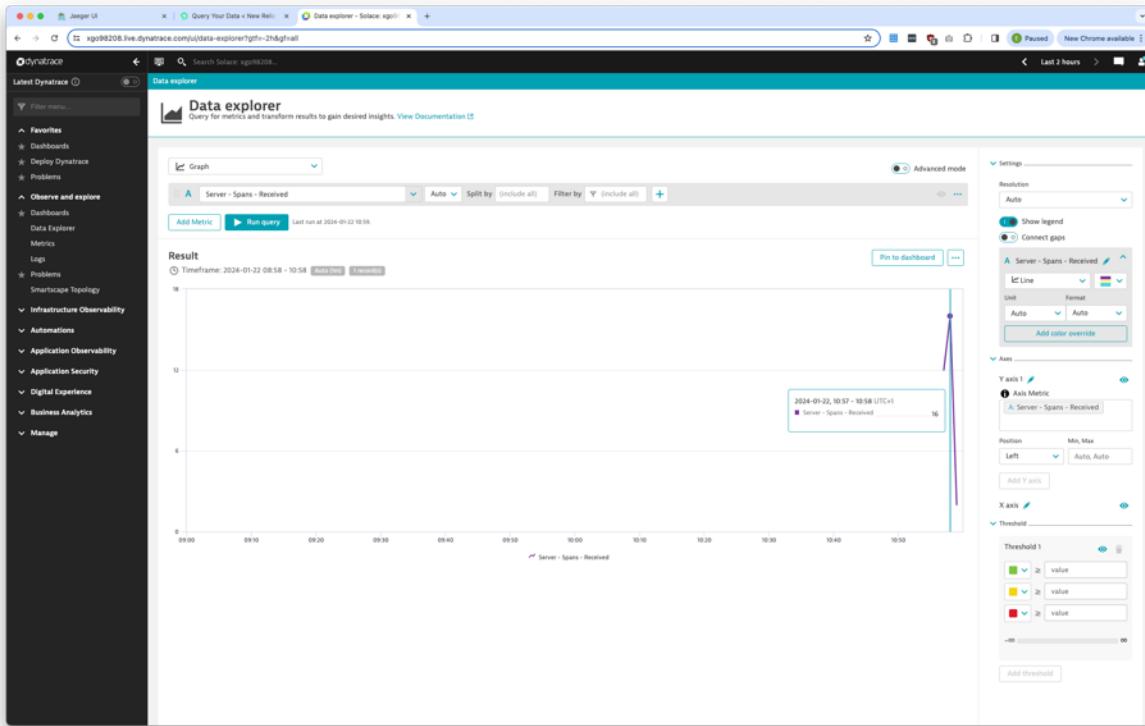
Now send some events with SDKPerf resulting in traces in Dynatrace.

The screenshot shows the Dynatrace Data explorer interface. On the left, the navigation menu is visible, including sections like Favorites, Dashboards, Deploy Dynatrace, Problems, Observe and explore (with Infrastructure Observability expanded), and Infrastructure Observability (with Application Observatory, Application Security, Digital Experience, and Business Analytics expanded). The main area is titled "Data explorer" with the sub-section "opentelemetry". A search bar at the top right says "Search for metric or transform results to gain desired insights. View Documentation". Below the search bar, there's a "Graph" section with a dropdown set to "opentelemetry". The main panel displays several metrics and visualizations:

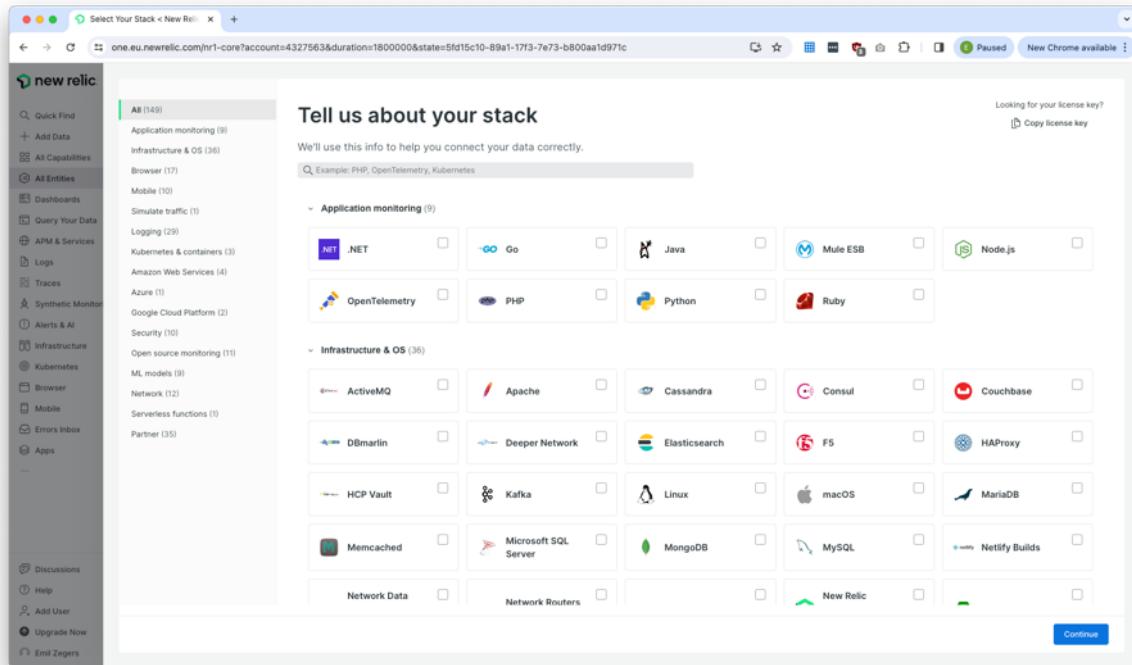
- Server - Spans - Received**: A chart showing the count of spans received over time, with a timestamp of Jan 22, 2024 10:58.
- Kubernetes clusters by CPU**: A heatmap showing Kubernetes clusters in terms of CPU usage.
- Largest Kubernetes workload in terms of running pods**: A chart showing the largest Kubernetes workload.
- Kubernetes nodes CPU requests utilization overview**: A chart showing Kubernetes nodes CPU requests utilization.
- Disk space used %**: A chart showing disk space usage.
- Crash-free user rate**: A chart showing application satisfaction SLO.
- Average Actions per Session**, **Core web vitals**, **Core web vitals - LCP**, **Core web vitals - CLS**, **Core web vitals - FID**, **Application satisfaction SLO**: Various charts related to web performance and satisfaction.
- Key request availability SLO**: A chart showing key request availability.

This screenshot is similar to the one above, but the "Run query" button in the "opentelemetry" section is highlighted in blue. The rest of the interface and data visualization are identical to the first screenshot.

Click [Run query]



4.4 New Relic



Select Your Stack < New Relic

one.eu.newrelic.com/hr1-core?account=4327563&duration=1800000&state=f3ff545b-aee1-f7fb-e69b-48c575f39522

Tell us about your stack

We'll use this info to help you connect your data correctly.

Looking for your license key? [Copy license key](#)

spentelemetry

All (3)

- Application monitoring (1)
- Infrastructure & OS (0)
- Browser (0)
- Mobile (0)
- Simulate traffic (0)
- Logging (1)
- Kubernetes & containers (0)
- Amazon Web Services (0)
- Azure (0)
- Google Cloud Platform (0)
- Security (0)
- Open source monitoring (1)
- ML models (0)
- Network (0)
- Serverless functions (0)
- Partner (2)

Application monitoring (1)

OpenTelemetry

Logging (1)

Vercel

Open source monitoring (1)

OpenTelemetry

Partner (2)

Jenkins

Vercel

Continue

This screenshot shows the 'Select Your Stack' step in the New Relic setup process. The user has selected 'OpenTelemetry' under 'Application monitoring'. Other sections like 'Logging' and 'Open source monitoring' also have items selected. The sidebar on the left shows the full list of available stacks. A search bar at the top is populated with 'spentelemetry'. A 'Continue' button is at the bottom right.

Select Your Stack Installation

one.eu.newrelic.com/hr1-core?account=4327563&duration=1800000&state=a1644e1c-3229-15d5-079a-3964fe9ff947

Time to install

Instrument your hosts

Whether it's in the cloud or on-premise, you can monitor your data from our cloud-based platform — click on the tile to start installing.

Linux

Windows

Docker

Kubernetes

macOS

Don't have access to host

Connect your applications

Add other instrumentation

See your data

Manual Install Support

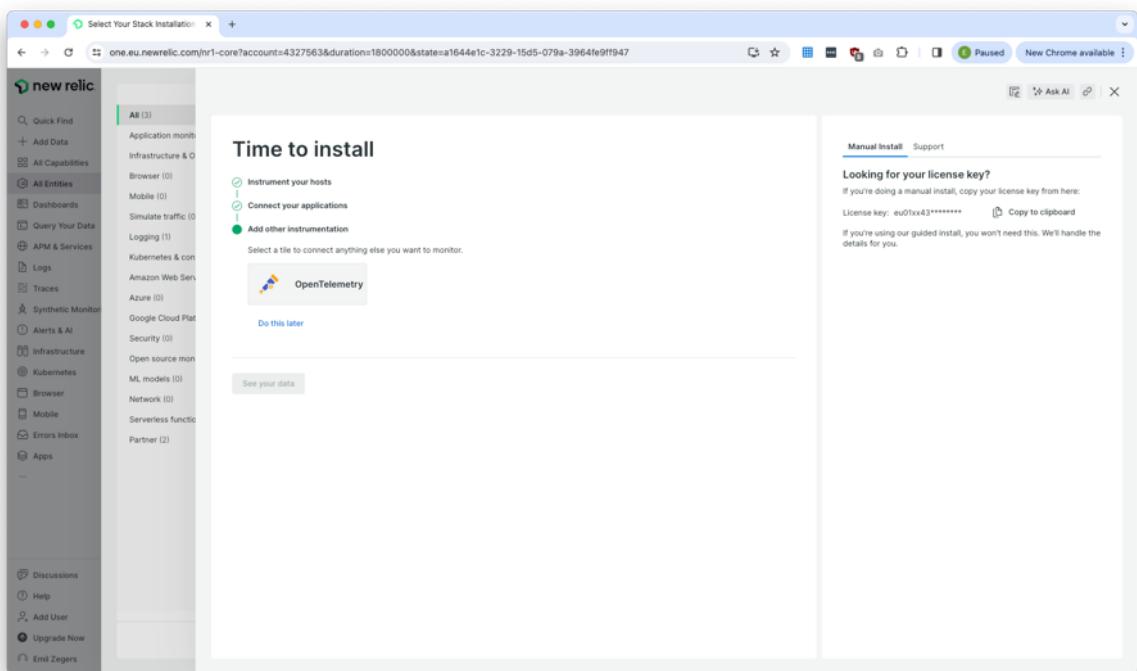
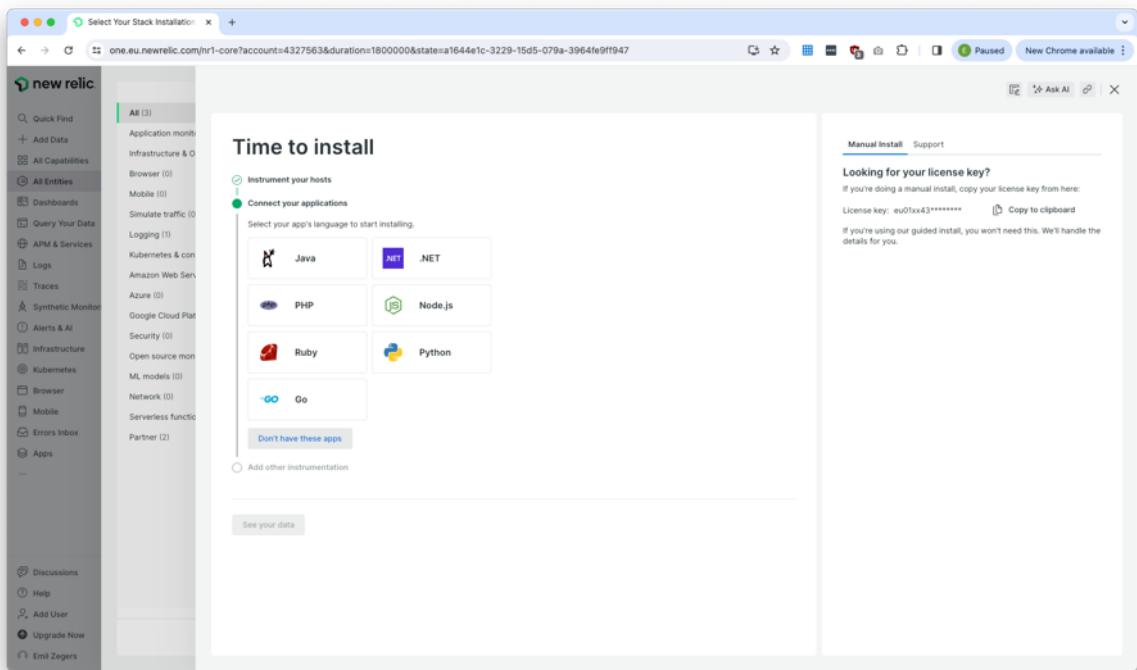
Looking for your license key?

If you're doing a manual install, copy your license key from here:

License key: eu0txx43***** [Copy to clipboard](#)

If you're using our guided install, you won't need this. We'll handle the details for you.

This screenshot shows the 'Time to install' step. It provides options for instrumenting hosts (Linux, Windows, Docker, Kubernetes, macOS) and connecting applications. A 'Don't have access to host' link is available. On the right, there's a 'Manual Install' section with a license key field containing 'eu0txx43*****' and a 'Copy to clipboard' button. A note indicates that if using a guided install, the license key is not needed as the details will be handled by the tool.



<https://docs.newrelic.com/docs/more-integrations/open-source-telemetry-integrations/opentelemetry/get-started/opentelemetry-get-started-intro/>

<https://docs.newrelic.com/docs/more-integrations/open-source-telemetry-integrations/opentelemetry/get-started/opentelemetry-get-started-intro/>

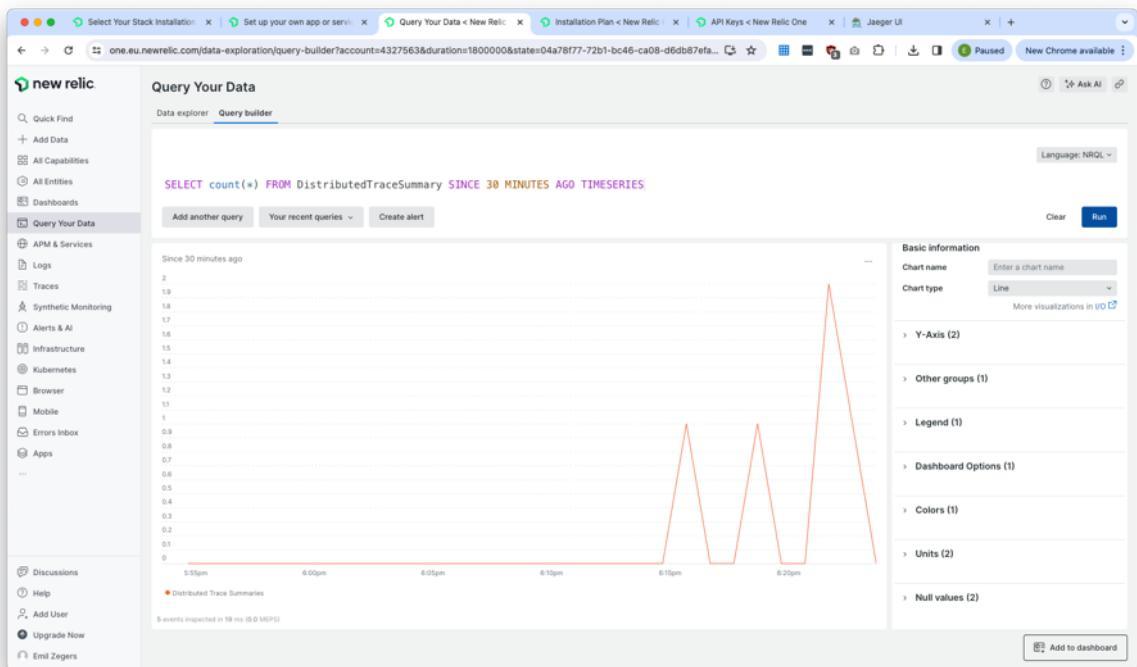
The screenshot shows the New Relic documentation website for OpenTelemetry. The main content area is titled 'Set up your own app or service'. It includes sections for 'Monitor a host', 'Monitor Kubernetes', and 'Try out a demo app'. A sidebar on the right contains links for 'On this page' such as 'Set up your own app or service', 'Monitor a host', and 'Monitor Kubernetes'. At the bottom right of the page is a 'Was this doc helpful?' poll with 'Yes' and 'No' options.

<https://docs.newrelic.com/docs/more-integrations/open-source-telemetry-integrations/opentelemetry/get-started/opentelemetry-set-up-your-app/>

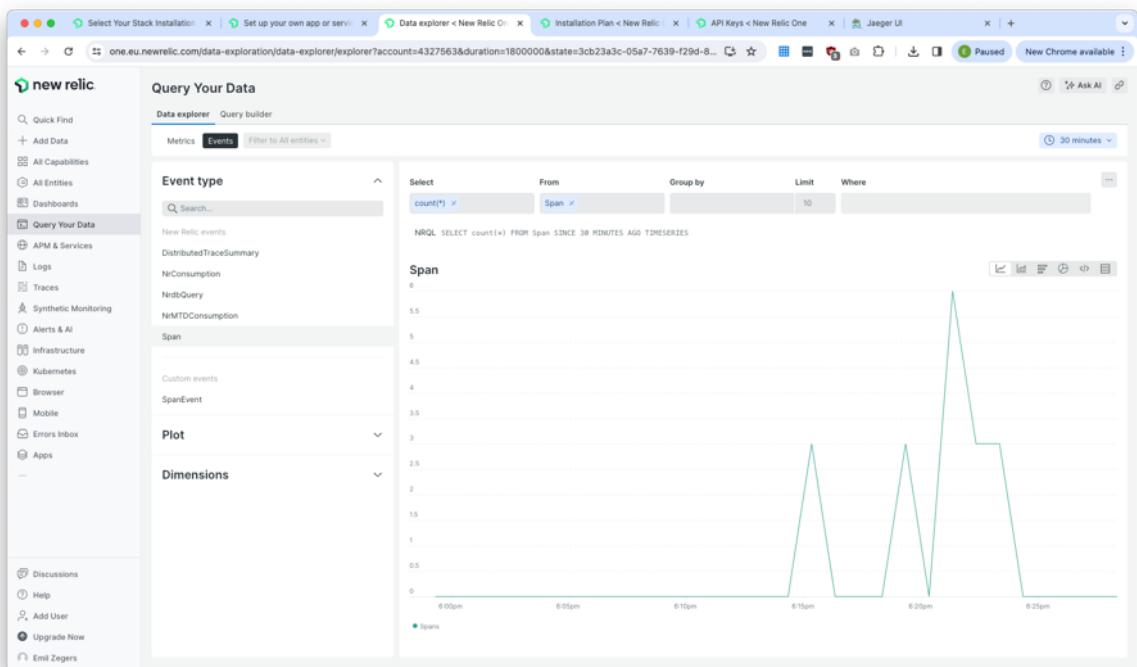
<https://docs.newrelic.com/docs/more-integrations/open-source-telemetry-integrations/opentelemetry/get-started/opentelemetry-set-up-your-app/>

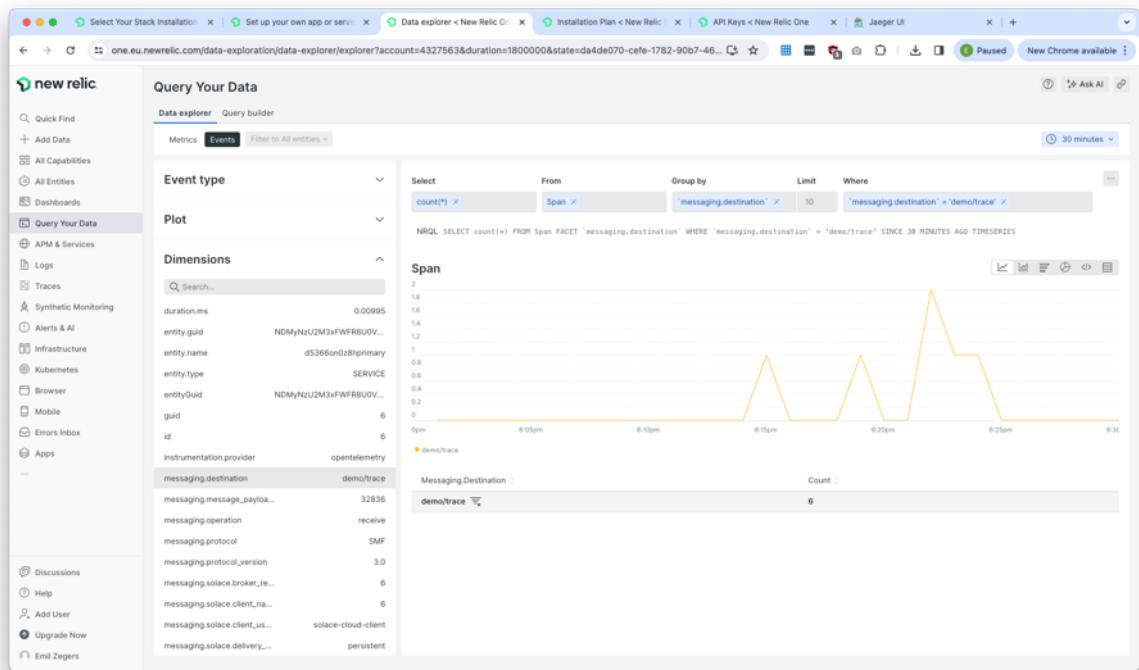
The screenshot shows the 'Review configurations for exporting telemetry data to New Relic' section. It discusses the OTLP protocol and provides two diagrams: one for direct export from an application and one for export via an OpenTelemetry Collector. The 'On this page' sidebar lists related topics like 'Instrument your app or service with OpenTelemetry' and 'Complete the export configuration steps'.

See yaml example at <https://docs.newrelic.com/docs/more-integrations/open-source-telemetry-integrations/opentelemetry/collector/opentelemetry-collector-basic/>



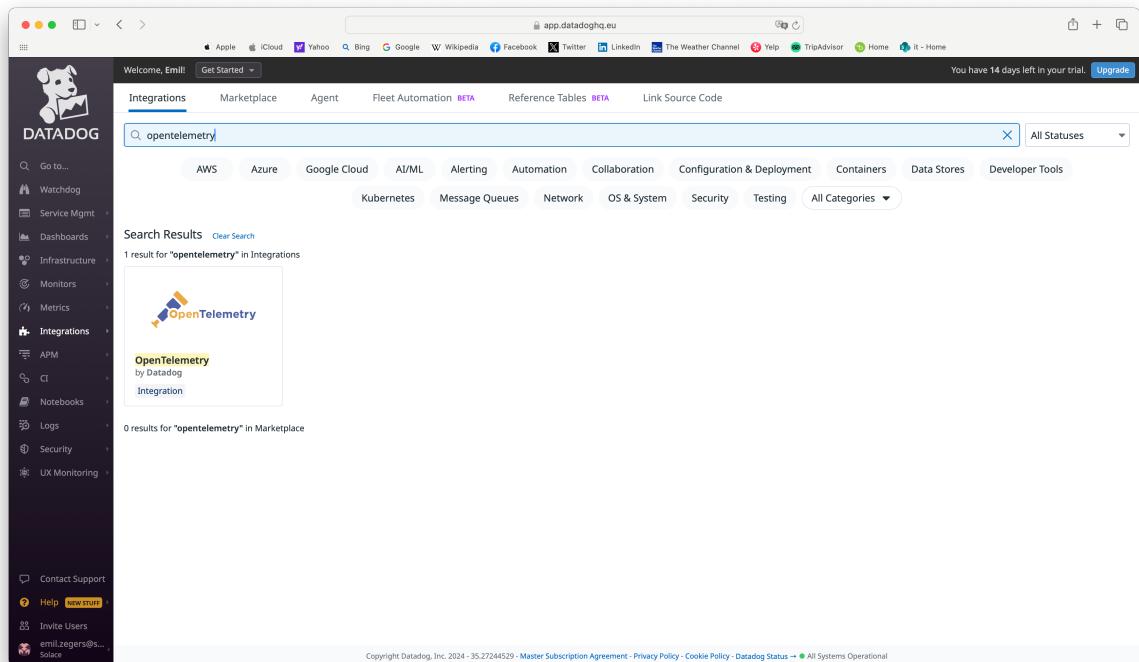
In Query Your Data select Event Type Span





4.5 DataDog

NOTE: no time yet to add... Keep an eye on updates.



Welcome, Emili! Get Started

OpenTelemetry
by Datadog
Get telemetry data from the OpenTelemetry Collector

AVAILABLE

Overview Configure Data Collected Monitoring Resources Support Release Notes

Category
Developer Tools Network
OS & System

Supported OS
Linux Windows macOS

OpenTelemetry is a vendor-agnostic standard for telemetry data. Datadog supports ingesting OpenTelemetry data through the OpenTelemetry Collector and the Datadog Agent. This tile documents how to export data to Datadog through the OpenTelemetry Collector with Datadog Exporter. [OpenTelemetry collector Datadog exporter](#). Also see [OTLP Ingest in Datadog Agent](#) for further information on ingesting OTLP traces with Datadog Agent.

The OpenTelemetry Collector is a vendor-agnostic agent process that, through the Datadog exporter, exports telemetry data directly to Datadog servers (no Agent installation required). It reports metrics and traces from instrumented applications as well as general system metrics.

Host metrics are shown in the OpenTelemetry Host Metrics default dashboard, but you can send arbitrary metrics to Datadog using the OpenTelemetry Collector. Metrics under `system.*` and `process.*`, such as those generated by the host metrics receiver, are renamed to `otel.system.*` and `otel.process.*` to prevent collisions with metrics from the Datadog Agent. Additionally, OpenTelemetry Collector metrics are shown in the OpenTelemetry Collector Metrics default dashboard.

Copyright Datadog, Inc. 2024 - 35.27244529 - Master Subscription Agreement - Privacy Policy - Cookie Policy - Datadog Status → All Systems Operational

Welcome, Emili! Get Started

OpenTelemetry
by Datadog
Get telemetry data from the OpenTelemetry Collector

INSTALLED

Overview **Configure** Data Collected Monitoring Resources Support Release Notes

This integration is working properly.

Configuration

Installation

Follow the [OpenTelemetry Collector documentation](#) to install the `opentelemetry-collector-contrib` distribution, or any other distribution that includes the Datadog Exporter.

The Datadog Agent is **not** needed to export telemetry data to Datadog in this setup. See [OTLP Ingest in Datadog Agent](#) if you want to use the Datadog Agent instead.

Configuration

To export telemetry data to Datadog from the OpenTelemetry Collector, add the Datadog exporter to your metrics and traces pipelines. The only required setting is [your API key](#).

A minimal configuration file to retrieve system metrics is as follows.

```
receivers:
  hostmetrics:
  scrapers:
  load:
  cpu:
  disk:
  filesystem:
  memory:
  network:
  paging:
  process:

processors:
  batch:
```

Copyright Datadog, Inc. 2024 - 35.27244529 - Master Subscription Agreement - Privacy Policy - Cookie Policy - Datadog Status → All Systems Operational

```
exporters:
  datadog:
    api:
      key: "<Your API key goes here>"
```

```
service:
  pipelines:
    metrics:
      receivers: [hostmetrics]
      processors: [batch]
      exporters: [datadog]
```

4.6 Splunk

NOTE: no time yet to add... Keep an eye on updates.

TODO: need for enterprise cloud package with Splunk?

5 Resources

<https://github.com/taatuut/clear-agnostic>

The repo also contains this document and the deck on Distributed Tracing used at the Solace Connect user group in Amsterdam on January 25th, 2024 <https://solace.com/event/solace-connect-user-group-benelux-2024/>

https://www.youtube.com/playlist?list=PLY1Ks8JEfJR7jWm3aafht9cou2oleB_Ef

This image shows a YouTube playlist page titled "Distributed Tracing in Event-Driven Architecture" by Solace. The page includes a thumbnail for the first video, "What is OpenTelemetry?", featuring a cartoon character of a man with arms raised. Below the thumbnail, the video title, duration (1:37), and channel information (Solace) are visible. To the left of the main content area, there is a sidebar with a "Play all" button and a "Shuffle" button. The sidebar also contains a brief description of the series, listing five episodes covering topics like Open Telemetry, EDA & Tracing Challenges, and Distributed Tracing Work With an Event Broker. At the bottom of the sidebar, there is a link to the Solace community website.

Distributed Tracing in Event-Driven Architecture

Solace

6 videos 1,566 views Last updated on 10 Nov 2023

Play all Shuffle

This playlist includes Episodes 1 through 5 of Tamimi's Distributed Tracing in Event-Driven Architecture (EDA) series. He will cover:

- 1: What is Open Telemetry?
- 2: How does Open Telemetry work?
- 3: The Basics of Distributed Tracing
- 4: Introduction to EDA & Tracing Challenges
- 5: How Distributed Tracing works with an Event Broker

Have questions? Drop into <https://solace.community>

Episode 1 - What is Open Telemetry?
Solace • 4.9K views • 10 months ago
1:37

Episode 2 - How Does Open Telemetry Work?
Solace • 2.8K views • 10 months ago
1:12

Episode 3 - The Basics of Distributed Tracing
Solace • 2K views • 10 months ago
1:24

Episode 4 - Introduction to EDA & Tracing Challenges
Solace • 1.4K views • 10 months ago
2:16

Episode 5 - How Does Distributed Tracing Work With an Event Broker?
Solace • 1.3K views • 10 months ago
4:14

OpenTelemetry e-commerce demo with Kafka and Solace PubSub+ Event Broker
Solace • 224 views • 2 months ago
9:22

OpenTelemetry e-commerce demo with Kafka and Solace PubSub+ Event Broker

<https://www.youtube.com/watch?v=RIHQGV5KNM>