Solace PubSub+ Distributed Tracing suing 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

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+ and SAP AEM event brokers. The Distributed Tracing part of 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/
In addition, you can also use Solace Syslog Forwarding to ingest Event, System and Command logs information.

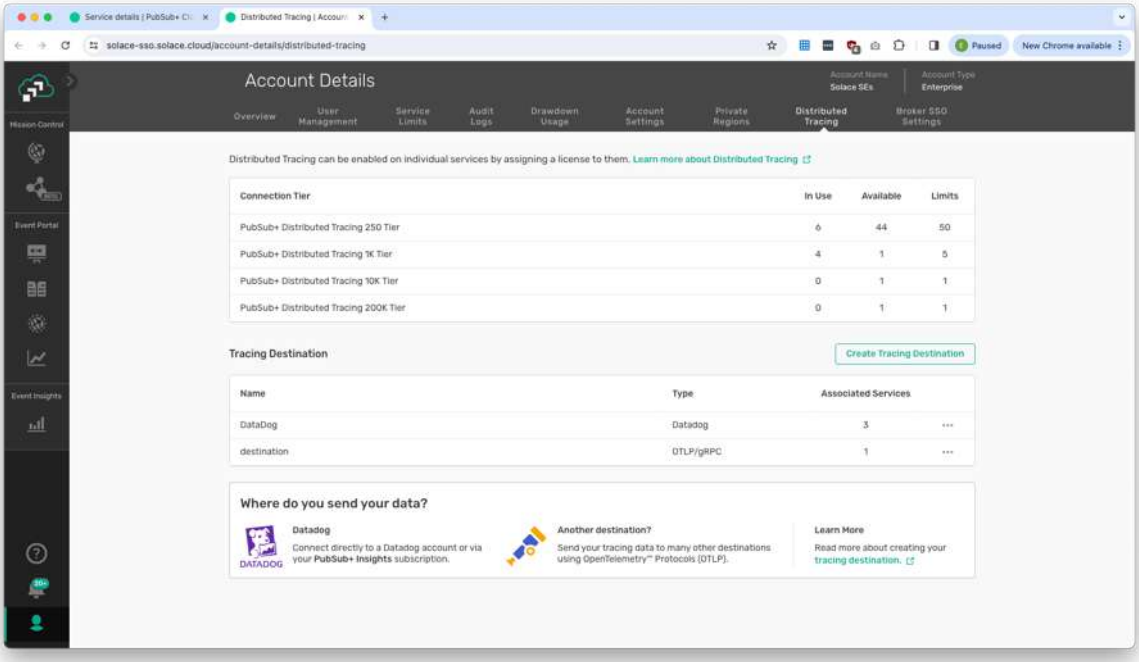
2 Setup

2.1 Prerequisites

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

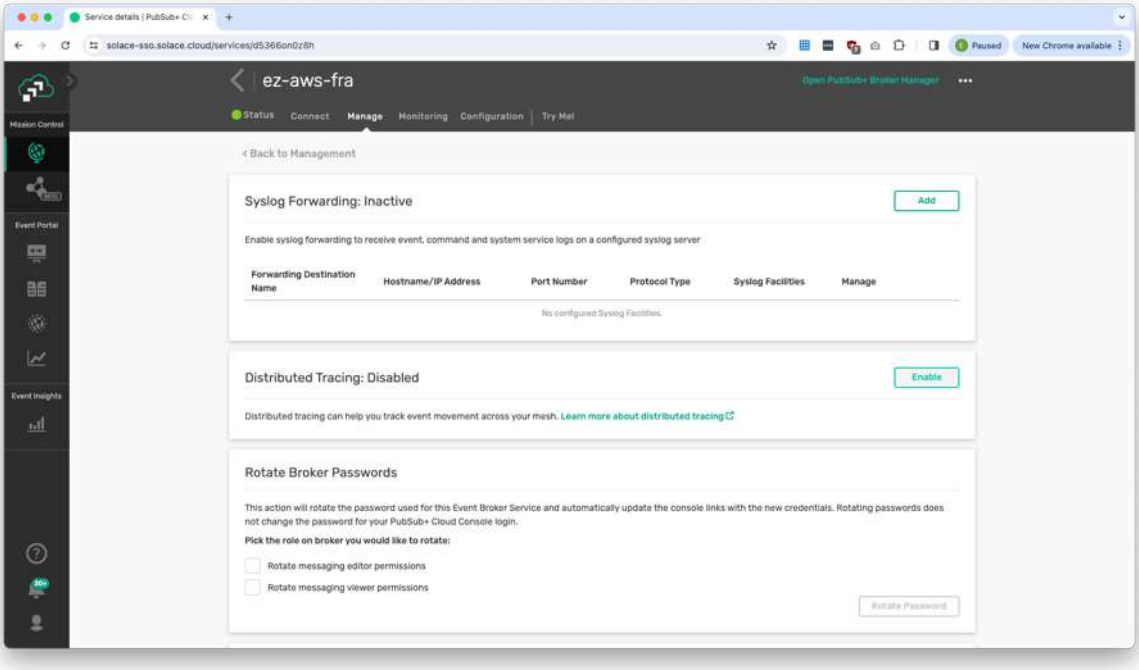
2.2 Configuration

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.

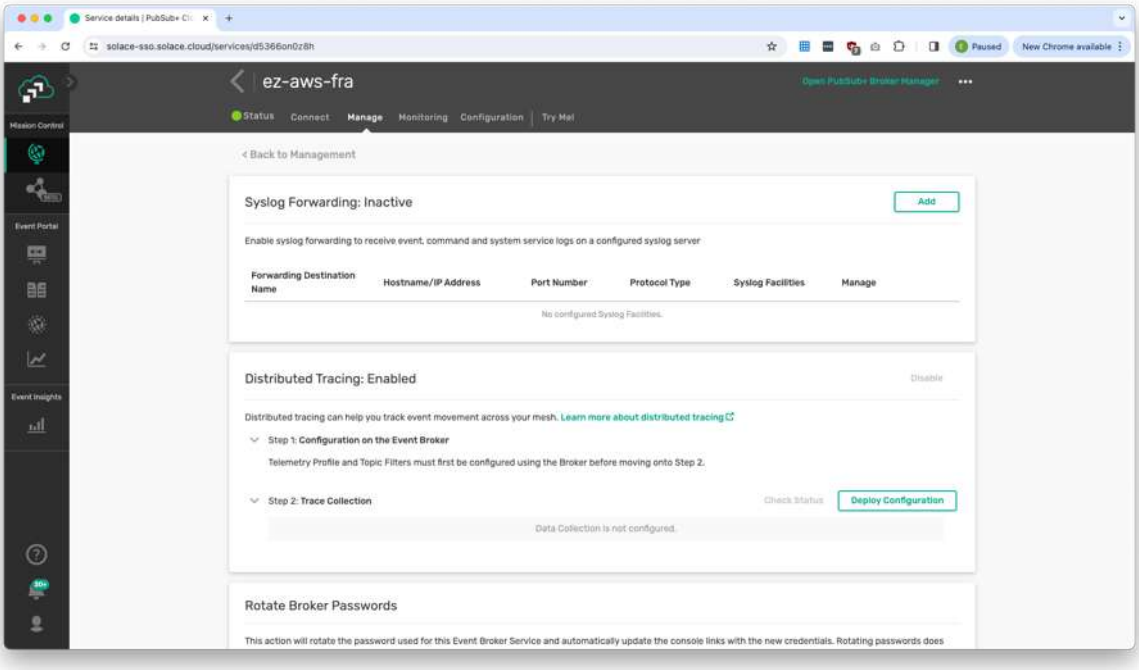


In this demo you will enable Distributed Tracing but you will not use the [Deploy Configuration] option to deploy a managed OTEL collector as you will use a self managed OTEL collector with local Jaeger and optional other destinations.

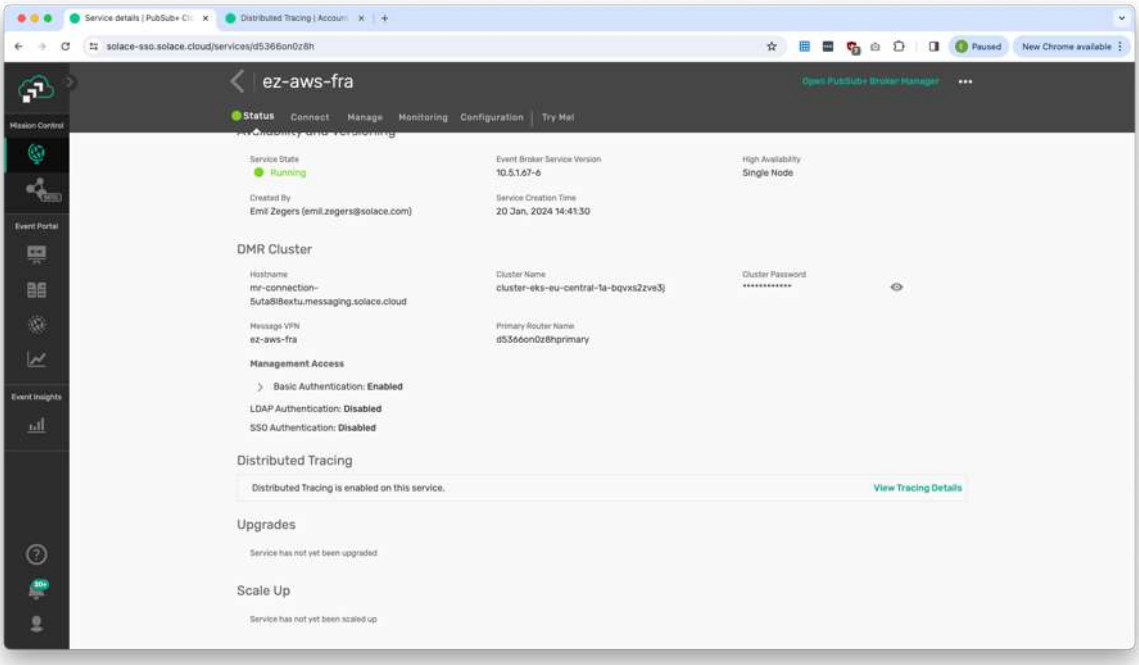
In Mission Control go to Cluster Manager and select the service where you want to use Distributed Tracing. In the service overview click Manage then Advanced Options. In the Distributed Tracing section click [Enable]



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

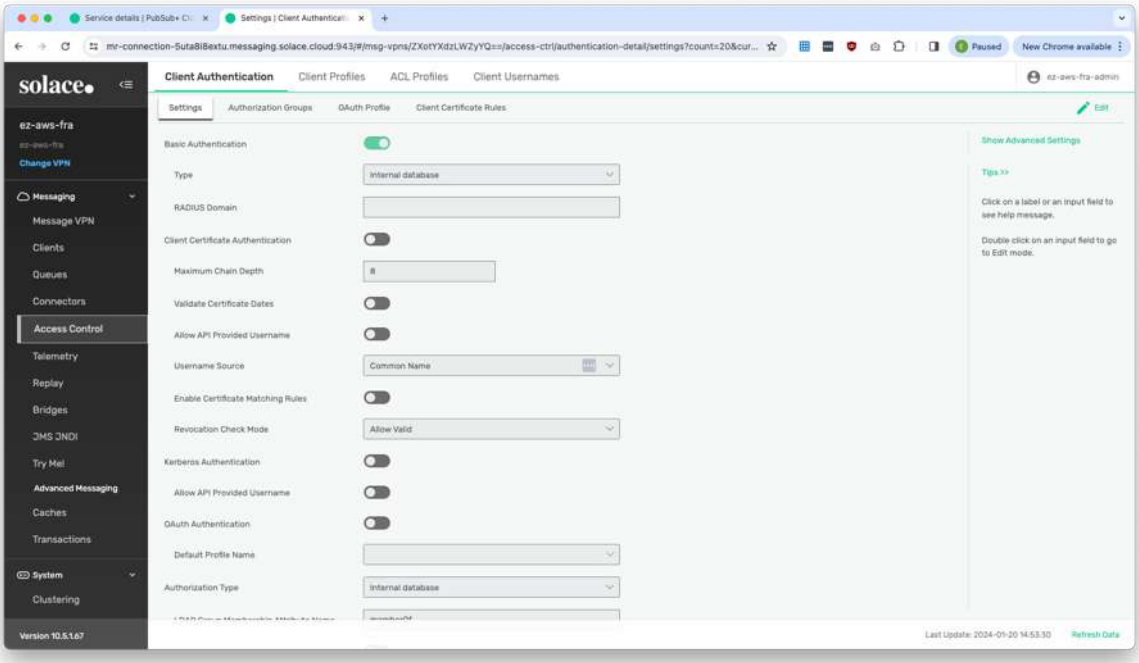


When enabled you'll find a section Distributed Tracing in the Status overview.

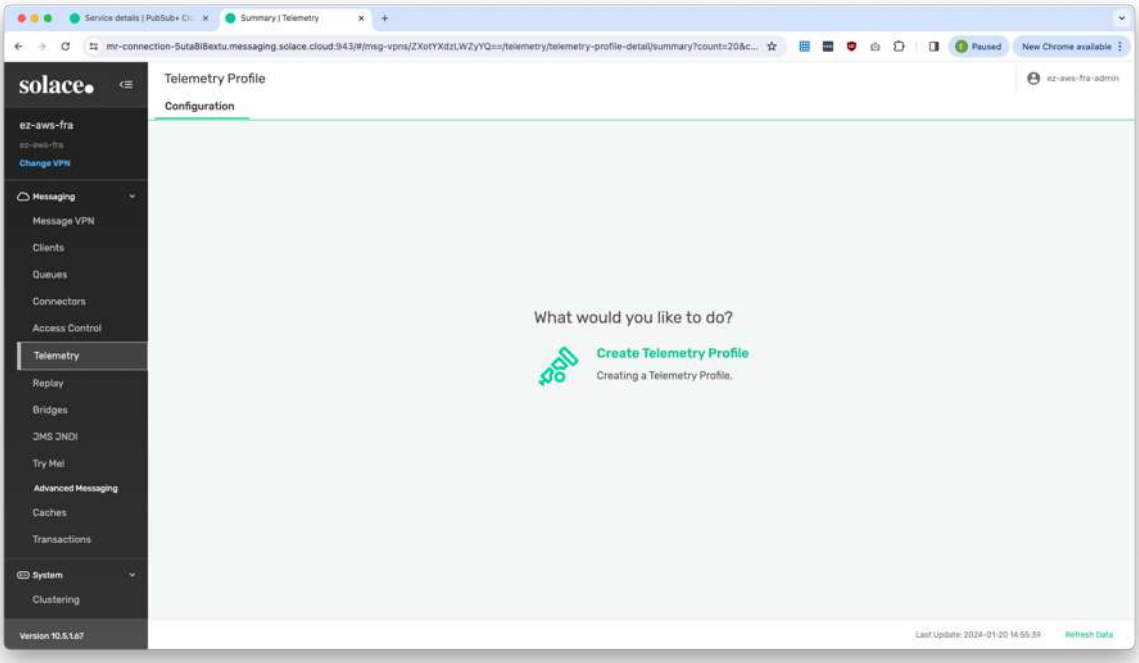


2.3 Configure Broker

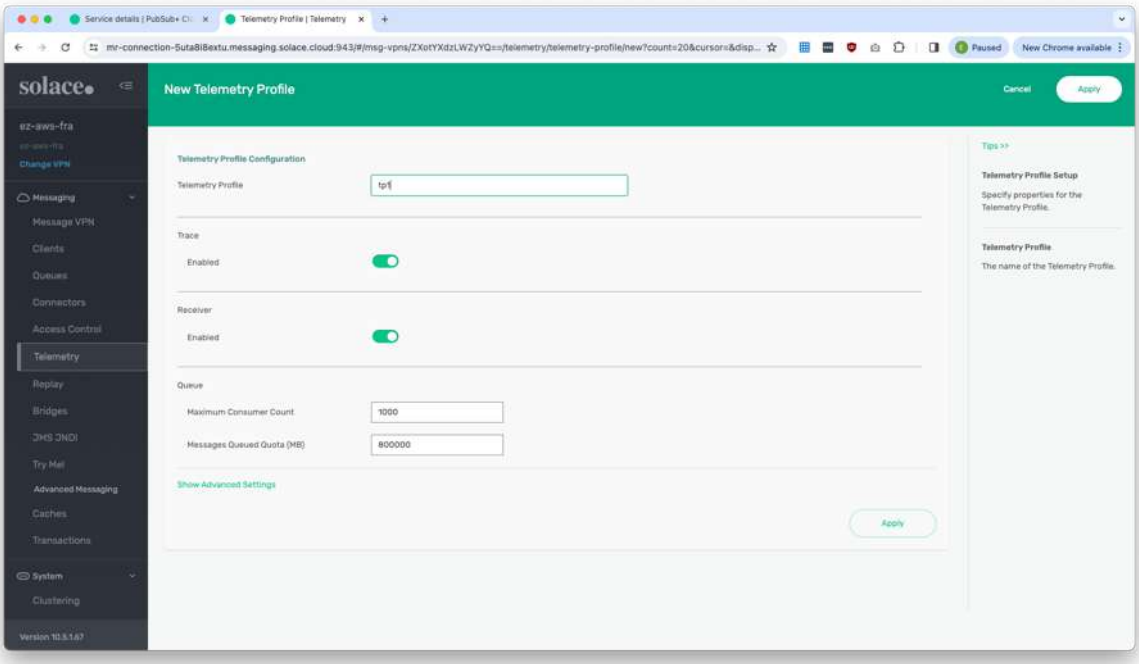
Open PubSub+ Broker Manager and verify under Access Control that Basic Authentication is enabled and set to Type Internal database.



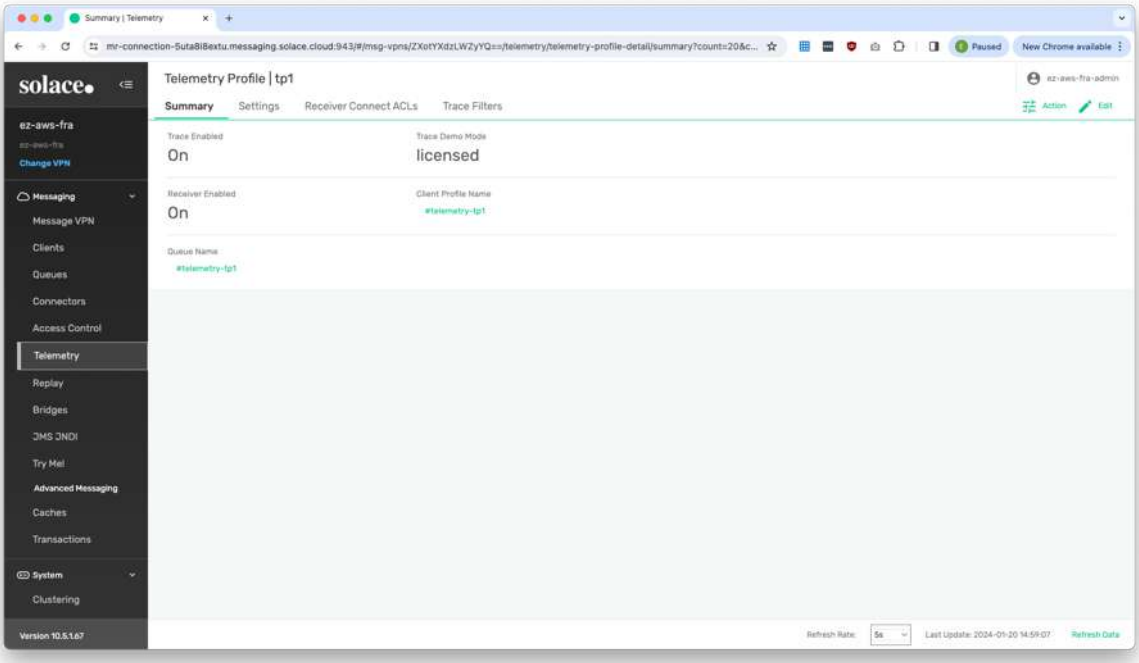
At Telemetry click Create a Telemetry Profile.



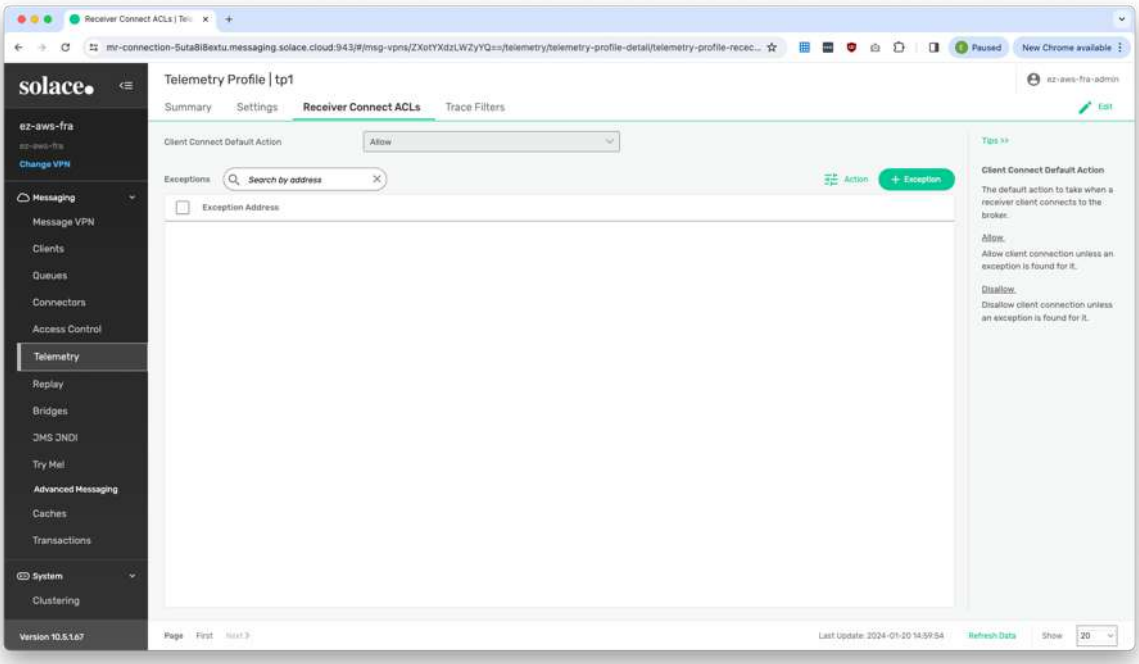
Use name "tp1" and check if Trace and Receiver are enabled then click [Apply]

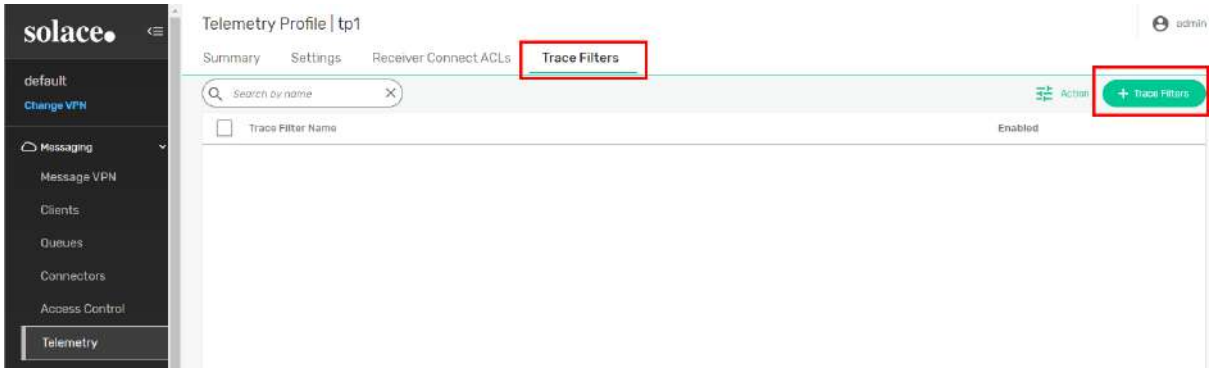


Telemetry Profile is displayed with Client Profile Name and Queue Name #telemetry-tp1

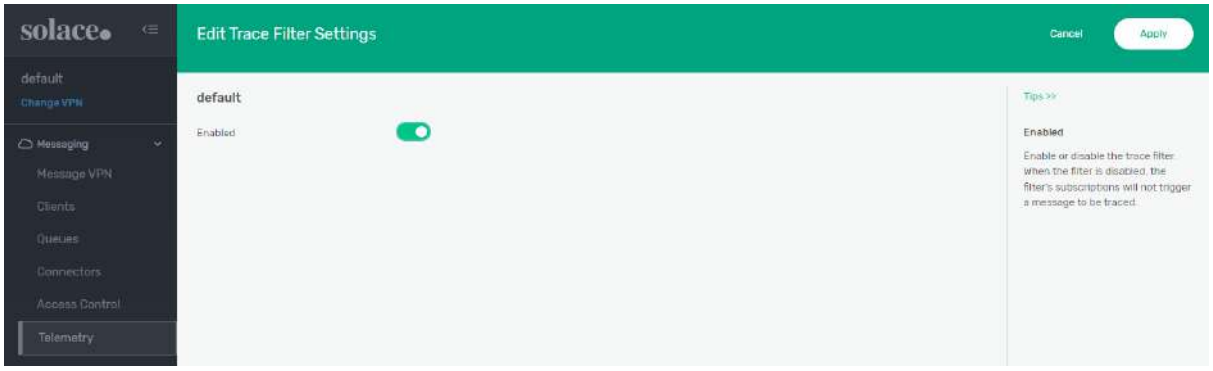


Click Edit at Receiver Connect ACLs to switch from Disallow to Allow and click [Apply]

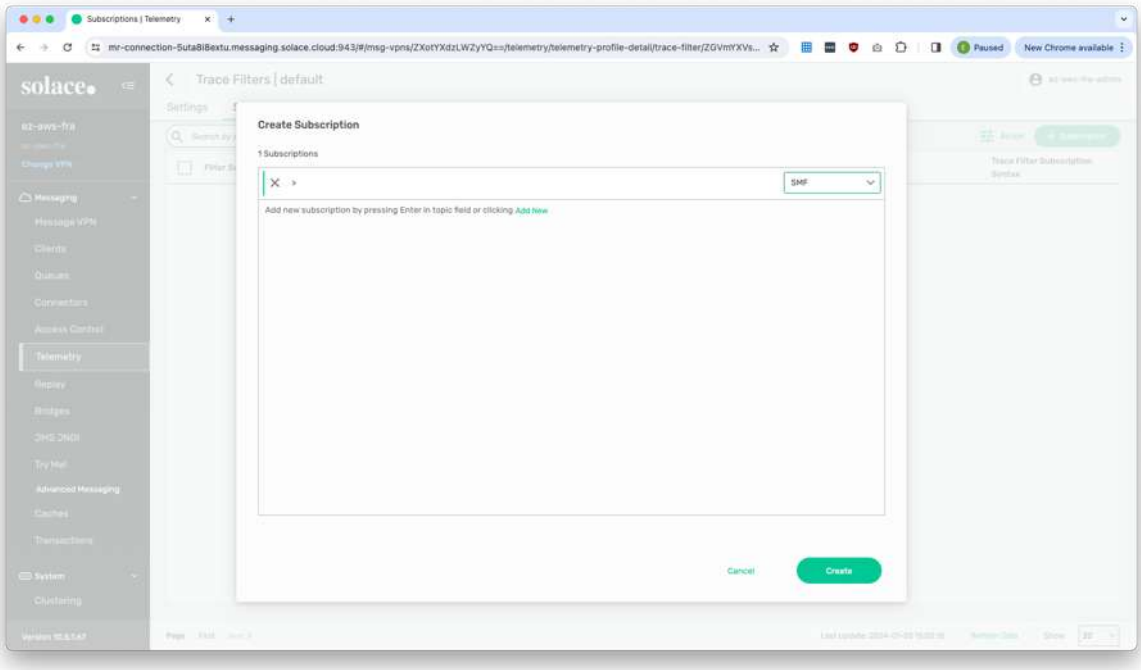


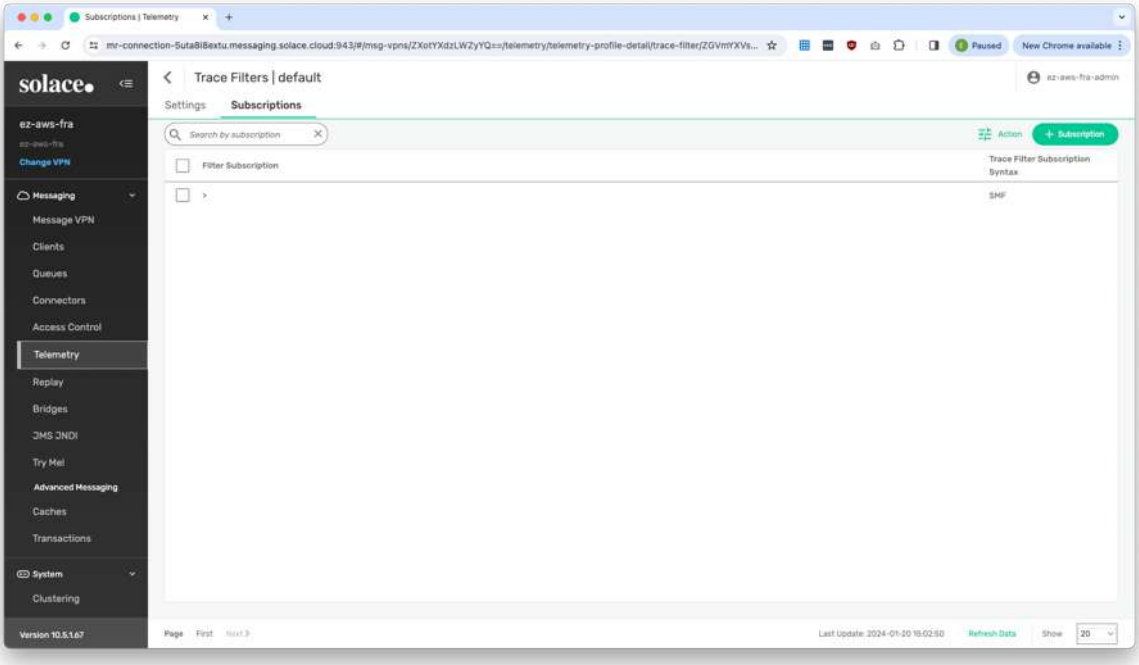


At Trace Filters click [+ Trace Filters] to add a filter with name "default" and set it to Enabled.

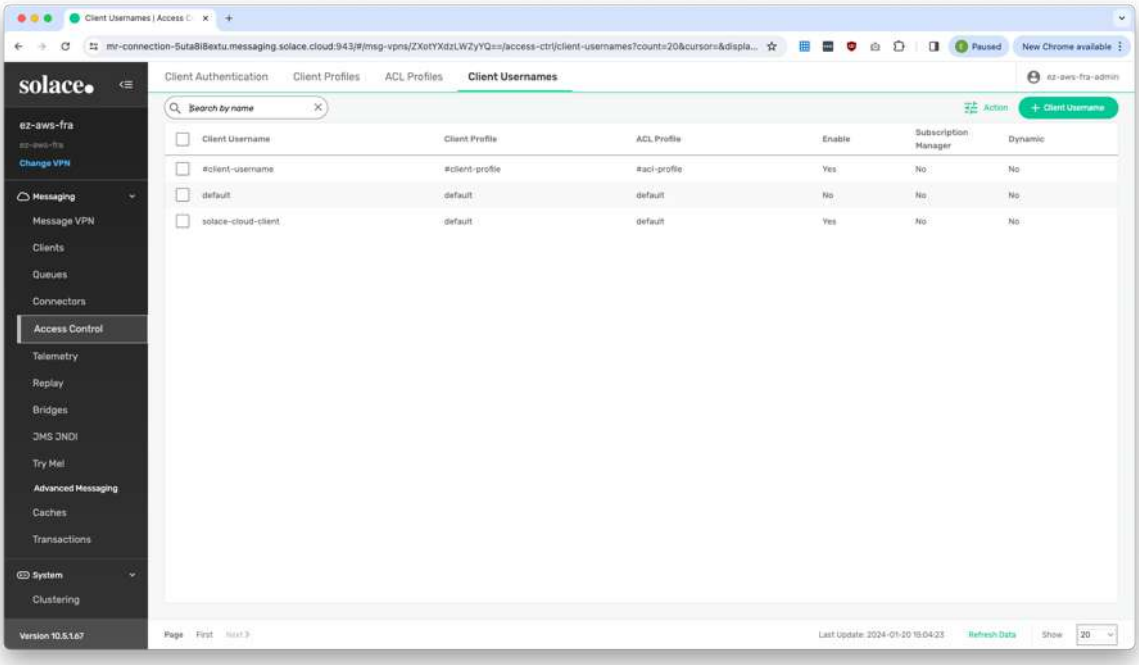


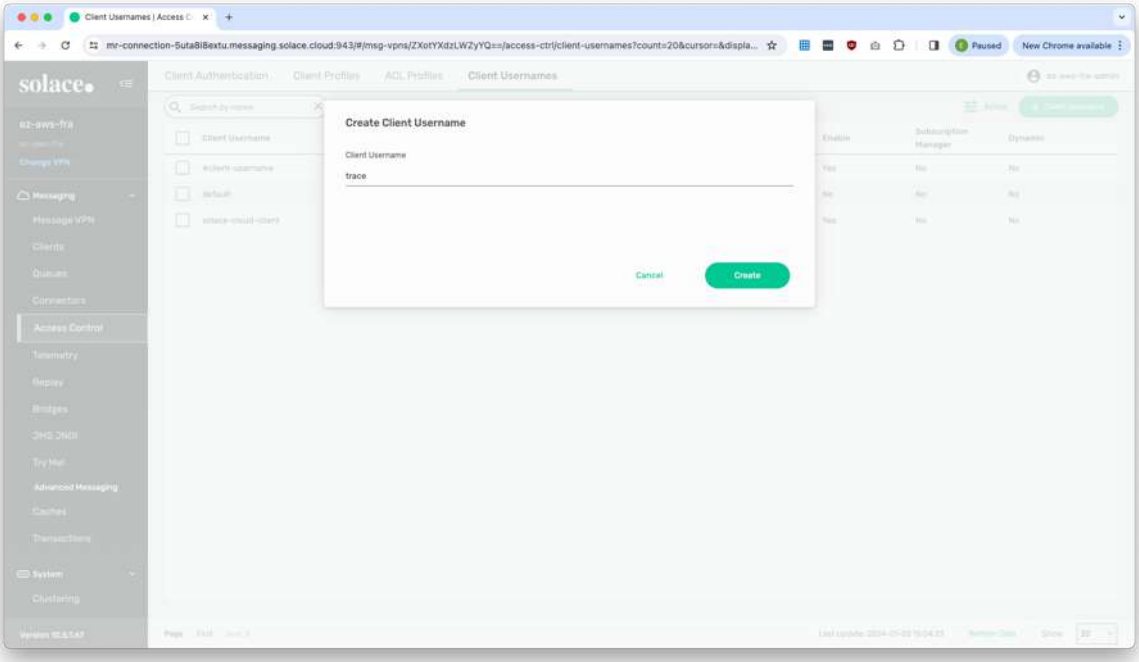
Open this filter and create a subscription using the *greater than wildcard sign* ">"

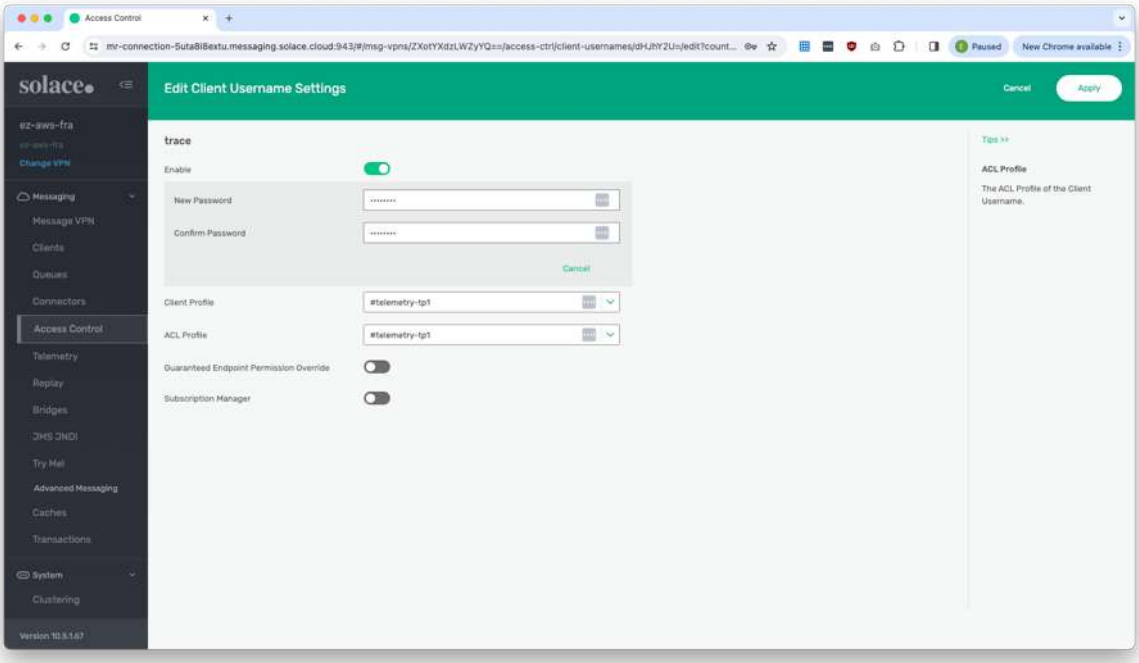




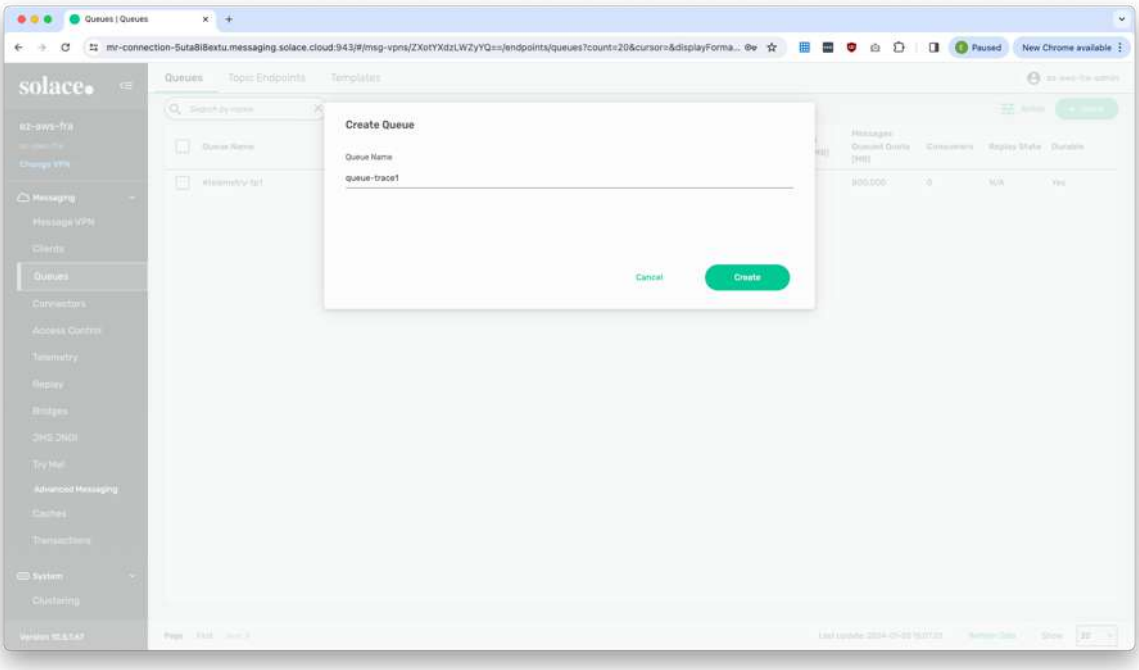
At Access Control go to Client Usernames and click [+ Client Username] to create a client username "trace" with password "trace123" and "#telemetry-tp1" profiles:

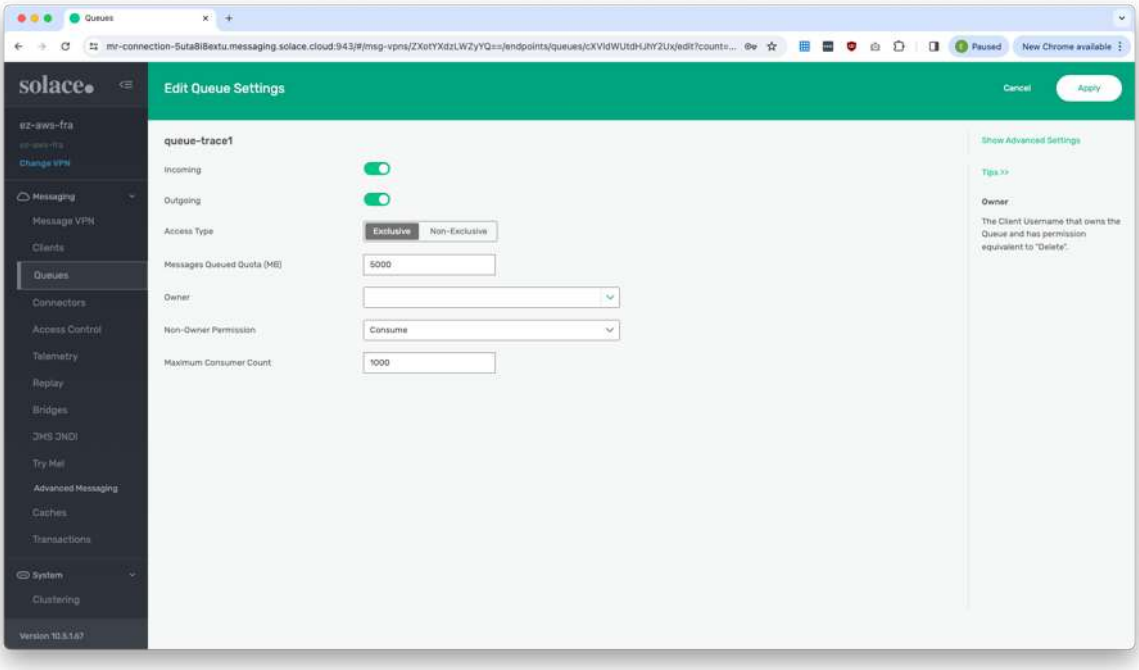


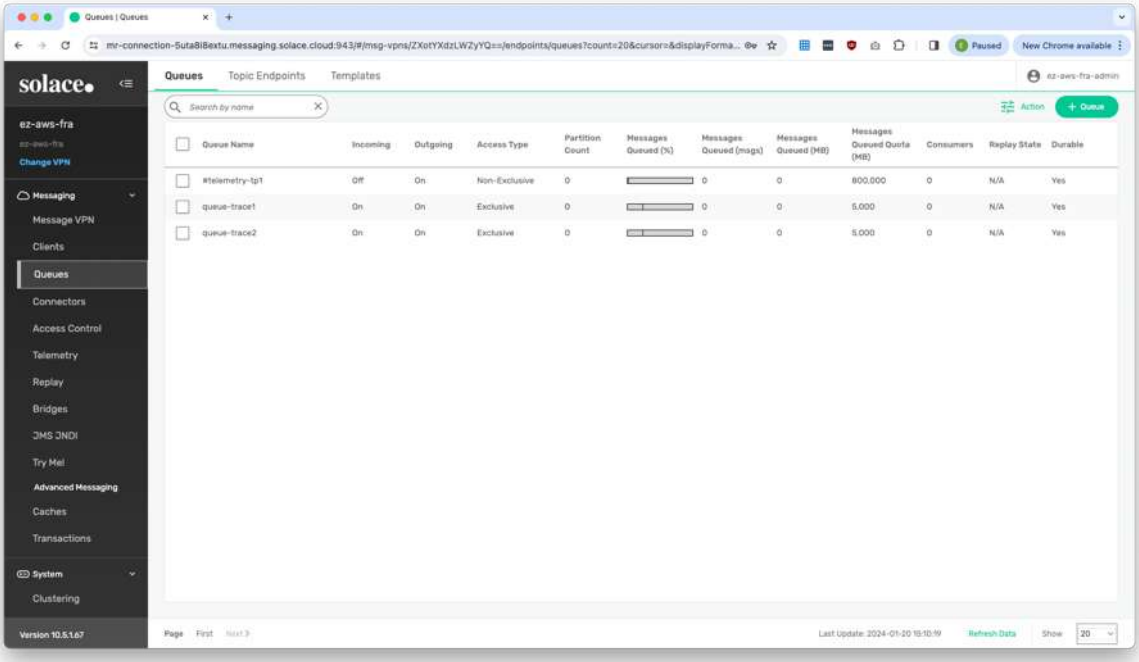

Client Usernames | Access C
mr-connection-5uta8i8extu.messaging.solace.cloud:943/#/msg-vpns/ZXotYXdzLWZyYQ==/access-ctrl/client-usernames?count=20&cursor=&displa...
Paused
New Chrome available
solace
ez-aws-fra
Change VPN
Messaging
Message VPN
Clients
Queues
Connectors
Access Control
Telemetry
Replay
Bridges
JMS JNDI
Try Me!
Advanced Messaging
Caches
Transactions
System
Clustering
Client Authentication
Client Profiles
ACL Profiles
Client Usernames
Create Client Username
Client Username
trace
Cancel
Create

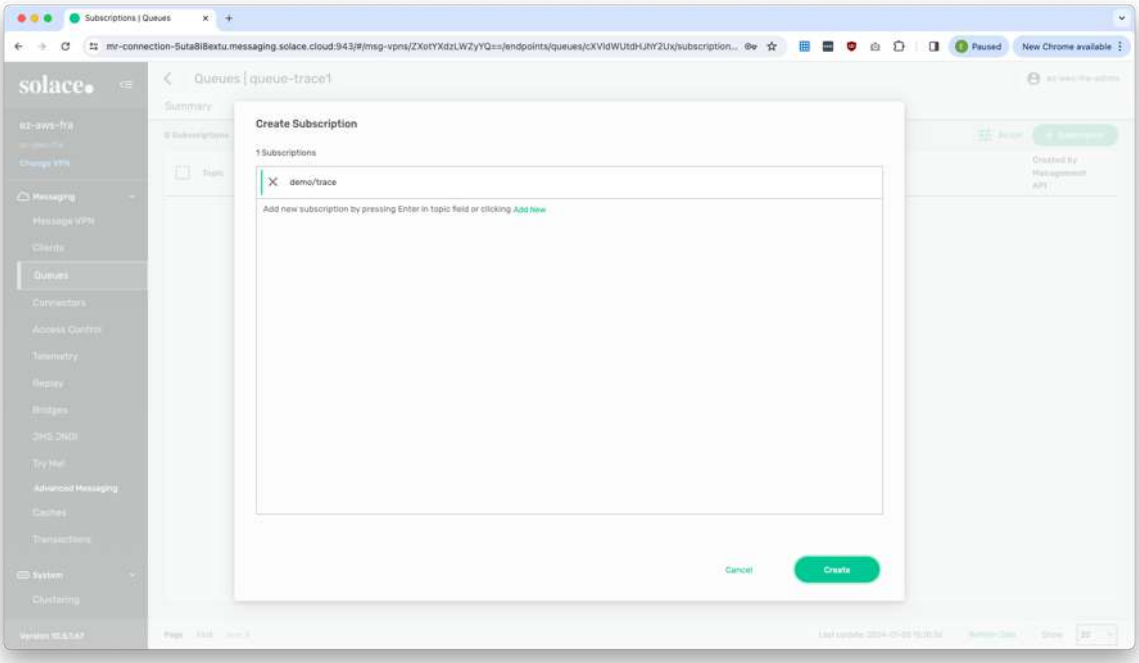

Access Control
mr-connection-5uta8i8extu.messaging.solace.cloud:943/#/msg-vpns/ZXotYXdzLWZyYQ==/access-ctrl/client-usernames/dHJhY2U=/edit?count...
Paused
New Chrome available
solace
Edit Client Username Settings
Cancel
Apply
ez-aws-fra
Change VPN
Messaging
Message VPN
Clients
Queues
Connectors
Access Control
Telemetry
Replay
Bridges
JMS JNDI
Try Me!
Advanced Messaging
Caches
Transactions
System
Clustering
Version 10.5.1.67
trace
Enable
New Password
Confirm Password
Cancel
Client Profile
#telemetry-tp1
ACL Profile
#telemetry-tp1
Guaranteed Endpoint Permission Override
Subscription Manager
Tips >>
ACL Profile
The ACL Profile of the Client Username.

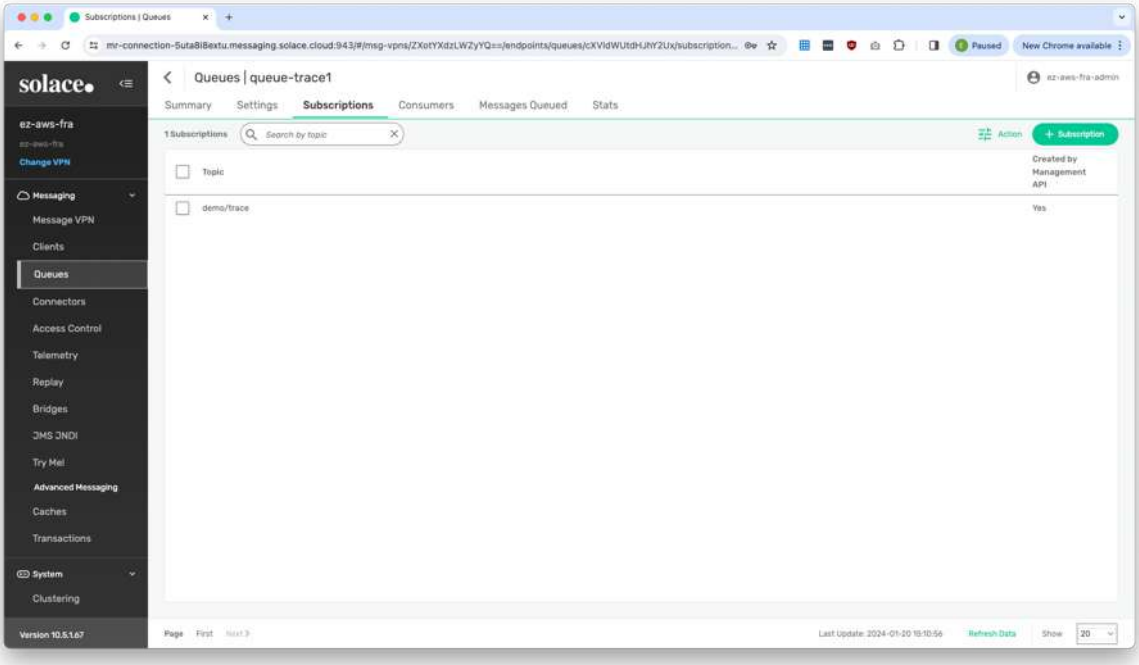
Create two or more queues (queue-trace1, queue-trace2, ...) with subscription "demo/trace". These are the messaging queues.






Queues | Queues
solace
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 10.5.1.67
Queues
Topic Endpoints
Templates
Search by name
Action
+ Queue
Queue Name
Incoming
Outgoing
Access Type
Partition Count
Messages Queued (%)
Messages Queued (msgs)
Messages Queued (MB)
Messages Queued Quota (MB)
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
First
Last Update: 2024-01-20 15:10:19
Refresh Data
Show
20

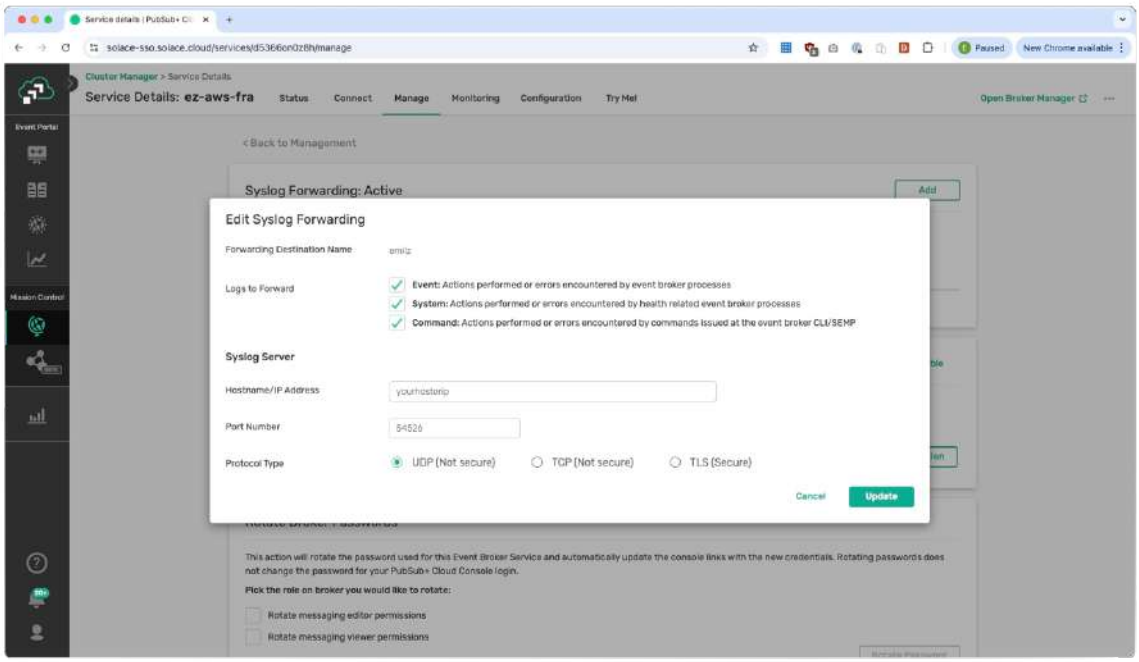

Subscriptions | Queues
solace
Queues | queue-trace1
Summary
Create Subscription
1 Subscriptions
demo/trace
Add new subscription by pressing Enter in topic field or clicking Add New
Cancel
Create



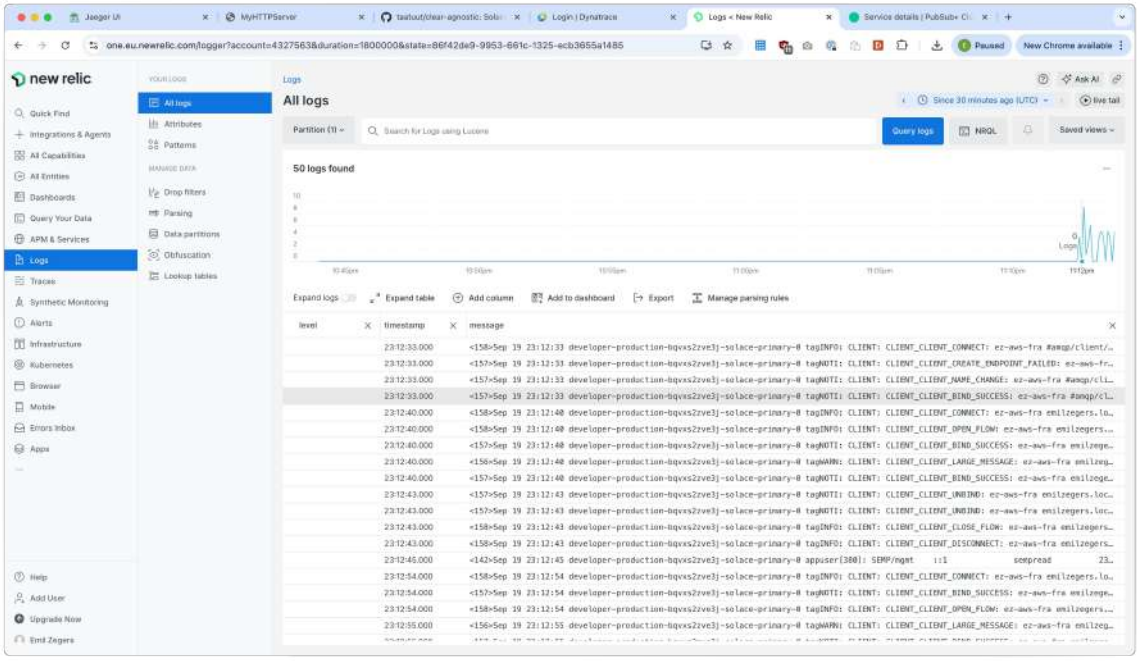
Configuration of the broker is now done.

2.4 Syslog Forwarding

See for configuration https://docs.solace.com/Cloud/cloud-syslog-forwarding.htm



Exported the log data to New Relic (you can have a permanent free account with some restrictions). Looks like Jaeger does not (directly) support ingesting log data? TODO: check, also explore Loki.



When using the JSON export, the log data can also be displayed by the Simple OTEL endpoint Python script. Example log data:


```
{"resourceLogs":[{"resource":{},"scopeLogs":[{"scope":{},"logRecords":[{"timeUnixNano":"17266
72533000000000","observedTimeUnixNano":"1726672533570971000","severityNumber":10,"severityTex
t":"notice","body":{"stringValue":"\u003c157\u003eSep 18 15:15:33 developer-production-
bqvxs2zve3j-solace-primary-0 tagNOTI: CLIENT: CLIENT_CLIENT_UNBIND: ez-aws-fra
emilzegers.local/60503/b12f5c57cd5958cd0001/vGOdYyr7dG Client (325)
emilzegers.local/60503/b12f5c57cd5958cd0001/vGOdYyr7dG username solace-cloud-client Unbind to
Flow Id (576), ForwardingMode(StoreAndForward), final statistics - flow(0, 0, 0, 0, 0, 1, 0,
0, 1, 0), isActive(No), Reason(Client issued
unbind)"},"attributes":[{"key":"facility","value":{"intValue":"19"}},{"key":"hostname","value
":{"stringValue":"developer-production-bqvxs2zve3j-solace-primary-
0"}},{"key":"message","value":{"stringValue":"CLIENT: CLIENT_CLIENT_UNBIND: ez-aws-fra
emilzegers.local/60503/b12f5c57cd5958cd0001/vGOdYyr7dG Client (325)
emilzegers.local/60503/b12f5c57cd5958cd0001/vGOdYyr7dG username solace-cloud-client Unbind to
Flow Id (576), ForwardingMode(StoreAndForward), final statistics - flow(0, 0, 0, 0, 0, 1, 0,
0, 1, 0), isActive(No), Reason(Client issued
unbind)"}},{"key":"priority","value":{"intValue":"157"}},{"key":"appname","value":{"stringVal
ue":"tagNOTI"}}],"traceId":"","spanId":""},{"timeUnixNano":"1726672533000000000","observedTim
eUnixNano":"1726672533602244000","severityNumber":10,"severityText":"notice","body":{"stringV
alue":"\u003c157\u003eSep 18 15:15:33 developer-production-bqvxs2zve3j-solace-primary-0
tagNOTI: CLIENT: CLIENT_CLIENT_UNBIND: ez-aws-fra
emilzegers.local/60503/b12f5c57cd5958cd0001/vGOdYyr7dG Client (325)
emilzegers.local/60503/b12f5c57cd5958cd0001/vGOdYyr7dG username solace-cloud-client Unbind to
Flow Id (1074), ForwardingMode(StoreAndForward), final statistics - flow(0, 0, 0, 0, 0, 1, 0,
0, 1, 0), isActive(No), Reason(Client issued
unbind)"},"attributes":[{"key":"priority","value":{"intValue":"157"}},{"key":"facility","valu
e":{"intValue":"19"}},{"key":"hostname","value":{"stringValue":"developer-production-
bqvxs2zve3j-solace-primary-
0"}},{"key":"appname","value":{"stringValue":"tagNOTI"}},{"key":"message","value":{"stringVal
ue":"CLIENT: CLIENT_CLIENT_UNBIND: ez-aws-fra
emilzegers.local/60503/b12f5c57cd5958cd0001/vGOdYyr7dG Client (325)
emilzegers.local/60503/b12f5c57cd5958cd0001/vGOdYyr7dG username solace-cloud-client Unbind to
Flow Id (1074), ForwardingMode(StoreAndForward), final statistics - flow(0, 0, 0, 0, 0, 1, 0,
0, 1, 0), isActive(No), Reason(Client issued
unbind)"}}],"traceId":"","spanId":""},{"timeUnixNano":"1726672533000000000","observedTimeUnix
Nano":"1726672533631824000","severityNumber":9,"severityText":"info","body":{"stringValue":"\
u003c158\u003eSep 18 15:15:33 developer-production-bqvxs2zve3j-solace-primary-0 tagINFO:
CLIENT: CLIENT_CLIENT_CLOSE_FLOW: ez-aws-fra
emilzegers.local/60503/b12f5c57cd5958cd0001/vGOdYyr7dG Client (325)
emilzegers.local/60503/b12f5c57cd5958cd0001/vGOdYyr7dG username solace-cloud-client Pub flow
session flow name 50e5f354f0154822a429ca719cdb87ac (1052), transacted session id -1,
publisher id 1001, last message id 117988, window size 50, final statistics - flow(0, 0, 0,
```

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1)"},"attributes":[{"key":"appname","value":{"stringValue":"tagINFO"}},{"key":"priority","val
ue":{"intValue":"158"}},{"key":"facility","value":{"intValue":"19"}},{"key":"hostname","value
":{"stringValue":"developer-production-bqvxs2zve3j-solace-primary-
0"}},{"key":"message","value":{"stringValue":"CLIENT: CLIENT_CLIENT_CLOSE_FLOW: ez-aws-fra
emilzegers.local/60503/b12f5c57cd5958cd0001/vGOdYyr7dG Client (325)
emilzegers.local/60503/b12f5c57cd5958cd0001/vGOdYyr7dG username solace-cloud-client Pub flow
session flow name 50e5f354f0154822a429ca719cdb87ac (1052), transacted session id -1,
publisher id 1001, last message id 117988, window size 50, final statistics - flow(0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1)"}}],"traceId":"","spanId":""},{"timeUnixNano":"1726672533000000000","observedTimeUnixNano"
:"1726672533639511000","severityNumber":9,"severityText":"info","body":{"stringValue":"\u003c
158\u003eSep 18 15:15:33 developer-production-bqvxs2zve3j-solace-primary-0 tagINFO: CLIENT:
CLIENT_CLIENT_DISCONNECT: ez-aws-fra emilzegers.local/60503/b12f5c57cd5958cd0001/vGOdYyr7dG
Client (325) emilzegers.local/60503/b12f5c57cd5958cd0001/vGOdYyr7dG username solace-cloud-
client WebSessionId (N/A) reason(Peer TCP Closed) final statistics - dp(14, 9, 1, 2, 15, 11,
974, 884, 33001, 66116, 33975, 67000, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) conn(0, 0,
95.97.192.6:52946, ESTAB, 0, 0, 0) zip(0, 0, 0, 0, 0.00, 0.00, 0, 0, 0, 0, 0, 0, 0, 0) web(0,
0, 0, 0, 0, 0, 0), SslVersion(TLSv1.2), SslCipher(ECDHE-RSA-AES256-SHA384 TLSv1.2 Kx=ECDH
Au=RSA Enc=AES(256)
Mac=SHA384)"},"attributes":[{"key":"facility","value":{"intValue":"19"}},{"key":"message","va
lue":{"stringValue":"CLIENT: CLIENT_CLIENT_DISCONNECT: ez-aws-fra
emilzegers.local/60503/b12f5c57cd5958cd0001/vGOdYyr7dG Client (325)
emilzegers.local/60503/b12f5c57cd5958cd0001/vGOdYyr7dG username solace-cloud-client
WebSessionId (N/A) reason(Peer TCP Closed) final statistics - dp(14, 9, 1, 2, 15, 11, 974,
884, 33001, 66116, 33975, 67000, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) conn(0, 0, 95.97.192.6:52946,
ESTAB, 0, 0, 0) zip(0, 0, 0, 0, 0.00, 0.00, 0, 0, 0, 0, 0, 0, 0, 0) web(0, 0, 0, 0, 0, 0, 0),
SslVersion(TLSv1.2), SslCipher(ECDHE-RSA-AES256-SHA384 TLSv1.2 Kx=ECDH Au=RSA Enc=AES(256)
Mac=SHA384)"}},{"key":"priority","value":{"intValue":"158"}},{"key":"hostname","value":{"stri
ngValue":"developer-production-bqvxs2zve3j-solace-primary-
0"}},{"key":"appname","value":{"stringValue":"tagINFO"}}],"traceId":"","spanId":""}]}]}]}

Some yaml excerpts below.

Add New Relic exporter to exporters section:

```
 otlphttp/newrelic:
   endpoint: ${NEW_RELIC_EXPORTER_OTLP_ENDPOINT}
   tls:
     insecure: false
   headers:
     api-key: ${NEW_RELIC_LICENSE_KEY}
```

Service section:

```
service:
 telemetry:
   logs:
     level: "debug"
 pipelines:
   logs:
     receivers: [syslog]
     exporters: [logging, otlphttp/newrelic, otlphttp/jsontest]
   traces:
     receivers: [solace/broker1, otlp]
     processors: [batch]
#      exporters: [logging, otlp/jaeger, otlphttp/newrelic, otlphttp/dynatrace, otlphttp/jsontest]
     exporters: [logging, otlp/jaeger, otlphttp/newrelic, otlphttp/jsontest]
```

See section OTEL Collector for full YAML configuration information.

2.5 Jaeger

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

2.6 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:

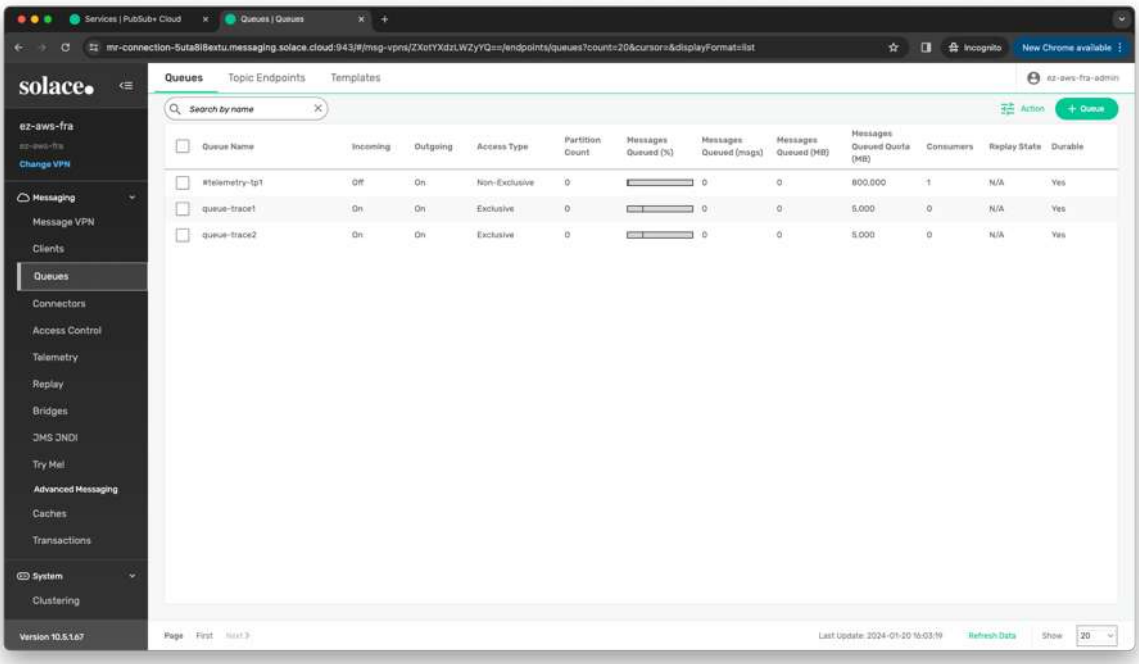
```
sudo spctl --master-disable
```

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:



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

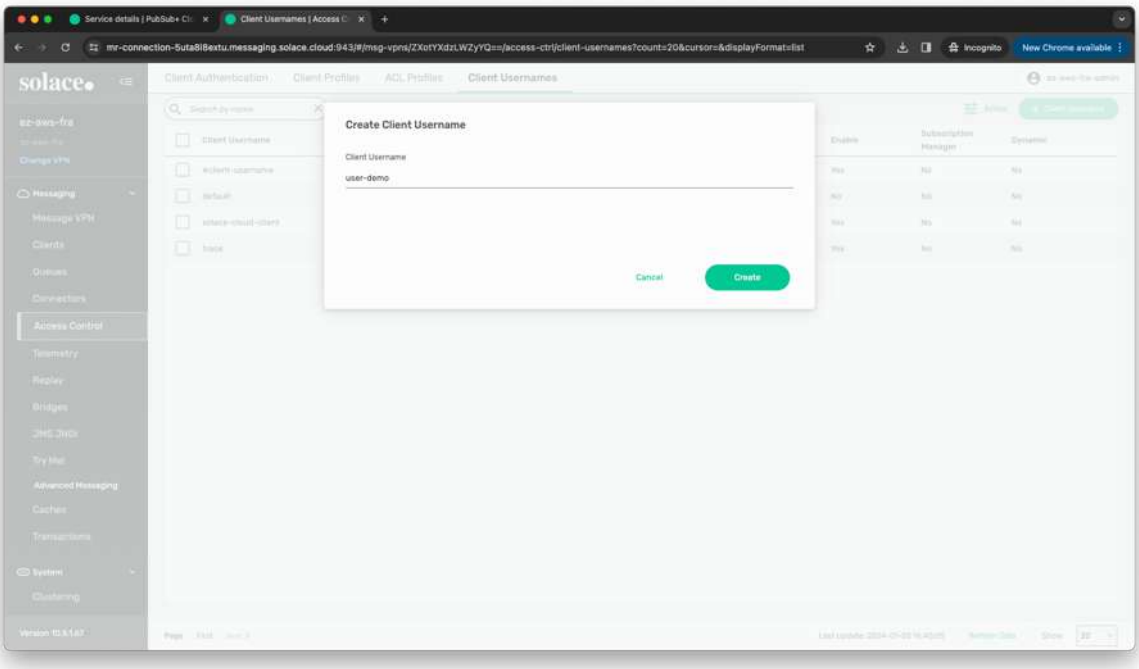
```
mkdir -p ~/sdkperf
cd ~/sdkperf
tar xzvf <your download directory>/sdkperf-jcsmp-8.4.14.10.zip
tar xzvf <your download directory>/sdkperf-mqtt-8.4.15.5.zip
```

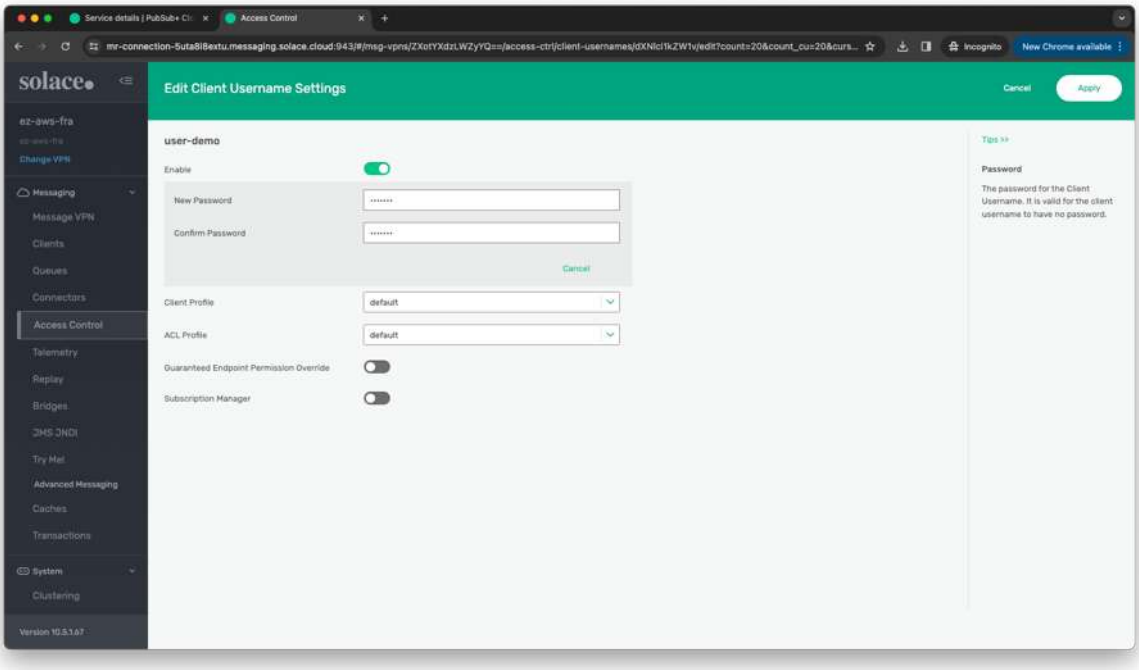
Gives output like:

```
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.





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=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"
```

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 Simple OTEL endpoint from top to bottom).

4 Results

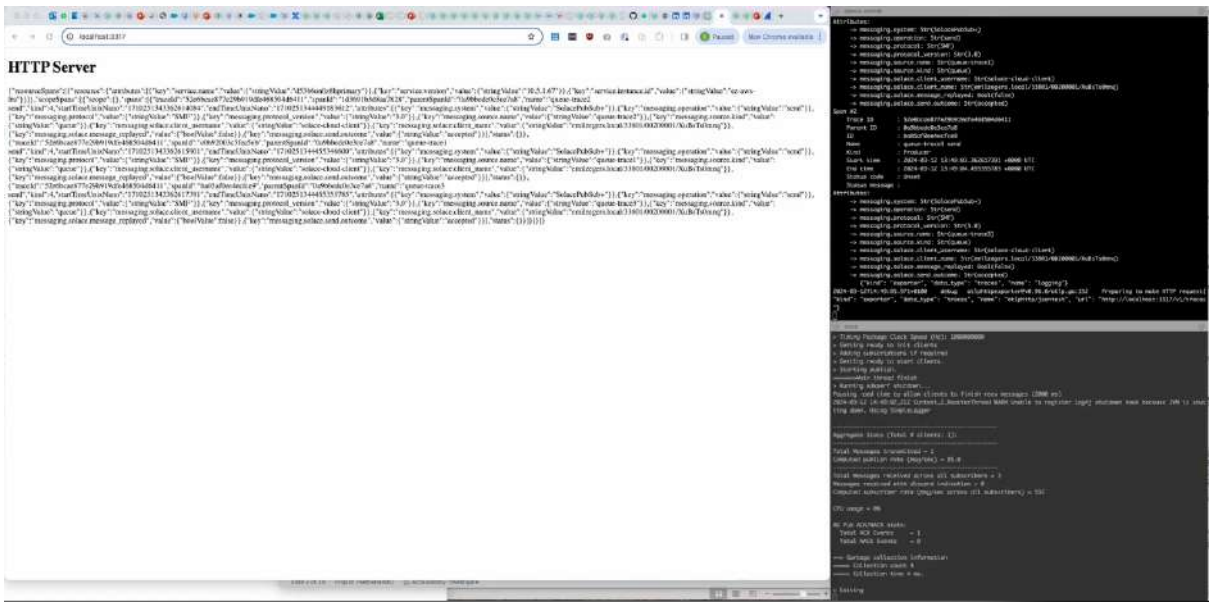
4.1 Simple OTEL endpoint

The Simple OTEL endpoint Python script processes POST requests from OTEL collector exporter with metrics in JSON and just displays the data received. See the simpleotelendpoint.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).

JAEGER UI Search Compare System Architecture Monitor

About Jaeger

sdkperf_java: demo/trace sdkperf_java_create d0c5a08

Trace Timeline Archive Trace

Trace Start January 21 2024, 11:06:36.000 Duration 3.12s Services 2 Depth 4 Total Spans 6

Service & Operation 0µs 789.84ms 1.58s 2.38s 3.17s

sdkperf_java demo/trace sdkperf_java_create 1.78ms

demo/trace sdkperf_java_create Service: sdkperf_java Duration: 1.78ms Start Time: 0µs

Tags

internal.span.format	otlp
span.kind	producer

Process

otel.library.name	sdkperf_java_create
telemetry.sdk.language	java
telemetry.sdk.name	opentelemetry
telemetry.sdk.version	1.20.0

d5360un0r0hprimary queue receive 91µs

(topic) receive Service: d5360un0r0hprimary Duration: 16µs Start Time: 2.07s

Tags

internal.span.format	otlp
messaging.destination	demo/trace
messaging.message_payload_size_bytes	32836
messaging.operation	receive
messaging.protocol	SMF
messaging.protocol_version	3.0
messaging.solace.broker.receive_time_utc_nano	1705823590778724750
messaging.solace.client_name	emilsepers.local/50820/0020001/1200mbLvy
messaging.solace.client_username	solace-cloud-client
messaging.solace.delivery_mode	persistent
messaging.solace.dmq_eligible	false
messaging.solace.dropped_application_message_properties	false
messaging.solace.dropped_propsvce_events_failed	0
messaging.solace.dropped_propsvce_events_success	0

Process

service.instance.id	mr-aws-1ra
service.version	10.5.1.47

Logs (2)

2.07s

event	queue-trace2 enqueue
messaging.solace.destination_type	queue
messaging.solace.rejects_all_producers	false

2.07s

event	queue-trace1 enqueue
messaging.solace.destination_type	queue
messaging.solace.rejects_all_producers	false

Log timestamps are relative to the start time of the full trace.

Warnings (1)

clock skew adjustment disabled; not applying calculated delta of -2071424394

Span: d0ca940354d2d8f

d5360un0r0hprimary queue trace2 send

sdkperf_java demo/trace sdkperf_java_process 1.30µs

demo/trace sdkperf_java_process Service: sdkperf_java Duration: 21µs Start Time: 120.08ms

Tags: internal.span.format = otlp span.kind = consumer

Process: otel.library.name = sdkperf_java_process telemetry.sdk.language = java telemetry.sdk.name = opentelemetry telemetry.sdk.version = 1.29.0

Warnings (1)

clock skew adjustment disabled; not applying calculated delta of 2503092026

Span: 9f9f8fc0d1b1ecf

d5360un0r0hprimary queue trace1 send

queue-trace1 send Service: d5360un0r0hprimary Duration: 1.1s Start Time: 2.07s

Tags

internal.span.format	otlp
messaging.operation	send
messaging.protocol	SMF
messaging.protocol_version	3.0
messaging.solace.client_name	emilsepers.local/50820/0020001/L2n8bdavy
messaging.solace.client_username	solace-cloud-client
messaging.solace.message_replayed	false
messaging.solace.send.outcome	accepted
messaging.source.kind	queue
messaging.source.name	queue-trace1
messaging.system	SolacePubSub+
span.kind	producer

Process

service.instance.id	mr-aws-1ra
service.version	10.5.1.47

Span: e6b7eccc1bc8b8b

sdkperf_java demo/trace sdkperf_java_process 1.40µs

demo/trace sdkperf_java_process Service: sdkperf_java Duration: 94µs Start Time: 123.21ms

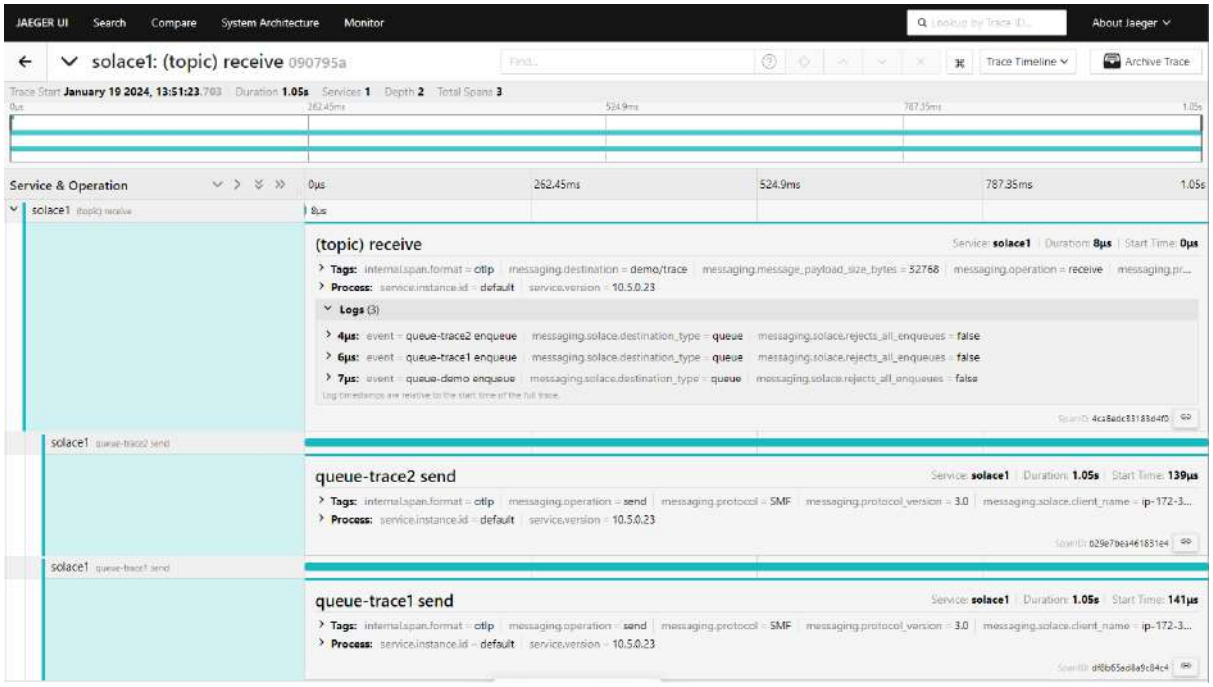
Tags: internal.span.format = otlp span.kind = consumer

Process: otel.library.name = sdkperf_java_process telemetry.sdk.language = java telemetry.sdk.name = opentelemetry telemetry.sdk.version = 1.29.0

Warnings (1)

clock skew adjustment disabled; not applying calculated delta of 2503202475s

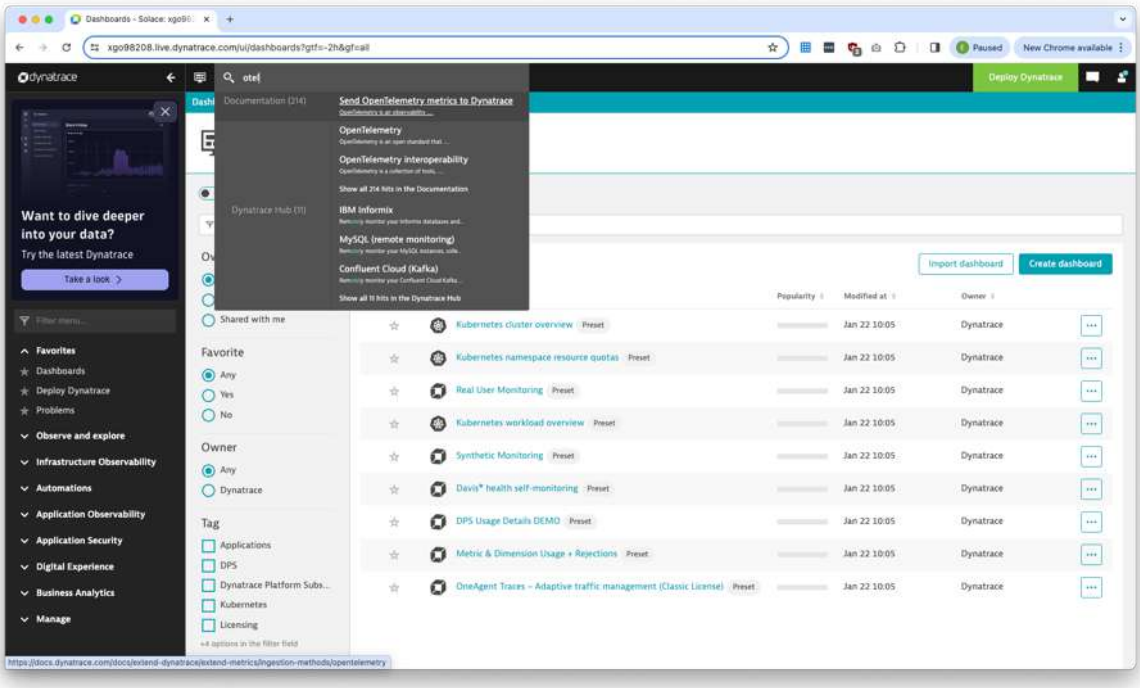
Span: 0d0852f207b4b5f3



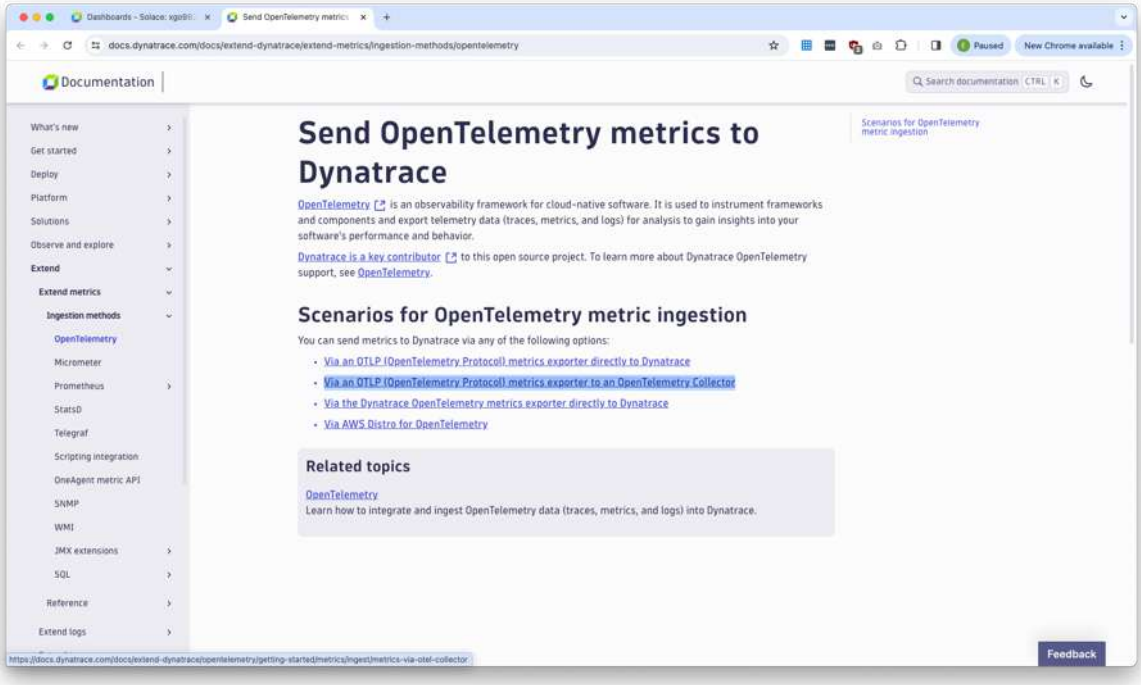
4.2.1 Loki

TODO

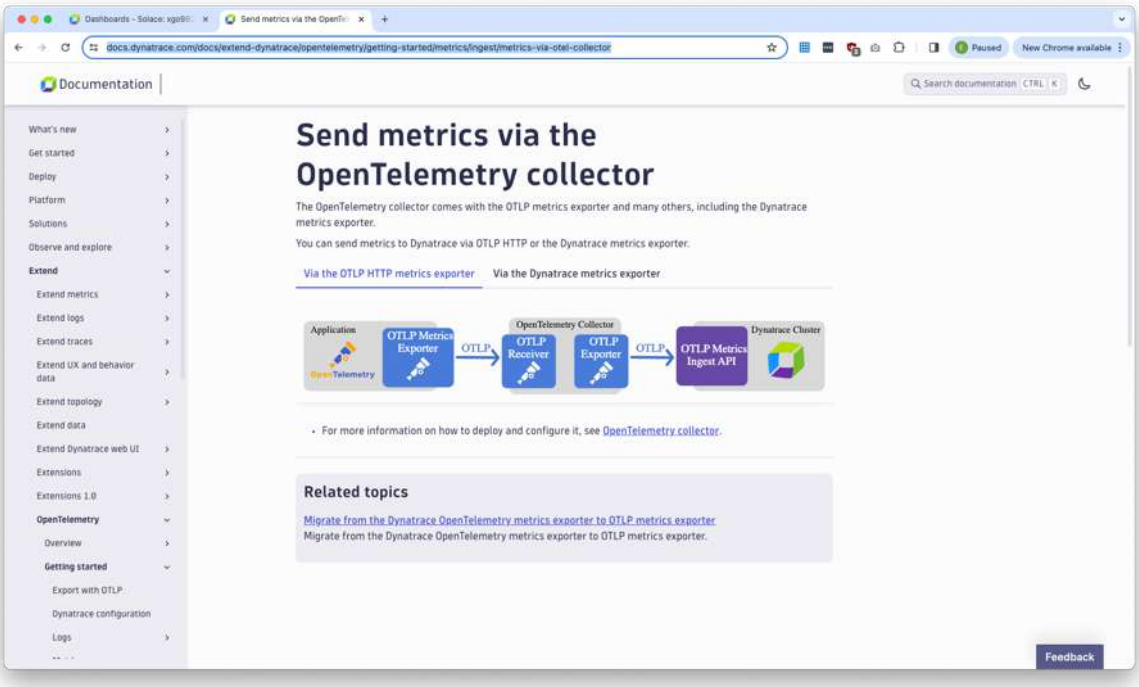
4.3 Dynatrace



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



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



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

Dashboards - Solace: xgo98 | Collector deployment - Dyna | Collector Configuration - Dy

docs.dynatrace.com/docs/extend-dynatrace/opentelemetry/collector/configuration

Paused | New Chrome available

Documentation | Search documentation CTRL K

What's new
Get started
Deploy
Platform
Solutions
Observe and explore
Extend
Extend metrics
Extend logs
Extend traces
Extend UX and behavior data
Extend topology
Extend data
Extend Dynatrace web UI
Extensions
Extensions 1.0
OpenTelemetry
Overview
Getting started
Integration walk-throughs
Collector
Deploy

Configuration example

Here is an example YAML file for a very basic Collector configuration that can be used to export OpenTelemetry traces, metrics, and logs to Dynatrace.

```
      grpc:
      http:

processors:
  cumulativetodelta:

exporters:
  otlphttp:
    endpoint: "https://{your-environment-id}.live.dynatrace.com/api/v2/otlp"
    headers:
      Authorization: "Api-Token ${API_TOKEN}"

service:
  pipelines:
    traces:
      receivers: [otlp]
      processors: []
      exporters: [otlphttp]
    metrics:
      receivers: [otlp]
      processors: [cumulativetodelta]
      exporters: [otlphttp]
    logs:
      receivers: [otlp]
      processors: []
      exporters: [otlphttp]
```

In this YAML file, we configure the following components:

- An OTLP 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)

Configuration example
Where to place the config
Delta metrics
Chained and load-balanced Collectors
API tokens

Feedback

Dashboards - Solace: xgo98 | Migrate from the Dynatrace | Collector deployment - Dyna | Collector Configuration - Dy

xgo98208.live.dynatrace.com/ui/dashboards?gtf=-2h&gf=all

Paused | New Chrome available

dynatrace | Search Solace: xgo98208... | Deploy Dynatrace

Latest Dynatrace

Favorites
Dashboards
Deploy Dynatrace
Problems
Observe and explore
Infrastructure Observability
Automations
Application Observability
Application Security
Digital Experience
Business Analytics
Manage

Dashboards

Dashboards

Overview of all dashboards you are permitted to view or edit.

Show all tenant dashboards (for admin users only)

Filter by

Ownership: Any, Mine, Shared with me
Favorite: Any, Yes, No
Owner: Any, Dynatrace
Tag: Applications, DPS, Dynatrace Platform Subs..., Kubernetes, Licensing
+4 options in the filter field

9 Dashboards

Favorite	Name	Popularity	Modified at
☆	Kubernetes cluster overview Preset		Jan 22 10:05
☆	Kubernetes namespace resource quotas Preset		Jan 22 10:05
☆	Real User Monitoring Preset		Jan 22 10:05
☆	Kubernetes workload overview Preset		Jan 22 10:05
☆	Synthetic Monitoring Preset		Jan 22 10:05
☆	Davis® health self-monitoring Preset		Jan 22 10:05
☆	DPS Usage Details DEMO Preset		Jan 22 10:05
☆	Metric & Dimension Usage + Rejections Preset		Jan 22 10:05
☆	OneAgent Traces – Adaptive traffic management (Classic License) Preset		Jan 22 10:05

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

Support Resources
Support Center
Release notes
Documentation
University
Community
Product ideas

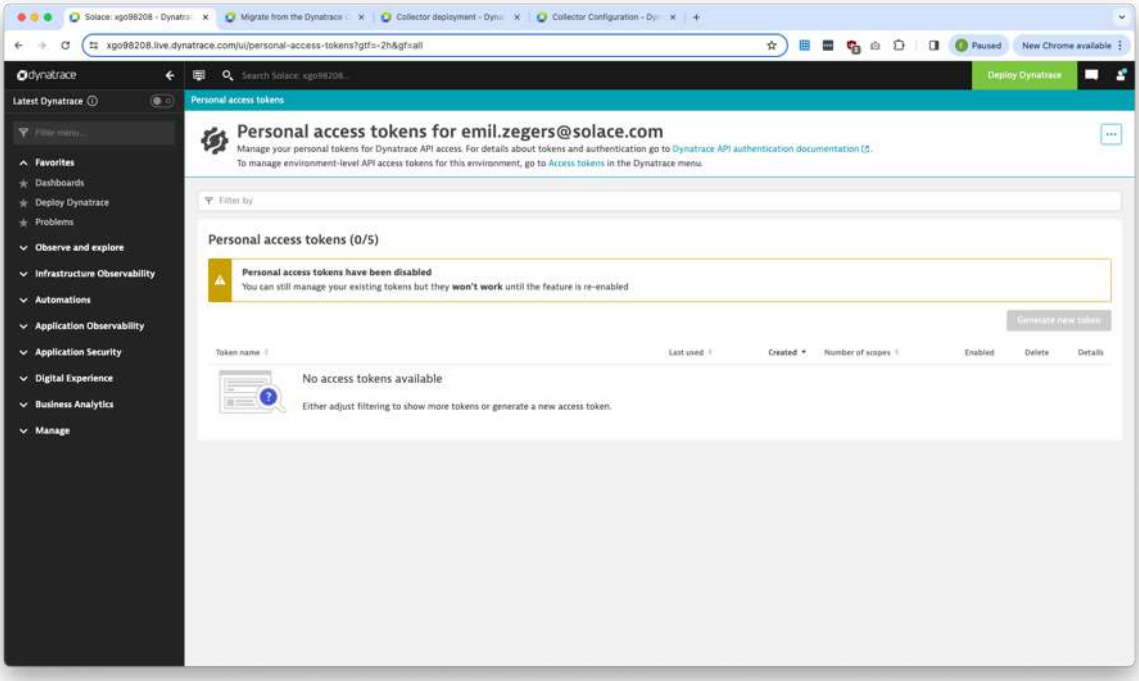
Dynatrace API
Environment API v2
Environment API v1
Configuration API
Personal access tokens

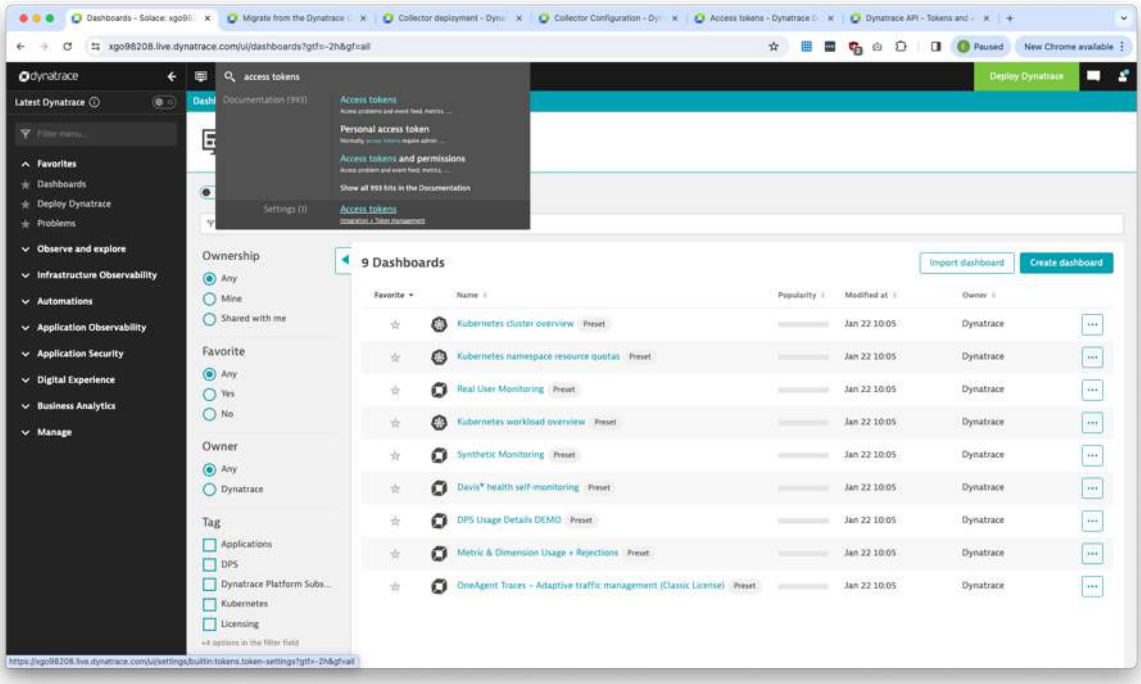
Mobile apps
Receive alerts via mobile app

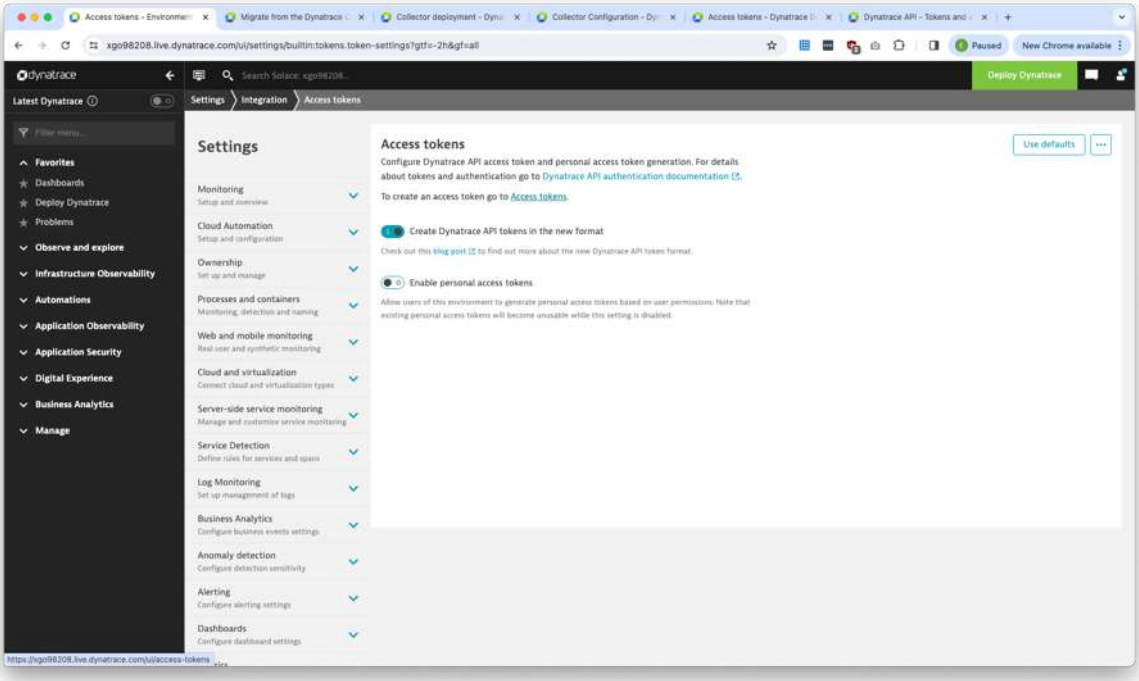
Sign out

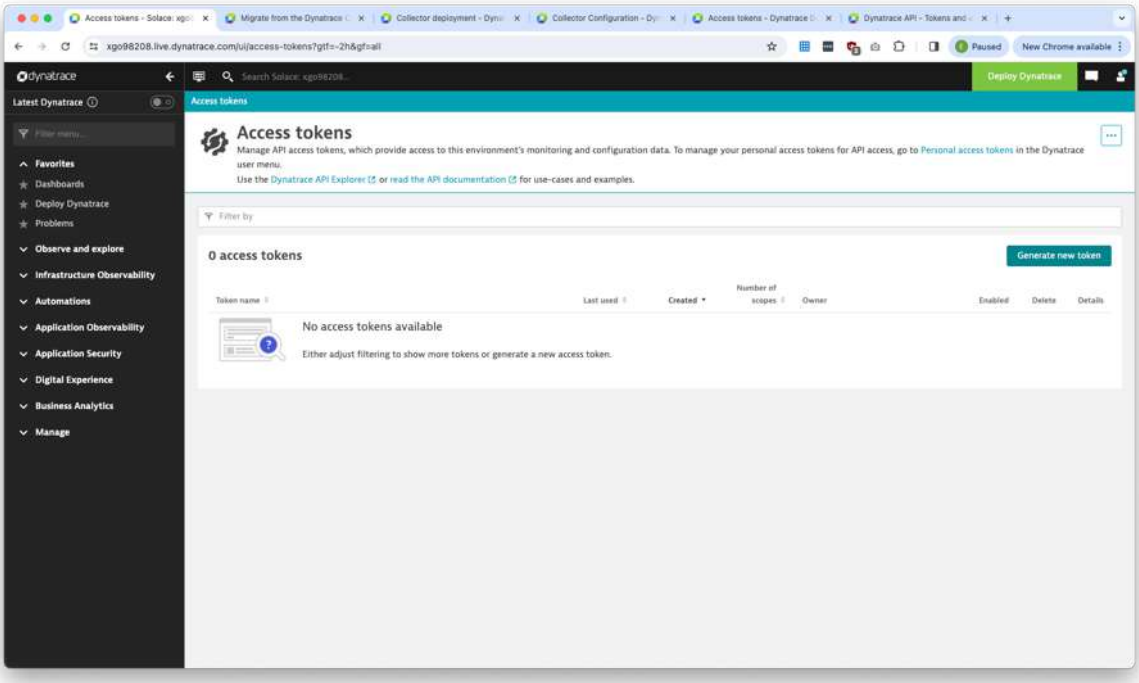
Dynatrace version 1.263.64

https://xgo98208.live.dynatrace.com/ui/personal-access-tokens?gtf=-2h&gf=all


xgo98208.live.dynatrace.com/ui/personal-access-tokens?gtf=-2h&gf=all
Personal access tokens
Personal access tokens for emil.zegers@solace.com
Manage your personal tokens for Dynatrace API access. For details about tokens and authentication go to Dynatrace API authentication documentation.
To manage environment-level API access tokens for this environment, go to Access tokens in the Dynatrace menu.
Personal access tokens (0/5)
Personal access tokens have been disabled
You can still manage your existing tokens but they won't work until the feature is re-enabled
Generate new token
Token name
Last used
Created
Number of scopes
Enabled
Delete
Details
No access tokens available
Either adjust filtering to show more tokens or generate a new access token.

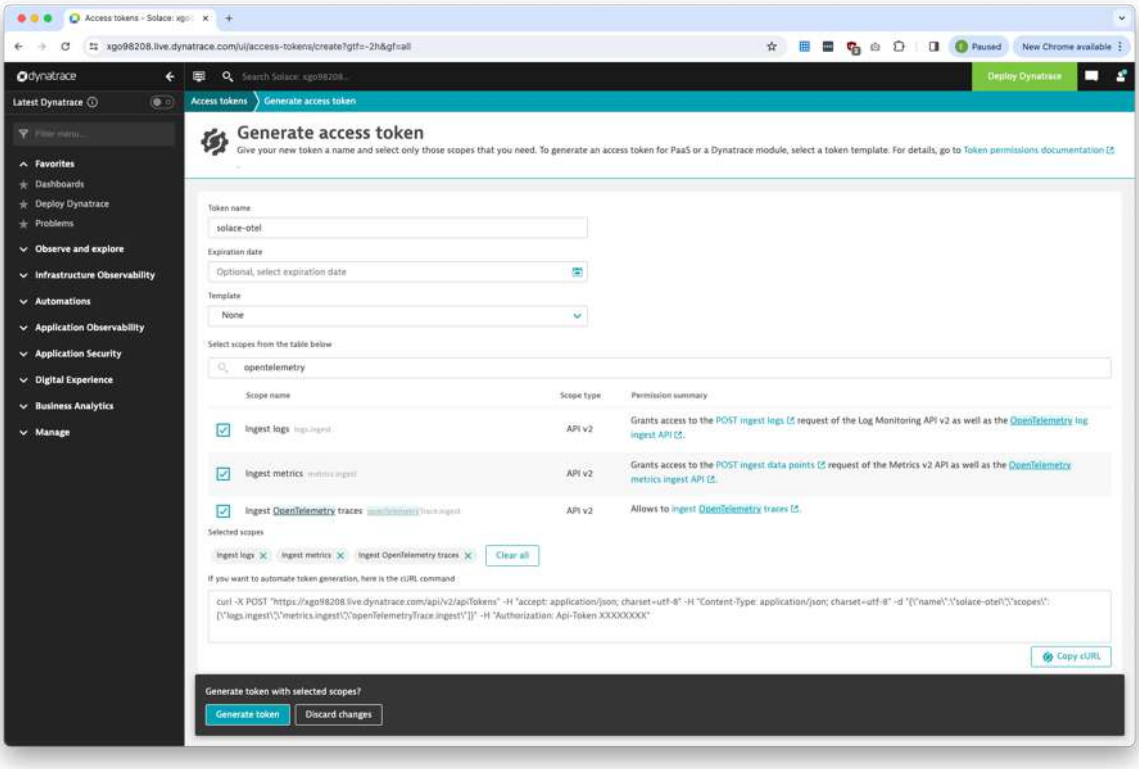

xgo98208.live.dynatrace.com/ui/dashboards?gtf=-2h&gf=all
access tokens
Access tokens
Personal access token
Access tokens and permissions
Access tokens
Ownership
Any
Mine
Shared with me
Favorite
Any
Yes
No
Owner
Any
Dynatrace
Tag
Applications
DPS
Dynatrace Platform Subs...
Kubernetes
Licensing
9 Dashboards
Import dashboard
Create dashboard
Favorite
Name
Popularity
Modified at
Owner
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)
Jan 22 10:05
Dynatrace



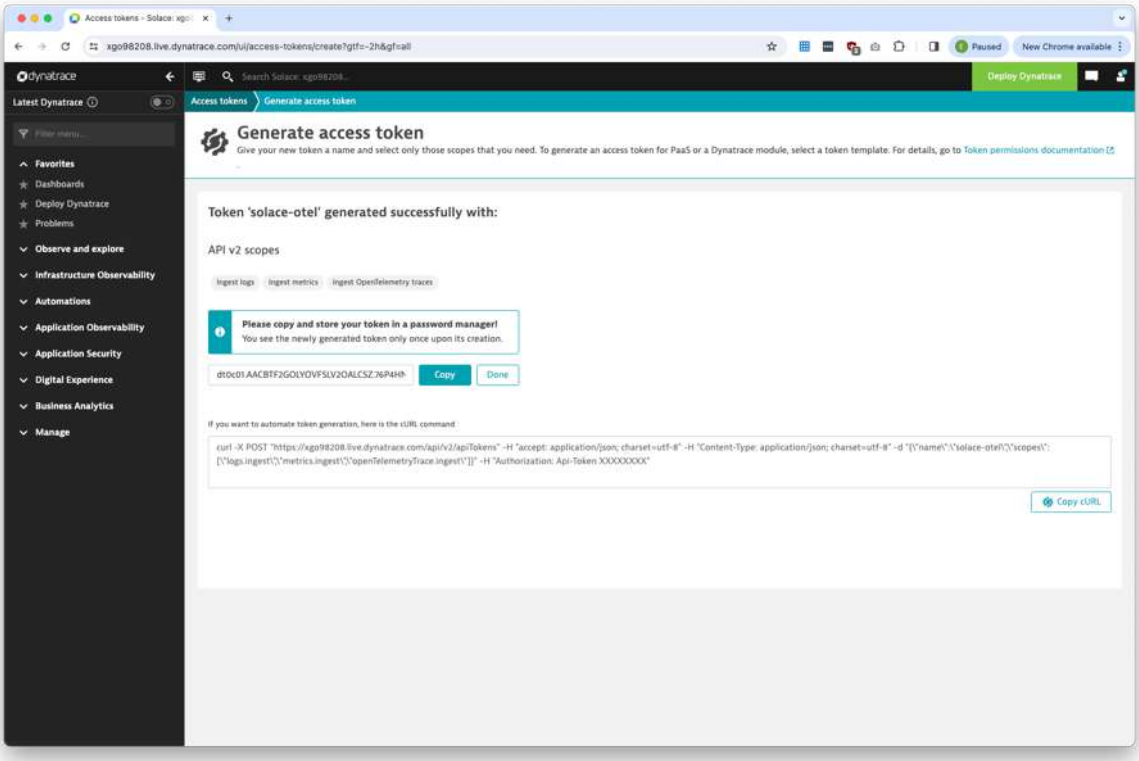


Token name: solace-otel

Search for OpenTelemetry and add Scopes

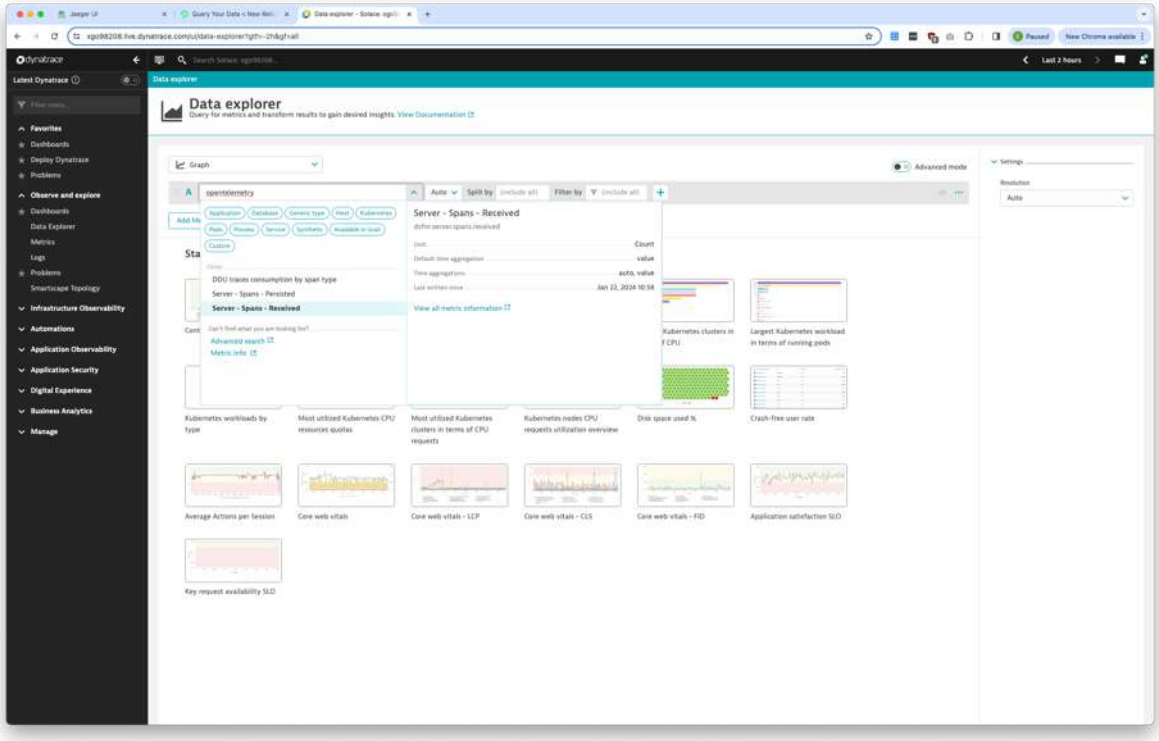


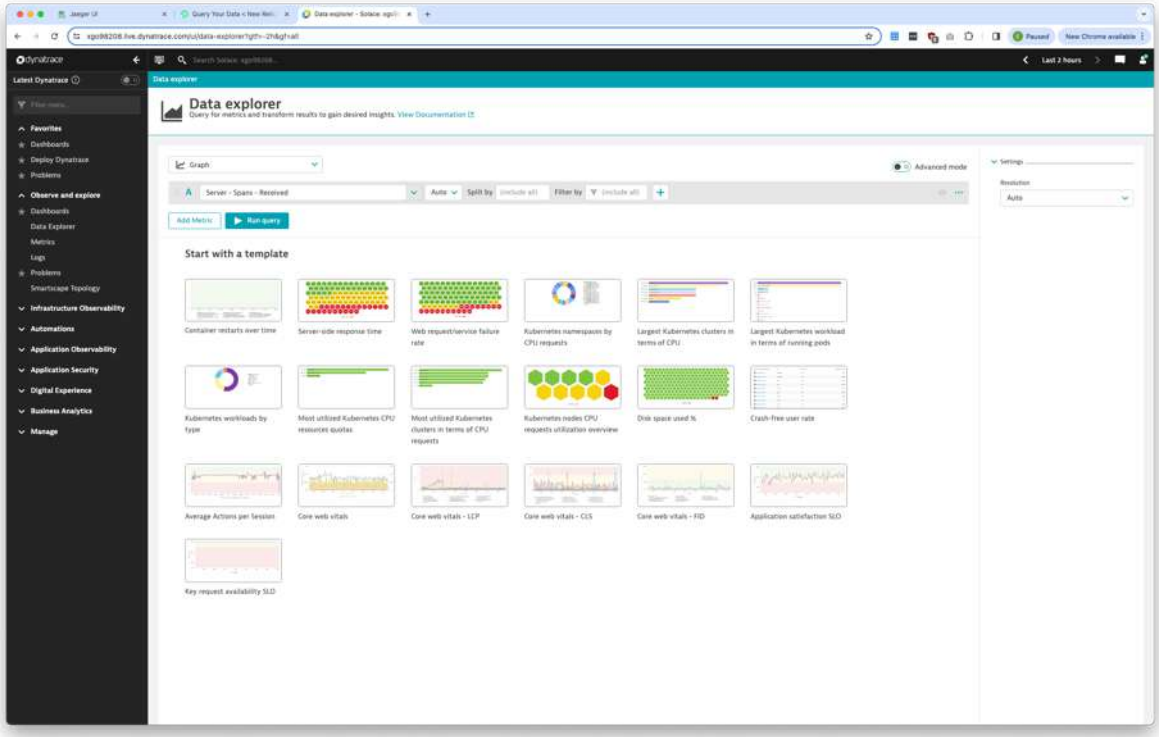
Click [Generate token]



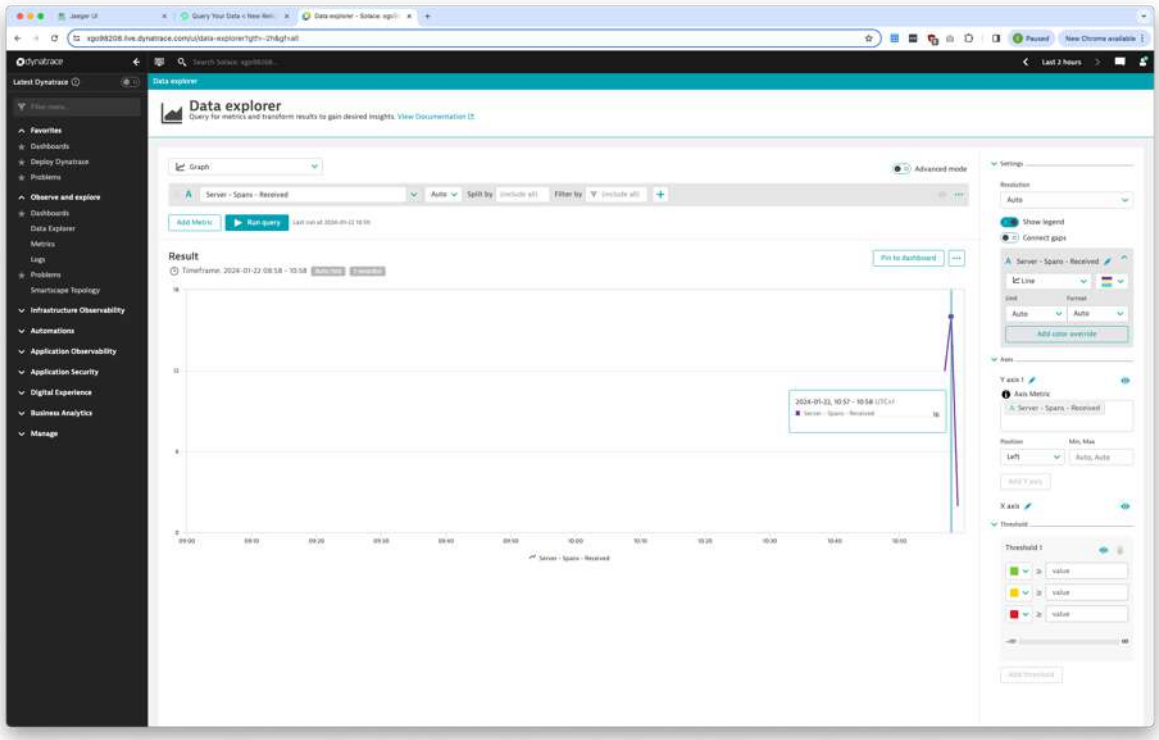
Copy token and add to YAML file.

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

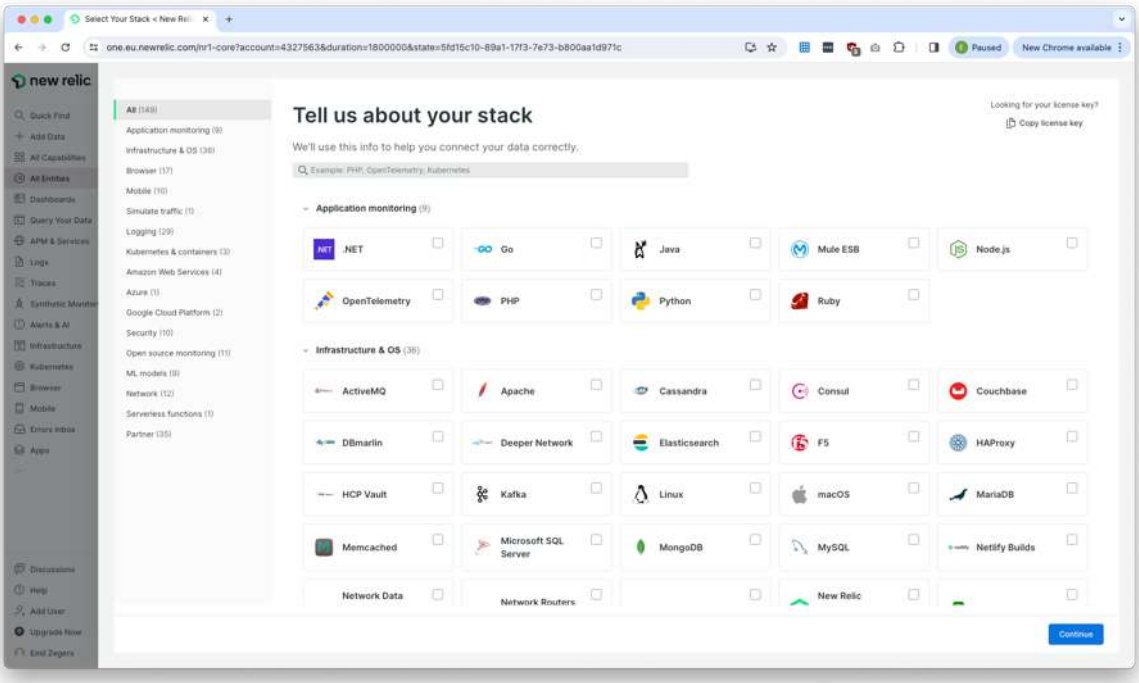


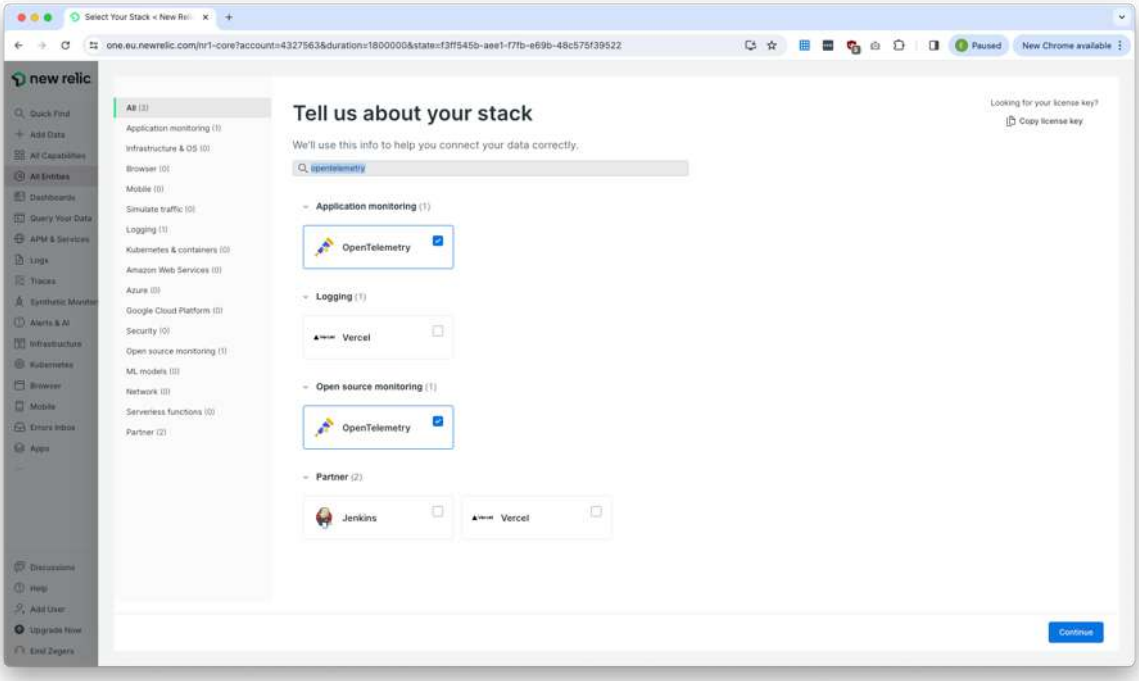


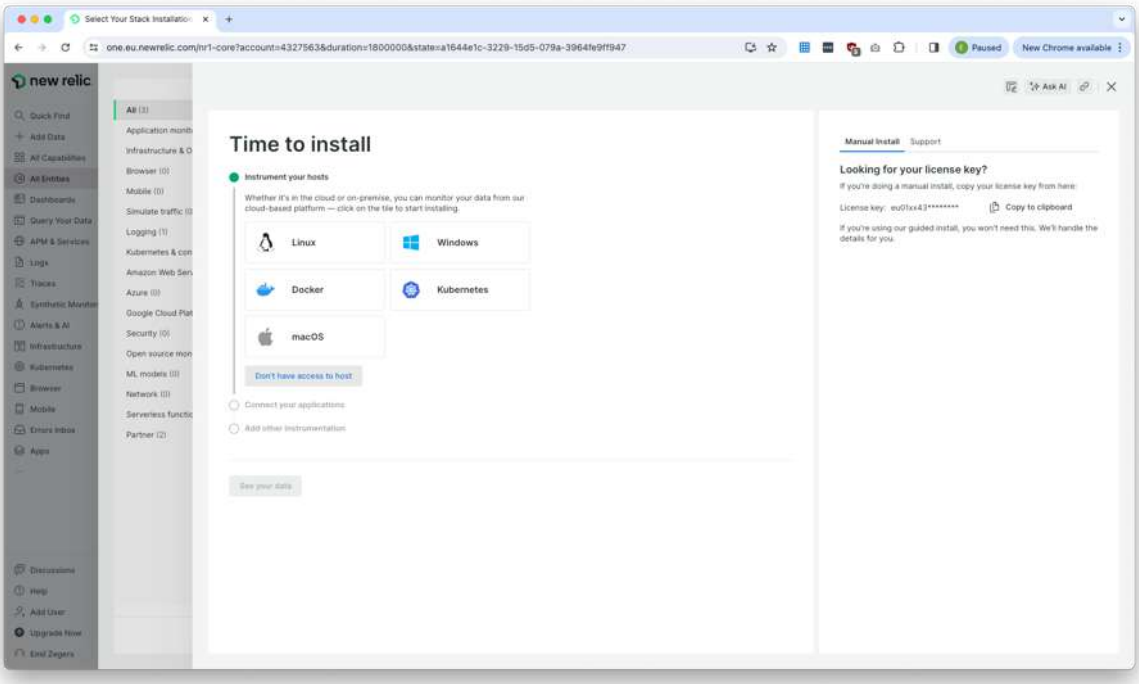
Click [Run query]

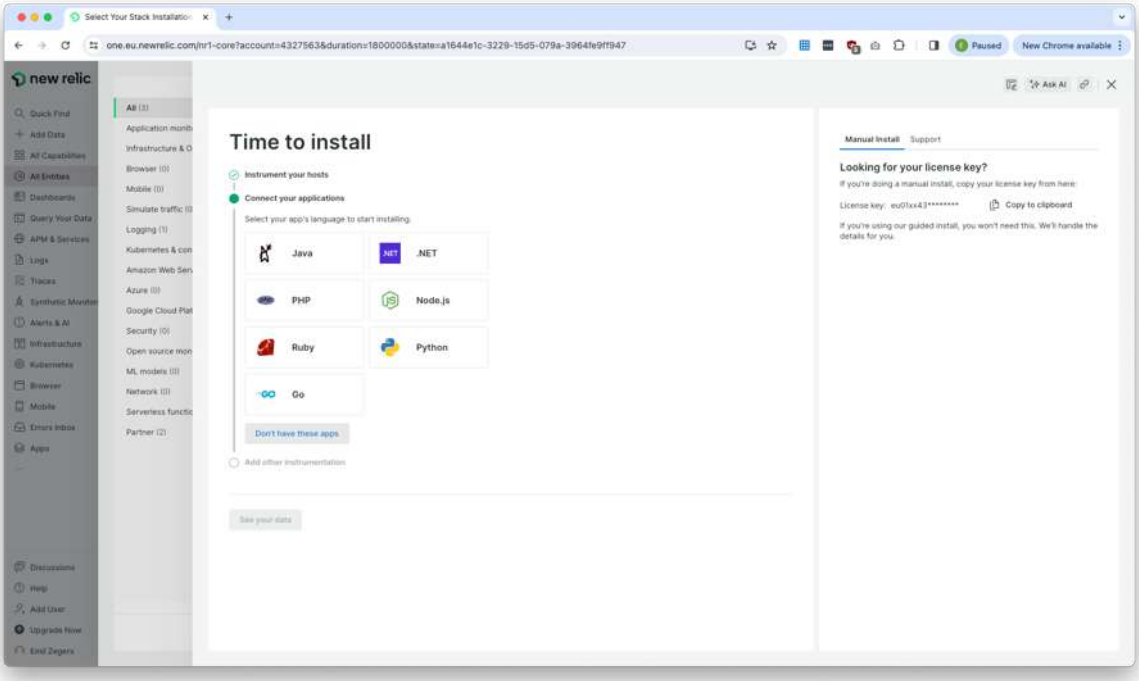

Data explorer
Query for metrics and transform results to gain desired insights.
Graph
Server - Spans - Received
Add Metric
Run query
Result
Pin to dashboard
Advanced mode
Settings
Show legend
Connect gaps
Line
Add color override
Y axis 1
Axis Metric
X axis
Threshold 1

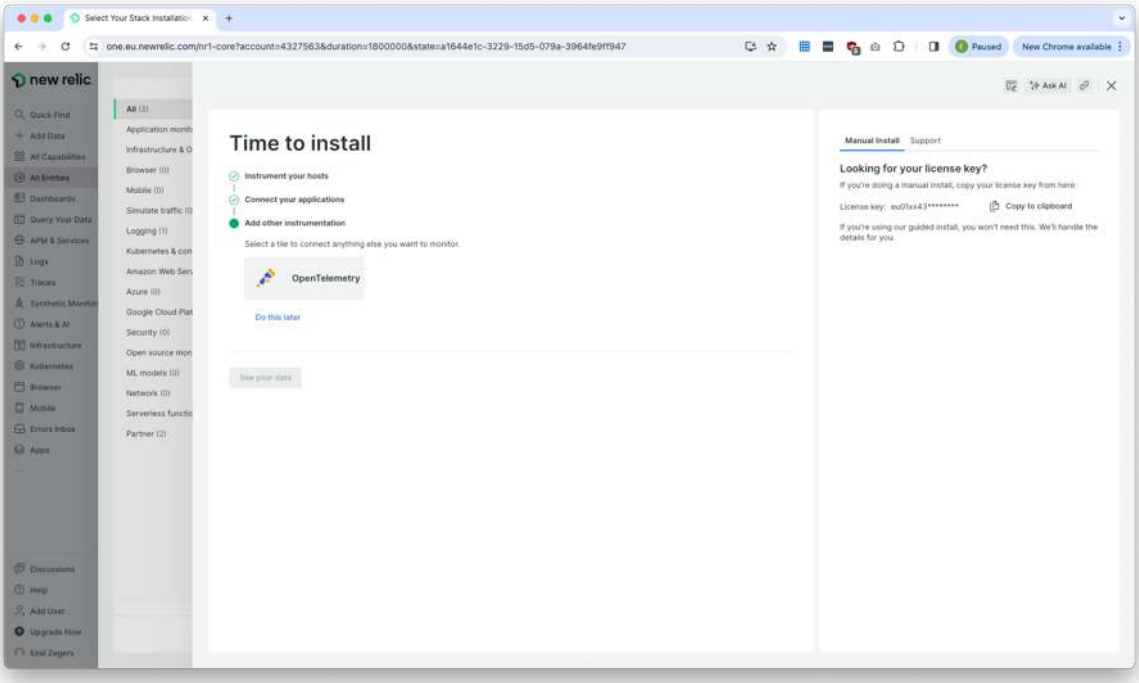
4.4 New Relic


new relic
Tell us about your stack
We'll use this info to help you connect your data correctly.
Example: PHP, OpenTelemetry, Kubernetes
Looking for your license key?
Copy license key
Application monitoring (9)
.NET
Go
Java
Mule ESB
Node.js
OpenTelemetry
PHP
Python
Ruby
Infrastructure & OS (36)
ActiveMQ
Apache
Cassandra
Consul
Couchbase
DBmarlin
Deeper Network
Elasticsearch
F5
HAProxy
HCP Vault
Kafka
Linux
macOS
MariaDB
Memcached
Microsoft SQL Server
MongoDB
MySQL
Netlify Builds
Network Data
Network Routers
New Relic
Continue

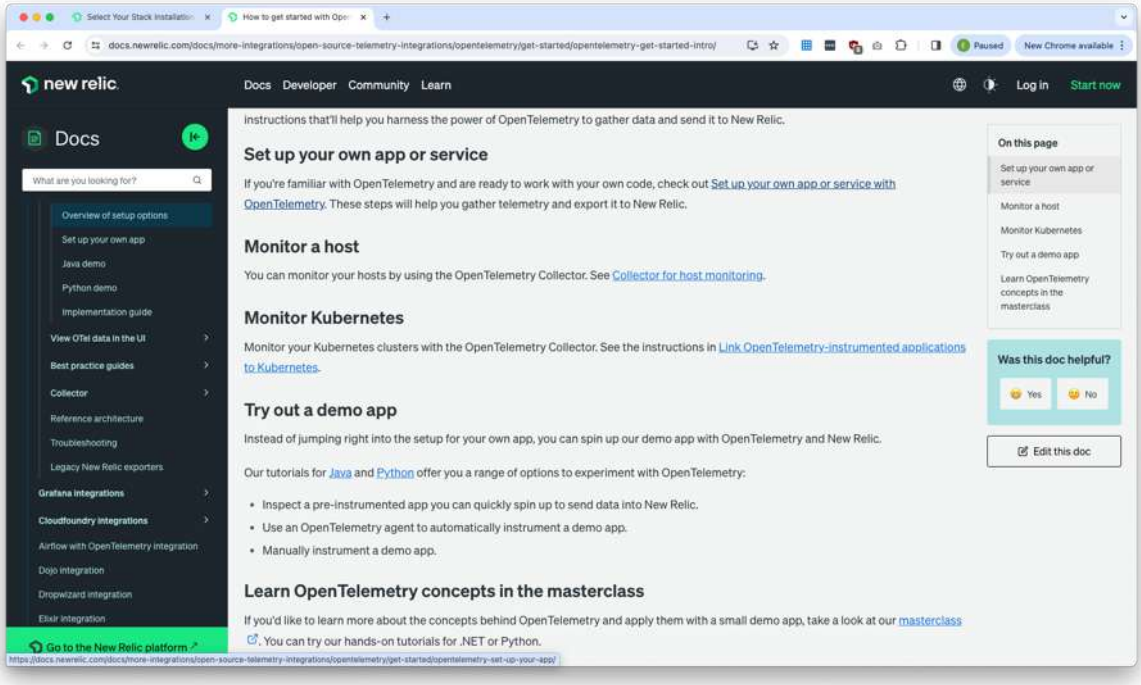

Select Your Stack < New Relic
one.eu.newrelic.com/nr1-core?account=4327563&duration=1800000&state=f3ff545b-aee1-f7fb-e69b-48c575f39522
Paused
New Chrome available
new relic
Tell us about your stack
We'll use this info to help you connect your data correctly.
Looking for your license key?
Copy license key
opentelemetry
Application monitoring (1)
OpenTelemetry
Logging (1)
Vercel
Open source monitoring (1)
OpenTelemetry
Partner (2)
Jenkins
Vercel
Continue


Select Your Stack Installation
one.eu.newrelic.com/nr1-core?account=4327563&duration=1800000&state=a1644e1c-3229-15d5-079a-3964fe9ff947
Paused
New Chrome available
new relic
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: eu01xx43********
Copy to clipboard
If you're using our guided install, you won't need this. We'll handle the details for you.

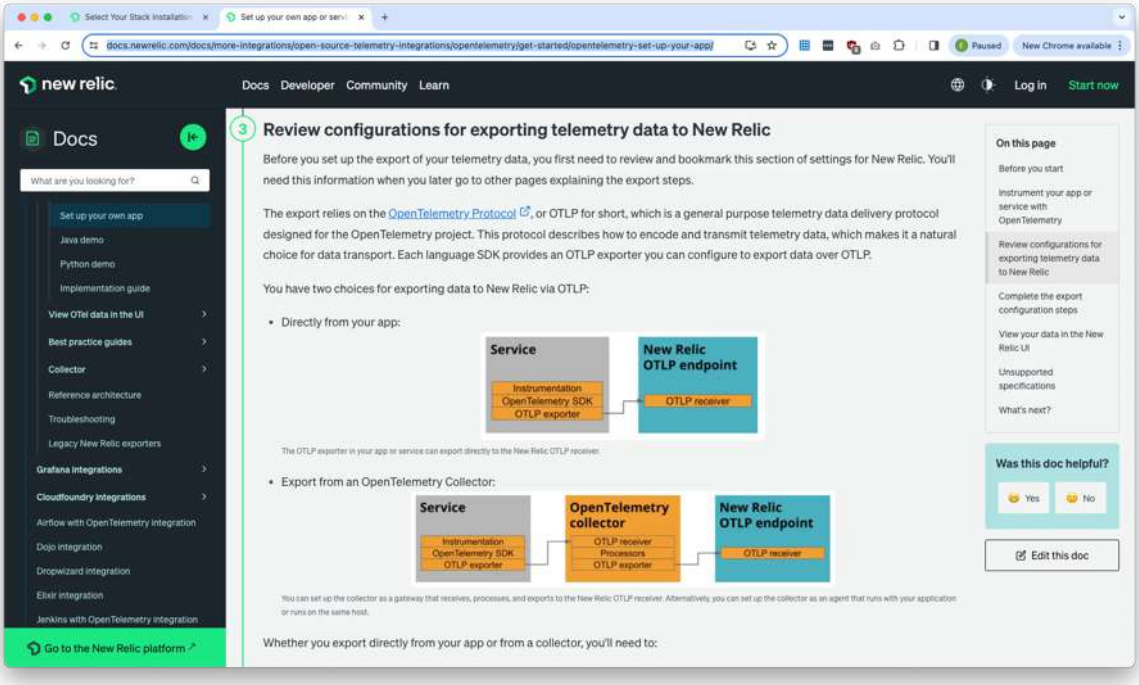




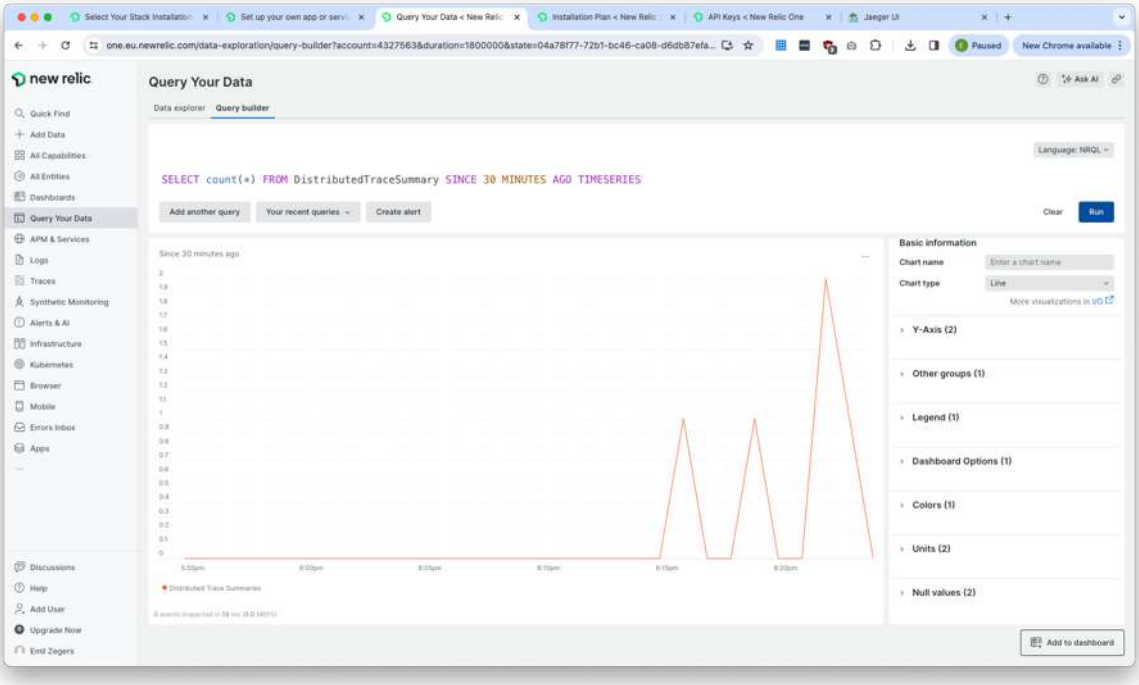
https://docs.newrelic.com/docs/more-integrations/open-source-telemetry-integrations/opentelemetry/get-started/opentelemetry-get-started-intro/



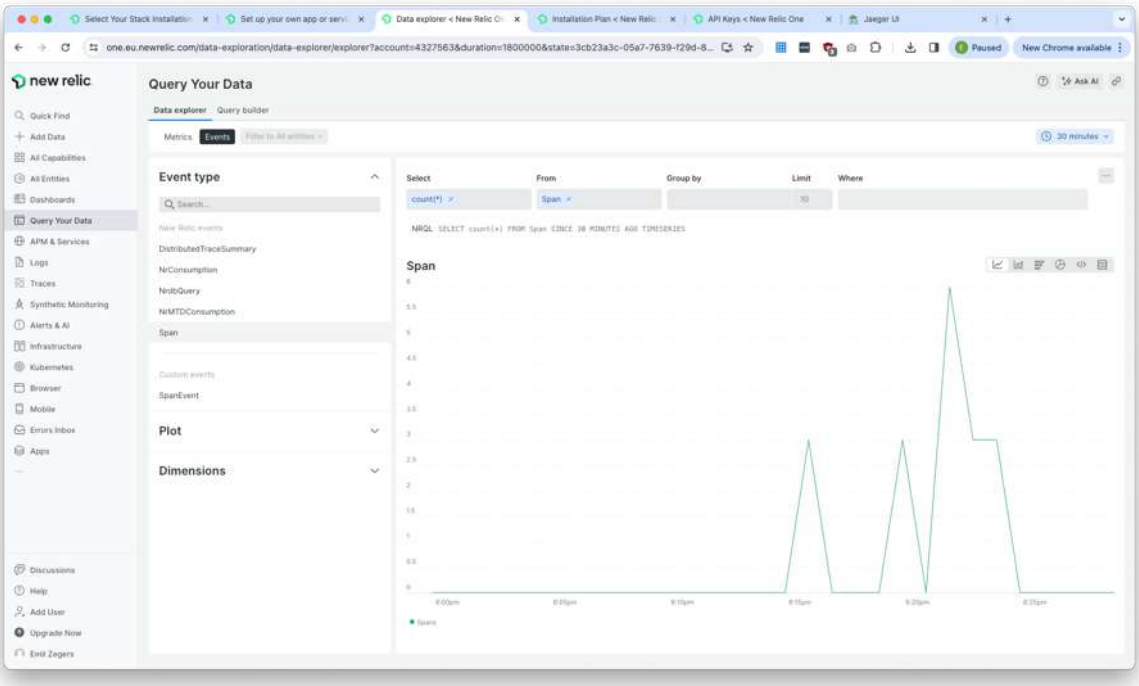
https://docs.newrelic.com/docs/more-integrations/open-source-telemetry-integrations/opentelemetry/get-started/opentelemetry-set-up-your-app/

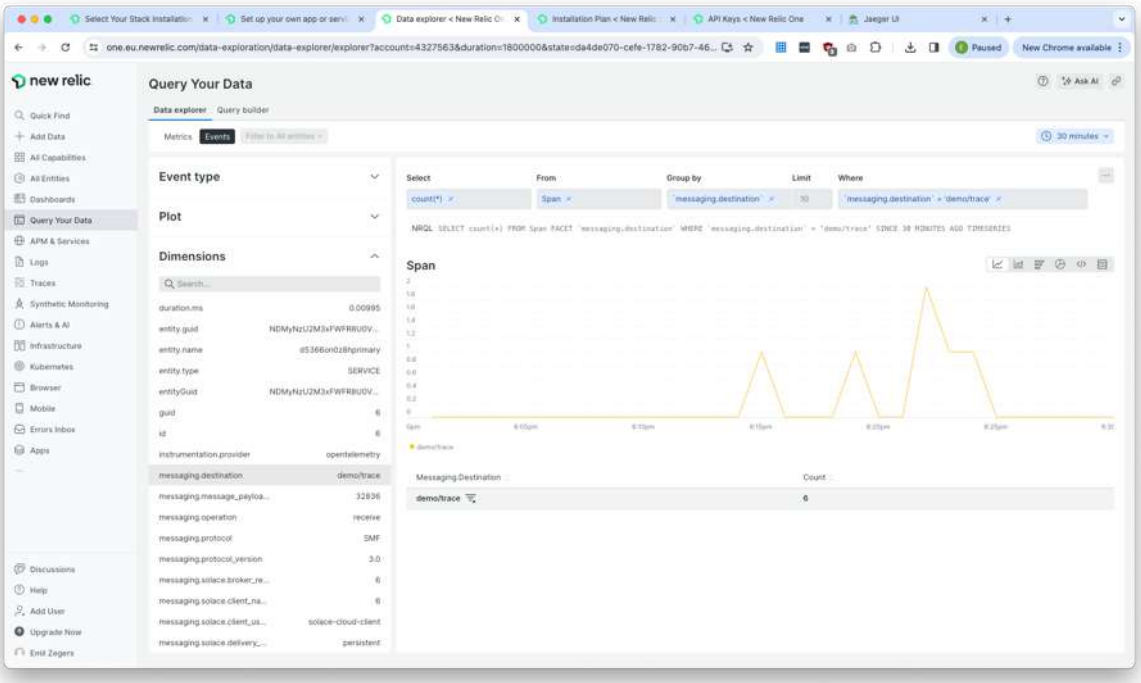


See YAML example at https://docs.newrelic.com/docs/more-integrations/open-source-telemetry-integrations/opentelemetry/collector/opentelemetry-collector-basic/



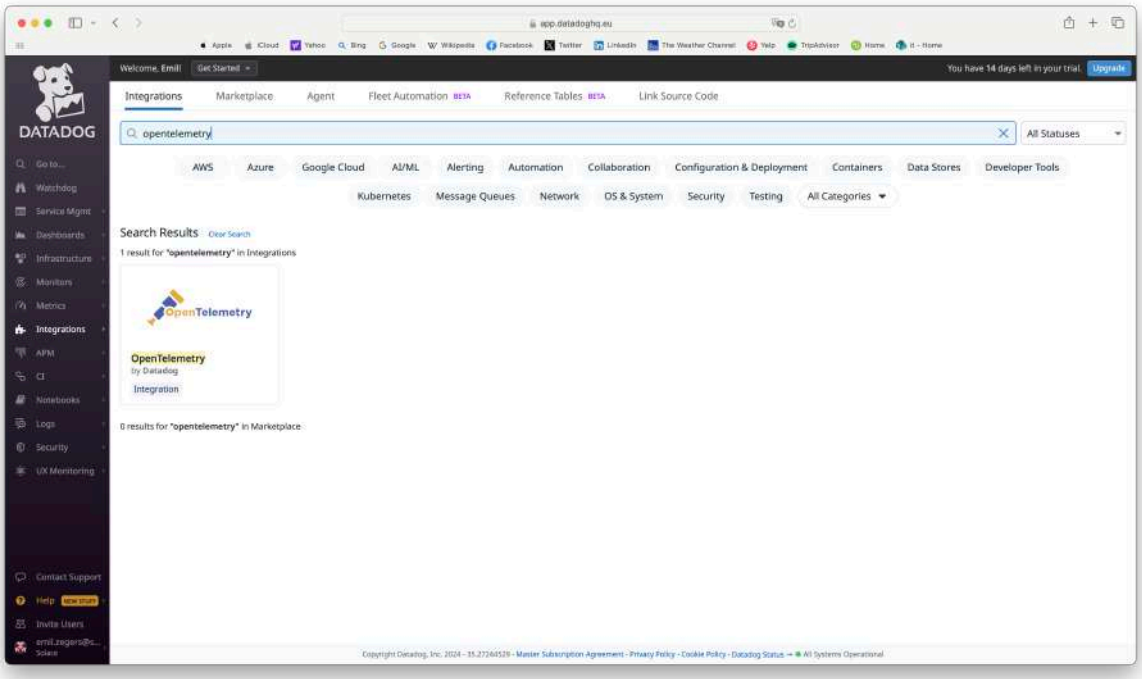
In **Query Your Data** select **Event Type Span**

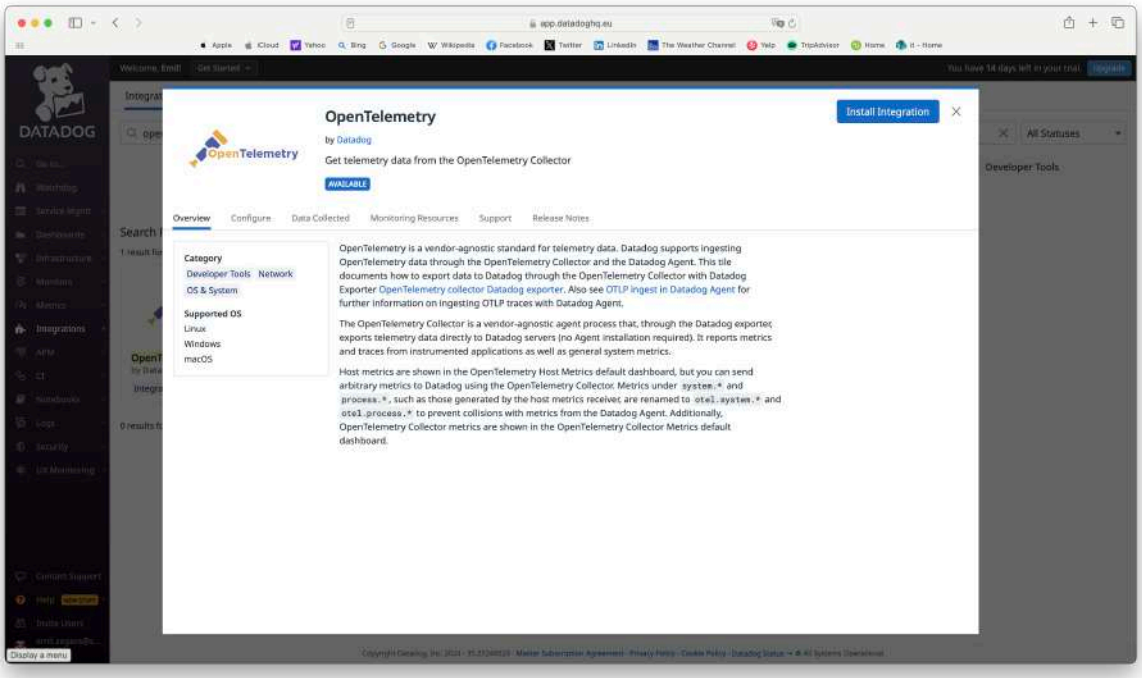


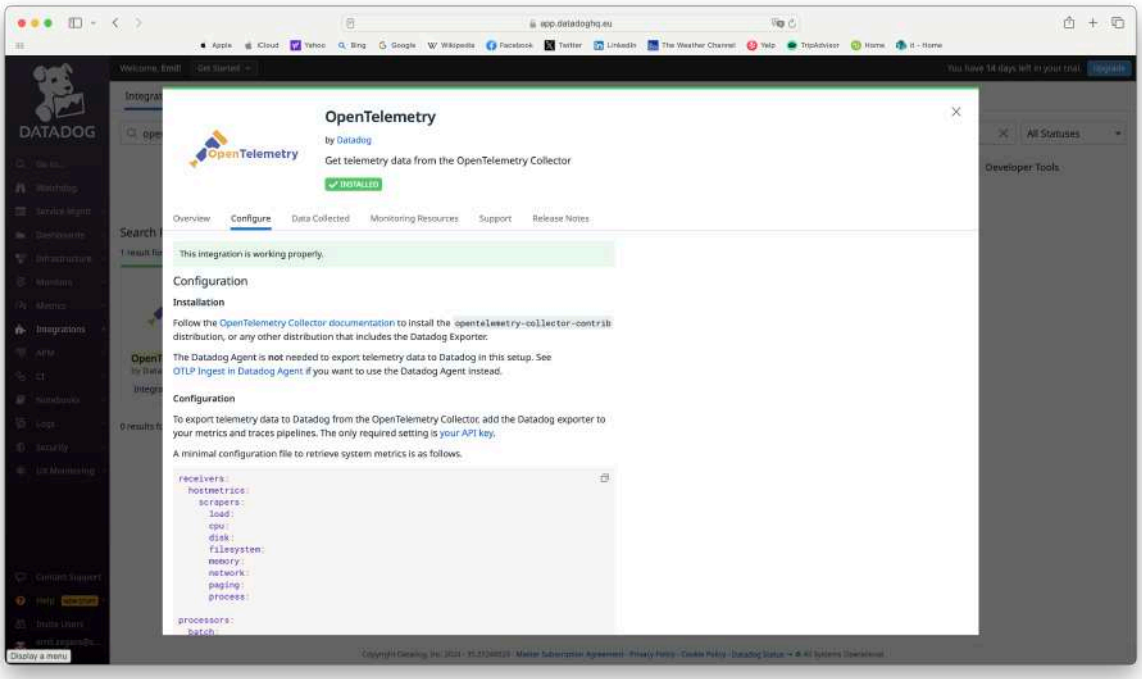


4.5 DataDog

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








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

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

4.6 Splunk

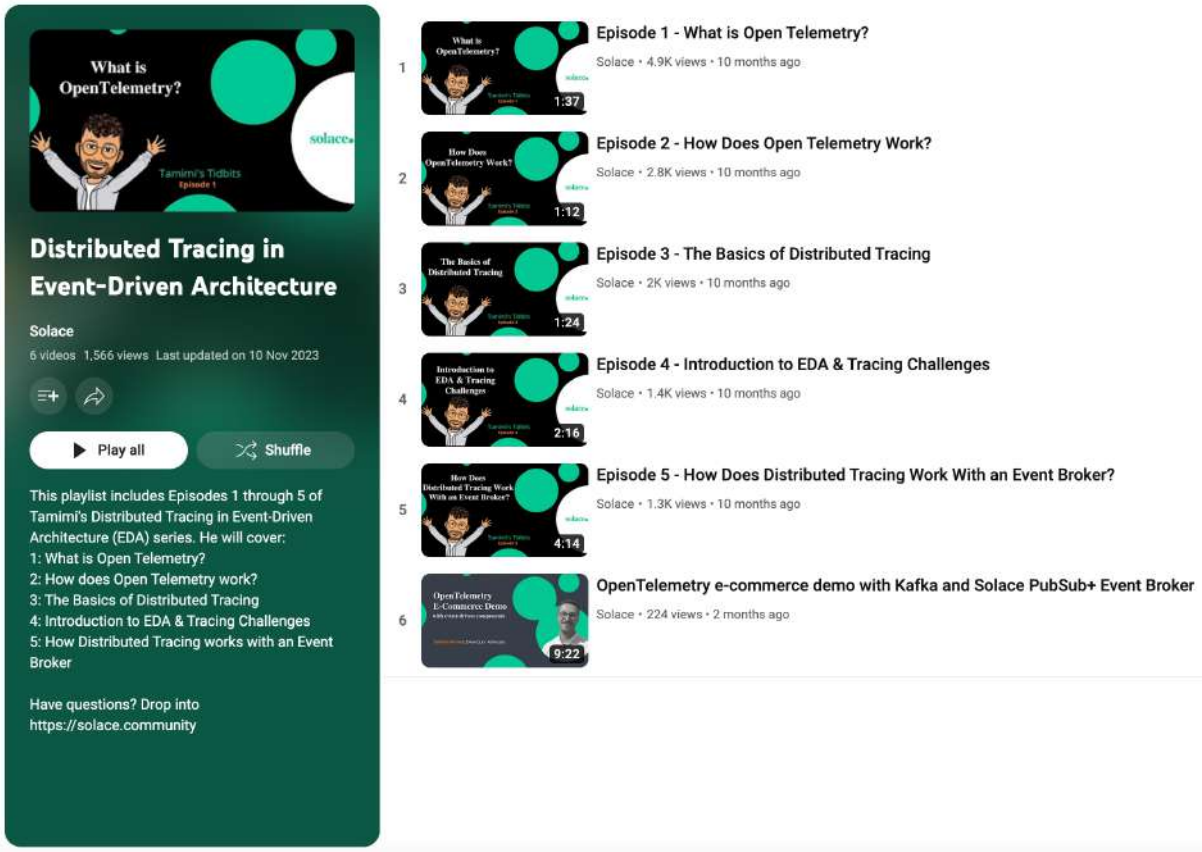
NOTE: no time yet to add… Keep an eye on updates.

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



OpenTelemetry e-commerce demo with Kafka and Solace PubSub+ Event Broker
https://www.youtube.com/watch?v=RIHQGVS5KNM