#

# From osprofiler

#

#

# Enable the profiling for all services on this node.

#

# Default value is False (fully disable the profiling feature).

#

# Possible values:

#

# \* True: Enables the feature

# \* False: Disables the feature. The profiling cannot be started via this

# project

# operations. If the profiling is triggered by another project, this project

# part will be empty.

# (boolean value)

# Deprecated group/name - [profiler]/profiler\_enabled

#enabled = false

{%- if \_data.enabled is defined %}

enabled = {{ \_data.enabled }}

{%- endif %}

#

# Enable SQL requests profiling in services.

#

# Default value is False (SQL requests won't be traced).

#

# Possible values:

#

# \* True: Enables SQL requests profiling. Each SQL query will be part of the

# trace and can the be analyzed by how much time was spent for that.

# \* False: Disables SQL requests profiling. The spent time is only shown on a

# higher level of operations. Single SQL queries cannot be analyzed this way.

# (boolean value)

#trace\_sqlalchemy = false

#

# Secret key(s) to use for encrypting context data for performance profiling.

#

# This string value should have the following format:

# <key1>[,<key2>,...<keyn>],

# where each key is some random string. A user who triggers the profiling via

# the REST API has to set one of these keys in the headers of the REST API call

# to include profiling results of this node for this particular project.

#

# Both "enabled" flag and "hmac\_keys" config options should be set to enable

# profiling. Also, to generate correct profiling information across all

# services

# at least one key needs to be consistent between OpenStack projects. This

# ensures it can be used from client side to generate the trace, containing

# information from all possible resources.

# (string value)

#hmac\_keys = SECRET\_KEY

#

# Connection string for a notifier backend.

#

# Default value is ``messaging://`` which sets the notifier to oslo\_messaging.

#

# Examples of possible values:

#

# \* ``messaging://`` - use oslo\_messaging driver for sending spans.

# \* ``redis://127.0.0.1:6379`` - use redis driver for sending spans.

# \* ``mongodb://127.0.0.1:27017`` - use mongodb driver for sending spans.

# \* ``elasticsearch://127.0.0.1:9200`` - use elasticsearch driver for sending

# spans.

# \* ``jaeger://127.0.0.1:6831`` - use jaeger tracing as driver for sending

# spans.

# (string value)

#connection\_string = messaging://

#

# Document type for notification indexing in elasticsearch.

# (string value)

#es\_doc\_type = notification

#

# This parameter is a time value parameter (for example: es\_scroll\_time=2m),

# indicating for how long the nodes that participate in the search will

# maintain

# relevant resources in order to continue and support it.

# (string value)

#es\_scroll\_time = 2m

#

# Elasticsearch splits large requests in batches. This parameter defines

# maximum size of each batch (for example: es\_scroll\_size=10000).

# (integer value)

#es\_scroll\_size = 10000

#

# Redissentinel provides a timeout option on the connections.

# This parameter defines that timeout (for example: socket\_timeout=0.1).

# (floating point value)

#socket\_timeout = 0.1

#

# Redissentinel uses a service name to identify a master redis service.

# This parameter defines the name (for example:

# ``sentinal\_service\_name=mymaster``).

# (string value)

#sentinel\_service\_name = mymaster

#

# Enable filter traces that contain error/exception to a separated place.

#

# Default value is set to False.

#

# Possible values:

#

# \* True: Enable filter traces that contain error/exception.

# \* False: Disable the filter.

# (boolean value)

#filter\_error\_trace = false