Мониторинг решения через метрики

Оглавление

Все сервисы в составе решения RooX UIDM предоставляют API-интерфейс для сбора счетчиков мониторинга различными системами типа Prometheus, Zabbix, Nagios.

API предоставляет данные в формате Prometheus, но поскольку это простой текстовый формат, то сбор может быть настроен любой системой.

Сервисы не выполняют отправку метрик, необходимо периодически опрашивать сервисы (PULL, а не PUSH модель).

Почти все метрики собираются раздельно для успешных вызовов (success) и для неуспешных вызовов (error).

О чём могут говорить метрики:

1. Соотношение success.rate и error.rate говорит о том, какая доля поступающих запросов заканчивается успешно, а какая ошибкой.
2. Процентили в отличие от среднего, лучше говорят о том, насколько быстро обрабатываются запросы. Например, среднее может выглядеть очень высоким, но при это 95 процентиль будет низкой. Это значит, что почти все запросы заканчиваются быстро, но было несколько запросов, которые выполнялись очень долго и они подняли значение среднего времени выполнения.
3. Если все значения таймеров для error близки к заданному в настройках какому-нибудь таймауту, то значит, что все ошибочные обращения заканчиваются ошибкой именно из-за таймаута.

Автоматическое инструментирование

Некоторые части системы покрываются метриками автоматически. Например инструментируются все методы всех контроллеров, все методы Spring Data-репозиториев, все методы SOAP-клиентов, все методы клиентов к REST-сервисам.

Автоматическое инструментирование работает за счет сканирования spring-контекста. В фазе постпроцессинга WebAPI monitoring-plugin сканирует контекст на предмет наличия бинов (bean) определенного класса либо реализующие определенный интерфейс. При помощи Reflection перечисляются методы и регистрируются метрики. Имя регистрируемой метрики составляется по правилам, описанным ниже.

Далее навешивается интерцептор, который собственно и увеличивает метрики при вызове инструментированного метода.

Срезы

В Prometheus используется подход, когда имя метрики — короткое, простое и не строится по шаблону, например http_request.

Обычно метрика содержит много значений в зависимости от характеристик вызова: URL, код ответа, имя сервера. Prometheus решает это через понятие меток В RooX UIDM для каждой метрики определены label, которые и предоставляют различные срезы данных.

Включение мониторинга

1. Изменить конфигурацию в env.properties `com.rooxteam.monitoring.prometheus.enabled` - установить в true
   Опционально `com.rooxteam.monitoring.prometheus.port` - прописать порт, на который будут выставляться метрики
2. Перезапустить сервис
3. Проверить, что при старте записано сообщение

```
RX_MONIT______6001: Started Prometheus exporter on port
```

Если такого сообщения нет, поискать сообщения:

```
RX_MONIT______6002: Prometheus exporter is not enabled and thus is not started.
```

Проверить, что настройка com.rooxteam.monitoring.prometheus.enabled внесена

```
RX_MONIT______4003: Not starting Prometheus exporter because port is not configured
propertly.
```

Проверить, что порт указан корректно.

Конечная точка

```
POST https://{sso_host}:{prometheus_port}/{query}
```

{sso_host} - имя хоста или IP-адрес сервера RooX UIDM. Конечная точка мониторинга не выводится на балансировщик, поэтому используйте внутренние адреса.

Каждый хост в кластерной инсталляции требуется опрашивать отдельно.

Порт используется по умолчанию из таблицы, либо может переопределяться администратором.

Таблица 1. Точные адреса для сбора метрик у сервисов

Сервис	Адрес	Примечание
sso-server	http://{sso_host}:15012	
oauth2-consumer	http://{sso_host}:27102	
federation-webapi	http://{sso_host}:13212	
webapi-otp-settings	http://{sso_host}:29102	
customer-webapi	http://{sso_host}:33102	
roox-identity-service-adapter	http://{sso_host}:31102	
roox-token-storage	http://{sso_host}:43102	
webapi-notifier	http://{sso_host}:34102	

API работает согласно спецификации https://prometheus.io/docs/instrumenting/exposition_formats

ЗАМЕТКА

Внутреннее устройство: в сервисы включен стандартный Exporter под Java, работающий через Micrometer.

В случае успешного выполнения запроса, HTTP статус ответа будет 20X, в случае проблем, код статуса будет - 4XX-5XX, в теле ответа будет описание ошибки.

В каждом запросе необходимо использовать Basic авторизацию, например:

```
Authorization: Basic dGVzdGxvZ2luOnRlc3RwYXNzd29yZA==
```

Методы API

Получение всех метрик

curl 127.0.0.1:15012

При запросе данных от сервиса без параметра {query} сервис отдает все имеющиеся данные по метрикам в text-based формате

```
GET / HTTP/1.1
User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.19.1 Basic ECC
zlib/1.2.3 libidn/1.18 libssh2/1.4.2
Host: 127.0.0.1:15012
Accept: */*

HTTP/1.1 200 OK
Date: Tue, 30 Jun 2020 14:52:30 GMT
Content-type: text/plain; version=0.0.4; charset=utf-8
Content-length: 2885051

# HELP jvm_memory_pool_allocated_bytes_total Total bytes allocated in a given JVM memory
pool. Only updated after GC, not continuously.
# TYPE jvm_memory_pool_allocated_bytes_total counter
jvm_memory_pool_allocated_bytes_total{pool="G1 Old Gen",} 1.86078648E8
jvm_memory_pool_allocated_bytes_total{pool="Code Cache",} 3.4545408E7
jvm_memory_pool_allocated_bytes_total{pool="G1 Eden Space",} 1.27401984E9
jvm_memory_pool_allocated_bytes_total{pool="G1 Survivor Space",} 5.24288E7
jvm_memory_pool_allocated_bytes_total{pool="Compressed Class Space",} 1.2598752E7
jvm_memory_pool_allocated_bytes_total{pool="Metaspace",} 1.10633728E8
...
```

Строки начинающиеся с # считаются закомментированными, кроме случаев если после них указаны токены HELP или TYPE.

HELP - после себя принимает, как минимум, один токен являющийся названием метрики. Все последующие данные являются описанием данной метрики

TYPE - после себя принимает, два дополнительных токена. Первый является названием метрики, а второй определяет тип метрики с данным именем, например, counter, gauge, histogram, summary, или untyped.

Далее идут метрики со срезами.

Получение метрик по фильтру имени

curl -g 127.0.0.1:15012/?name[]=jvm_buffer_pool_used_bytes

Также при запросе можно указать конкретное имя метрики используя конструкцию name[]=jvm_buffer_pool_used_bytes, где вместо jvm_buffer_pool_used_bytes можно указать имя интересующей метрики

```
GET /?name[]=jvm_buffer_pool_used_bytes HTTP/1.1
User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.19.1 Basic ECC
zlib/1.2.3 libidn/1.18 libssh2/1.4.2
Host: 127.0.0.1:15012
Accept: */*

HTTP/1.1 200 OK
Date: Tue, 30 Jun 2020 15:30:25 GMT
Content-type: text/plain; version=0.0.4; charset=utf-8
Content-length: 224
```

```
# HELP jvm_buffer_pool_used_bytes Used bytes of a given JVM buffer pool.
# TYPE jvm_buffer_pool_used_bytes gauge
jvm_buffer_pool_used_bytes{pool="direct",} 1.5097425E7
jvm_buffer_pool_used_bytes{pool="mapped",} 5.72559294E8
```

Что мониторится

Запросы к СУБД

Имя метрики

`rx_rdbms_repository`

Устаревшее имя метрики

`com.rooxteam.webapi.repository..methods..*`

Значение метрики

Время выполнения запроса, мс.

Срезы

Label	Срез	Примеры значений
repository	Имя репозитория (репозиторий это термин из архитектуры веб-приложений; существует правило: каждая бизнес-сущность сохраняется через свой одноименный репозиторий)	partnerMappingRepository
method	Имя метода	hashCode()
status	Результат операции	success/error
quantile	Квантиль (Квантиль в математической статистике — значение, которое заданная случайная величина не превышает с фиксированной вероятностью.)	0.5,0.75,0.95,0.98,0.99,0.999

Время выполнения запросов к внешним системам по HTTP протоколу

Имя метрики

`rx_http_client`

Устаревшее имя метрики

`com.rooxteam.webapi.clients..methods..*`

Значение метрики

Метрика отображает время выполнения запроса к внешней системе, мс.

Срезы

Label	Срез	Примеры значений
client	Имя системы, к которой идёт обращение	ssoCacRestTemplate
method	Имя метода	delete(String, Map)
status	Результат операции	success/error

Ожидание получения из пула подключения к СУБД

Имя метрики

`rx_rdbms_pool_wait`

Устаревшее имя метрики

`*.pool.Wait`

Значение метрики

Метрика отображает время ожидания, мс.

Срезы

Label	Срез	Примеры значений
pool	Имя подключения	hikari-pool-audit
quantile	квантиль	0.5,0.75,0.95,0.98,0.99,0.999

Утилизация пула к СУБД

Имя метрики

`rx_rdbms_pool_usage`

Устаревшее имя метрики

`*.pool.Usage`

Значение метрики

Метрика отображает абстрактную величину — уровень загруженности пула.

Срезы

Label	Срез	Примеры значений
pool	Имя подключения	hikari-pool-audit
quantile	квантиль	0.5,0.75,0.95,0.98,0.99,0.999

Ответы REST API

Имя метрики

`rx_http_endpoint`

Устаревшее имя метрики

`com.rooxteam.webapi.methods....`

Значение метрики

Метрика отображает время ответа API, мс.

Срезы

Label	Срез	Примеры значений
api	Имя контроллера API (контроллер — термин из архитектуры веб-приложений; в случае REST один базовый URL обрабатывается одним контроллером, например /credentials)	sessionList
endpoint	Логическое имя метода	/sessionList/@me/active
method	HTTP-метод	DELETE
status	Результат операции	success/error

Работа API аутентификации

Имя метрики

`rx_authnz_endpoint`

Устаревшее имя метрики

`com.rooxteam.monitoring.filter.StatisticalMonitoringFilter...*`

Значение метрики

Метрика отображает время ответа API аутентификации и авторизации, мс.

Срезы

Label	Срез	Примеры значений
apigroup	Тип API	authentication/authorization
api	Тип API	oauth1/oauth2/mlk
status	Результат операции	success/error

Метрики JVM

Имя метрики

`jvm_buffer_pool_used_bytes`

Значение метрики

Метрика отображает количество байт пула JVM буфера определенного типа

Срезы

Label	Срез	Примеры значений
pool	Тип пула	direct/mapped

Имя метрики

`jvm_buffer_pool_capacity_bytes`

Значение метрики

Метрика отображает вместительность пула JVM буфера определенного типа в байтах

Срезы

Label	Срез	Примеры значений
pool	Тип пула	direct/mapped

Имя метрики

`jvm_buffer_pool_used_buffers`

Значение метрики

Метрика отображает количество использованных пулов JVM буфера определенного типа

Срезы

Label	Срез	Примеры значений
pool	Тип пула	direct/mapped

Имя метрики

`jvm_info`

Значение метрики

Информация о JVM

Срезы

Label	Срез	Примеры значений
version	Версия JVM	1.8.0_242-b08
vendor	Вендор	Oracle Corporation
runtime	Среда выполнения	OpenJDK Runtime Environment

Имя метрики

`jvm_gc_collection_seconds_sum`

Значение метрики

Метрика отображает суммарное количество времени потраченного в выбранном JVM сборщике мусора, с.

Срезы

Label	Срез	Примеры значений
gc	Раздел кучи	G1 Young Generation/G1 Old Generation

Имя метрики

`jvm_threads_current`

Значение метрики

Метрика отображает текущее количество потоков JVM

Имя метрики

`jvm_threads_peak`

Значение метрики

Метрика отображает пиковое количество потоков JVM

Имя метрики

`jvm_threads_deadlocked`

Значение метрики

Cycles of JVM-threads that are in deadlock waiting to acquire object monitors or ownable synchronizers

Имя метрики

`jvm_threads_state`

Значение метрики

Метрика отображает текущее количество потоков в определенных состояниях

Срезы

Label	Срез	Примеры значений
state	Состояние потока	NEW / TIMED_WAITING / BLOCKED / RUNNABLE / TERMINATED / WAITING

Алгоритм работы с метриками

1. Включить отправку метрик prometheus.
2. Собирать данные по метрикам (рекомендуется собирать количество запросов в срезах по признакам status success/error, квантили 95 и 99, min, max и mean).

Собранные данные позволят проводить анализ при возникновении проблем.