Loway представляет

Система слишком медленная сегодня…но чем она занята?

Как контролировать QueueMetrics через JMX.

Иногда мы сталкиваемся с проблемами производительности в запущенных нами системах: не всегда легко понять, что на самом деле происходит. Приложение работает медленно.
Но почему? Часто решение основывается на методе проб и ошибок; если повезет, проблема может быть устранена. Конечно, это не лучший способ работы. Часто люди советуют предоставить системе больше памяти, или установить более быстрый диск, или выделить больше ресурсов процессора, и иногда это срабатывает.
Но определенно можно поступить более разумно.

Одна из самых замечтельных вещей в среде Java, на которой основаны QueueMetrics и WombatDialer, заключается в том, что она позволяет получить очень точное и подробное представление о том, что на самом деле происходит. Вам не нужно использовать специальную сборку программного обеспечения, которая предназначена для сбора статистики, и вам не нужно платить высокую цену за понимание того, что делает система, так как вполне доступно собирать статистику в реальном времени в продуктивной среде.
Более того, JVM созданы для того, чтобы быть контролируемыми, поэтому существует стандартизированная экосистема инструментов, которые помогут вам понять, что на самом деле происходит. Этот подход называется JMX.

Оборотная сторона - как вы понимаете, всегда есть обратная сторона - то, что настроить JMX не так-то просто. Особенно, если нужно контролировать систему с другого компьютера, например мониторить продуктивный сервер с рабочей станции, настройка заставит вас рвать волосы, прежде чем вы поймете все параметры, необходимые для настройки рабочего соединения.

Чтобы облегчить вам жизнь, мы подготовили возможность установки полностью работающего JMX-подключения просто с помощью раскомментирования одной строки и создания SSH туннеля, который позволит безопасно получать доступ к данным.
Вам понадобится последняя версия Tomcat RPM (8.5.64 или новее).

Итак, как только он установлен, редактируем `/etc/sysconfig/qm-tomcat6`, и указываем:

```
# Remove the comments below to activate JMX monitoring

# JAVA_RMI_ADDRESS=localhost

JAVA_JMX_PORT="30000"
```

Раскомментируйте строку с `JAVA_RMI_ADDRESS`; сохраните файл и перезапустите. Вам нужно сделать это всего один раз, и это можно безопасно сделать даже в продуктивных системах.
При необходимости можно изменить порт JMX - неважно, какой порт выберете, если он совпадает на ПК и на сервере.

Теперь переходите на свою рабочую станцию и запускаете новый сеанс SSH со следующим заклинанием, чтобы создать локальный туннель для порта 30000 и для следующего тоже:

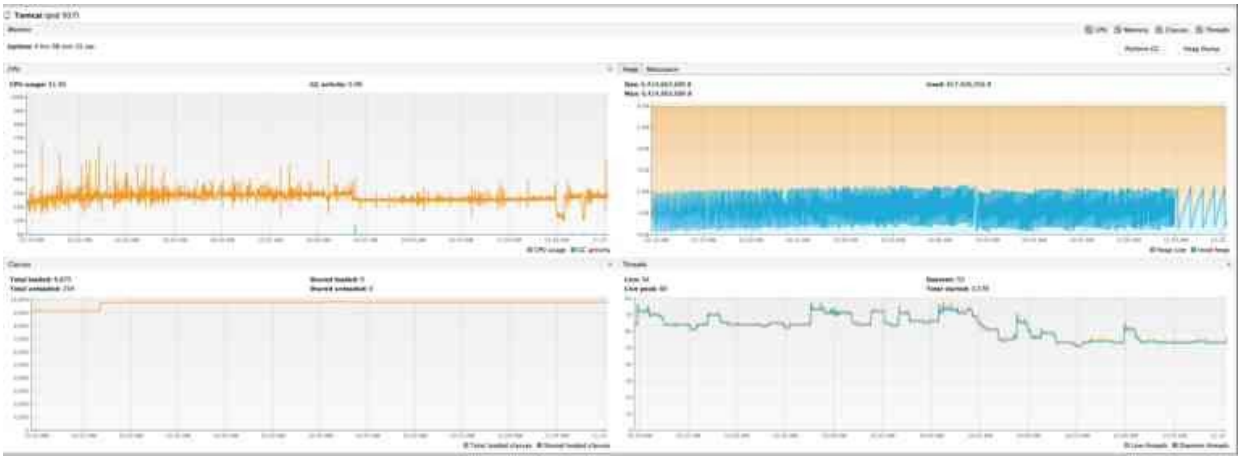
```
ssh root@my.queuemetrics.server \

        -L 30000:localhost:30000 \

        -L 30001:localhost:30001
```

Теперь можно запустить VisualVM на рабочей станции - скорее всего, он уже будет установлен, если стоит Java SDK; в противном случае можно скачать его отсюда: https://visualvm.github.io

Заметьте, что хотя в приведенном выше примере мы выполняем вход через SSH как «root», подойдет любой пользователь.

При запуске VisualVM нужно щелкнуть по кнопке «Добавить соединение JMX» (‘Add JMX connection’), и создать новое соединение с «localhost: 30000» без пароля и без шифрования (поскольку безопасность и шифрование обеспечивает SSH).

Когда вы это сделаете, нажмите на вновь созданное соединение, чтобы начать мониторинг.



Например, на графиках выше видим, что:

- Использование ЦП составляет около 30% и стабильно (график слева). Очень маленькие сборки мусора, которые обычно являются основными виновниками проблем с производительностью Java.
- Нет проблем с памятью. Система использует около 2 ГБ (график справа)
- Система несколько простаивала последние несколько минут - она не использовала много памяти, но что-то вычисляла.
- Можно узнать, что она делает, получив дамп потока.

Ряд интересных вещей можно сделать с помощью VisualVM:

- «Узнайте свою JVM»: можно посмотреть настройки JVM в «Обзор» / «Аргументы JVM» (‘Overview’ / ‘JVM arguments’).
- «Мониторинг памяти»: можно увидеть текущее использование ЦП, памяти и потоков на странице «Монитор» (‘Monitor’). Заметьте, что для большинства настроек это нормально, что вся память будет израсходована прежде, чем будет выполнена сборка мусора; поэтому увидите скачки и падения на графике. Также можно принудительно выполнить сборку мусора, если хотите увидеть «истинное» использование памяти, но это может быть неразумно на сильно загруженных продуктивных серверах.
- «Мониторинг потоков»: можно получить текстовый дамп потока, подобный рассмотренному выше, выбрав «Потоки» / «Дамп потоков» (‘Threads’ / ‘Thread dump’).
- Можно использовать «Sampler», чтобы получить разбивку использования памяти и ЦП по классам (сначала необходимо установить плагин «VisualVM-Sampler» из меню Plugins, если это еще не было сделано).
- Можно держать сервер открытым с несколькими экземплярами VisualVM, чтобы контролировать несколько серверов QM.

Удачного мониторинга!

Ссылки

Программное обеспечение QueueMetrics доступно «on premise» или как облачный хостинг для FreePBX, Yeastar S PBX, Grandstream PBX, Issabel, FusionPBX и других дистрибутивов Asterisk.

Более подробную техническую информацию смотрите в User Manual.

Посетите www.queuemetrics.com для получения бесплатной полнофункциональной пробной версии.

Обратите внимание на Free Webinars с живой демонстрацией QueueMetrics.