14 JMX мониторинг

Обзор

Мониторинг JMX можно использовать для наблюдения за счетчиками JMX в Java приложениях.

В Zabbix 2.0 добавлена встроенная поддержка мониторинга JMX, был выпущен новый Zabbix демон, так называемый “Zabbix Java gateway”.

Когда Zabbix сервер хочет узнать значение конкретного счетчика JMX у узла сети, он опрашивает Zabbix **Java gateway**, который в свою очередь используя API управление JMX, удаленно опрашивает интересующее приложение.

Для получения более подробных сведений, включая где можно взять Zabbix Java gateway и как его настроить, смотрите этот раздел руководства.

Связь между Java gateway и наблюдаемым JMX приложением не должна быть закрыта брандмауэром.

Включение удаленного JMX мониторинга для Java приложений

Приложению Java не требуется какое-либо дополнительно установленное программное обеспечение, но для поддержки удаленного мониторинга JMX приложение должно быть запущено с указанными ниже параметрами командной строки.

Как минимум, если вы просто хотите начать наблюдение за простым приложением Java на локальном хосте без каких либо защиты, запустите его со следующими опциями:

```
java \
-Dcom.sun.management.jmxremote \
-Dcom.sun.management.jmxremote.port=12345 \
-Dcom.sun.management.jmxremote.authenticate=false \
-Dcom.sun.management.jmxremote.ssl=false \
-jar /usr/share/doc/openjdk-6-jre-headless/demo/jfc/Notepad/Notepad.jar
```

С этими аргументами Java будет слушать входящие соединения JMX на порту 12345, только с локальных хостов, без обязательных аутентификации или SSL.

Если вы хотите разрешить подключения с другого интерфейса, укажите параметр -Djava.rmi.server.hostname равным IP адресу этого интерфейса.

Если вы хотите иметь более строгую проверку в плане безопасности, есть много других опций в Java, которые вам доступны. Например, следующая иллюстрация запускает приложение с более универсальным набором опций и открывает это приложение для более широкой сети, не только для локального компьютера.

```
java \
-Djava.rmi.server.hostname=192.168.3.14 \
-Dcom.sun.management.jmxremote \
```

```
-Dcom.sun.management.jmxremote.port=12345 \
-Dcom.sun.management.jmxremote.authenticate=true \
-Dcom.sun.management.jmxremote.password.file=/etc/java-6-
openjdk/management/jmxremote.password \
-Dcom.sun.management.jmxremote.access.file=/etc/java-6-
openjdk/management/jmxremote.access \
-Dcom.sun.management.jmxremote.ssl=true \
-Djavax.net.ssl.keyStore=$ВАШЕ_ХРАНИЛИЩЕ_КЛЮЧЕЙ \
-Djavax.net.ssl.keyStorePassword=$ВАШ_ПАРОЛЬ_К_ХРАНИЛИЩУ_КЛЮЧЕЙ \
-Djavax.net.ssl.trustStore=$ВАШЕ_ДОВЕРЕННОЕ_ХРАНИЛИЩЕ \
-Djavax.net.ssl.trustStorePassword=$ВАШ_ПАРОЛЬ_К_ДОВЕРЕННОМУ_ХРАНИЛИЩУ \
-Dcom.sun.management.jmxremote.ssl.need.client.auth=true \
-jar /usr/share/doc/openjdk-6-jre-headless/demo/jfc/Notepad/Notepad.jar
```

Значительное количество (если не все) этих настроек можно указать в /etc/java-6-openjdk/management/management.properties (или там, где этот файл расположен на вашем компьютере).

Обратите внимание, если вы желаете использовать SSL, то вы должны изменить startup.sh скрипт Java gateway, добавив в него опции `-Djavax.net.ssl.*` так, чтобы он знал где искать хранилище ключей и доверенное хранилище.

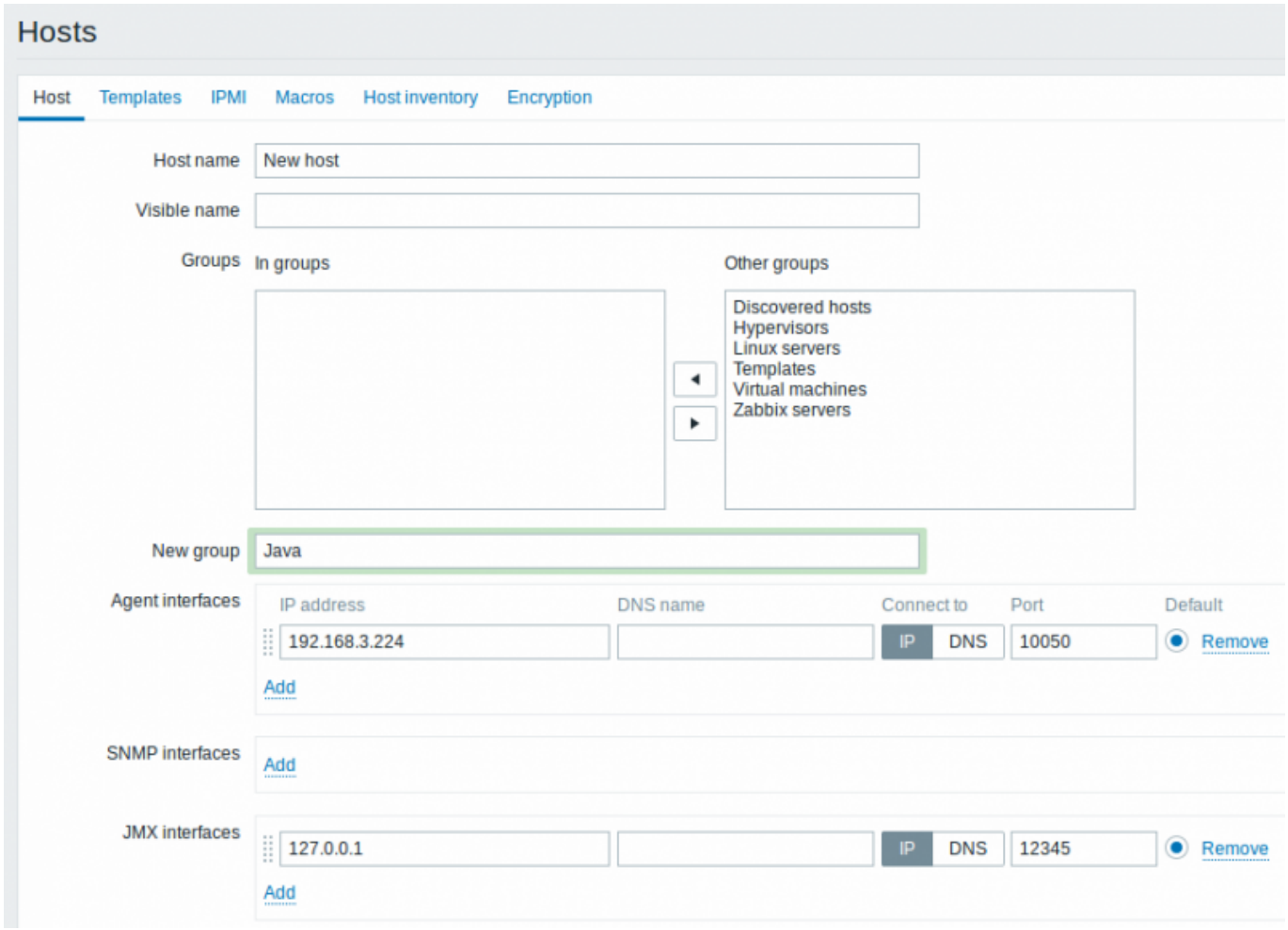
Смотрите Мониторинг и Управление с использованием JMX [en] для получения более подробной информации.

Настройка JMX интерфейсов и элементов данных в веб-интерфейсе Zabbix

Когда Java Gateway запущен, сервер знает где его искать и Java приложение запущено с поддержкой удаленного JMX мониторинга, самое время настроить интерфейсы и элементы данных в Веб-интерфейсе Zabbix.

Настройка JMX интерфейса

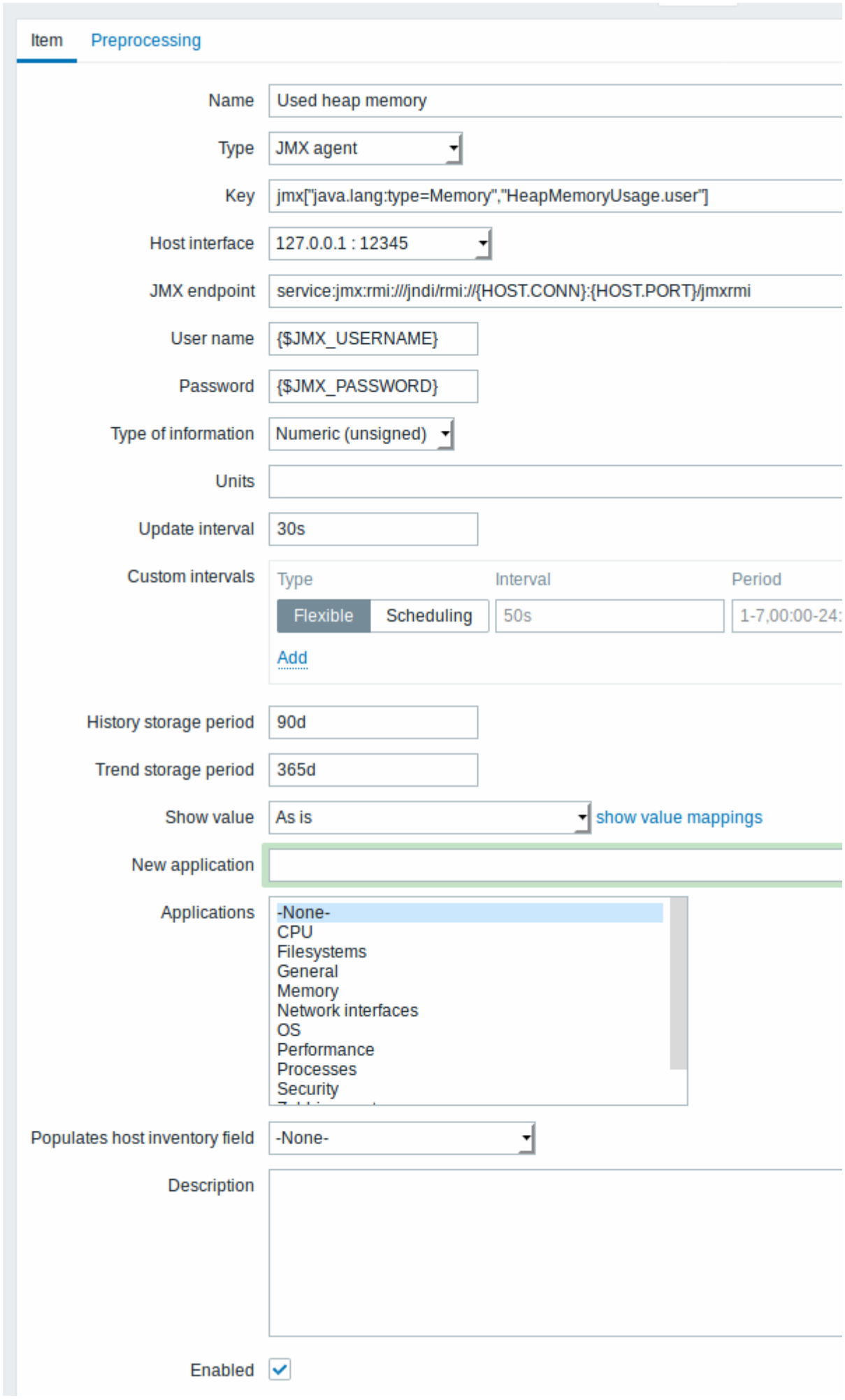
Начнем с создания интерфейса JMX-типа у интересующего узла сети:



Добавление элемента данных JMX агента

Для каждого интересующего вас счетчика JMX вам необходимо добавить элемент данных с типом **JMX агент** присоединенный к этому интерфейсу.

Ключ на снимке экрана ниже имеет следующий вид `jmx["java.lang:type=Memory","HeapMemoryUsage.used"]`.


Item
Preprocessing
Name
Used heap memory
Type
JMX agent
Key
jmx["java.lang:type=Memory","HeapMemoryUsage.user"]
Host interface
127.0.0.1 : 12345
JMX endpoint
service:jmx:rmi:///jndi/rmi://{HOST.CONN}:{HOST.PORT}/jmxrmi
User name
{$JMX_USERNAME}
Password
{$JMX_PASSWORD}
Type of information
Numeric (unsigned)
Units
Update interval
30s
Custom intervals
Type
Interval
Period
Flexible
Scheduling
50s
1-7,00:00-24:
Add
History storage period
90d
Trend storage period
365d
Show value
As is
show value mappings
New application
Applications
-None-
CPU
Filesystems
General
Memory
Network interfaces
OS
Performance
Processes
Security
Populates host inventory field
-None-
Description
Enabled

Поля, требующие специфичной информации для JMX элементов данных:

Тип	Укажите здесь **JMX агент**.
Ключ	Ключ элемента данных `jmx[]` состоит из двух параметров: **имя объекта** - имя объекта MBean; **имя атрибута** - имя атрибута MBean с опциональными составными данными имен полей, разделенных точками. Смотрите ниже для получения более подробных сведений о ключах элементов данных JMX. Начиная с Zabbix 3.4, вы можете обнаруживать MBeans и MBean атрибуты, используя элемент данных `jmx.discovery[]` низкоуровневого обнаружения.
JMX endpoint	Вы можете указать пользовательский JMX endpoint. Убедитесь, что параметры подключения JMX endpoint совпадают с JMX интерфейсом. Это можно сделать при помощи макросов {HOST.*}, как это сделано в JMX endpoint по умолчанию. Это поле поддерживается начиная с 3.4.0. Поддерживаются макросы {HOST.*} и пользовательские макросы.
Имя пользователя	Укажите имя пользователя, если вы настроили аутентификацию у вашего Java приложения. Поддерживаются пользовательские макросы.
Пароль	Укажите пароль, если вы настроили аутентификацию у вашего Java приложения. Поддерживаются пользовательские макросы.

Если вы хотите наблюдать за Логическим счетчиком, который может быть “true” или “false”, вы должны указать тип информации “Числовой (целое положительное)” и “Логический” тип данных. Сервер будет записывать Логические значения как 1 или 0, соответственно.

Детальная информация о ключах JMX элементов данных

Простые атрибуты

Имя объекта MBean неважно, кроме строки, которую вы определили в вашем Java приложении. Имя атрибута, с другой стороны, может быть более сложным. В случае, если атрибут возвращает простой тип данных (число, строку и т.п.), то не стоит волноваться об этом, ключ будет выглядеть примерно так:

```
jmx[com.example:Type=Hello,weight]
```

В этом примере именем объекта является “com.example:Type=Hello”, именем атрибута будет являться “weight” и, скорее всего, тип возвращаемого значения должен быть “Числовой (с плавающей точкой)”.

Атрибуты возвращающие составные данные

Ключ становится более сложным, когда ваш атрибут возвращает составные данные. Например:

именем вашего атрибута является “apple” и он возвращает хэш представляющих его параметров, таких как “weight”, “color” и прочее. Тогда ваш ключ может выглядеть примерно так:

```
jmx[com.example:Type=Hello,apple.weight]
```

Этот пример показывает как разделяются с помощью точки имя атрибута и ключ хэша. Точно также, если атрибут возвращает часть вложенных составных данных, их нужно снова разделить точкой:

```
jmx[com.example:Type=Hello,fruits.apple.weight]
```

Проблема с точками

Пока все хорошо. Но что, если имя атрибута или ключ хэша содержит символ точки? Вот пример:

```
jmx[com.example:Type=Hello,all.fruits.apple.weight]
```

Это проблема. Как сказать Zabbix'y, что имя атрибута “all.fruits”, а не просто “all”? Как отличить точку, которая является частью имени, от точки которая разделяет имя атрибута и ключи хэшей?

До **2.0.4** Zabbix Java gateway был не способен справится с такими ситуациями и пользователи оставались с НЕПОДДЕРЖИВАЕМЫМИ элементами данных. Начиная с 2.0.4 проблема была исправлена, все что вам требуется сделать - экранировать точки, которые являются частью имени, обратной косой чертой:

```
jmx[com.example:Type=Hello,all\.fruits.apple.weight]
```

Аналогично, если ваш ключ хэша содержит точку вам необходимо её экранировать:

```
jmx[com.example:Type=Hello,all\.fruits.apple.total\.weight]
```

Другие проблемы

Символ обратной косой черты тоже должен быть экранирован:

```
jmx[com.example:type=Hello,c:\\documents]
```

Для обработки любых других символов в ключе JMX элемента данных, пожалуйста, смотрите этот раздел.

На самом деле это все, что нужно сделать. Успешного мониторинга JMX!

Пример пользовательского endpoint с JBoss EAP 6.4

Пользовательские endpoint позволяют работать с различными транспортными протоколами, которые отличаются от протокола по умолчанию RMI.

Для иллюстрации этой возможности в качестве примера давайте попытаемся настроить JBoss EAP 6.4. Во-первых, давайте сделаем некоторые предположения:

- У вас уже имеется установленный Zabbix Java gateway. Если нет, тогда вам нужно сделать это в соответствии с документацией.
- Zabbix сервер и Java gateway установлены с префиксом /usr/local/.
- JBoss уже установлен в /opt/jboss-eap-6.4/ и запущен в автономном режиме.
- Мы будем считать, что все эти компоненты работают на одном и том же хосте.
- Брандмауэр и SELinux отключены (или настроены соответствующим образом).

Давайте выполним некоторые простые настройки в zabbix_server.conf:

```
JavaGateway=127.0.0.1
StartJavaPollers=5
```

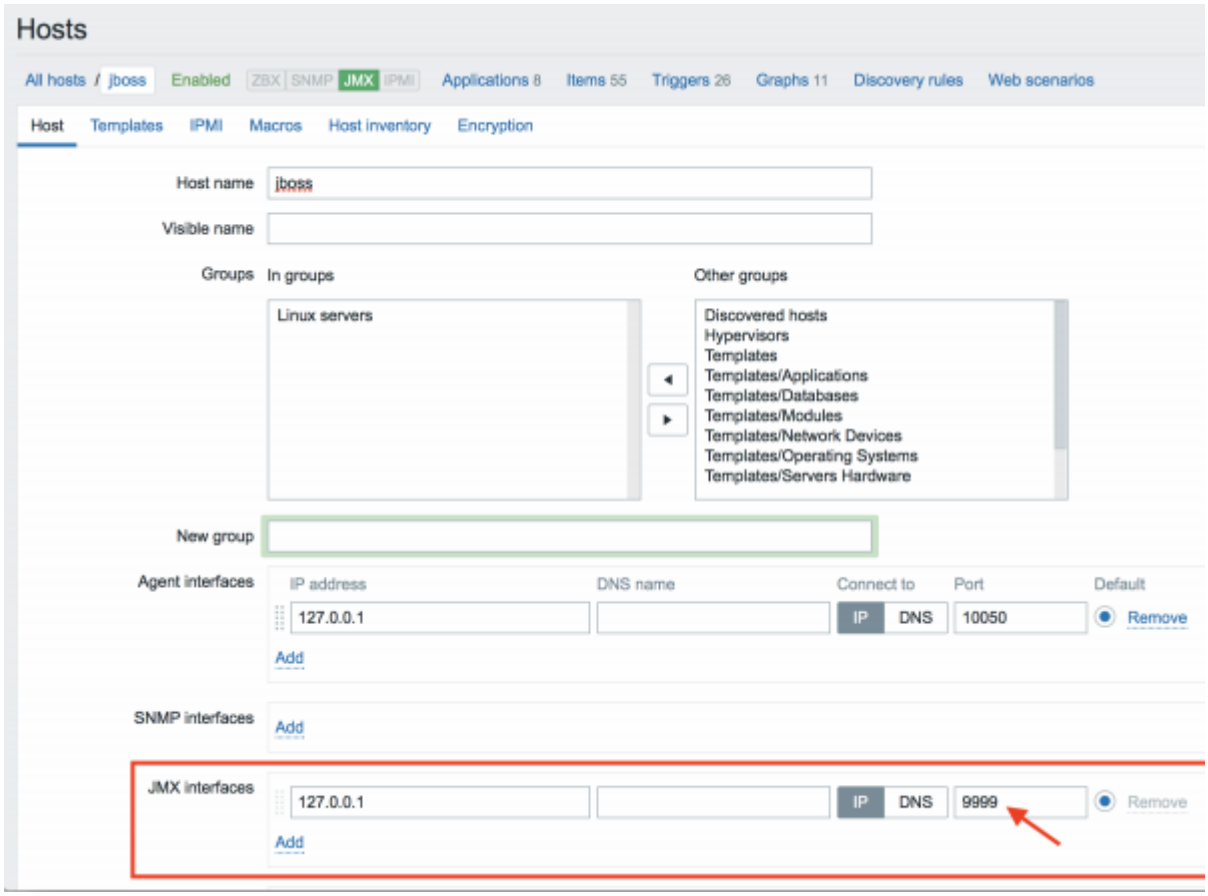
И в файле конфигурации zabbix_java/settings.sh (или zabbix_java_gateway.conf):

```
START_POLLERS=5
```

Проверьте, что JBoss слушает свой стандартный порт управления:

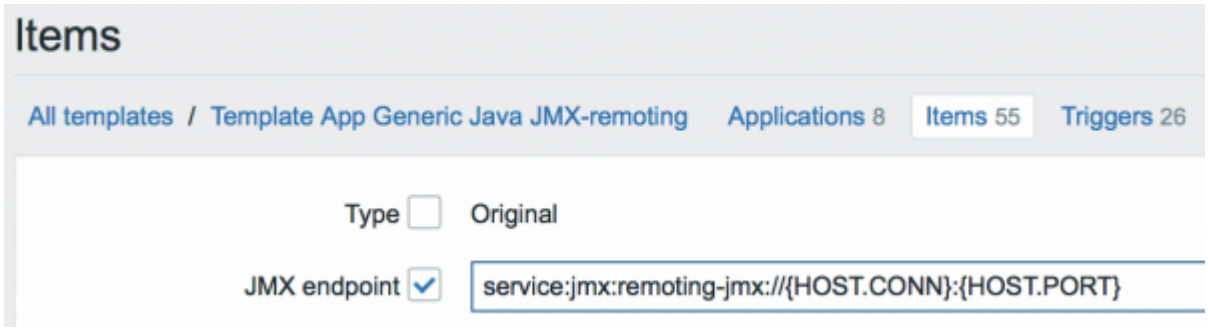
```
$ netstat -natp | grep 9999
tcp        0      0 127.0.0.1:9999          0.0.0.0:*               LISTEN
10148/java
```

Теперь давайте создадим в Zabbix узел сети с JMX интерфейсом 127.0.0.1:9999.



Как мы знаем эта версия JBoss использует протокол JBoss Remoting всесто RMI, мы можем использовать массовое обновление параметра JMX endpoint в нашем шаблоне JMX в соответствии:

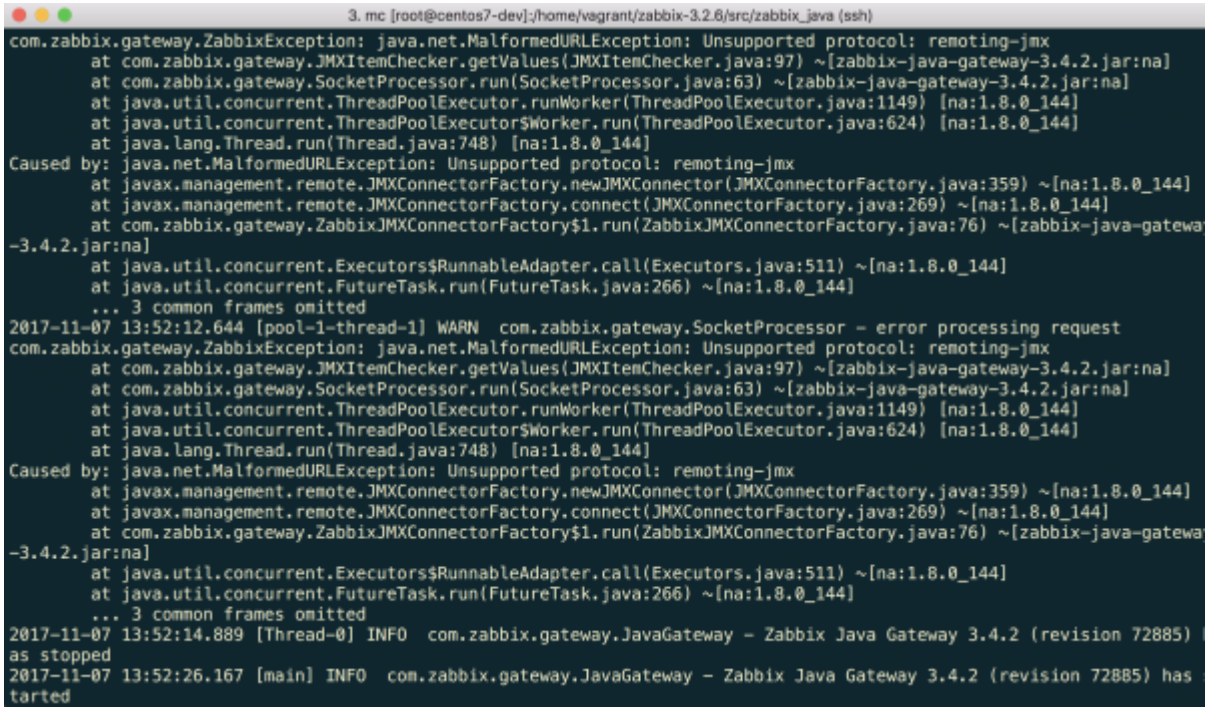
```
service:jmx:remoting-jmx://{HOST.CONN}:{HOST.PORT}
```



Давайте обновим кэш конфигурации:

```
$ /usr/local/sbin/zabbix_server -R config_cache_reload
```

Обратите внимание, что сначала может возникнуть ошибка.



“Unsupported protocol: remoting-jmx” означает, что Java gateway не знает как работать с указанным протоколом. Эту ошибку можно исправить создав файл ~/needed_modules.txt со следующим содержимым:

```
jboss-as-remoting
jboss-logging
jboss-logmanager
jboss-marshalling
jboss-remoting
jboss-sasl
jcl-over-slf4j
jul-to-slf4j-stub
log4j-jboss-logmanager
remoting-jmx
slf4j-api
xnio-api
xnio-nio</pre>
```

и затем выполнив эту команду:

```
$ for i in $(cat ~/needed_modules.txt); do find /opt/jboss-eap-6.4 -iname
${i}*.jar -exec cp {} /usr/local/sbin/zabbix_java/lib/ \; ; done
```

Таким образом, у Java gateway будут в наличии все необходимые модули для работы с jmx-remoting. Осталось только перезапустить Java gateway, немного подождать и, если вы все сделали правильно, вы увидите, что эти данные JMX мониторинга начинают поступать в Zabbix: