Система проверки доступности Watchdog

Отслеживание сбоев

WCS имеет встроенную подсистему проверки доступности сервера с отправлением отчетов на почту - Watchdog.

Watchdog работает по JMX и проверяет три вещи:

1. Доступность серверного процесса - отвечает ли серверный процесс на запросы.
2. Работоспособность SIP-стека.
3. Тестовую регистрацию на SIP-сервере методом SIP REGISTER.

Если какая-либо из проверок не проходит, Watchdog отправляет предупреждение на почту и создает на сервере отчет, содержащий диагностическую информацию,
которую можно передать в техподдержку для выяснения причин сбоя.

Далее Watchdog пытается устранить проблему путем перезапуска SIP-стека или всего серверного процесса.
После того как сервер перезапущен и все проверки пройдены успешно, Watchdog отправляет дополнительное письмо-уведомление о том что сервер был перезапущен и работает нормально.

Работу Watchdog можно описать следующим образом:

1. Есть три критических события:

a) CoreProcessDown - Требует перезагрузки процесса.
b) EventScannerDown - Не требует перезагрузки процесса, только восстановление SIP-стека.
c) SIPRegDoesNotWork - Требует перезагрузки процесса.

2. Последовательность действий если произошло одно из критических событий.

a) Создание отчета с дампами.
b) Отправка уведомления о сбое.
c) Перезагрузка процесса если требует событие (опционально).
d) Новый цикл проверки и отправка уведомления о восстановлении после сбоя.

Настройки и запуск

Настройки Watchdog хранятся в конфиге watchdog.properties. Запуск и остановка Watchdog производится из CLI командами watchdog start и watchdog stop.
Кроме этого можно настроить автоматический запуск Watchdog. Для этого используется настройка watchdog_autorun=true в конфиге watchdog.properties.

Если вы все настроили верно и запустили Watchdog, то в случае проблем с сервером на указанную в настройках почту будут приходить отчеты о работе Watchdog.

Примеры уведомлений:

1) Уведомление о сбое.

```
[T16] WCS core process is down or unavailable.

To change Watchdog behavior please edit watchdog.properties file.
To start or stop Watchdog please use 'watchdog start' and 'watchdog stop' CLI commands.
To start Watchdog automatically, set watchdog_autorun=true in the watchdog.properties.

Best regards,
Flashphoner Watchdog
```

2) Уведомление о восстановлении после сбоя.

```
[T16] WCS is up and running

To change Watchdog behavior please edit watchdog.properties file.
To start or stop Watchdog please use 'watchdog start' and 'watchdog stop' CLI commands.
To start Watchdog automatically, set watchdog_autorun=true in the watchdog.properties.

Best regards,
Flashphoner Watchdog
```

Тест с недоступностью процесса

Для тестирования доступности сервера можно убить рабочий серверный процесс командой 'kill', например:

```
kill -9 19522
```

где 19522 - это ID процесса(PID), который можно увидеть командой:

```
ps aux | grep WebCallServer
```

Через некоторое время Watchdog должен перезагрузить сервер и отправить отчет о проблеме на почту. После этого Watchdog должен выполнить повторную проверку и отправить на почту сообщение о доступности сервера и готовности к работе.

Тест SIP

Чтобы включить проверку SIP регистраций, нужно добавить событие SIPRegDoesNotWork, чтобы настройка выглядела так: watchdog_events=CoreProcessDown,EventScannerDown,SIPRegDoesNotWork.

Для проведения теста отключим исходящий SIP-порт на Firewall для того чтобы не проходили SIP REGISTER запросы и Watchdog начал реагировать.

```
iptables -A OUTPUT -p udp -m udp --dport 5060 -j DROP
```

Через некоторое время Watchdog пришлет на почту отчет о том, что он не смог подключиться к SIP серверу и начал перезагрузку WCS Core. Перезагрузка WCS сервера здесь не поможет, т.к. SIP порт был закрыт на Firewall.

Удаляем правило и открываем SIP порт:

```
iptables -D OUTPUT 1
```

SIP сервер снова доступен и WCS регистрирует тестового SIP-клиента для валидации SIP соединения. Далее на почту будет отправлено сообщение Watchdog о том, что WCS снова готов к работе.

Логирование Watchdog

Подсистема проверки доступности Watchdog ведет отдельные логи в файле logs/watchdog/watchdog.log.

```
15:08:06,406 check CoreProcessDown - ok
15:08:06,410 check EventScannerDown - ok
15:08:36,414 watchdog - next cycle
15:08:36,433 check CoreProcessDown - ok
15:08:36,437 check EventScannerDown - ok
15:09:06,439 watchdog - next cycle
```

Так выглядит лог Watchdog в котором отображаются успешные проверки.
Для конфигурации логгирования используется стандартный конфиг watchdog.log4j.properties. Логи пишутся с почасовой ротацией в формате

```
'%d{HH:mm:ss,SSS} %-5p %20.20c{1} - %t %m%n'
```