# Домино8 в режиме WEB-сервера. Замеры производительности.

## Тест производительности

### Описание теста

Тест выполняется локально на одном компьютере, чтобы исключить влияние скорости передачи данных по сети.

Один экземпляр Домино8 запускается в режиме WEB- сервера с полностью отключённым протоколированием. На проекте написан простейший метод, который без задержек отвечает на входящий http-запрос. Ответ представляет собой текст в формате xml длиной 108 байт.

Второй экземпляр Домино8 запускается в обычном режиме (клиента). На проекте написана процедура, которая в течении 60 секунд непрерывно в цикле вызывает описанный выше метод web-сервера. По окончании работы процедура показывает количество запросов, которое web- сервер успел обработать за прошедшее время.

В первом варианте теста запускается один экземпляр Домино-клиента, который посылает запросы на сервер. Во втором варианте теста запускается пять экземпляров Домино-клиента, которые одновременно посылают запросы на сервер.

Тестируется два варианта запуска WEB- сервера - с прослушиванием localhost и с прослушиванием реального ip-адреса сервера. В первом случае запросы от клиента к серверу идут через внутренний loopback, во втором - через сетевой интерфейс, что требует больших ресурсов.

## Тестовые платформы

1. Intel Core2 Duo E8400 @3GHz, 8 Gb (одно ядро + HT)

2. Intel Xeon E5 2690 @3GHz, 16 Gb (четыре ядра без HT)

# Результаты тестирования

Измерялась производительность web-сервера, запросов в секунду

| Платформа | | Intel Core2 Duo E8400 | Intel Xeon E5 2690 |
|---|---|---|---|
| Один клиент | localhost | 3600 | 2900 |
| | ip | 3200 | 2500 |
| Пять клиентов одновременно | localhost | 7400 | 9300 |
| | ip | 6600 | 8600 |

# Тест на размер очереди входящих сообщений

## Описание теста

Тест выполняется локально на одном компьютере, чтобы исключить влияние скорости передачи данных по сети. Тестируется вариант запуска WEB- сервера с прослушиванием localhost.

Экземпляр Домино8 запускается в режиме WEB- сервера с полностью отключённым протоколированием. На проекте написан простейший метод (аналогично предыдущему тесту), который отвечает на входящий http-запрос с задержкой в 0.1 сек. Ответ представляет собой текст в формате xml длиной 108 байт.

В качестве клиента используется специально написанное многопоточное приложение. Каждый поток в цикле без задержек последовательно посылает 5 http- запросов к тестируемому серверу и получает от него ответы. Результаты и время выполнения записываются в отдельный протокол потока. Всего параллельно запускается 500 потоков, таким образом в очереди на сервере будет постоянно находится 500 запросов, а ожидаемое время ответа сервера на запрос будет примерно 500 * 0.1 = 50 сек.

На втором этапе последовательно увеличивается число потоков клиента до появления ошибок в ответах сервера. Так определяется максимальное количество входящих соединений для сервера и его реакция на перегрузку.

# Тестовая платформа

Intel Core2 Duo E8400 @3GHz, 8 Gb (одно ядро + HT)

# Результаты тестирования

По логам проверялось отсутствие ошибок в журналах сервера и клиента, среднее время выполнения запроса и количество выполненных запросов.

Ошибки в логах сервера и клиента отсутствуют.

Среднее время выполнения запроса 50-55 секунд, что примерно соответствует расчётному 0.1 сек. на запрос при 500 запросах в очереди.

Количество выполненных запросов во всех логах клиента одинаковое - 5.

Предельное количество потоков, при котором система работает без ошибок - 1000. При дальнейшем увеличении числа потоков начинают регистрироваться ошибки закрытия соединения со стороны сервера «WebException : The underlying connection was closed: An unexpected error occurred on a receive». Таким образом можно предположить, что количество входящих соединений web-сервера равно 1000 при настройках «по умолчанию».

Для увеличения числа входящих соединений нужно менять настройки Microsoft IIS (смотрим документацию).

# Тест связки NGINX + пул web-серверов Домино

## Описание теста

Сервер NGINX развернут на VM с CentOS7. Никаких специфических настроек ОС не делалось.

После установки linux рекомендуется отключить SELinux. В противном случае возникает масса проблем с настройками nginx, когда отклоняются те или иные операции. Можно также отключить SE только для домена httpd_t

semanage permissive -a httpd_t

В настройках /etc/nginx/nginx.conf делаем следующее:

Контекст main

```
# Увеличиваем общее число файлов для рабочих процессов. Так как мы работаем
# в режиме proxy, то минимум число входящих соединений x2 + резерв
worker_rlimit_nofile 65535;

events {
  # Увеличиваем число соединений на рабочий процесс
  worker_connections 4096;
 # Включаем возможность одновременного приёма множества подключений
  multi_accept on;
}
```

Контекст http

```
# Включаем буферизацию логов доступа
access_log ... buffer=32k flush=10s;

# Описываем пул web- серверов Домино
upstream DominoWeb {
  server 192.168.1.150:8081;
  server 192.168.1.150:8082;
  server 192.168.1.150:8083;
  ...
  server 192.168.1.150:8098;
}
```

Контекст server

```
# Слушаем порт 8080
listen 8080;

# Локация для перенаправления запросов
location /domino/ {
  # Проксируем запросы на пул DominoWeb по методу round-robin
  proxy_pass http://DominoWeb;
```

```
}
```

Пул из 16 WEB- серверов Домино развернут на VM с Win19 Server. Память сервера 8Гб, процессоры 2x Intel Xeon E5 2680 @2.4GHz (по два ядра без HT). Суммарно web- сервера требуют 16 * 300Мб = 4,8Гб оперативной памяти.

В настройках firewall добавили правило, разрешающее входящие tcp соединения на порты 8080-8099.

Каждый экземпляр Домино8 WEB- сервера запускается на отдельном порту и с полностью отключённым протоколированием.

```
start C:\Domino\WebService\BIN\domino8.exe
    C:\Domino\WebService\PROJECT\RETAIL-STORE-2010
    /SERVER
    LISTEN=http://192.168.1.150:8081/domino/
    DBSERVER=oracle4
    SCHEME=MEGAITALIA
    USERNAME=Администратор
    TOKEN=81
```

На проекте написан простейший метод (аналогично предыдущему тесту), который отвечает на входящий http-запрос с регулируемой задержкой (значение задержки передаётся параметром запроса). Ответ представляет собой текст в формате xml длиной 300 байт

В качестве клиента для тестирования используется специально написанное многопоточное приложение. Каждый поток в цикле без задержек последовательно посылает 100 http-запросов к тестируемому серверу и получает от него ответы. задержка ответа сервера задаётся параметром запроса delay. Результаты и время выполнения записываются в отдельный протокол потока. Общее число потоков задаётся параметром JOBS. Таким образом в очереди на сервере будет постоянно находится JOBS запросов, которые будут распределяться между 16 web-серверами Домино. Ожидаемое время ответа сервера на запрос будет примерно JOBS * delay / 16

В процессе тестирования определяются значения параметров JOBS и delay, при которых появляются ошибки в ответах сервера. Так как число JOBS ограничено 1000, то для проверки работы бОльшего числа одновременных соединений с сервером запускается несколько экземпляров тестового приложения-клиента.

PS: При работающем SELinux могут быть такие проблемы:

Если после настройки прокси в nginx.conf  location ... { proxy_pass домино_web_сервер} мы получаем ошибку 502 Bad Gateway, а в логах nginx видим (13: Permission denied) while

connecting to upstream, то нужно разрешить в SELinux upstream http соединения:

> setsebool -P httpd_can_network_connect 1

Для того, чтобы при включённом SELinux увеличить число файлов на рабочий процесс (worker_rlimit_nofile) нужно разрешить эту операцию:

> setsebool -P httpd_setrlimit 1

# Результаты тестирования

По логам проверялось отсутствие ошибок в журналах сервера и клиента, среднее время выполнения запроса и количество выполненных запросов при изменении параметров JOBS и delay.

Стабильная работа описанной связки наблюдается только при JOBS < 200. Далее в логах клиента начинают регистрироваться периодические ошибки невозможности связи с сервером. При этом в логах сервера ошибок нет. При увеличении delay до 100ms безошибочная работа достигается при JOBS порядка 500. Предварительная гипотеза - исчерпание каких-то ресурсов, связанных с сетевыми протоколами (по аналогии с DDos атаками). Источником ошибок может быть как сервер nginx, так и тестовый клиент.

Тема требует дальнейшего исследования с привлечением профильных специалистов по nginx и сетевым протоколам.

---

Версия #3

Демонов Сергей создал Mon, Jul 3, 2023 7:44 AM

Демонов Сергей обновил Mon, Jul 3, 2023 7:53 AM