Тестирование производительности веб-протокола SPDY при помощи собственного клиента.

КН-301, Колмогорцев Егор.

Введение, постановка проблемы.

Как известно, доступ к веб-страницам осуществляется через прокол HTTP, однако высокая производительность не является его достоинством. Google выделяет следующие проблемы HTTP, негативно сказывающиеся на скорости:

1. Один запрос на одно соединение. Из-за этого ради каждого нового запроса приходится заново устанавливать соединение.
2. Запросы может инициировать только клиент. Нет механизма, по которому сервер мог бы в чем-либо уведомить клиента.
3. Заголовки HTTP передаются несжатыми. Размер заголовков при запросе может достигать существенных значений (до нескольких килобайт) за счет cookies, что может замедлить работу у клиентов с несимметричным каналом (высокая скорость загрузки и низкая скорость отдачи).
4. Заголовки, которые не изменяются (например, User-Agent) приходится заново посылать при каждом запросе.
5. Сжатие передаваемых данных возможно только опционально. Оптимальным является сжимать данные всегда.

Для решения данных проблем, в корпорации Google был разработан протокол SPDY (подробнее о нем ниже). Целью данной работы было создание своего клиента, работающего по протоколу SPDY и тестирование его скорости работы.

Существующие решения.

- HTTP pipelining, введенный в стандарте HTTP/1.1 отчасти решает проблему 1, предоставляя очередь HTTP запросов через одно TCP соединение, однако он не реализован в большинстве современных браузеров.
- Параллельные соединения - одновременная загрузка нескольких ресурсов в разных соединениях.
- Stream Control Transmission Protocol (SCTP), Structured Stream Transport (SST) - протоколы транспортного уровня, предлагаемые на замену TCP, и, как следствие, испытывающие огромные проблемы в массовом внедрении.
- MUX and SMUX - Протоколы сессионного уровня, были предложены в одно время с HTTP/1.1, на практике не получили широкого распространения.

Что такое SPDY?

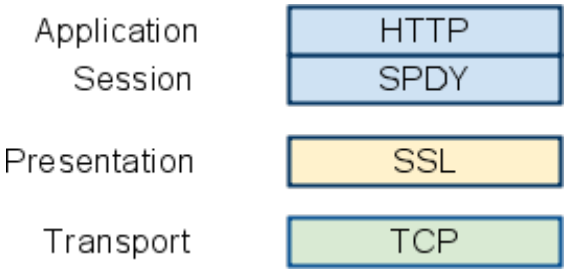
SPDY - экспериментальный протокол, разработанный в корпорации Google для уменьшения времени загрузки веб-страниц.

Цели, поставленные при разработке:

- Главная цель - уменьшить время загрузки веб - страниц на 50%
- Протокол не должен требовать изменений в TCP и HTTP, для облегчения внедрения
- Обеспечить возможность нескольких параллельных HTTP - запросов через одно TCP соединение
- Предоставить серверу возможность самому инициировать дополнительные соединения.
- Обеспечить возможность передачи данных через зашифрованное соединение.

Реализация:

SPDY работает на сессионном уровне стэка протоколов, при передаче данных через зашифрованное SSL соединение, SPDY работает поверх SSL.



Для обеспечения одновременный работы сервера как с SPDY, так и со стандартным HTTP, было разработано расширение протокола TSL под названием NPN (Next Protocol Negotiation), которое позволяет клиенту и серверу указывать список протоколов, с которыми они могут работать поверх зашифрованного соединения.

Основные достоинства:

- **Параллельные соединения**
  SPDY позволяет создавать параллельные HTTP - соединения через одно TCP - соединение, что позволяет избежать дополнительных расходов на установку и обслуживание TCP - соединений.
- **Приоритеты запросов.**
  Возможно явно задать приоритет обработки множества одновременных запросов.
- **Сжатие HTTP - заголовков.**
  В SPDY HTTP - заголовки запросов и ответов сжимаются через один поток Zlib со специально подобранным словарем, что позволяет уменьшить размер одинаковых заголовков.
- **Server push & server hint.**
  SPDY расширяет протокол HTTP залоговками X-Subresources и X-Associated-Content. X-Subresources реализует так называемый “server hint”, который позволяет серверу информировать клиента о ресурсах, которые ему точно понадобятся. Server push позволяет серверу самому инициировать SPDY поток с HTTP - заголовком X-Associated-Content и загружать данные, до того как клиент их запросит.

Первая спецификация протокола была опубликована 11 Ноября 2009г. На данный момент используется вторая, третья находится в стадии разработки.

Тестирование.

Для проведения собственного тестирования протокола был написан SPDY - клиент на языке Perl.

Конфигурация:

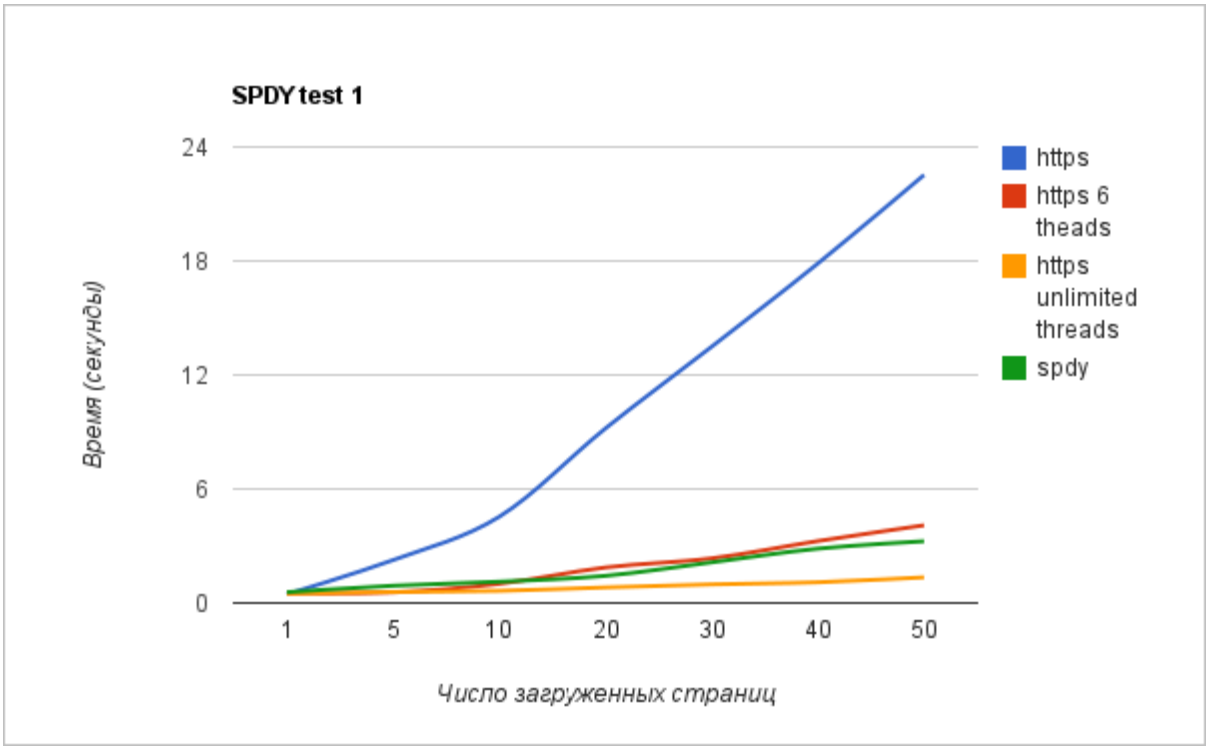
- Ubuntu 10.10
- Perl v. 5.10.1
- Пропускная способность интернет-соединения: 20 Mbps отдача/загрузка
- Загружаемый ресурс: запросы вида encrypted.google.com/search?q=n, где n принимает значения от 1 до количества загружаемых страниц в тесте.
- Скорость до загружаемого ресурса: ping = 237 ms, загрузка - 6,51 Mbps, отдача - 4.3 Mbps.

При тестировании загрузка страниц производилась 4 способами:

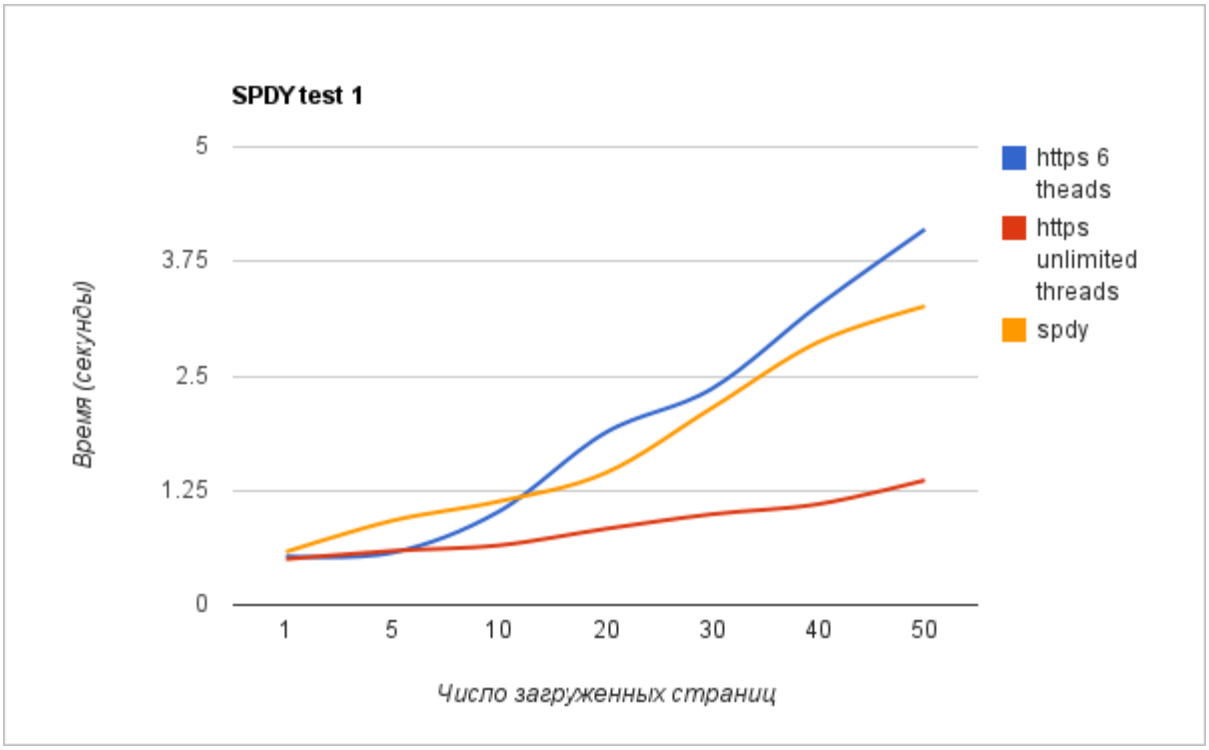
- Последовательная загрузка - самый простой вариант, не использующий каких либо оптимизаций (осуществлялась через функцию get_https модуля Net::SSLeay)
- Параллельная загрузка в 6 потоков (как в текущих реализациях популярных веб-браузеров)
- Параллельная загрузка без ограничения на число потоков - потребляющий наибольшее число ресурсов вариант.
- SPDY - соединение

Тест 1:
Загрузка только HTML страницы (размер около 70 кб).

кол-во страниц	последовательный https	https 6 threads	https unlimited threads	spdy
1	0.46	0.52	0.49	0.57
5	2.26	0.56	0.58	0.91
10	4.54	1.01	0.64	1.12
20	9.19	1.87	0.82	1.43
30	13.49	2.35	0.98	2.14
40	17.87	3.26	1.09	2.86
50	22.53	4.09	1.35	3.25



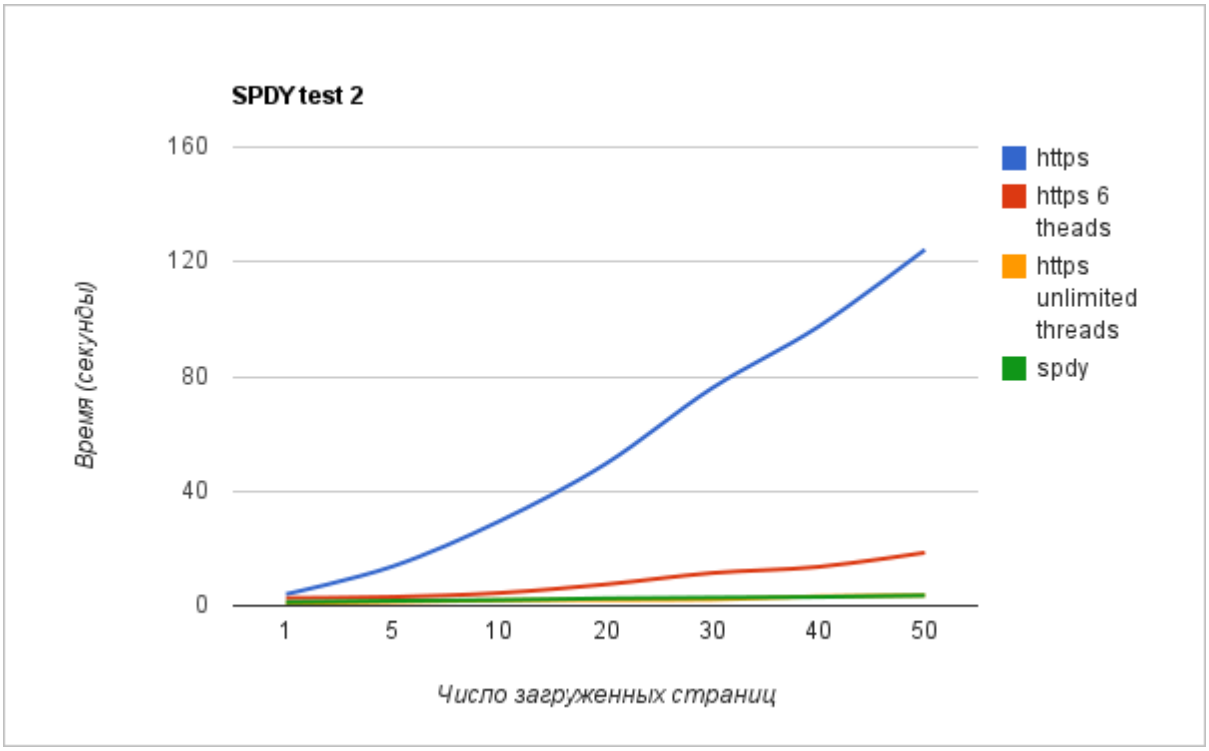
Крупным планом, без последовательного https:



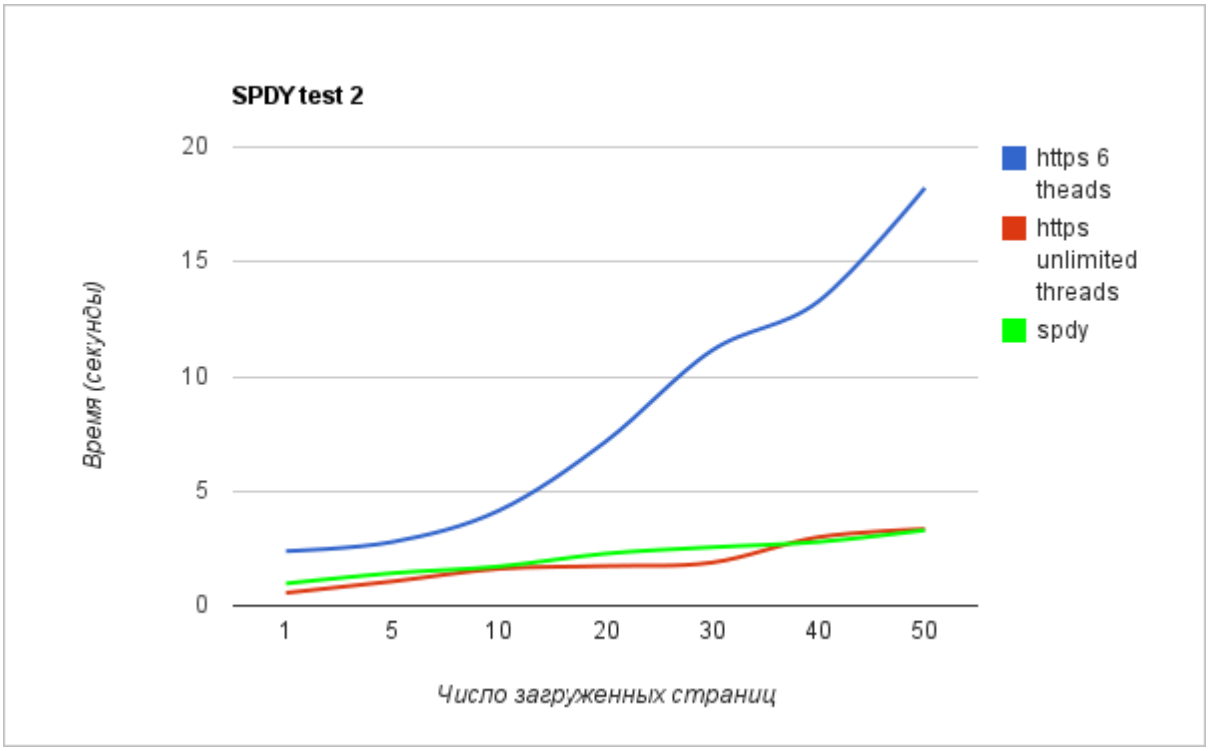
Тест 2:

Загрузка страницы вместе с доп. контентом (изображения, скрипты итд), суммарный размер примерно 250 kb. Парсинг html - страницы и извлечение сслылок осуществлялось через модуль HTML::LinkExtor.

Кол-во страниц	последовательный https	https 6 threads	https unimited threads	spdy
1	3.71	2.34	0.52	0.93
5	13.38	2.74	1.02	1.38
10	29.1	4.11	1.57	1.67
20	49.13	7.11	1.69	2.23
30	75.56	11.09	1.84	2.51
40	97.04	13.23	2.95	2.74
50	123.9	18.18	3.3	3.25



Крупным планом, без последовательного https:



Выводы, заключение:

Как видно из тестов, даже относительно примитивная реализация spdy клиента значительно опережает загрузку страниц через https при числе загружаемых ресурсов большем, чем число одновременных соединений.
SPDY также имеет преимущества перед параллельной загрузкой всех ресурсов: экономия памяти (как на клиентской, так и серверной стороне) и траффика (установка и поддержание https - сессии оказывает нагрузку на канал передачи данных), а возможность инициации потоков со стороны сервера позволяет начать загрузку всего дополнительного контента до получения и разбора html - страницы. На данный момент SPDY используется в браузере Google Chrome и во всех сервисах Google, доступ к которым осуществляется через https, планируется реализация SPDY клиента в Mozilla Firefox, также SPDY используется в Amazon Kindle.