**УДК 004.057.4**

*Старший викладач Грицька Т.С.,*

*старший викладач Кубракова К.М.,*

*Овчаренко А.І., студент гр. 401-ТК*

*Полтавський національний технічний університет*

*імені Юрія Кондратюка*

ДОСЛІДЖЕННЯ МЕРЕЖЕВОГО ПРОТОКОЛУ HTTP/2

*У статті розглянуто нововведення у мережевому протоколі HTTP/2, досліджено переваги у порівнянні із попередніми протоколами, проаналізовано приріст у швидкості завантаження сторінки при використанні нового протоколу.*

**Ключові слова:** мережа, протокол, HTTP, HTTP/2, SPDY, стиснення, шифрування.

Вступ

Протокол HTTP (англ. HyperText Transfer Protocol – «протокол передачі гіпертексту») – протокол прикладного рівня передачі даних (розроблявся для передачі гіпертекстових сторінок, зараз використовується для передачі будь-яких даних).

HTTP базується на технології «клієнт-сервер», обмін повідомленнями реалізований за звичайною схемою «запит-відповідь». HTTP може ідентифікувати дані за допомогою URI (англ. Uniform Resource Identifier) – уніфікований ідентифікатор ресурсу, такими ресурсами можуть виступати як звичайні файли так і логічні об’єкти.

Зараз HTTP використовується у Всесвітній павутині для отримання інформації з веб-сайтів. Вже у 2007 році у Північній Америці частка HTTP-трафіку перевищила частку P2P-мереж і склала 46%, з яких майже половина – це передача потокового відео і звуку [1].

Оскільки остання специфікація HTTP1.1 була прийнята в 1997, а HTTP/2 була опублікована як RFC 7540 у травні 2015, на даний момент, за даними W3Techs, лише 7% всіх веб сайтів підтримують протокол HTTP/2 [2]. Метою даної статті є дослідження протоколу HTTP/2, ознайомлення з його перевагами, проведення порівняльного аналізу даного протоколу з попередньою версією, виділення основних змін у принципі роботи. Кінцевою метою є проведення порівняльної характеристки швидкості завантаження сайтів з різними мережевими протоколами HTTP1.1, SPDY, HTTP/2.

Основна частина

Оскільки протокол HTTP став протоколом, що використовується буквально для всього в мережі Інтернет, то сьогодні простіше запустити щось поверх HTTP, ніж створювати щось нове. Коли HTTP був створений, він передавав лише гіпертекст і сприймався як простий протокол, але розробникам стандарту вдалося додати велику кількість опцій, що можна змінювати, це й дозволило йому залишатись основним протоколом протягом двадцяти років. Але за цей час веб-сторінки набули багатьох змін, тепер вони мають динамічно оновлюваний контент, багато мультимедійних форматів, які є ресурсозатратними, кількість файлів для відправки досягає сотень, а загальний розмір в середньому сягає 1,5Мб. А оскільки протокол HTTP дуже чутливий до затримок, то ці тенденції спочатку змусили Google створити протокол SPDY, перша версія якого вийшла в 2009, і хоча й мала багато інновацій, але базувалась на HTTP1.1. Згодом був створений протокол HTTP/2, який в свою чергу багато в чому базується на SPDY і має сумісність з HTTP1.1.

Основні відмінності від HTTP1.1:

* протокол HTTP/2 є бінарним;
* підтримує мультиплексування;
* пріоритезація (кожному запиту можна призначити пріоритет);
* стиснення заголовків;
* стиснення даних;
* Server push.

Розглянемо детальніше кожну особливість.

**HTTP/2 є бінарним,** він відправляє фрейми. Усі фрейми мають однакову структуру: тип, довжина, прапори, ідентифікатор потоку і самі дані фрейма. Існує десять різних типів фреймів, визначених у специфікації HTTP/2, і, в тому числі, два, можливо найбільш важливих, які звязують з HTTP1.1: DATA (дані) і HEADERS (заголовки).

HTTP/2 є бінарним для того, щоб зробити формування пакетів простіше. Визначення початку і кінця пакету – одне з найскладніших завдань в HTTP1.1 і в усіх текстових протоколах в принципі. Крім того, це дозволяє набагато простіше розділяти частини, повʼязані з самим протоколом і пакетом даних, що в HTTP1 безладно перемішано [4].

Протокол дозволяє використовувати стиснення і часто працює поверх TLS, що знижує цінність тексту, оскільки в будь-якому випадку більше не видно відкритого тексту, а лише набір зашифрованих символів. Щоб зʼясувати, що відбувається на рівні протоколу HTTP/2, треба використовувати аналізатор Wireshark або щось схоже.

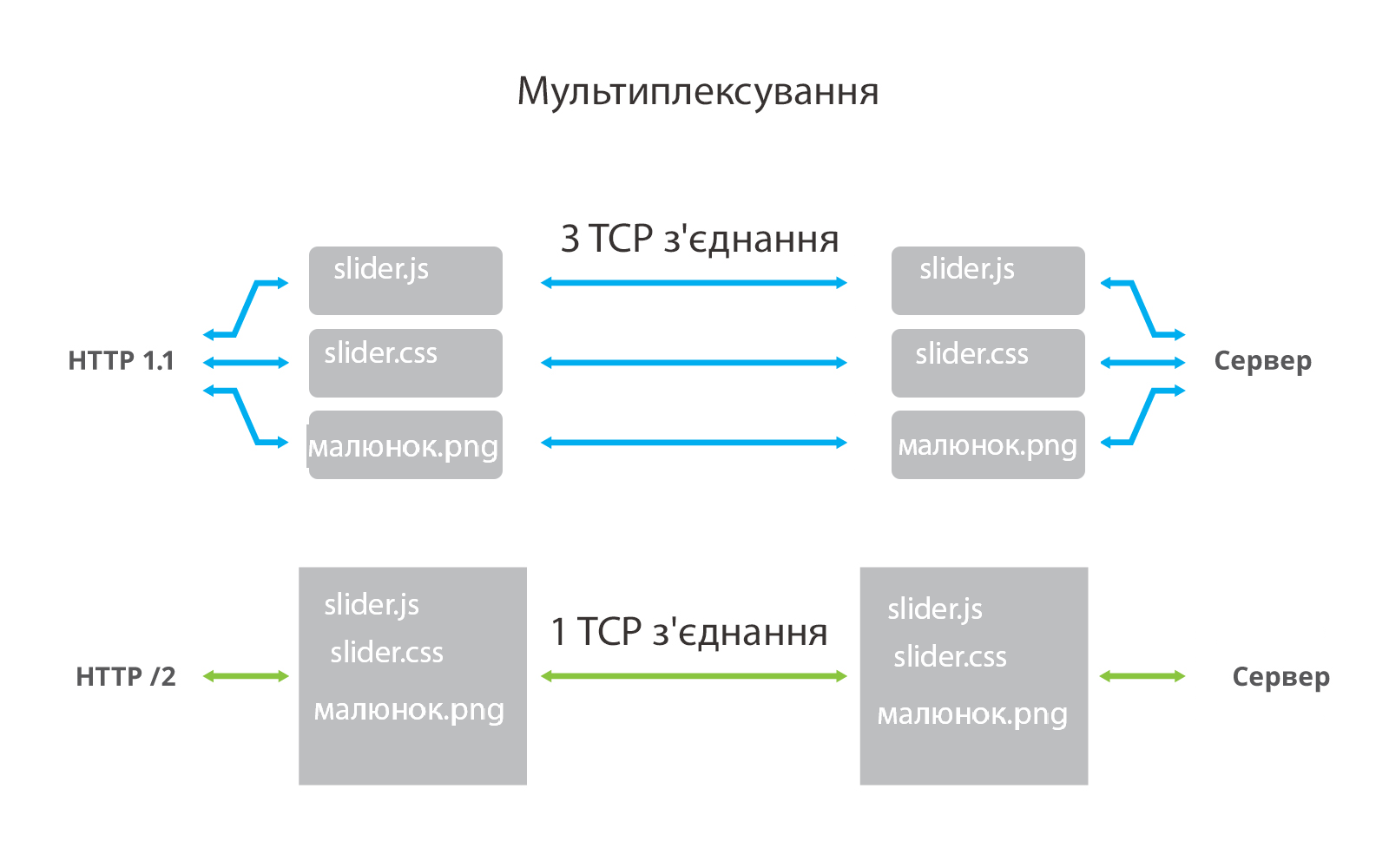
Використання бінарних команд має досить багато переваг:

* менші витрати на парсинг даних;
* нижча схильність до помилок;
* ефективе використання мережевих ресурсів;
* усунення проблем безпеки, що були повʼязані з текстовою природою HTTP1.1;
* можливість існування інших нових функцій протоколу (мультиплексування, стиснення, пріоритезація).

**Мультиплексування потоків**

Ідентифікатор потоку привʼязує кожен фрейм, який передається поверх HTTP/2, до так званого «потоку». Потік – це логічна асоціація, незалежна двостороння послідовність фреймів, якими обмінюються клієнт з сервером всередині HTTP/2-зʼєднання. Одне HTTP/2-з'єднання може містити безліч одночасних відкритих потоків від будь-якої зі сторін. Потоки можуть бути встановлені і використані в односторонньому порядку або спільно використані як клієнтом, так і сервером, і можуть бути закриті кожною зі сторін. Важливий порядок, в якому відправляються фрейми. Отримувач обробляє фрейми в порядку їх отримання. Мультиплексування потоків означає, що пакети безлічі потоків змішані в рамках одного з'єднання.

Усі ці можливості дають змогу пакетам даних із різних потоків змішуватись і передаватись по одному TCP-з'єднанню. Потім ці пакети розділяються і одержувач отримує окремі потоки



*Рис.1 Мультиплексування потоків*

Можливо, це найголовніша перевага HTTP/2 над HTTP1.1, оскільки у HTTP1.1 для передачі декількох паралельних запитів необхідно було відкривати декілька TCP-з'єднань, що відбивалось на загальній продуктивності мережі, незважаючи на передачу більшої кількості потоків за меншу кількість часу.

Використання мультиплексування має наступні переваги:

– паралельні запити не блокують один-одного ;

– одне TCP-з’єднання забезпечує ефективне використання мережевих ресурсів;

– зменшує затримки, дозволяє відмовитись від прийомів оптимізації.

**Пріоритети і залежності**

Усі потоки мають пріоритет, використовуваний для того, щоб показати іншому учаснику обміну, які потоки вважати більш важливими у тому випадку, якщо є обмеження в ресурсах.

Пріоритети можуть змінюватися динамічно, що дозволяє користувачеві при відкритті у браузері сторінки, заповненої зображеннями, вказати, які саме зображення є найбільш важливими, або при переключенні сторінки, підвищити пріоритет потокам, які несподівано потрапили у фокус.

На практиці сервер рідко має контроль над ресурсами. Також пріоритезація має багато проблем з упровадженням і потребує багато доповнень від авторів протоколу.

У HTTP/2 пріоритетизація є не обов'язковим, але бажаним методом. Однак мультиплексування без неї працювати належним чином не буде.

**Стиснення HTTP-заголовків**

Сучасні веб-сторінки мають у своєму складі багато контенту та графіки, а оскільки HTTP є протоколом без стану, що означає що кожен клієнтський запит повинен супроводжуватися великою кількістю інформації, потрібної серверу для виконання необхідної операції. Цей механізм викликає потоки, що несуть кадри повторюваної інформації, оскільки сам сервер не зберігає інформацію з попереднього запиту клієнту. Оскільки сучасні сторінки мають велику кількість елементів, то це викликає передачу великої кількості майже однакових фреймів, що викликає затримки та зайве навантаження на мережу. HTTP/2 вирішує ці проблеми тим, що заголовки передаються в стислому вигляді.

Стиснення HTTPS і SPDY виявилися вразливими до атак типу BREACH і CRIME. Шляхом вставки відомого тексту в потік і спостереження за тим, як змінюється зашифрований висновок, атакуючий міг зʼясувати, що було надіслано. Виконання стиснення для динамічного контенту в протоколі без ризику бути атакованим вимагає серйозних спеціальних заходів. Так з'явився алгоритм HPACK, призначений спеціально для стиснення HTTP/2-заголовків. Новий формат містить, наприклад, спеціальні прапори, які просять посередників не стискати певні заголовки і опціонально додавати у фрейми зайві порожні дані, щоб ускладнити атаку на стиснення [4].

**Server push (посилання сервера)**

Ця можливість також відома як «посилання до кеш». Ідея в тому, що сервер надсилає додаткову інформацію клієнту, яка ще не була запитана, але очікується її запит у майбутньому. Дана інформація поміщається у кеш і зберігається там до її запиту.

Переваги використання Server push:

* клієнт зберігає отримані дані в кеш;
* клієнт може використати ці дані на інших сторінках;
* сервер може мультиплексувати передачу разом із запитом на одному TCP-зʼєднанні.

**Порівняння протоколів**

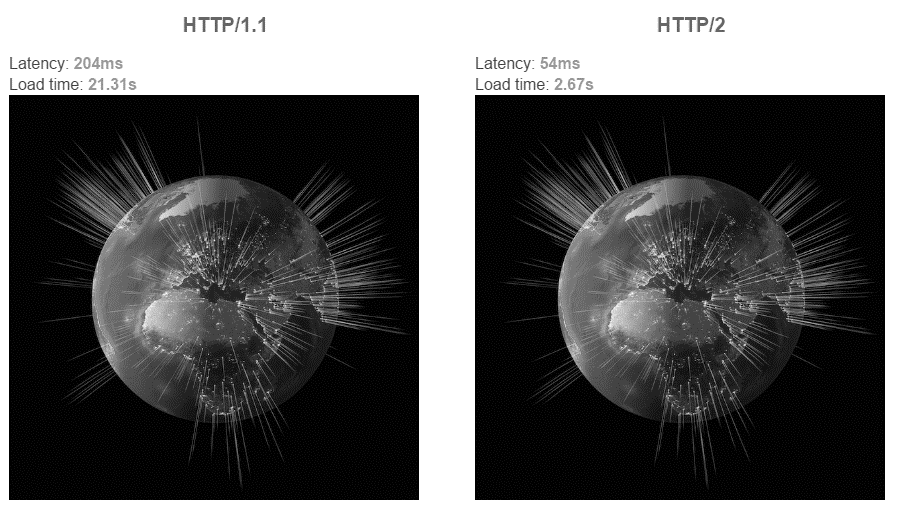
HTTP/2 заснований на SPDY, альтернативі HTTP1.1 від Google. Реальні відмінності полягають у механізмах, що використовуються для обробки запитів клієнт-сервер. У наведеній нижче таблиці визначені деякі схожості і відмінності між HTTP1.1, SPDY і HTTP/2.

*Таблиця.1. Порівняння основних протоколів*

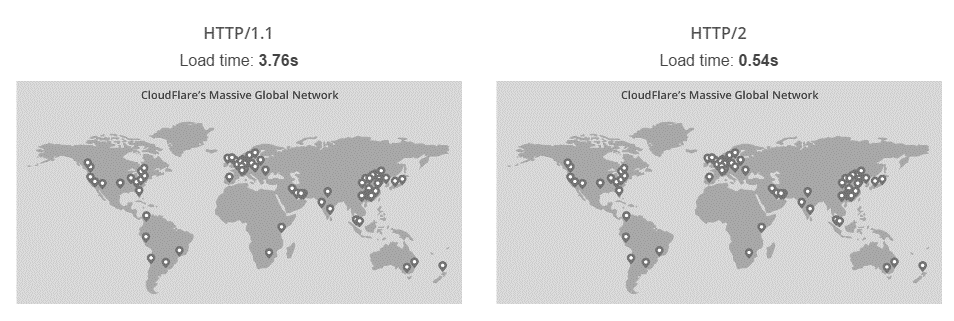
|  |  |  |
| --- | --- | --- |
| HTTP1.1 | SPDY**S** | HTTP/2**2** |
| SSL не є обов’язковим | SSL є обов’язковим | SSL не є обов’язковим, але рекомендується |
| Повільне шифрування | Швидке шифрування | Покращене шифрування (у порівнянні із SPDY) |
| Один запит клієнт-сервер на одне TCP з’єднання | Декілька запитів клієнт-сервер за одне TCP з’єднання. Виконується на одному хості одночасно | Мульти-хост мультиплексування. Виконується на декількох хостах в одиницю часу |
| Немає стиснення заголовків | Стиснення заголовків присутнє | Стиснення заголовків з використанням покращених алгоритмів, які збільшують продуктивність і безпеку |
| Немає пріоритетизації потоків | Пріоритетизація потоків присутня | Використовуються покращені механізми потоку пріоритетизації |

**Порівняння швидкості завантаження сторінок**

Скористаємося кількома готовими демо та проаналізуємо сайт, що вже використовує HTTP/2



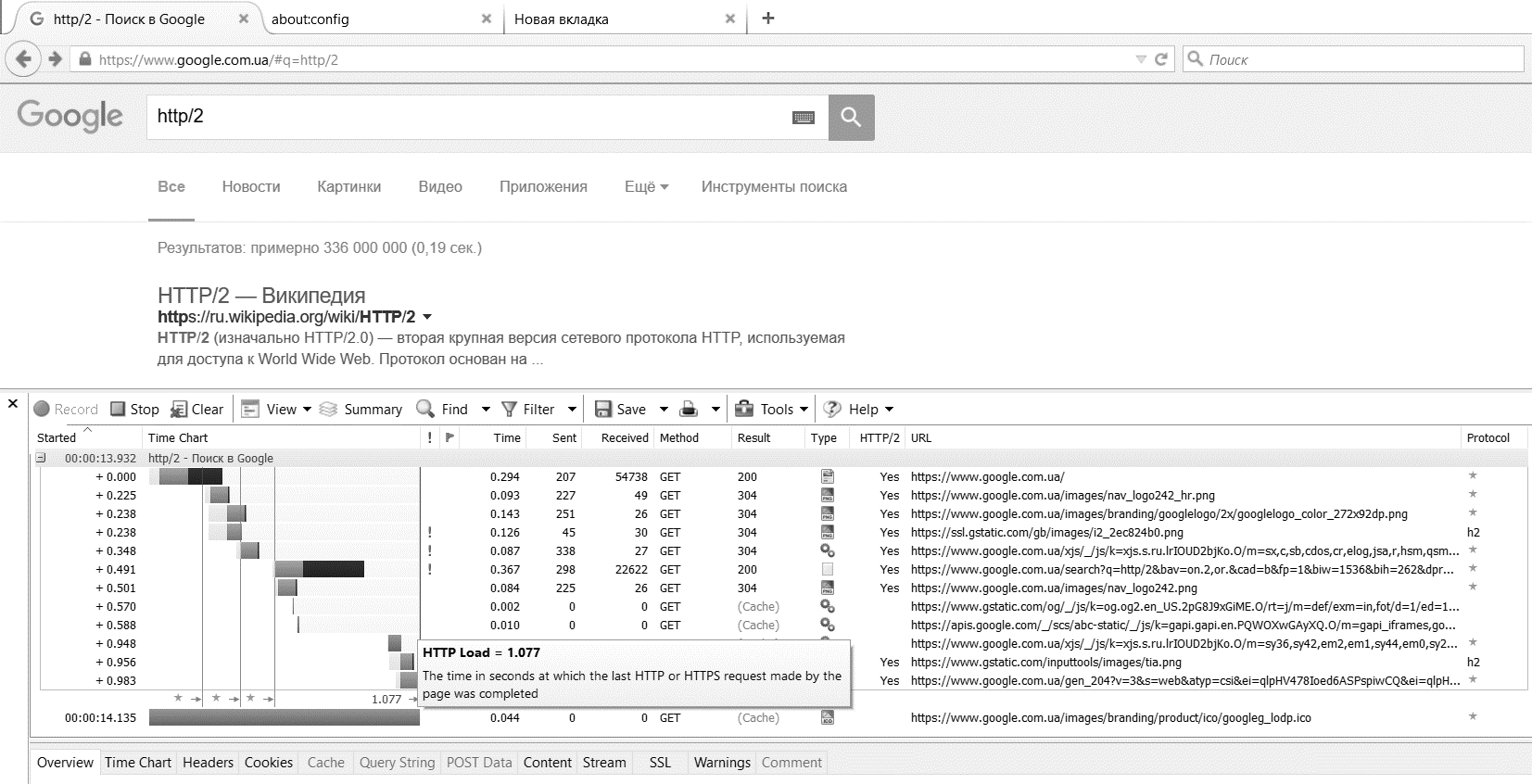
*Рис.2. Порівняльне демо від akamai.com [6]*



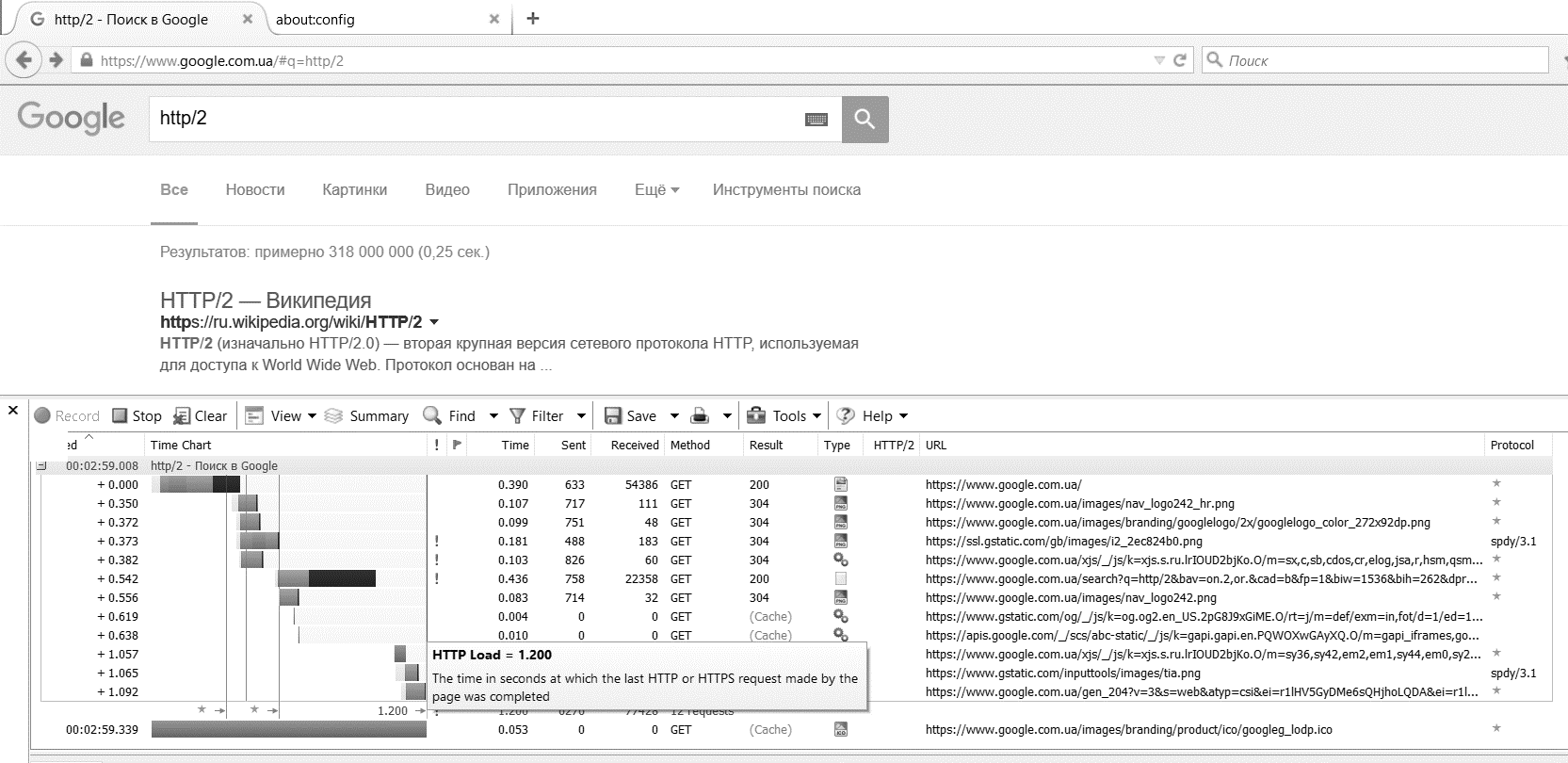
*Рис.3. Порівняльне демо від cloudflare [7]*

Дані демонстрації використовують ідеальні умови для HTTP/2, велику кількість зображень (для тесту використовувались 200 зображень), тому HTTP1.1 програє у швидкодії залежно від браузеру приблизно в 10-12 разів.

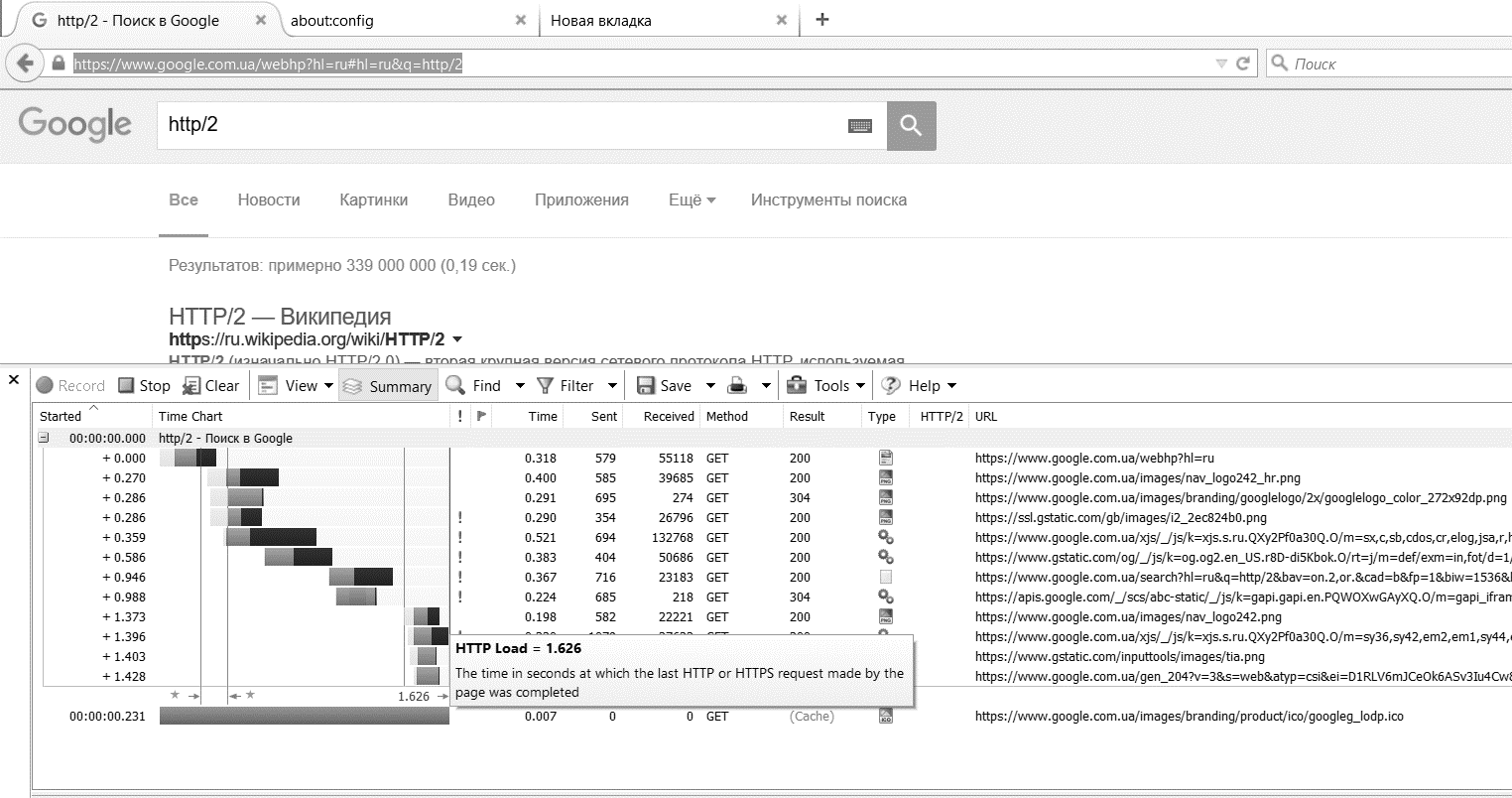
Хоча стандарт передбачає роботу і без обов’язкового шифрування, але на даний момент браузери не підтримують цю функцію, тому тестування на локальному сервері без ssl-сертифікату неможливе. Для тесту на реальному сайті використаємо <https://www.google.com>.ua, оскільки на серверах вони паралельно підтримуються HTTP/2 та SPDY. Скориставшись HttpWatch [5] у браузері Firefox, отримано наступні результати:



*Рис.4. Час завантаження при використанні HTTP/2*



*Рис.5. Час завантаження при використанні SPDY*



*Рис.6. Час завантаження при використанні HTTP1.1*

Провівши декілька тестів, в середньому приріст у швидкості у порівнянні з HTTP1.1 для HTTP/2 складає 60%. Як бачимо, різниця у швидкості для реальних веб-сторінок відрізняється від демонстраційних тестів, але чим більше на сторінці буде медіа контенту, тим більше буде різниця між часом завантаження. Також час завантаження залежить від налаштувань сервера, зокрема пріоритетів та server push. Але навіть у таких умовах HTTP/2 показує найкращий результат.

**Висновок**

Таким чином, протокол HTTP/2, найбільше оновлення протоколу з 1999 року, пристосовуючись до сучасних реалій, зберіг сумісність з першою версією в основних моментах. Перероблено механізми функціонування, належну увагу приділено швидкодії й захищеності. На даний момент усі сучасні сервери та браузери підтримують протокол HTTP/2, тому його використання є доцільним, оскільки при належному налаштуванні серверу виграш у швидкості завантаження та безпека підвищуються в декілька разів. Але слід зазначити, що на даний час ще не всі передові можливості нової версії використовуються належно, через бажання зберегти сумісність і можливість проксінгу трафіку між HTTP/2 та HTTP1.1, HTTP/2 має в своєму складі багато застарілого, що ускладнює подальший його розвиток і нововведення.

*Використані джерела*

1. *HTTP Overtakes P2P Traffic // DSL report – Режим доступу http://www.dslreports.com/shownews/85022 – 19.01.2007.*
2. *Usage of HTTP/2 for websites // W3Tech – Режим доступу https://w3techs.com/technologies/details/ce-http2/all/all– 14.05.2016*
3. *HTTP/2 // Github.io – Режим доступу http://http2.github.io/ – 12.06.2015*
4. [*https://github.com/bagder/http2-explained/tree/master/en*](https://github.com/bagder/http2-explained/tree/master/en) *– Режим доступу*
5. *http2 explained // Github.io – Режим доступу* [*http://www.httpwatch.com/download/*](http://www.httpwatch.com/download/) *– 11.02.2016*
6. *HTTP/2 is the future of the Web, and it is here! // Akamai – Режим доступу* [*https://http2.akamai.com/demo*](https://http2.akamai.com/demo)
7. *CloudFlare HTTP/2 // CloudFlare – Режим доступу https://www.cloudflare.com/http2/*

*Овчаренко А.И., студент гр. 401-ТК,*

*Грицкая Т.С., старший преподаватель кафедры*

*Полтавский национальный технический университет*

*имени Юрия Кондратюка*

ИССЛЕДОВАНИЕ сетевОГО протоколА HTTP/2

*В статье рассмотрены нововведения в сетевом протоколе HTTP/2, исследованы преимущества в сравнении с предыдущей версией протокола, проанализирован прирост в скорости загрузки страницы при использовании нового протокола.*

**Ключевые слова:** сеть, протокол, HTTP, HTTP/2, SPDY, сжатие, шифрование.

*Ovcharenko A., 401-TK student,*

*Gritska T., senior lecturer*

*Poltava National Technical University*

RESEARCH OF network protocol HTTP / 2

*The article discusses innovations in network protocols HTTP/2, investigated the benefits in comparison with previous protocols, analyzes the increase in page load speed when using the new protocol.*

***Keywords:*** network, protocol, HTTP, HTTP/2, SPDY, compression, encryption.