Министерство науки и высшего образования РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное Государственное бюджетное образовательное   
учреждение высшего образования

«Поволжский государсТвенный технологический  
университет» (ФГБОУ ВО «ПГТУ»)

Кафедра Информационно-вычислительных систем

ОТЧЁТ

по практической работе № 3

«Анализ протокола HTTP»

по дисциплине «Технологии программирования и создание WEB приложений»

Вариант №4

Выполнил: студент группы ИСТ-38

Двоенко О.Ю.

Проверил: преп. каф. Информатики

Уржумов Д.В.

Йошкар-Ола

2025

**Цели работы:**

- Изучение основ работы протокола HTTP: понять основные принципы функционирования HTTP, его методы (GET, POST и др.) и структуру запросов и ответов.

- Овладение навыками работы с Wireshark: научиться использовать Wireshark для захвата и анализа сетевого трафика.

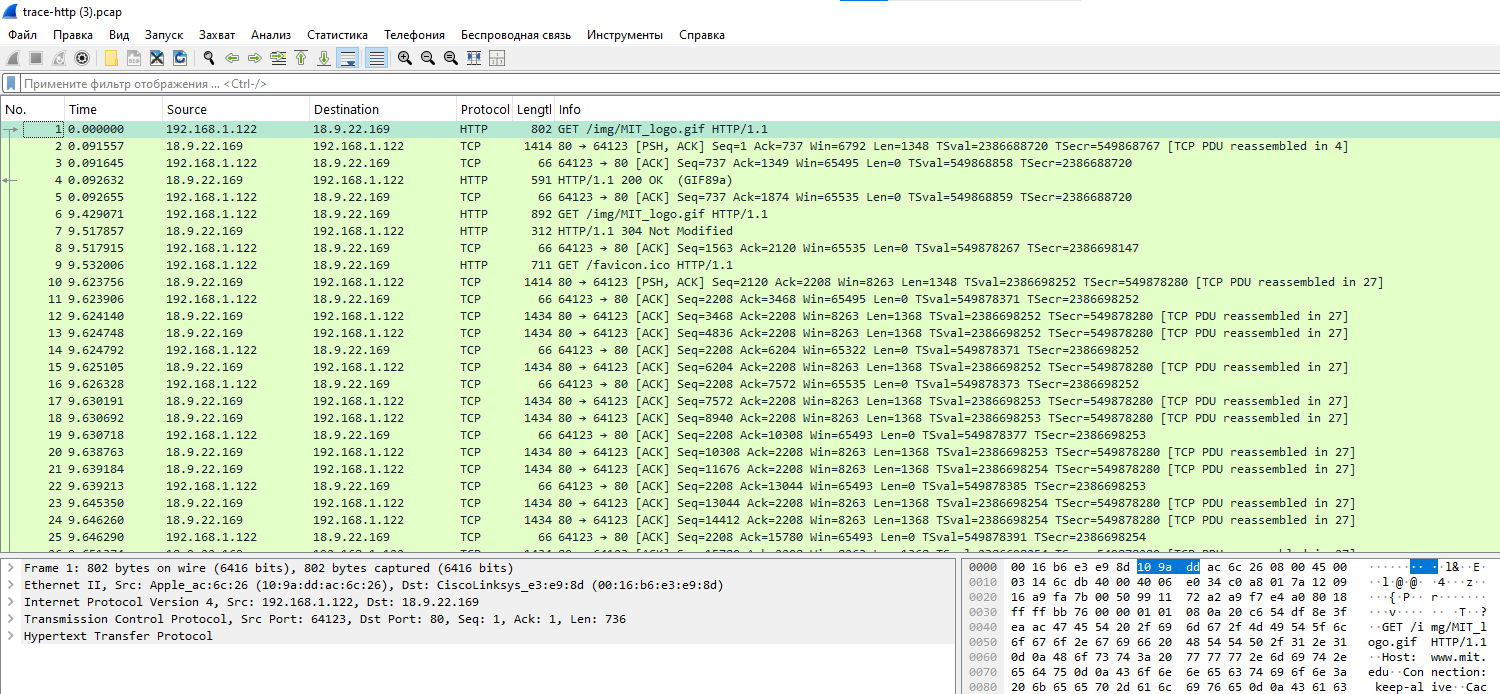
- Анализ HTTP-запросов и ответов: научиться идентифицировать и интерпретировать HTTP-запросы и ответы в захваченном трафике.

- Изучение кодов состояния HTTP: ознакомиться с различными кодами состояния ответов HTTP и их значениями.

- Исследование заголовков HTTP: понять роль заголовков в HTTP-запросах и ответах, изучить, как они влияют на поведение клиента и сервера.

**Задание 1. Загрузка файла трассировки http**

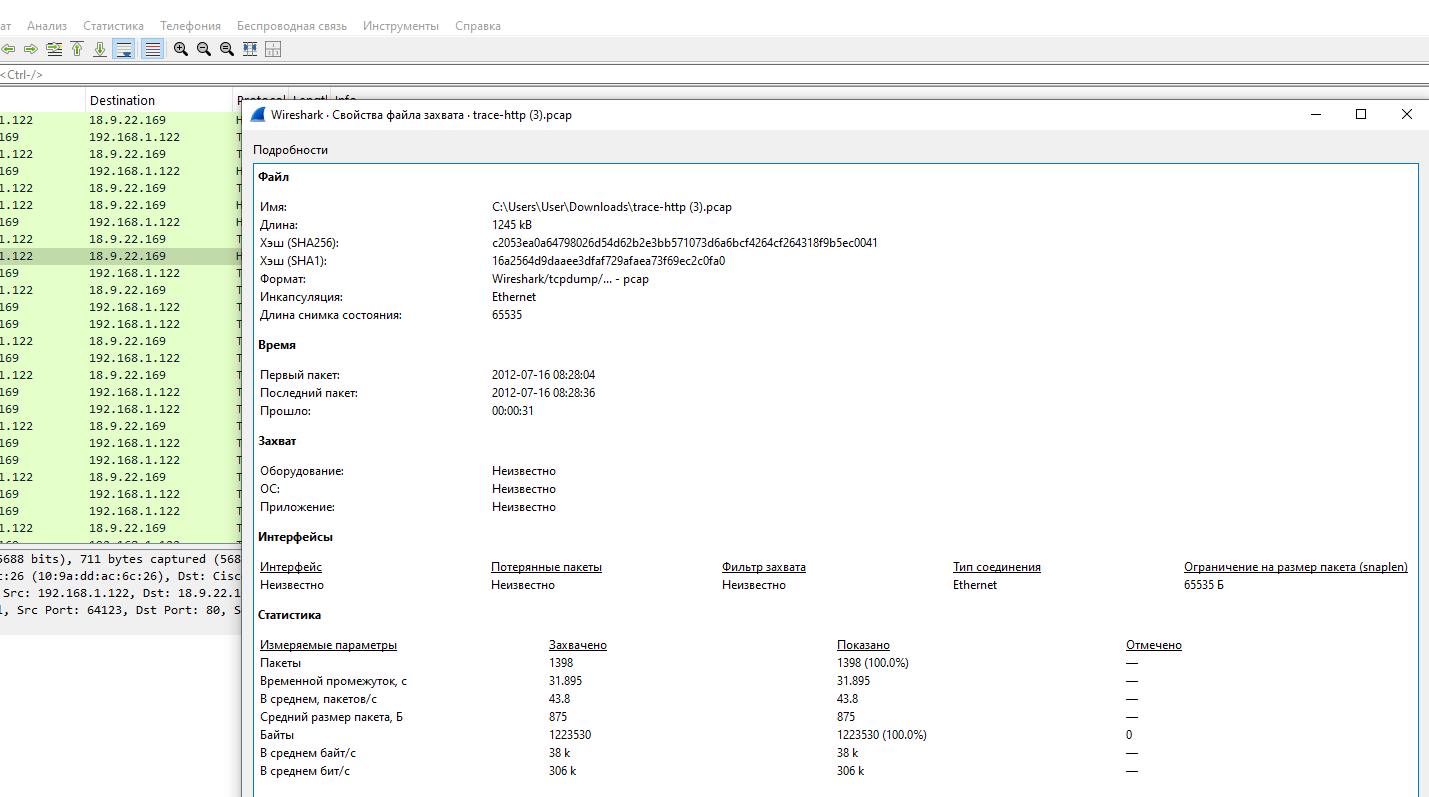
Поведение браузера может быть достаточно сложным, так как там используя больше функций HTTP, чем просто базовый обмен данными. Поэтому в этой работе мы будем использовать предохранённый файл трассировки, который можно скачать отсюда: trace-http.pcap. Откройте этот файл в Wireshark, должен появится экран, как показано ниже.



Используя доступный интерфейс отображения статистики, ответьте на следующие вопросы, подкрепив ответ скриншотом:

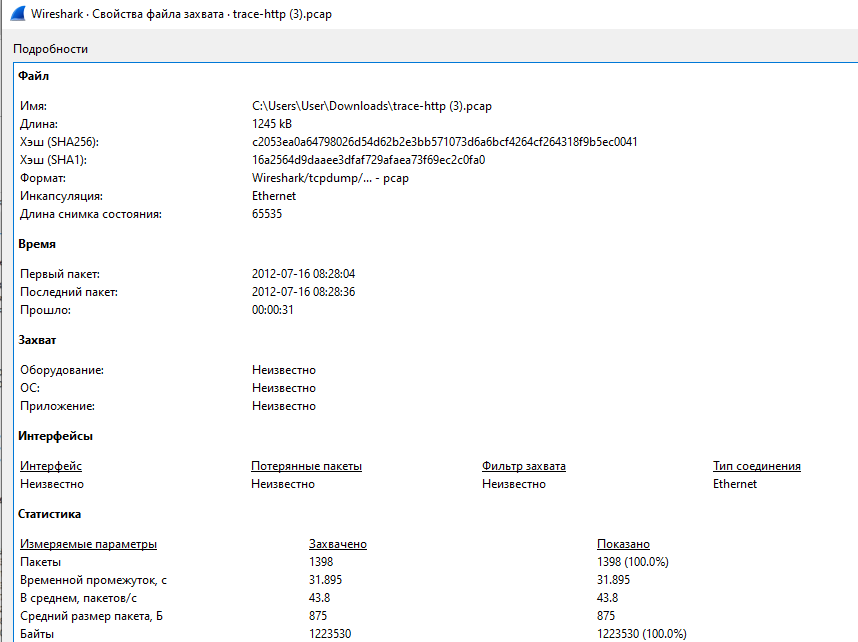
Количество пакетов, который были перехвачены Wireshark и присутствуют в этом файле?

1398 пакетов



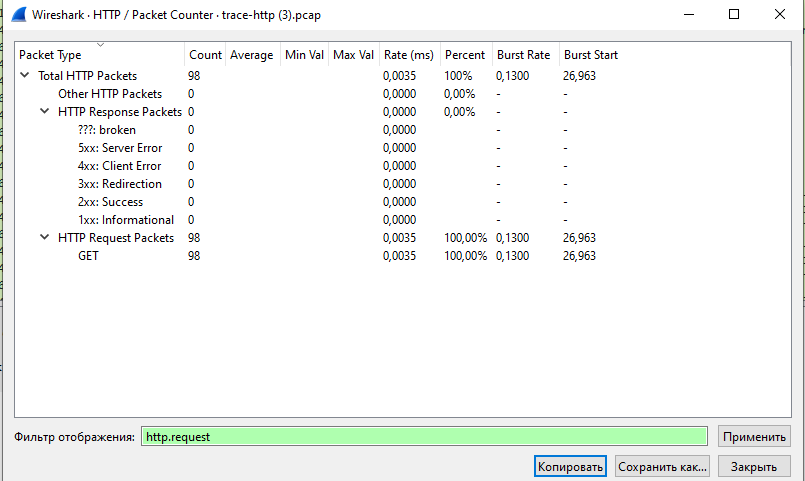
Какой общий объем перехваченных пакетов (в байтах)?

1223530 Байт



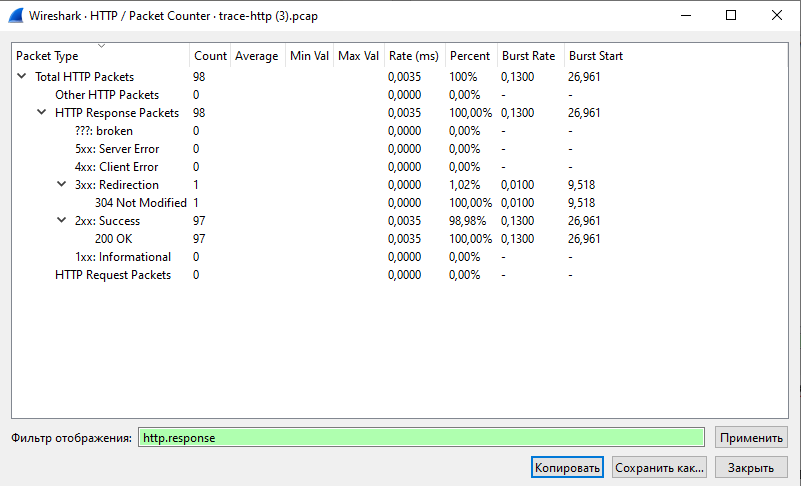
Количество пакетов HTTP Request?

98 пакетов



Количество пакетов HTTP Response?

98 пакетов

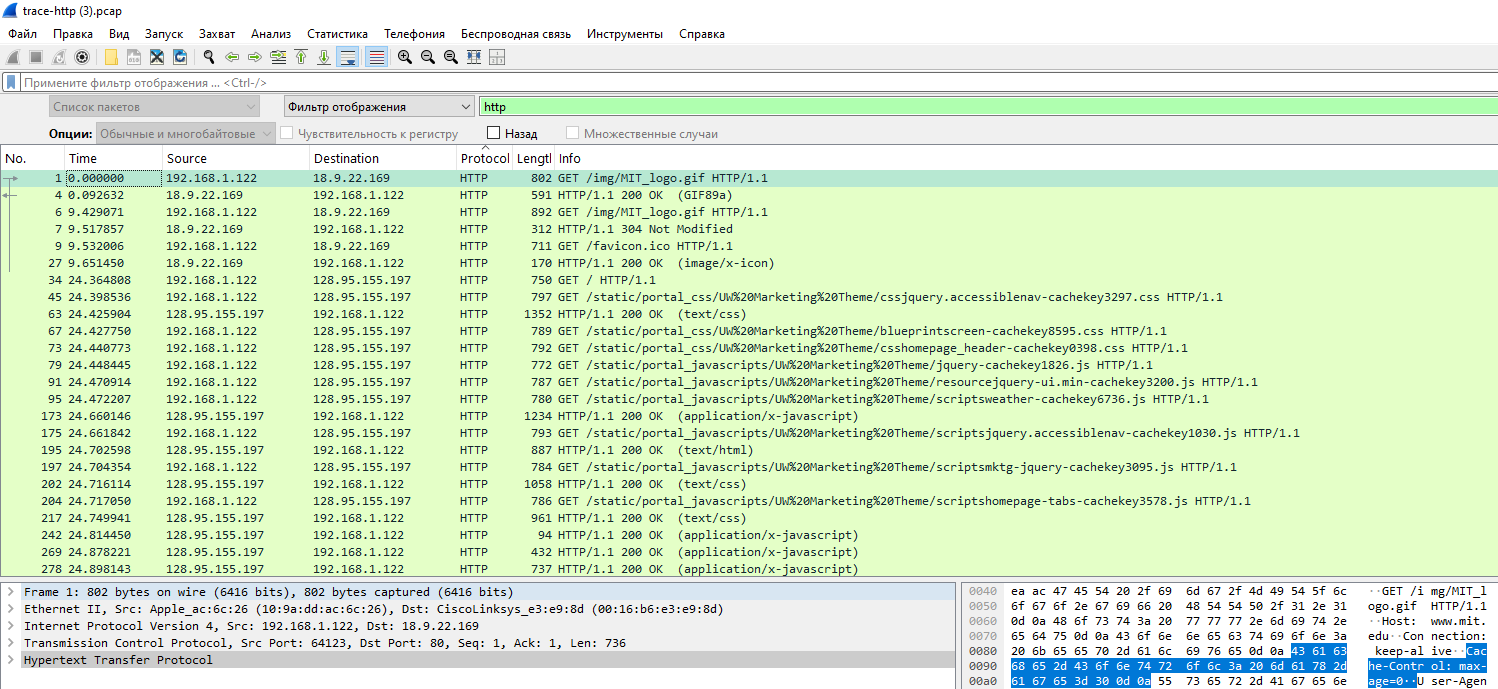


**Задание 2. Изучение трассировки запроса и ответа**

Применение фильтра: чтобы сосредоточиться на трафике HTTP, введем и применим выражение фильтра «http». Этот фильтр отображает запросы и ответы HTTP, но не отдельные пакеты, которые в них участвуют, а весь список.

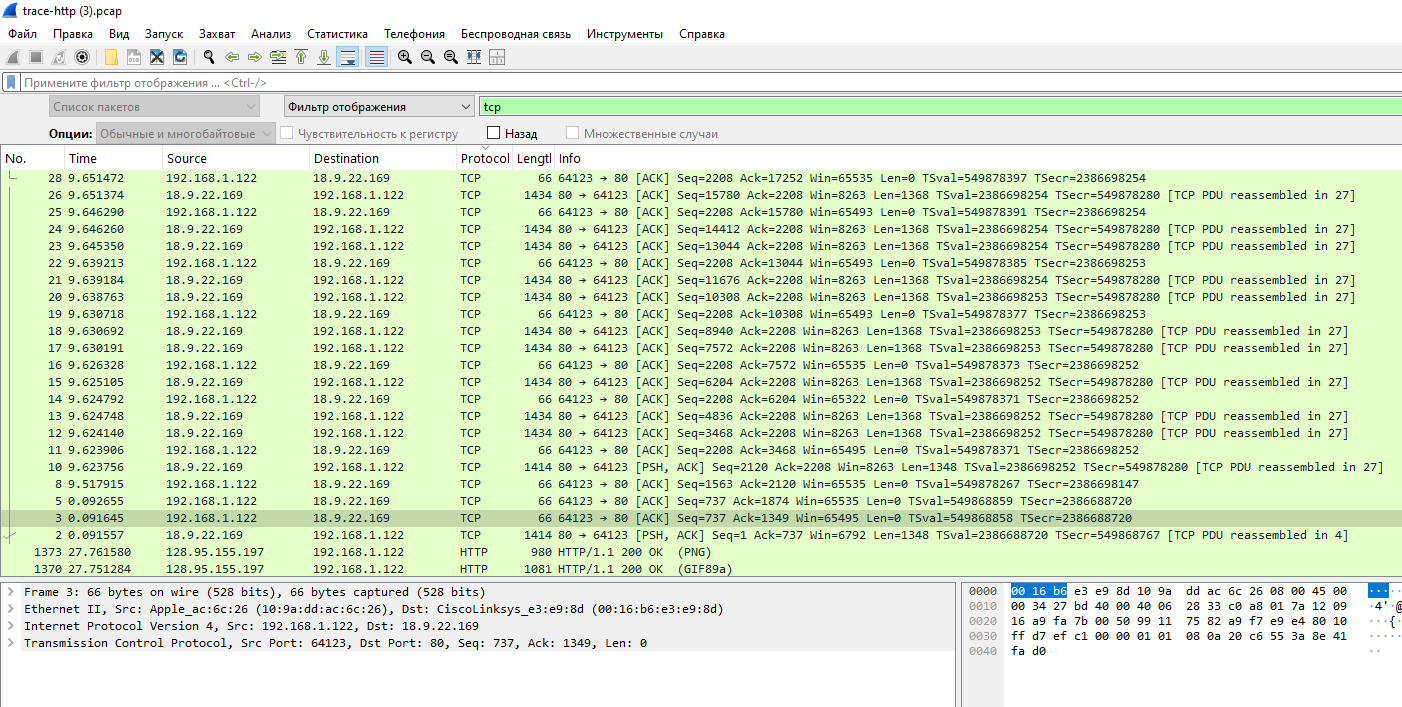
HTTP-ответ, несущий контент, обычно распределяется по нескольким пакетам. Когда приходит последний пакет в ответе, Wireshark собирает полный ответ и помечает пакет протоколом HTTP. Более ранние пакеты — это просто TCP-сегменты, несущие данные; последний пакет, помеченный HTTP, включает список всех более ранних пакетов, использованных для создания ответа. Аналогичный процесс происходит для запроса, но в этом случае запрос обычно помещается в один пакет.

С помощью выражения фильтра «http» мы скроем промежуточные TCP-пакеты и увидим только HTTP-запросы и ответы.

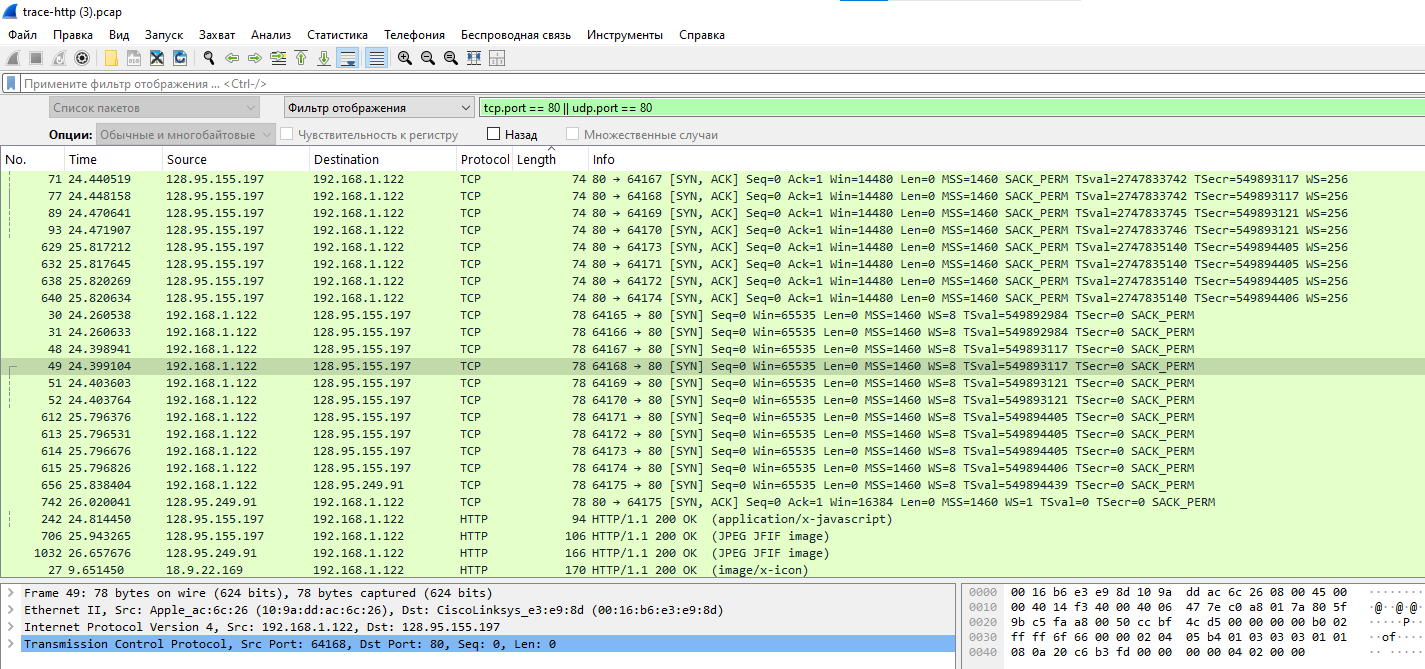


Также выполните фильтрацию и оформите результаты по протоколам:

- TCP

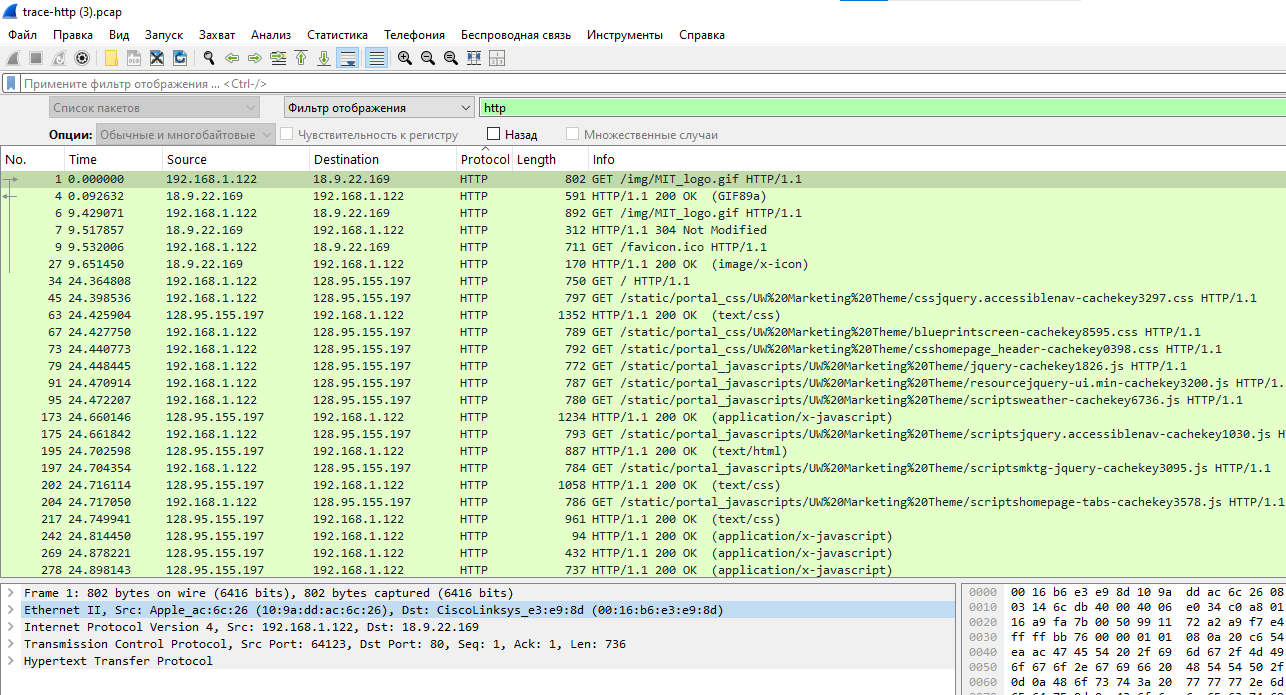


- TCP или UDP с использованием порта 80

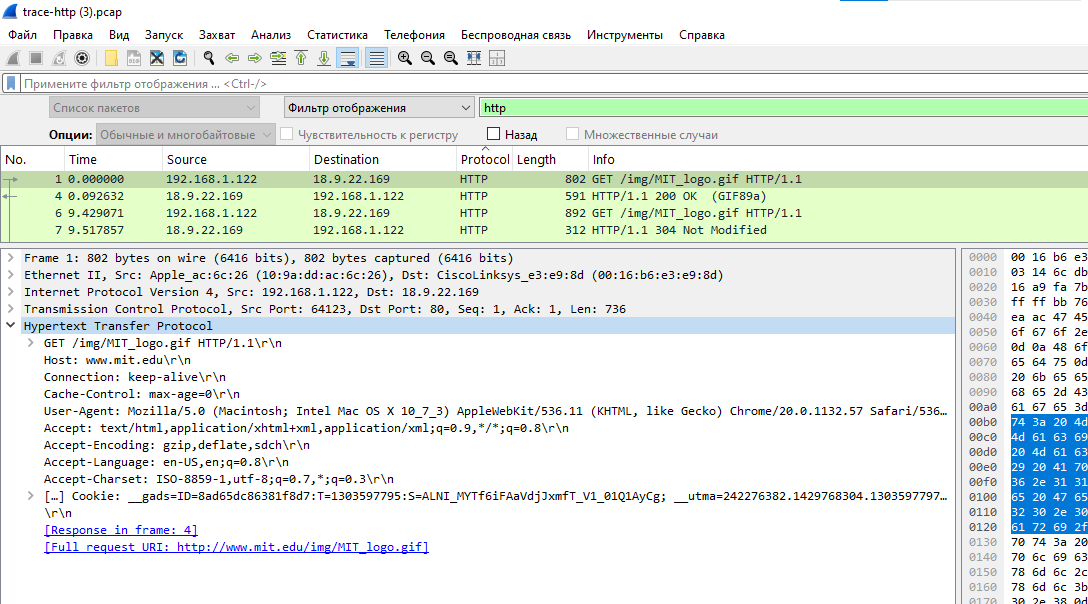


Просмотр GET запроса и изучение HTTP-заголовка.

Выберите первый GET в трассировке и разверните его HTTP-блок (это будет 802 GET /img/MIT\_logo.gif HTTP/1.1).



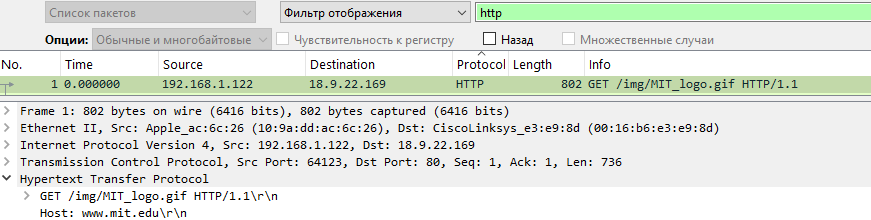
Это позволит нам изучить детали HTTP-запроса. Изучите HTTP-заголовок, развернув стрелку вниз рядом с Протокол передачи гипертекста (Hypertext Transfer Protocol) в нижней панели.



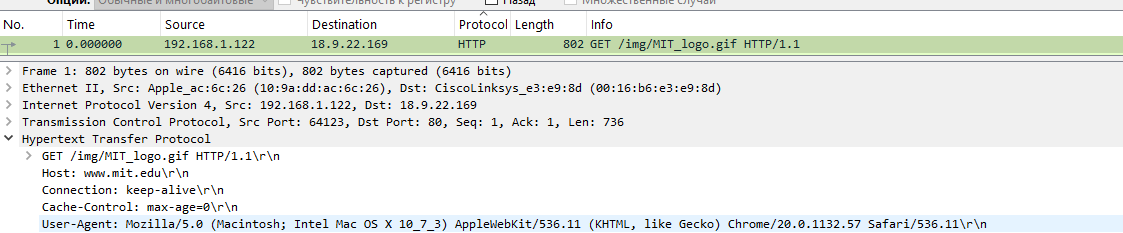
Изучите заголовки, которые отправляются вместе с запросом. Сначала мы увидим метод GET в начале запроса, включая такие сведения, как путь. Затем мы увидим ряд заголовков в виде помеченных параметров. Заголовков может быть много, и выбор заголовков и их значений варьируется от браузера к браузеру.

Посмотрите имеются ли в файле трассировки какие-либо из общих заголовков:

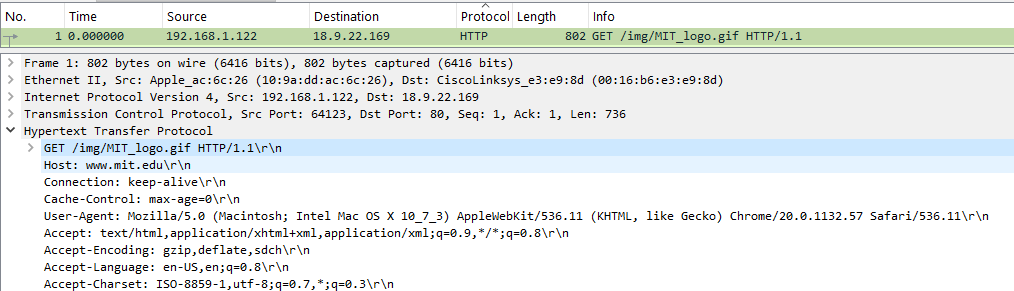
- Host. Обязательный заголовок, он идентифицирует имя (и порт) сервера.



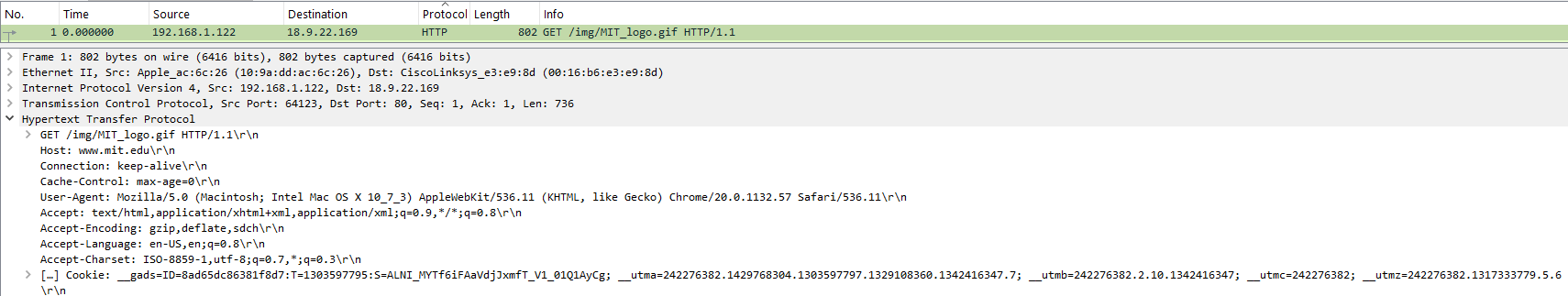
- User-Agent. Вид браузера и его возможности.



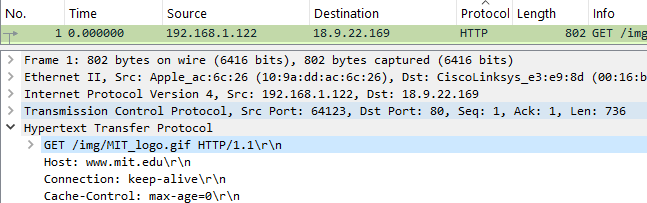
- Accept, Accept-Encoding, Accept-Charset, Accept-Language. Описания форматов, которые будут приняты в ответе, например, text/html, включая его кодировку, например, gzip, и язык.



- Cookie. Имя и значение cookie-файлов, которые браузер хранит для веб-сайта.

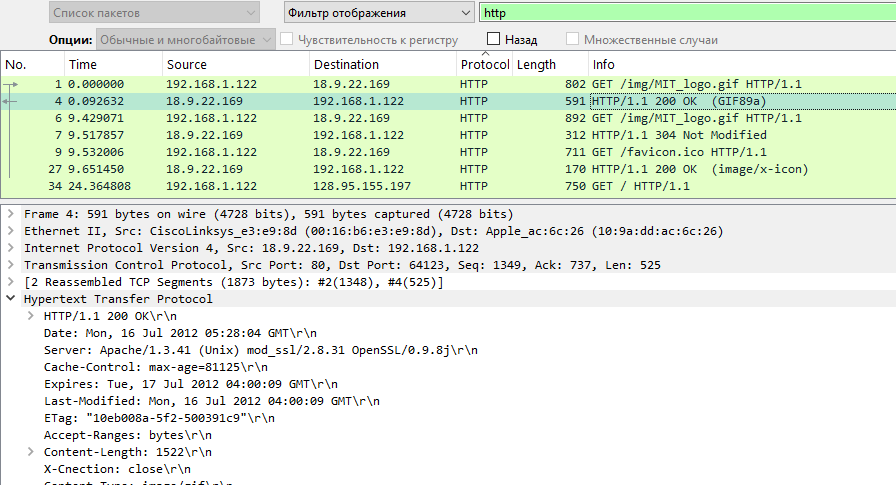


- Cache-Control. Информация о том, как ответ может быть кэширован.



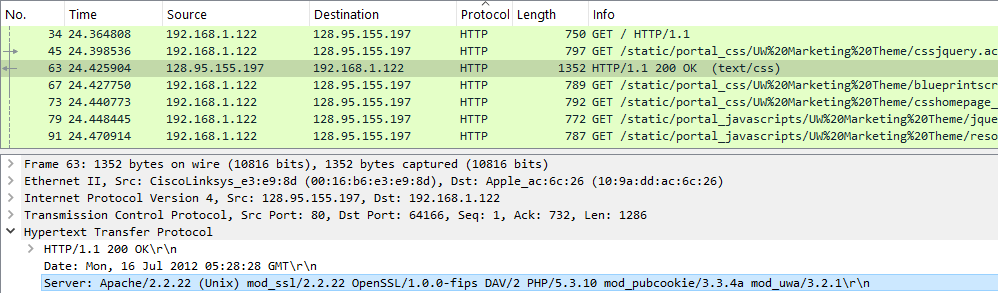
Просмотр GET ответа

Выберите ответ, который соответствует первому GET в трассировке, и разверните его HTTP-блок (см. ниже). Информация для этого пакета будет указывать «200 OK» в случае нормальной, успешной передачи.



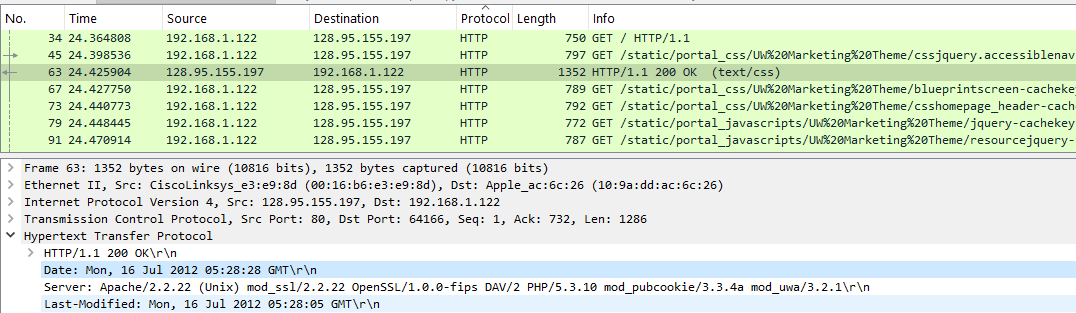
Изучите общие заголовки, такие как:

- Server. Тип сервера и его возможности.



**Server** - содержится информация о сервере, который обслуживает запрос. Помогает определить программное обеспечение сервера, что важно для диагностики и анализа уязвимостей.

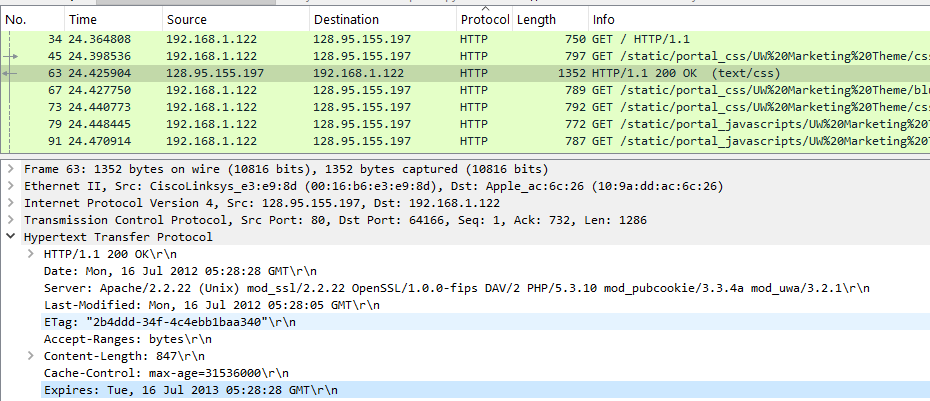
- Date, Last-Modified. Время ответа и время последнего изменения контента.



Date - время отправки ответа сервером. Используется для синхронизации и кэширования. Фиксирует время ответа, обеспечивает синхронизацию.

Last-Modified - Время последнего изменения содержимого ресурса. Бывает полезно для определения необходимости обновления кэша. Позволяет эффективно управлять кэшированием: браузеру и прокси-серверам знать, когда контент изменился.

- Cache-Control, Expires, Etag. Информация о том, как можно кэшировать ответ.



Cache-Control - указывает, как кэшировать ответ, в данном случае — 8760 часов. Управляет сроками кэширования, уменьшая нагрузку на сервер и ускоряя загрузку страниц.

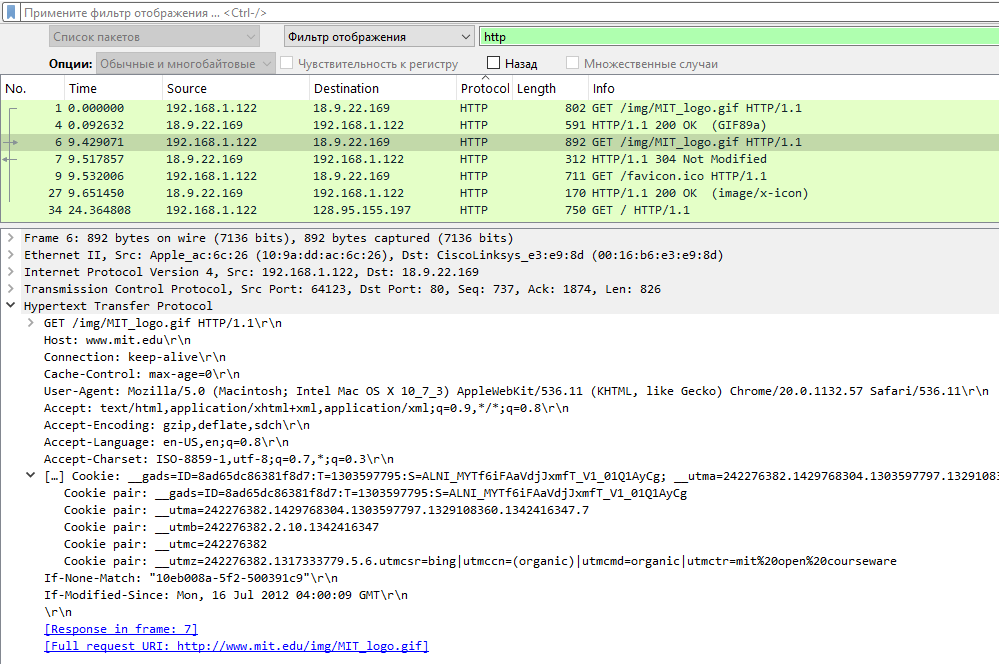
Expires - время истечения срока действия кэша, после него контент считается устаревшим. Управляет сроками кэширования, уменьшая нагрузку на сервер и ускоряя загрузку страниц.

Etag - уникальный идентификатор версии ресурса, используется для более точного определения изменений. Позволяет эффективно управлять кэшированием: браузеру и прокси-серверам знать, когда контент изменился.

**Задание 3. Кэширование контента**

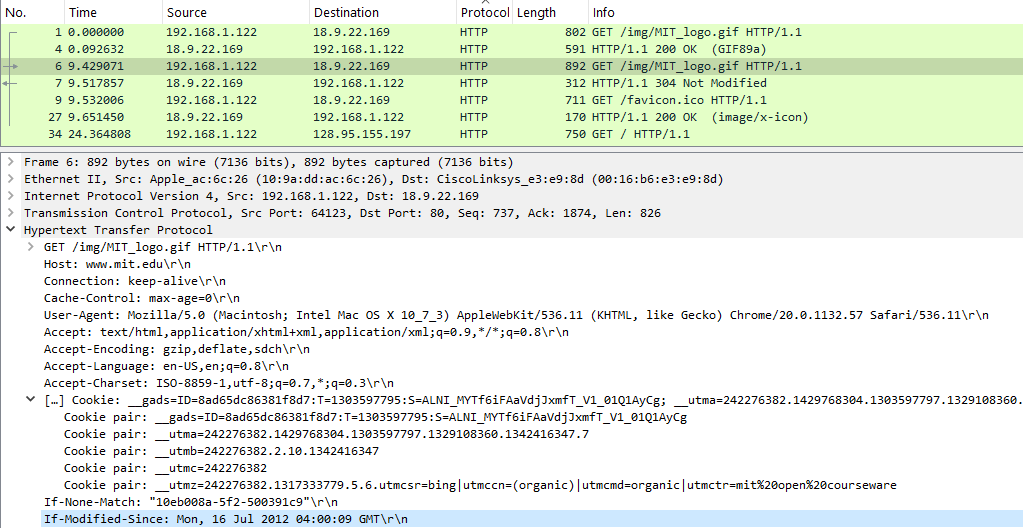
Второй блок записей в трассировке — это повторная выборка по первому URL. Этот блок дает нам возможность рассмотреть кэширование в действии, поскольку весьма вероятно, что изображение или документ не изменились и, следовательно, не нуждаются в повторной загрузке. Механизмы кэширования HTTP должны предоставить такую возможность.

Щелкните на третью строку в файле трассировки, т.е. на 892 GET /img/MIT\_logo.gif HTTP 1.1. Разверните HTTP-блок в нижней панели.

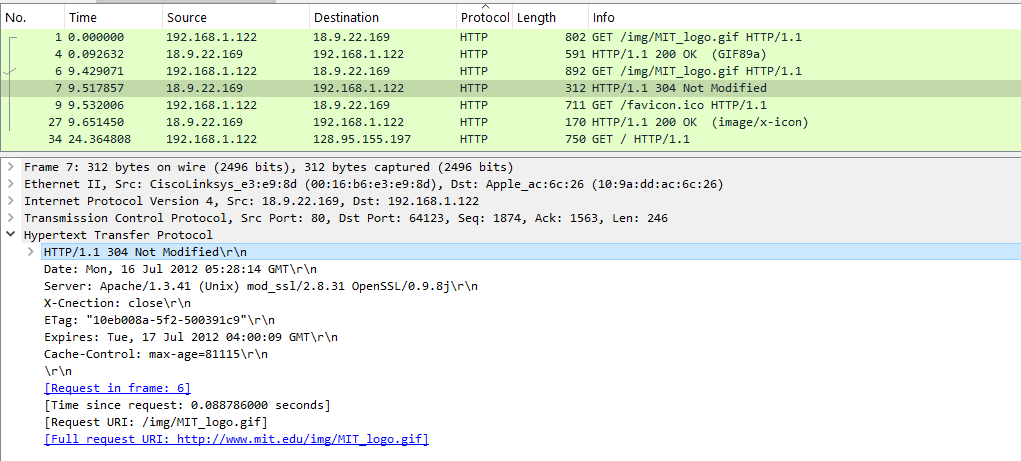


Найдите заголовок, который позволит серверу решить, нужно ли отправлять свежий контент. Серверу нужно будет отправлять свежий контент только в том случае, если контент изменился с момента последней загрузки браузером. Чтобы сделать это, браузер включает временную метку, взятую из предыдущей загрузки для контента, который он кэшировал. Этот заголовок отсутствовал в первом GET, поскольку кэш браузера был очищен, поэтому у браузера не было предыдущей загрузки контента, который он мог бы использовать. В большинстве других отношений этот запрос будет таким же, как и первый запрос.

Заголовок называется «If-Modified-Since», т.е. он просит сервер отправить контент, если он был изменен с определенного времени. Теперь выберем ответ на повторную выборку и развернем его HTTP-блок.



Мы видим, что кэширование сработало, как и ожидалось, этот ответ не будет содержать контент. Вместо этого код статуса ответа — «304 Not Modified». Это сообщает браузеру, что контент не изменился по сравнению с предыдущей копией, и кэшированный контент затем может быть отображен.



**«HTTP/1.1 304 Not Modified»** — это код состояния HTTP, который означает **«не изменялось»**. **Цель использования кода**— оптимизация загрузки страниц, так как браузер использует сохранённую версию ресурса вместо того, чтобы загружать его снова.

Date – фиксирует время отправки ответа сервером, обеспечивает синхронизацию. Для данной строки трассировки – Понедельник, 16 июля 2012 года, точное время - 05:28:14.

**Server** - содержится информация о сервере, который обслуживает запрос. Для данной строки трассировки – Apache/1.3.41 (Unix) mod\_ssl/2.8.31 OpenSSL/0.9.8j.

X-Cnection – пользовательский заголовок, используемый для идентификации и отслеживания соединений в процессе диагностики сети. Для данной строки трассировки – завершено текущее соединение.

Etag - уникальный идентификатор версии ресурса, используется для более точного определения изменений. Для данной строки трассировки – 10eb008a-5f2-500391c9.

Expires - время истечения срока действия кэша, после него контент считается устаревшим. Для данной строки трассировки – Вторник, 17 Июля 2012 года, точное время – 04:00:09.

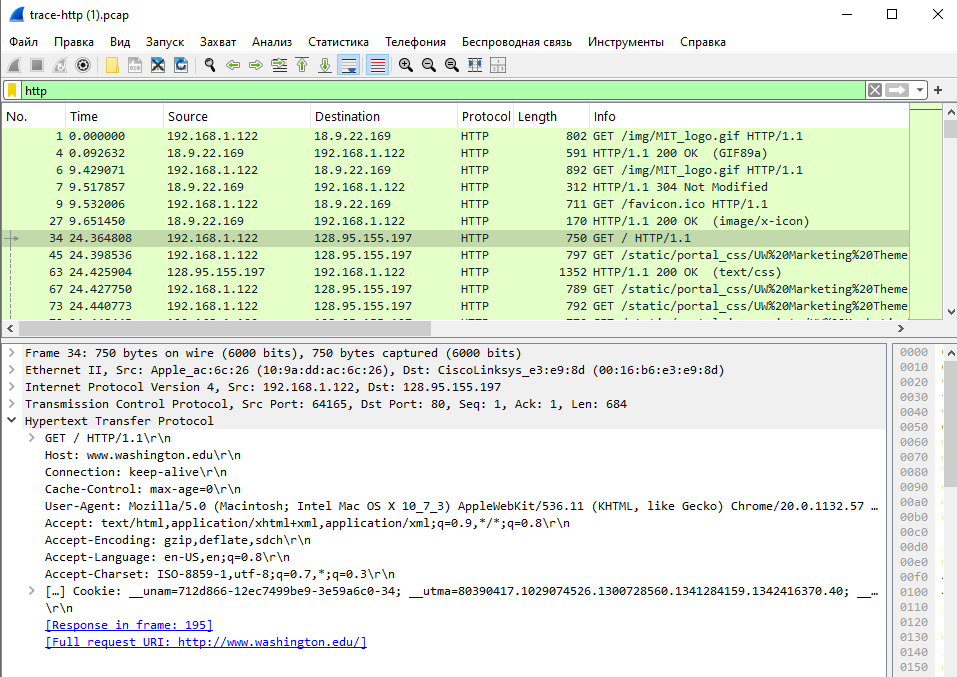
Cache-Control – управляет кешированием ресурсов, позволяя серверу и клиенту договариваться о том, как долго можно хранить ресурс в кеше. Для данной строки трассировки – max-age=81115, это указывает на время в секундах, в течение которого ресурс считается свежим после получения. 81115 секунд : 3600 = примерно 22,5 часа. В течение этого времени сервер не требуется повторно запрашивать или обновлять ресурс.

Значение временной метки берется из заголовка «Last-Modified» самой последней загрузки контента. Это временная метка сервера для последнего изменения контента, а не временная метка браузера или временная метка времени загрузки.

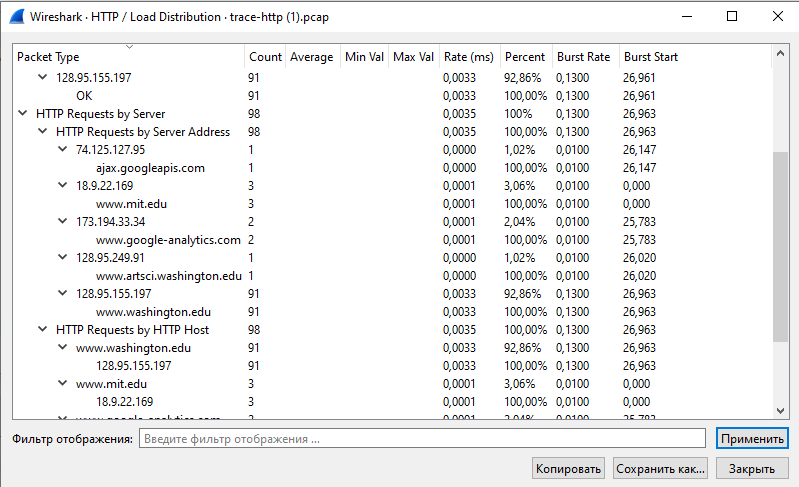
Добавьте в отчет описание заголовков для данной строки трассировки (строка 7) и добавьте скрин загруженного Gif изображения по которому обрабатывался запрос.

**Задание 4. Сложные страницы**

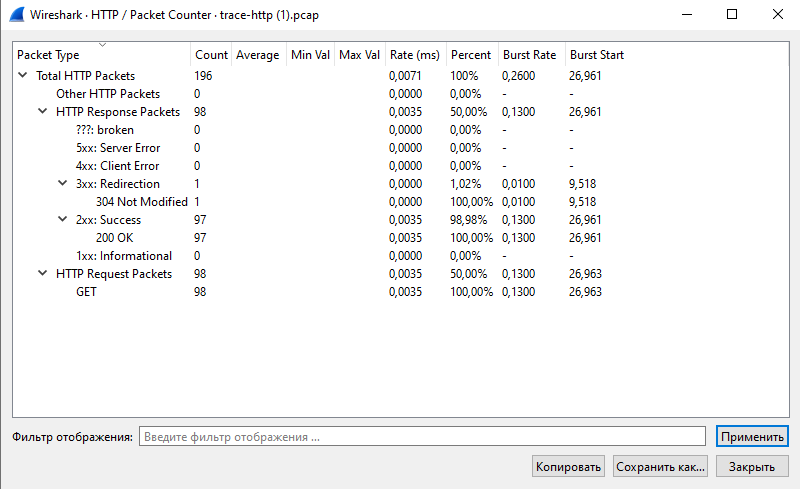
Рассмотрим четвертую выборку в трассировке. Эта выборка была для более сложной веб-страницы, которая, скорее всего, будет иметь встроенные ресурсы. Поэтому браузер загрузит исходный HTML плюс все встроенные ресурсы, необходимые для отображения страницы, плюс другие ресурсы, которые запрашиваются во время выполнения сценариев страницы. В этой выборке мы увидим, что одна страница может включать много GET. Щелкните по номеру 34, который является 750 GET / HTTP/1.1.



Обобщите GET для этой страницы, вызвав панель HTTP Load Distribution. Эту панель можно найти в разделах Статистика и HTTP. На этой панели показана информация, сколько запросов было сделано на какие серверы. При серфинге есть вероятность, что fetch запросит контент с других серверов, о которых, мы могли и не подозревать. Эти другие серверы могут включать третьи стороны, такие как сети распространения контента, рекламные и аналитические сети. При извлечении страницы было сделано 98 запросов на 5 разных серверов.

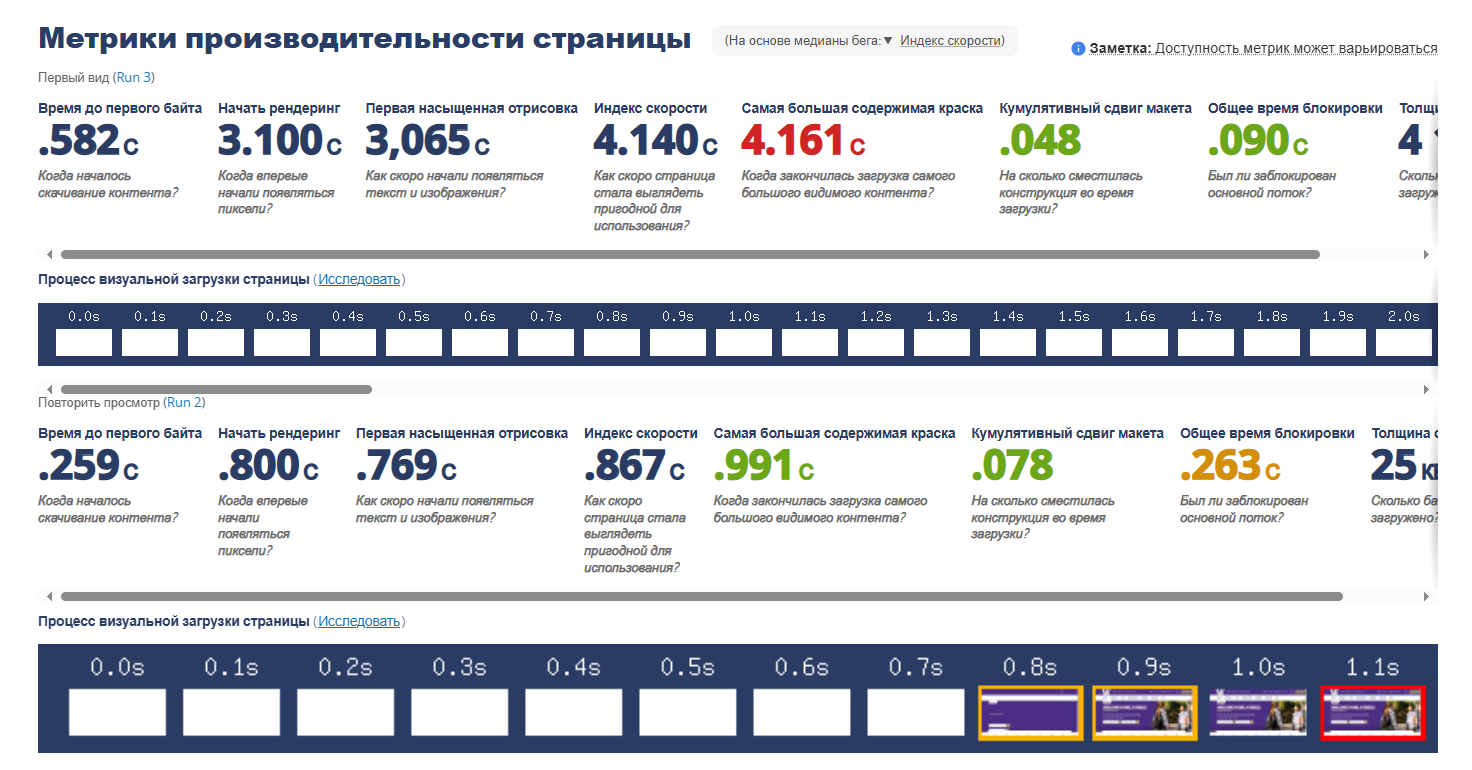


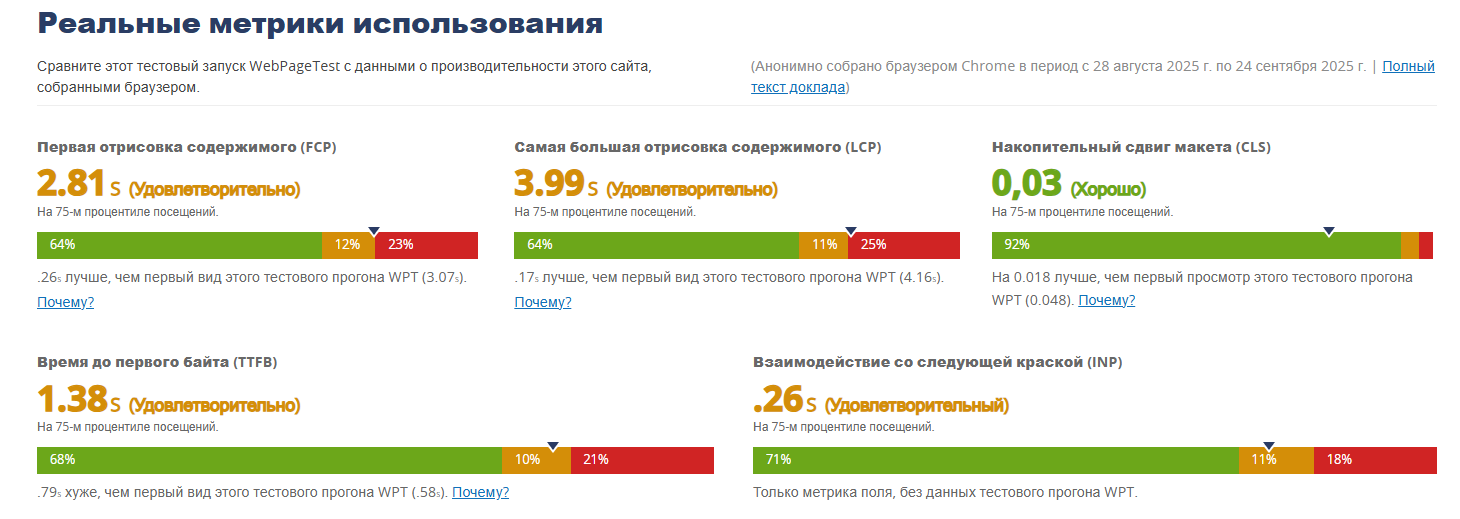
Для получения другого вида сводки GET-запросов откройте панель счетчика HTTP-пакетов. На панели отображаются типы запросов и ответов. Она состоит из запросов GET, которые соответствуют ответам 200 OK и 304 Not Modified. Однако есть множество других кодов ответов, которые наблюдаются в файле трассировки, например, когда ресурс уже кэширован.

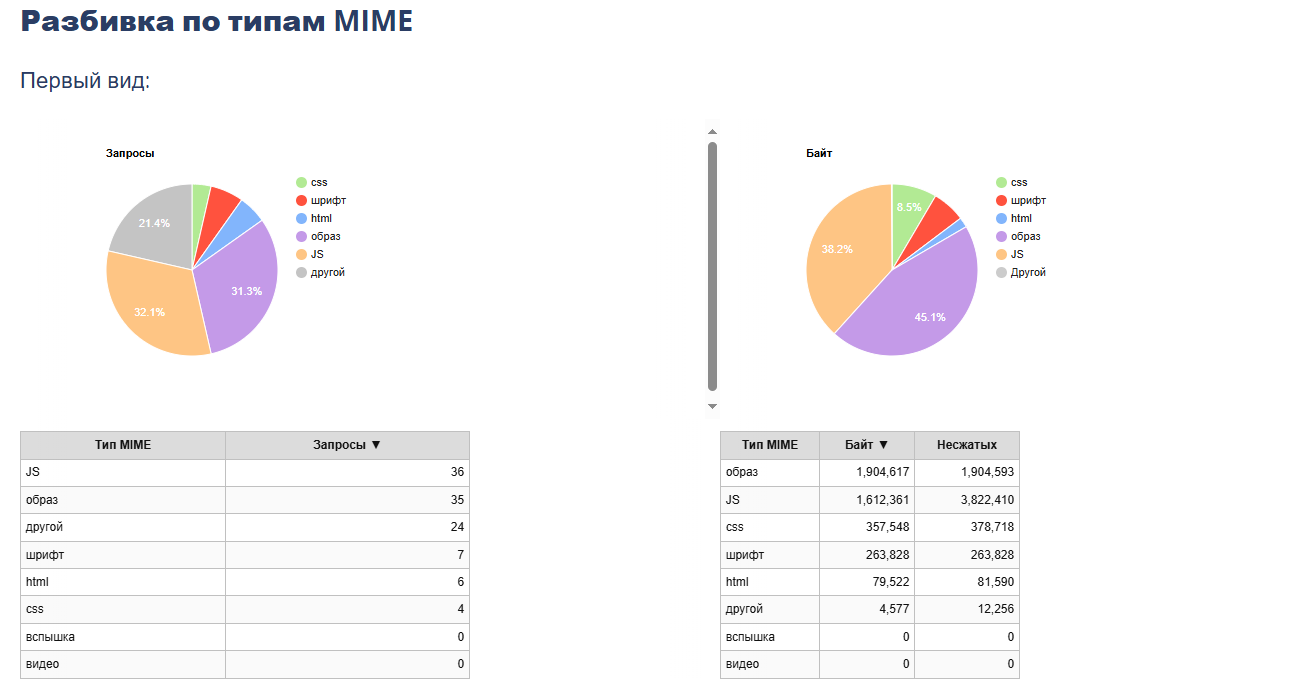


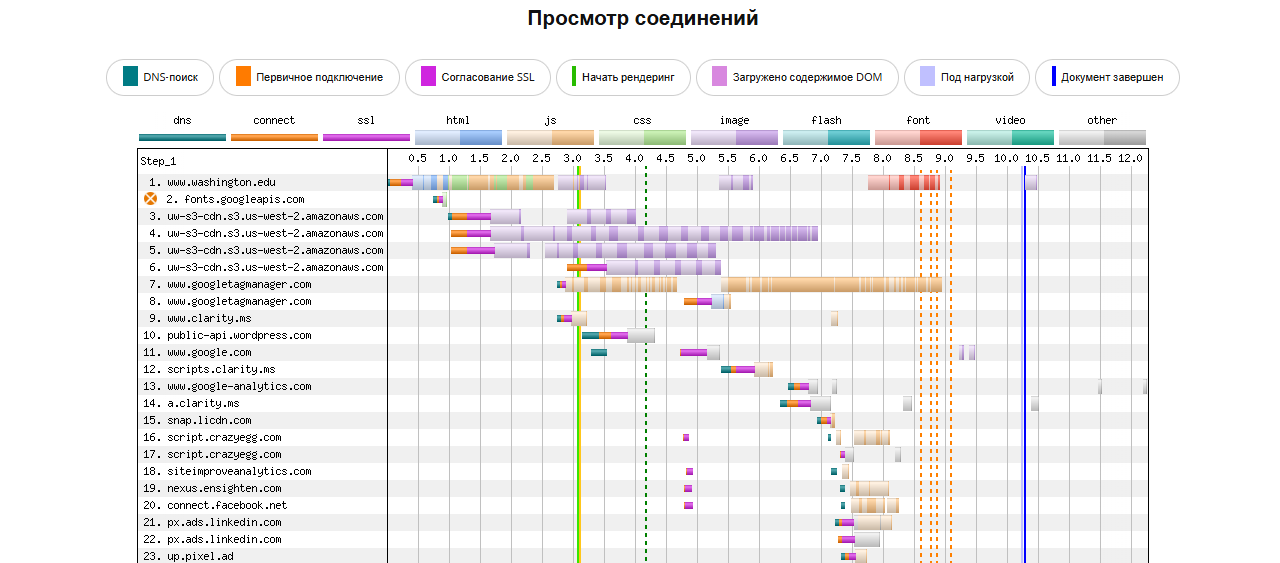
Для более подробного обзора общего процесса загрузки страницы используйте такой сайт, как PageSpeed ​​от Google или https://www.webpagetest.org/. Эти сайты проверят URL-адрес по выбору и сгенерируют отчет об активности загрузки страницы, сообщая, какие запросы были получены в какое время, и давая советы по сокращению общего времени загрузки страницы.

Начало «водопадной» диаграммы для загрузки страницы, соответствующей трассировке на www.ulster.ac.uk, показано на следующем рисунке. После извлечения исходного HTML-ресурса происходит множество последующих быстрых извлечений встроенных ресурсов, таких как скрипты JavaScript, таблицы стилей CSS, изображения и т. д.

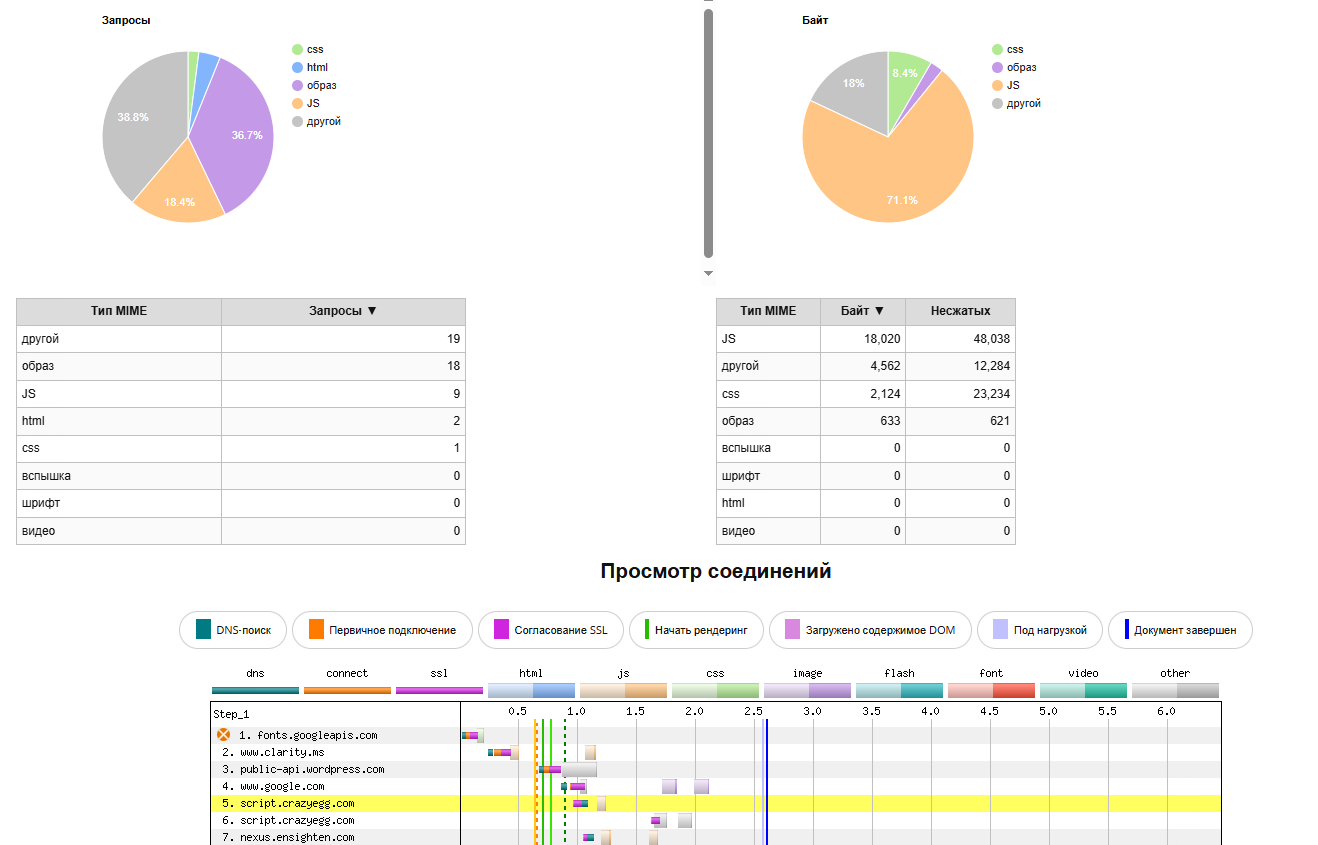






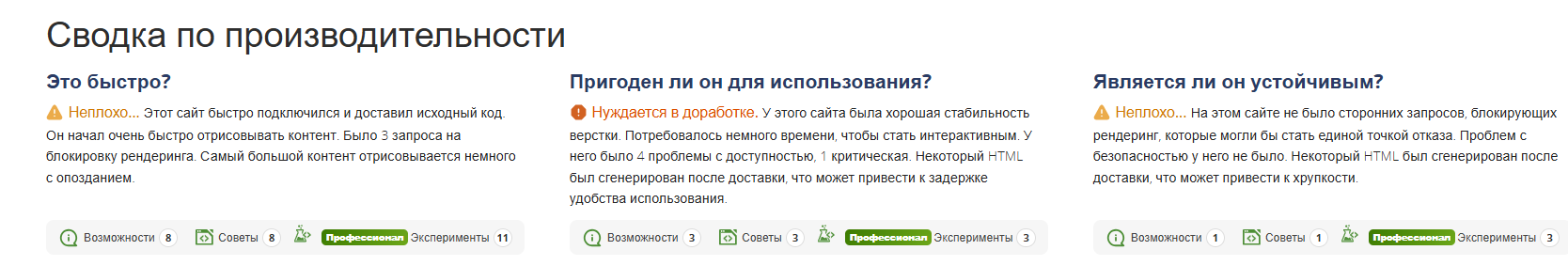


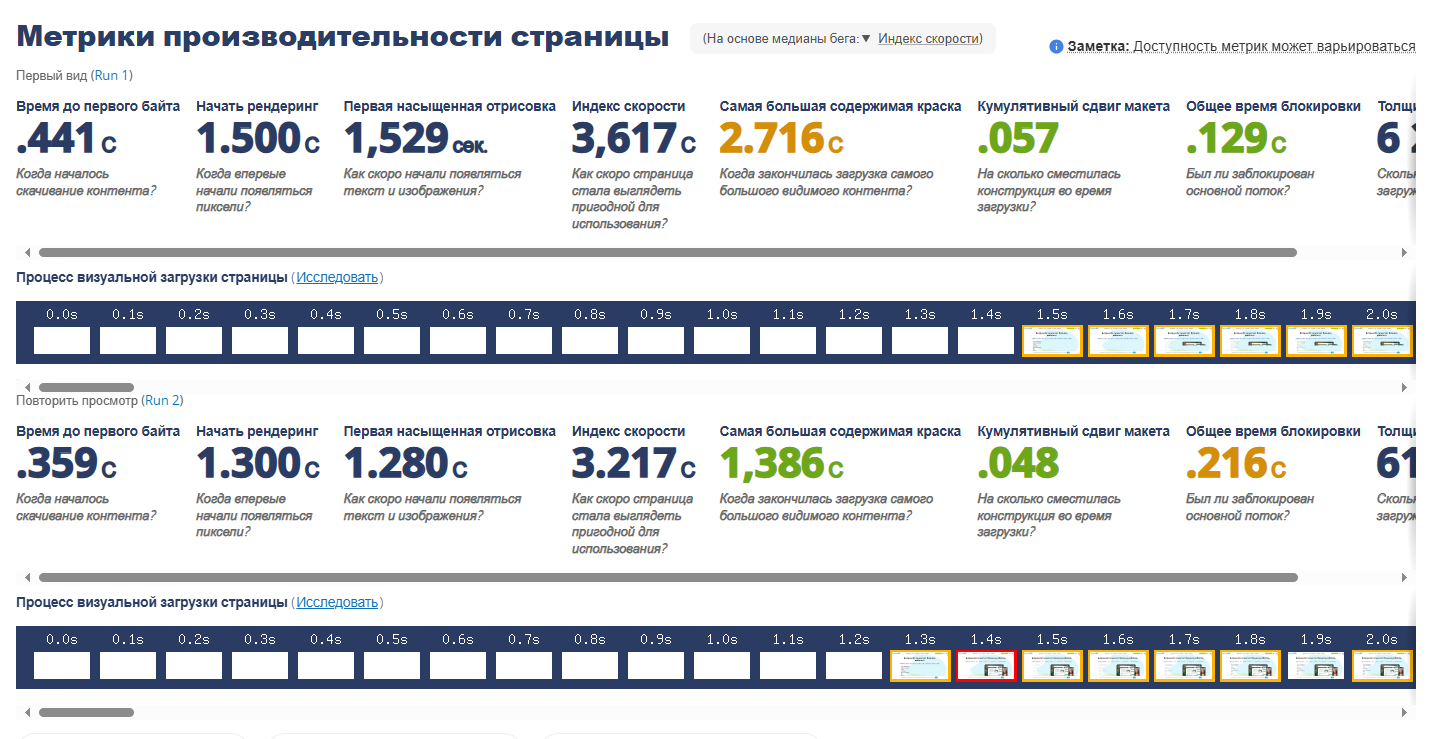
Повторный запрос:

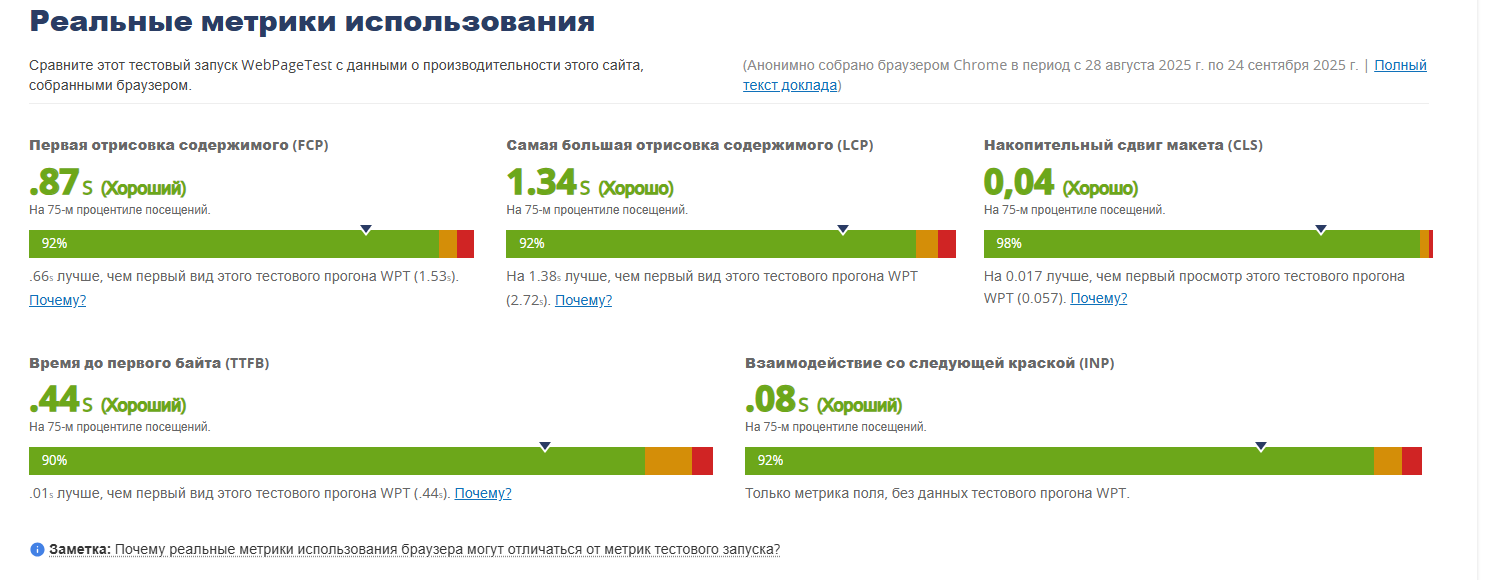


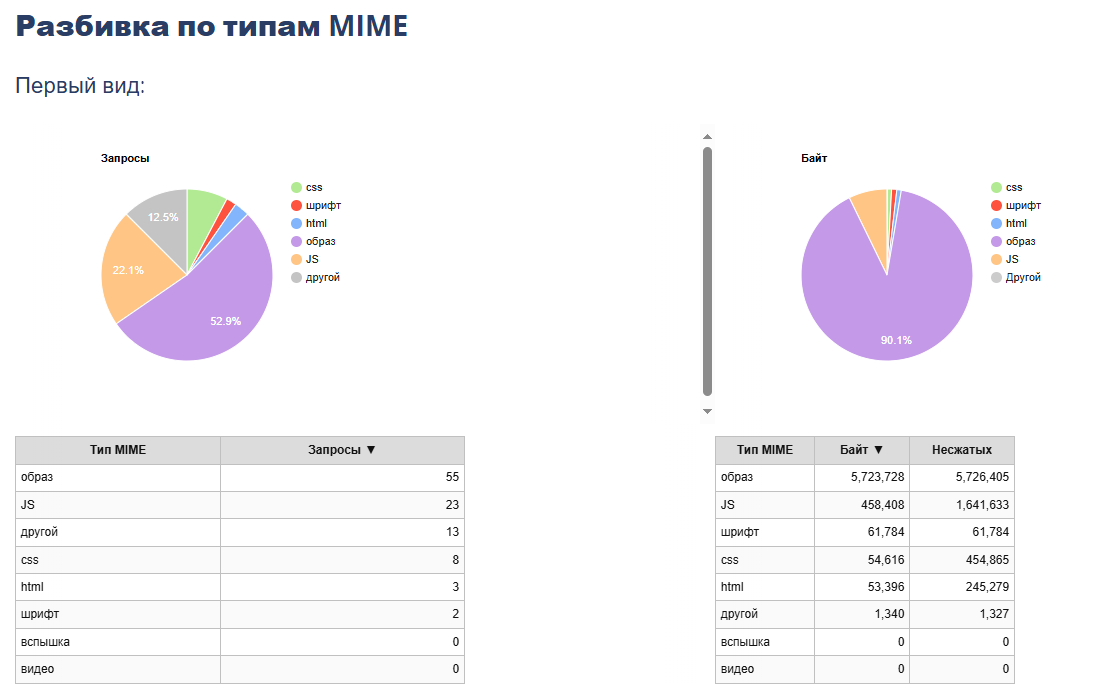
Попробуйте выполнить тестирование Интернет-ресурса, с которым мы работали при выполнении первой практической работы (сетевые утилиты) с помощью сервиса PageSpeed или https://www.webpagetest.org/. Сформируйте для отчета диаграммы демонстрирующие результаты тестирования. Объясните какую ценность имеет эта информация для веб-разработки.

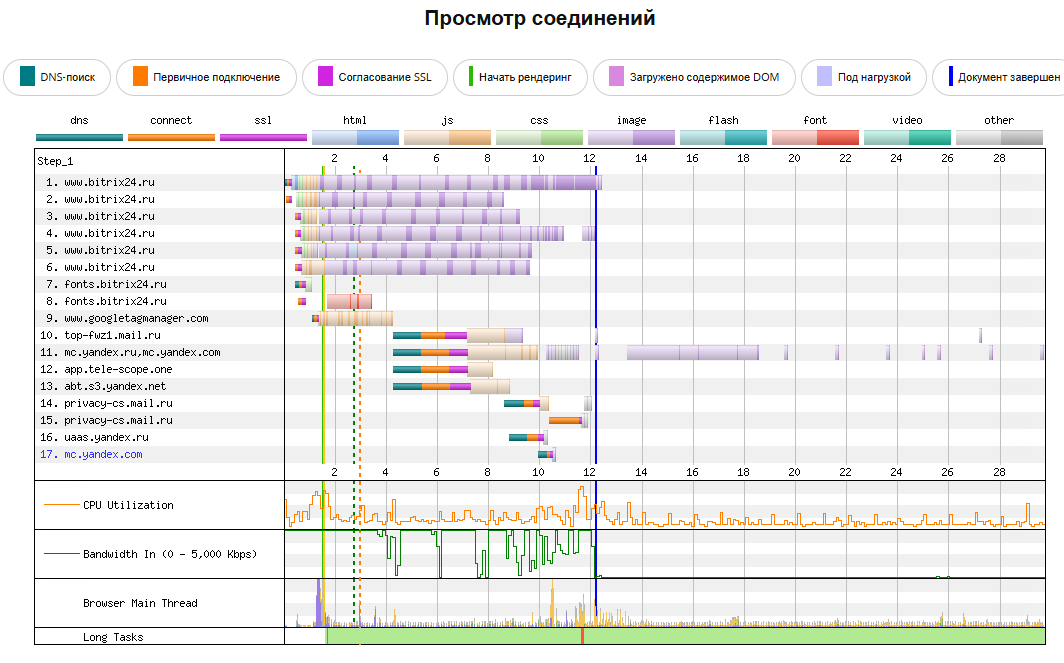
Результаты теста производительности веб-страницы Битрикс24:

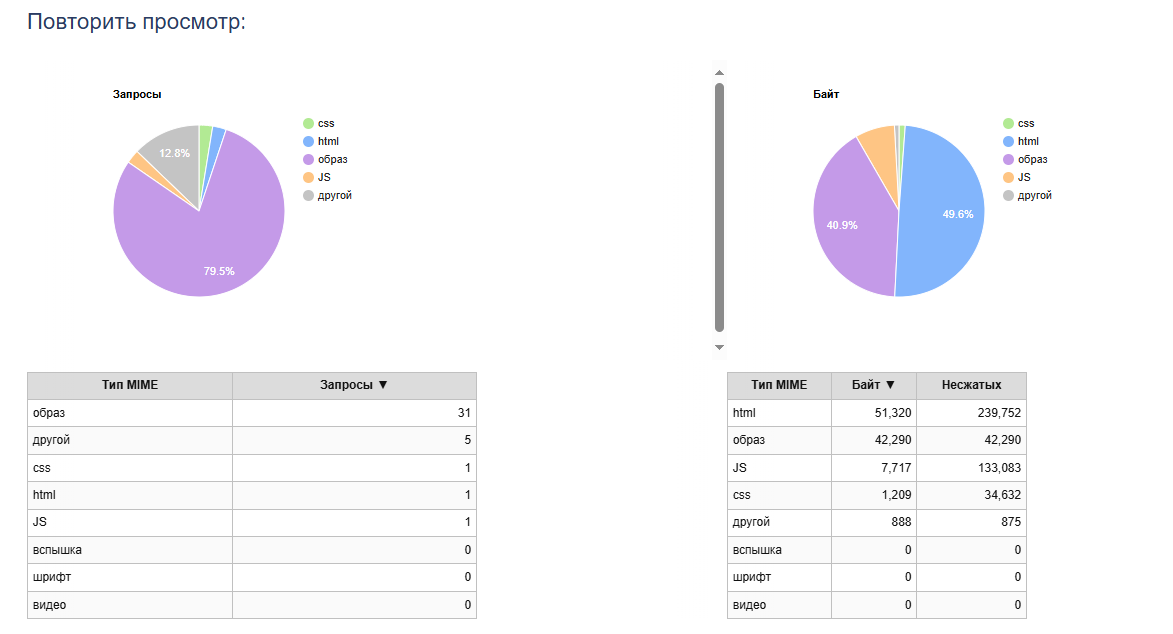


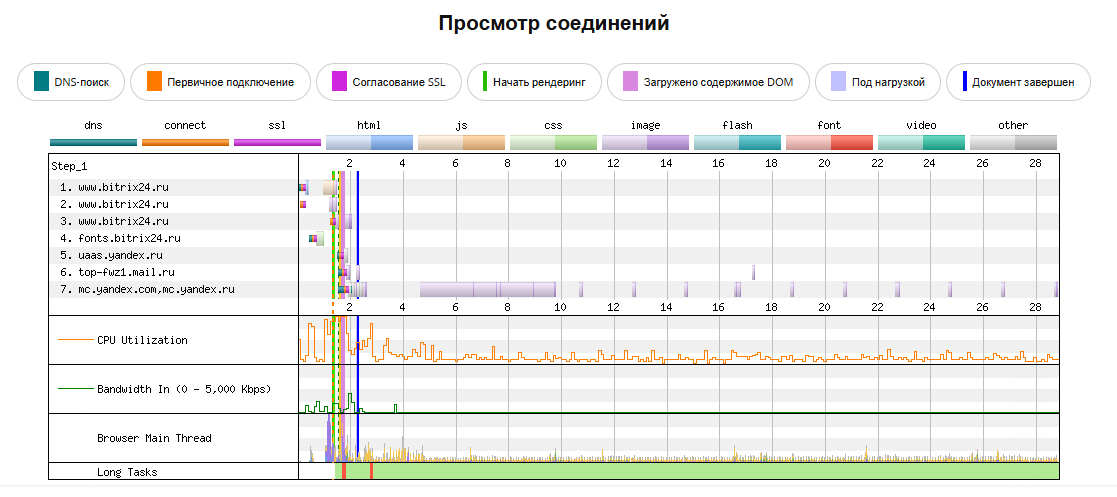












Данная информация имеет важное значение для веб-разработки по нескольким причинам:

1. Оценка производительности: тестирование интернет-ресурса Битрикс24 с помощью WebPageTest позволяет определить скорость загрузки страниц, время отклика сервера и другие показатели, влияющие на пользовательский опыт. Быстро загружающийся сайт повышает удовлетворенность пользователей и способствует лучшему индексу в поисковых системах.

2. Анализ проблемных аспектов: в результате тестирования можно выявить узкие места, такие как большие размеры изображений, неэффективный код, медленные серверные ответы или ошибки в настройке ресурса, что позволяет проводить целенаправленную оптимизацию.

3. Сравнение и мониторинг: регулярное проведение таких тестов позволяет отслеживать эффективность внесённых изменений, сравнивать показатели после оптимизаций и обеспечивать стабильную работу ресурса.

4. Визуализация данных: использование диаграмм и графиков в отчёте помогает лучше понять результаты тестирования, визуально выделить ключевые параметры и принять обоснованные решения по дальнейшим улучшениям.