**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский Авиационный Институт»**

**(Национальный Исследовательский Университет)**

**Институт: №8 «Информационные технологии   
и прикладная математика»   
Кафедра: 806 «Вычислительная математика   
и программирование»**

Лабораторная работа № 4   
по курсу «Криптография»

Группа: М8О-306Б-21

Студент(ка): О. А. Мезенин

Преподаватель: А. В. Борисов

Оценка:

Дата: 14.04.2024

Москва, 2024

**ОГЛАВЛЕНИЕ**

[1 **Тема**  3](#_Toc1)

[2 **Задание**  3](#_Toc2)

[3 **Теория** 4](#_Toc3)

[4 **Ход лабораторной работы** 5](#_Toc4)

[5 **Выводы** 13](#_Toc5)

[6 **Список используемой литературы** 14](#_Toc6)

# **Тема**

Аутентификация с асимметричными алгоритмами шифрования

# **Задание**

1. Выбрать не менее 2-ух web-серверов сети Интернет различной организационной и государственной принадлежности.

2. Запустить Wireshark и используя Firefox установить https соединение с выбранным сервером.

3. Провести анализ соединения.

4. Сохранить данные необходимы для последующего сравнительного анализа:

* Имя сервера, его характеристики.
* Версия TLS.
* Выбранные алгоритмы шифрования.
* Полученный сертификат: версия. Валидность сертификата, валидность ключа, удостоверяющий центр.
* Время установки соединения (от ClientHello до Finished)

5. Если список исследуемых серверов не исчерпан выбрать другой сервер и повторить соединение.

6. Если браузер поддерживал соединение TLS 1.2 принудительно изменить параметры TLS соединения в Firefox на TLS 1.0 (в браузере перейти по адресу “about:config” и изменить раздел SSL\TLS) и провести попытки соединения с выбранными серверами).

7. Провести сравнительный анализ полученной информации.

8. В качестве отчета представить результаты сравнительного анализа, выводы в отношении безопасности и корректности настройки веб-серверов с учетом их организационной и государственной принадлежности.

# **Теория**

Проблема протокола HTTP заключается в том, что данные передаются по сети в открытом виде, что делает возможным для злоумышленника прослушивать передаваемые пакеты и извлекать информацию из параметров, заголовков и тела сообщений. Для устранения этой уязвимости был разработан HTTPS, который представляет собой HTTP поверх SSL (а затем и TLS), позволяющий безопасный обмен данными. В отличие от HTTP, использующего стандартный TCP/IP порт 80, HTTPS использует порт 443.

Secure Sockets Layer (SSL) — это криптографический протокол, обеспечивающий безопасное взаимодействие между пользователем и сервером в небезопасной сети. Сегодня он считается устаревшим.

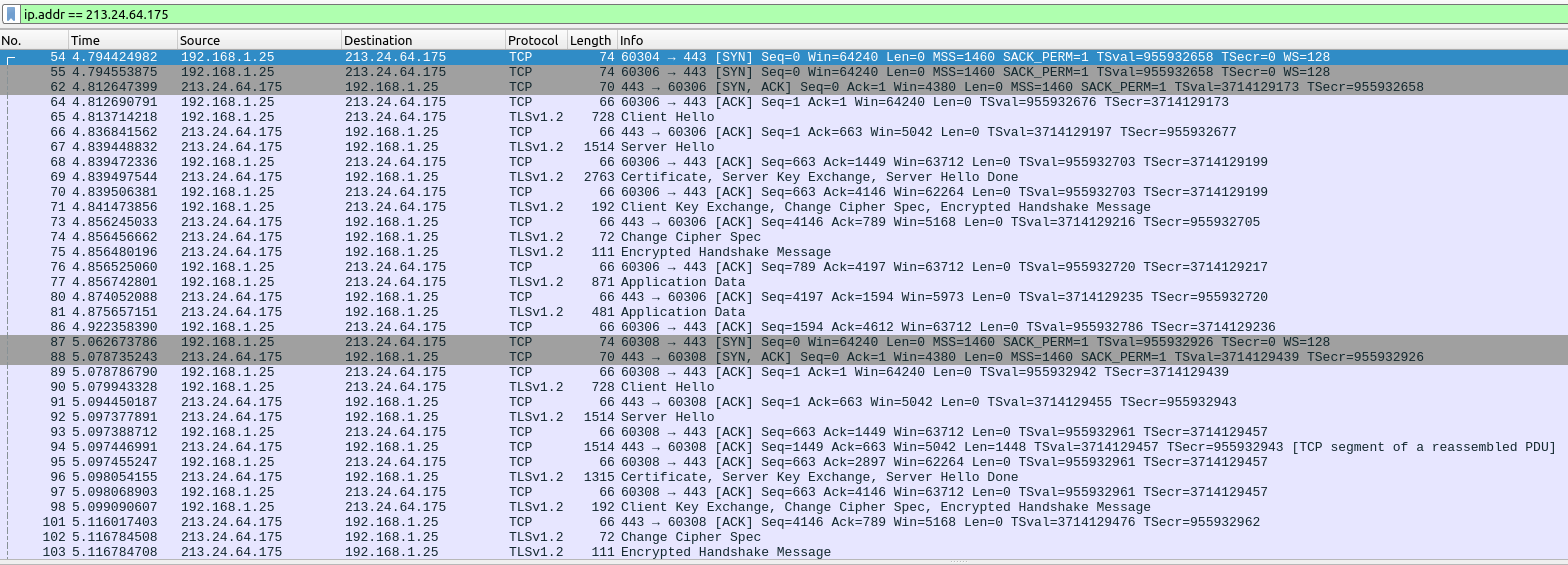
Transport Layer Security (TLS) — это развитие идей, лежащих в основе протокола SSL. Этот протокол обеспечивает приватность, целостность и аутентификацию. Он использует гибридное шифрование, сочетающее асимметричное и симметричное шифрование: общий ключ для симметричного шифрования данных передаётся от клиента к серверу зашифрованным открытым ключом сервера, после чего сервер расшифровывает его своим закрытым ключом и использует для обмена данными с клиентом.

# **Ход лабораторной работы**

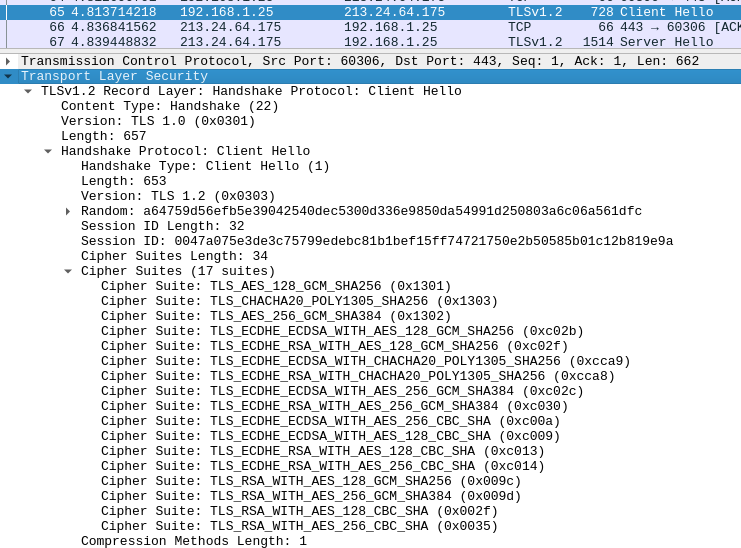
В качестве web-серверов были выбраны два ресурса: <https://lkfl2.nalog.ru> — личный кабинет налогоплательщика и <https://ya.ru/> — главная страница поискового сервиса «Яндекс».

**Первый ресурс**

Начнем с первого сайта. Зайдем через Firefox на lkfl2.nalog.ru и в Wireshark выставим фильтрацию *ip.addr == 213.24.64.175*.

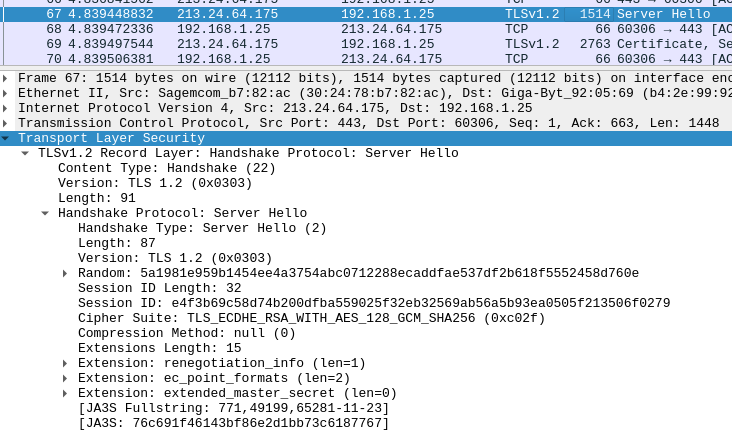


**Первый этап** соединения — сообщение от клиента Client Hello.



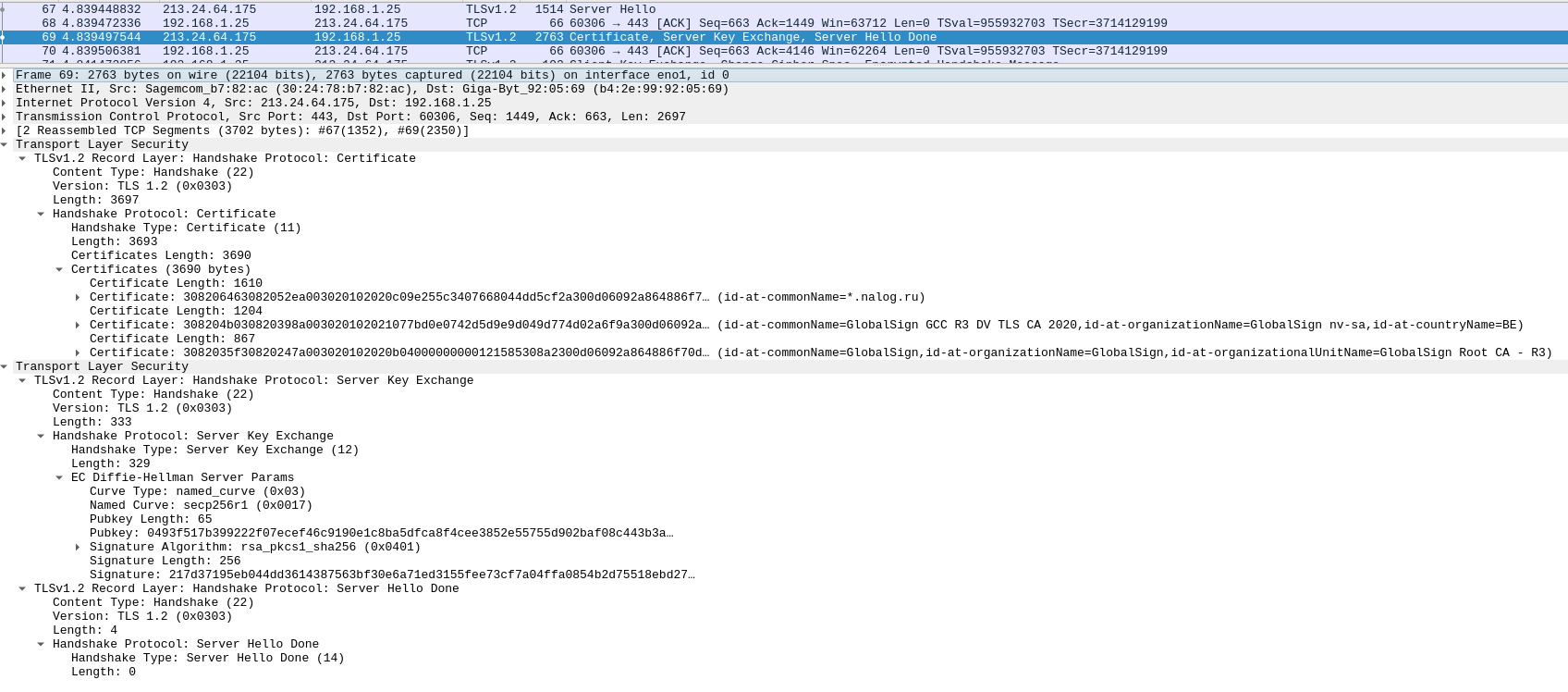
Здесь можно увидеть версию протокола — TLSv1.2, также набор поддерживаемых клиентом алгоритмов шифрования (Cipher Suites) и рандомное число (Random), которое будет использоваться для создания сеансового ключа.

**Второй этап** — сообщение от сервера Server Hello.



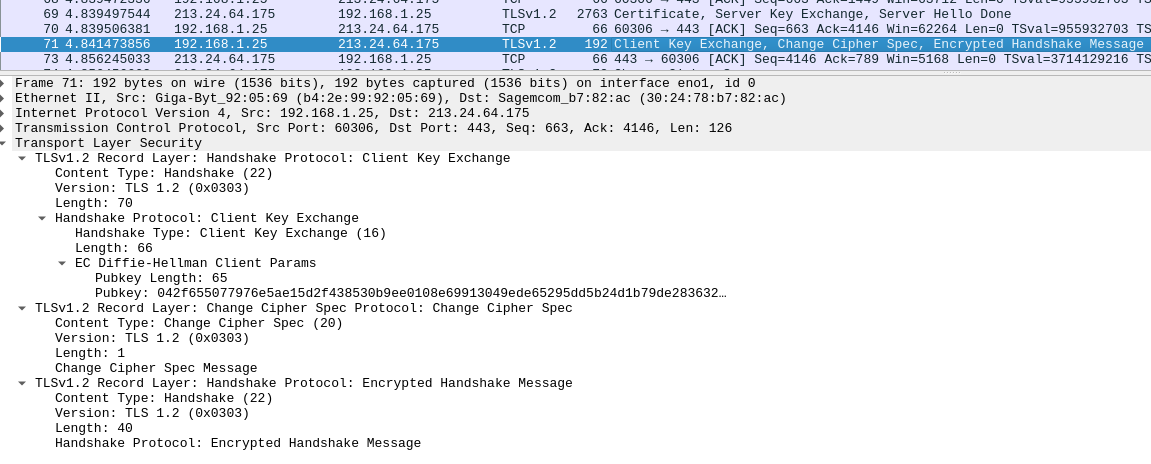
Здесь можно увидеть выбранный алгоритм шифрования (Cipher Suite) — TLS\_ECHDE\_RSA\_WITH\_AES\_128\_GCM\_SHA256 и рандомное число (Random) для создания сеансового ключа.

**Третий этап** — сообщение от сервера Certificate, Server Key Exchange, Server Hello Done.



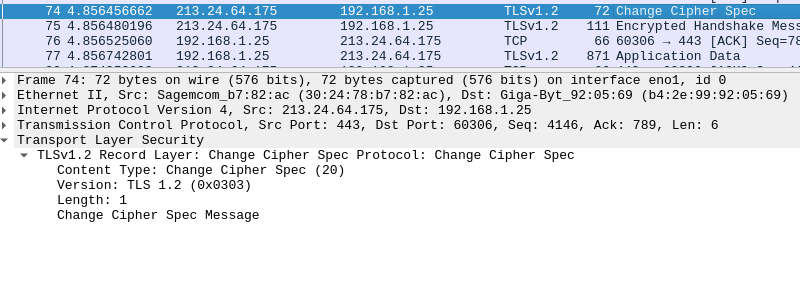
Здесь есть цепочка сертификатов (Certificate): первый — сервера, последний — центра сертификации. Server Key Exchange содержит публичный ключ для создания pre-master secret, который тоже используется для создания сессионного симметричного ключа. Server Hello Done говорит, что начальный этап установки соединения завершен.

**Четвертый этап** — сообщение от клиента Client Key Exchange, Change Cipher Spec, Finished.



В Client Key Exchange клиент передает свою часть публичного ключа для pre-master secret. В Change Cipher Spec клиент говорит, что готов перейти на защищенное соединение. В Finished (Encrypted Handshake Message) содержится первое защифрованное сообщение.

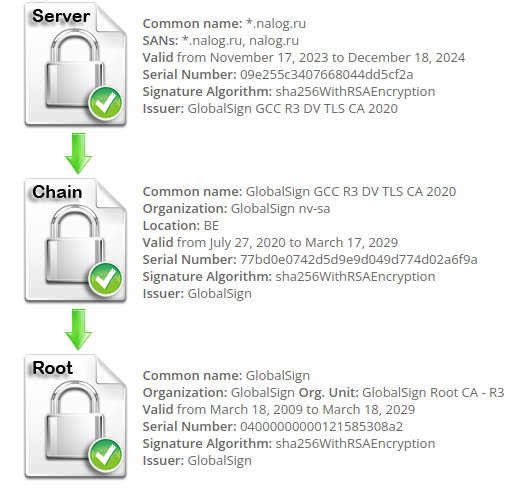
**Пятый этап** — сообщения от сервера Change Cipher Spec и Finished.



После этого этапа соединение считается установленным.

Запишем информацию для этого сайта:

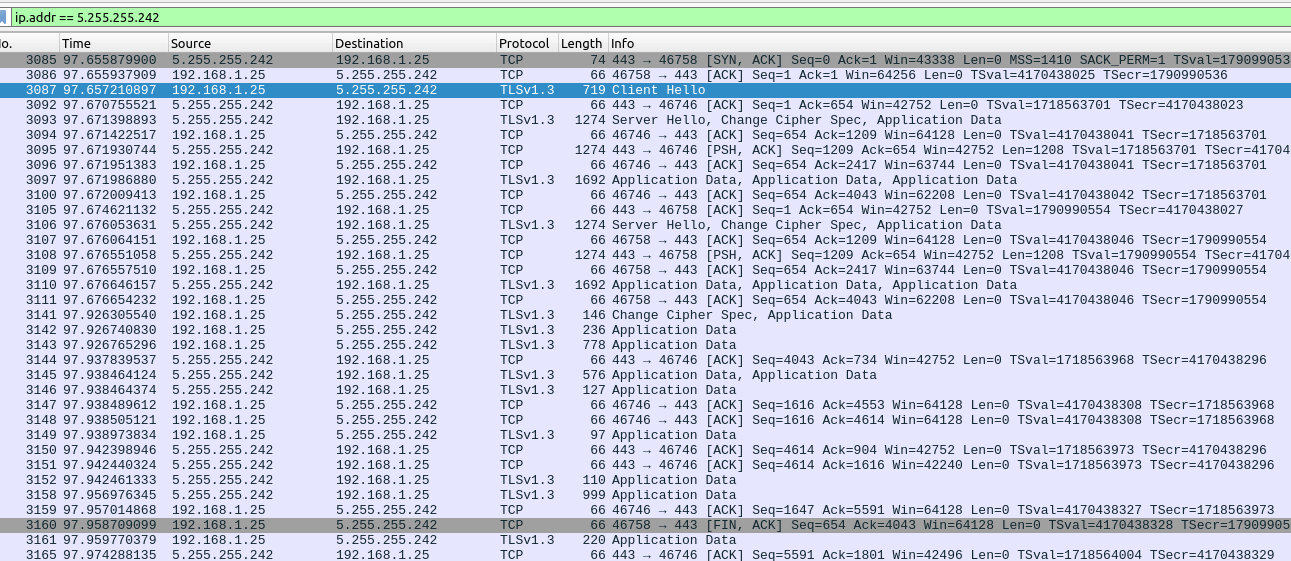
* Имя сервера: lkfl2.nalog.ru
* IP-адрес: 213.24.64.175
* Версия TLS: 1.2
* Выбранные алгоритмы шифрования: TLS\_ECHDE\_RSA\_WITH\_AES\_128\_GCM\_SHA256
* Сертификаты:



* Время установки соединения (от ClientHello до Finished): 0.042765978 секунды.

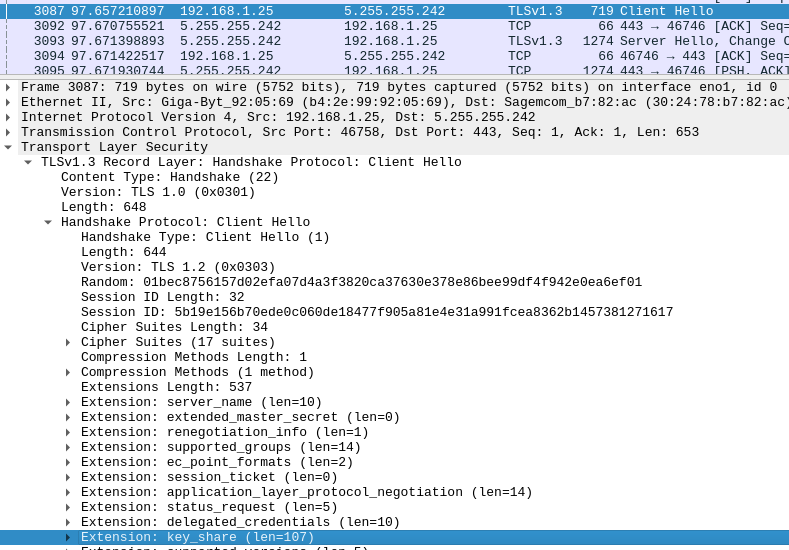
**Второй ресурс**

Перейдем ко второму сайту — ya.ru с ip5.255.255.242.



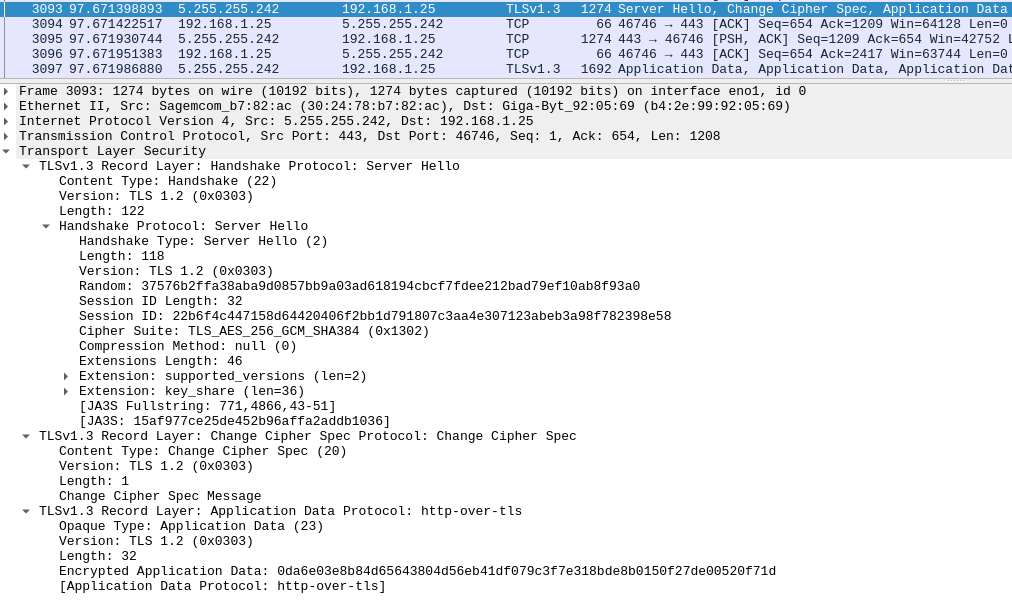
Сервер использует TLSv1.3. Здесь в отличии от TLSv1.2 будут три этапа.

**Первая этап** соединения — сообщение от клиента Client Hello.



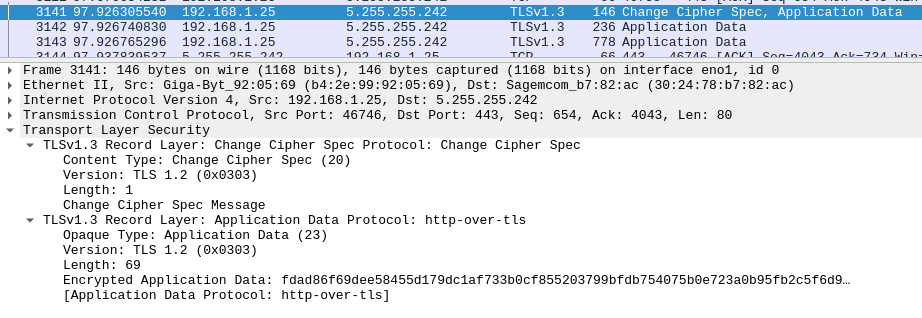
Здесь всё то же самое, как и в TLSv1.2. Но стоит обратить внимание на key\_share — это поле будет и в ответе сервера, это значение используется для создания pre-master secret.

**Второй этап** — сообщение от сервера Server Hello, Change Cipher Spec, Finished.



Видим здесь Change Cipher Spec. Значит, все остальные сообщения (например, Certificate, Finished) будут зашифрованы.

**Третий этап** — сообщение от клиента Change Cipher Sped, Finished.



Клиент тоже начинает шифровать сообщения и отправляет Finished.

Запишем информацию для этого сайта:

* Имя сервера: ya.ru
* IP-адрес: 5.255.255.242
* Версия TLS: 1.3
* Выбранные алгоритмы шифрования: TLS\_AES\_256\_GCM\_SHA384
* Сертификаты:



* Время установки соединения (от ClientHello до Finished): 0.269094643 секунды.

**TLS 1.0**

При смене версии TLS на клиенте удалось подключиться к обоим ресурсам. Но на сайте ya.ru появилось предупреждение о «незащищённом соединении». А на сайте lkfl2.nalog.ru никакого предупреждения не было.



# **Выводы**

В ходе лабораторной работы ознакомился с протоколом TLS, в частности со структурой и работой его версий: TLS 1.2 и TLS 1.3. Научился с помощью программы Wireshark анализировать трафик сети.

В качестве примеров были выбраны два веб-сервера и проведён сравнительный анализ. На его основе можно сделать следующие выводы:

1. Максимальная версия TLS, поддерживаемая **государственным** сайтом lkfl2.nalog.ru — 1.2, т.е. не самая актуальная. При этом сайт поддерживает TLS 1.0 и никак об этом не предупреждает — хотя TLS 1.0 является deprecated. Кажется, что здесь есть проблема с безопасностью.
2. Сайт ya.ru поддерживает TLS 1.3 — актуальную версию протокола. При этом сервер поддерживает и TLS 1.0 (видимо, для большего охвата аудитории), но честно предупреждает, что используется «незащищённое соединение».
3. Было замерено время установки соединении для двух веб-серверов. Сайт ya.ru с TSL 1.3 устанавливал соединение гораздо дольше, чем сайт lkfl2.nalog.ru с TSL 1.2. Но это не означает, что работа TSL 1.2 быстрее, т.к. тут дело ещё в самом сервере — в частности его местоположении относительно клиента. Хоть и обе версии используют гибридное шифрование, TSL 1.3 требует на установку соединения меньше запросов, из чего можно предположить, что эта версия протокола может работать быстрее версии 1.2.

# **Список используемой литературы**

1. Wireshark — подробное руководство по началу использования — https://habr.com/ru/articles/735866/
2. Основы HTTPS, TLS, SSL. Создание собственных X.509 сертификатов. Пример настройки TLSv1.2 в Spring Boot — <https://habr.com/ru/articles/593507/>
3. Decoding TLS v1.2 Protocol Handshake With Wireshark — <https://thesecmaster.com/blog/decoding-tls-v1-2-protocol-handshake-with-wireshark>
4. Decoding TLS 1.3 Protocol Handshake With Wireshark — <https://thesecmaster.com/blog/decoding-tls-1-3-protocol-handshake-with-wireshark>