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 № 5 по курсу

«Криптография»

Группа: М8О-306Б-20

Студент: И. П. Попов

Преподаватель: А. В. Борисов

Оценка:

Дата:

Москва, 2022

# **Задача**

Порядок выполнения лабораторной работы:

1. Выбрать не менее 5-ти веб-серверов 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)

1. Если список исследуемых серверов не исчерпан выбрать другой сервер и повторить соединение.
2. Если браузер поддерживал соединение TLS 1.2 принудительно изменить параметры TLS соединения в Firefox на TLS 1.0 (в браузере перейти по “about:config” и изменить раздел SSL\TLS) и провести попытки соединения с выбранными серверами).
3. Провести сравнительный анализ полученной информации.
4. В качестве отчета представить результаты сравнительного анализа, выводы в отношении безопасности и корректности настройки веб-серверов с учетом их организационной и государственной принадлежности.

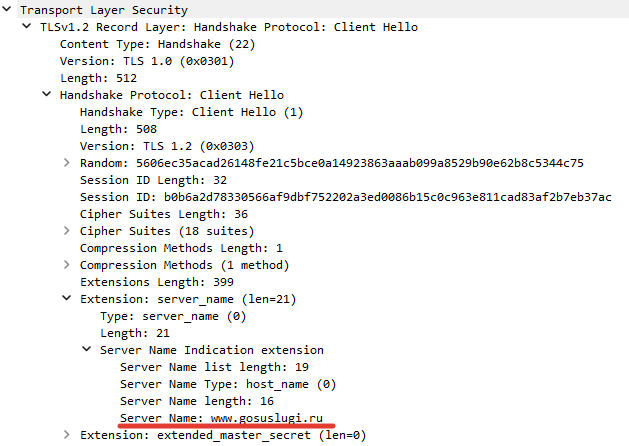
# 

Выбранные серверы:

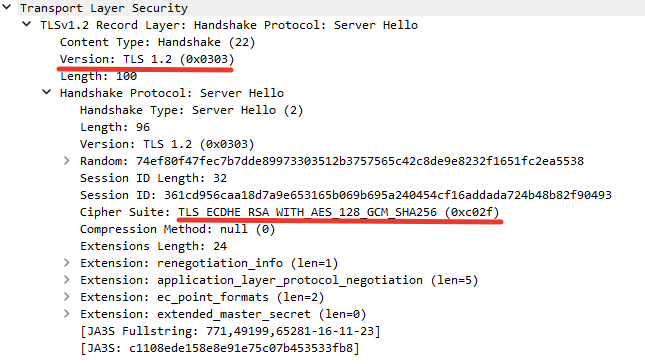
1. Сервер госуслуг России
2. Сервер госуслуг Москвы
3. Сервер МАИ
4. Сервер telegram.org
5. Сервер Сбера

# **www.gosuslugi.ru(109.207.1.118)**

**Имя сервера** находится в пакете Client Hello



Заметим, что клиент предлагает версию протокола TLS 1.0



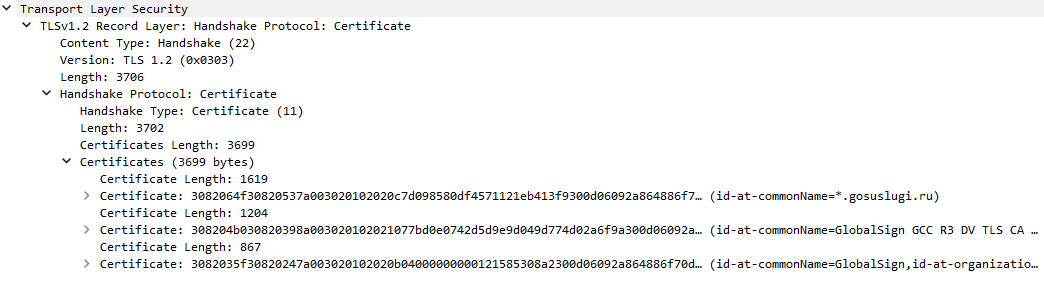
В пакете Server Hello можно найти следующую информацию:

1. **Версию TLS**, которая будет использована при установке соединения, заметим, что она выше, чем предлагал клиент, а именно TLS 1.2.
2. Сгенерированное значение Random для генерации разделяемого ключа, который необходим для алгоритма Диффи-Хеллмана.
3. Набор шифров Cipher Suite

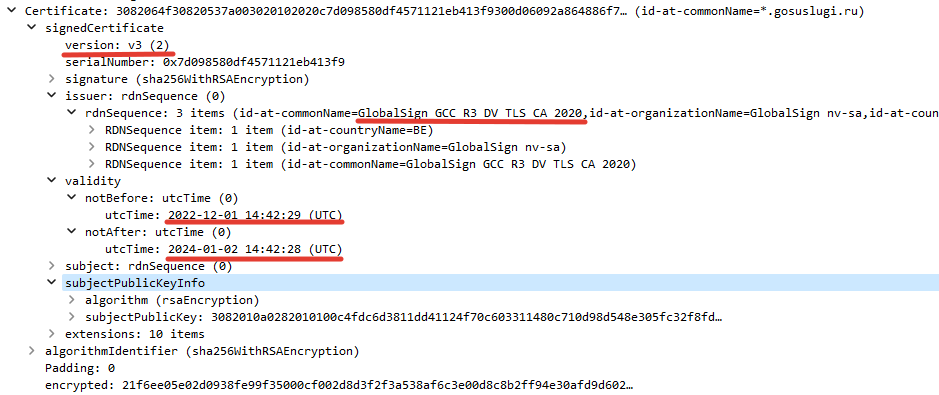
TLS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256 (0xc02f)

говорит о том, что будет использоваться **алгоритм Диффи-Хеллмана на эллиптических кривых ephemeral** (что говорит о том, что новые значения Random будут генерироваться и у сервера и у клиента заново при каждой новой сессии, если прошло достаточно времени после предыдущей, потому что в противном случае будет использован протокол восстановления сессии), шифрование выполняется с помощью алгоритма AES\_128, режим работы GCM и для генерации MAC используется функция SHA256.

Затем сервер отправляет клиенту свой сертификат:

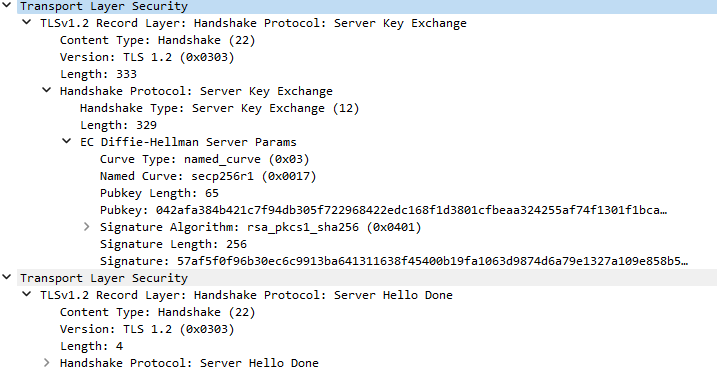


Здесь указаны три сертификата, для начала разберемся с сертификатом сервера.

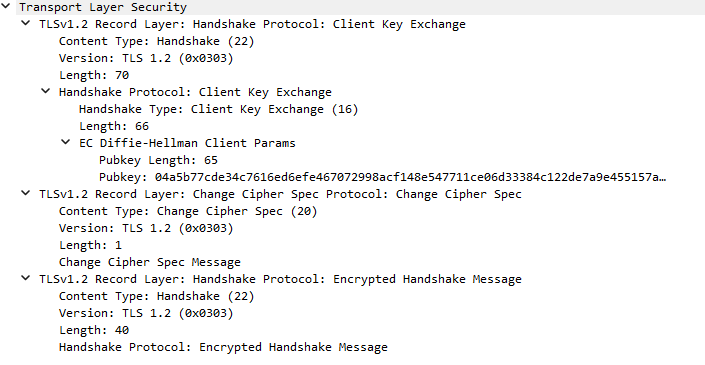


Из этого сертификата мы можем узнать его **версию**, номер, алгоритм записи RSA, информация о выдавшем сертификат удостоверяющем центре, срок действия сертификата (**сертификат валиден**, так как действует до 2024 года), открытый ключ сервера и в encrypted - цифровую подпись удостоверяющего центра. Наш **удостоверяющий центр** - GlobalSign GCC R3 DV TLS CA 2020, но он не является корневым, поэтому нам передаются сертификаты всех его предков, вплоть до одного из корневых, о которому мы, как клиент доверяем.

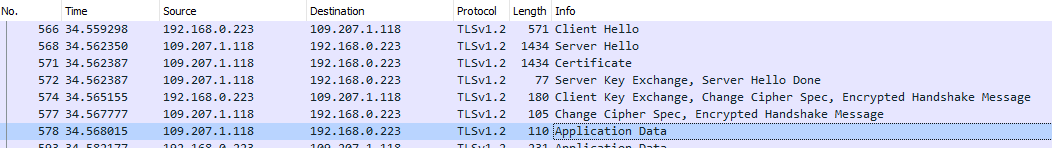
Далее следует обмен **ключами** от сервера к клиенту



И обратно



Рассчитаем **время соединения**

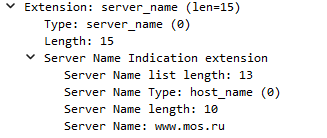


34.568015 - 34.559298 = 0,008717

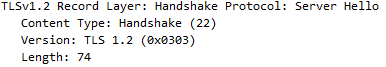
Первый пример я постарался разобрать подробно, последующие же будут содержать лишь информацию про рассматриваемые характеристики.

# **www.mos.ru(94.79.51.14)**

Имя сервера



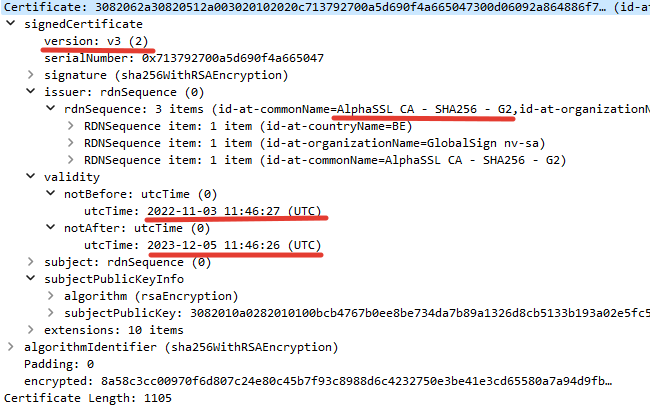
Версия TLS



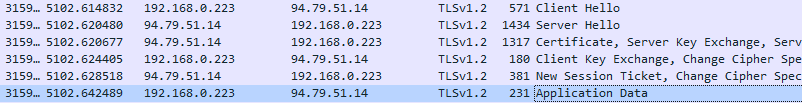
Выбранный алгоритмы шифрования

Cipher Suite: TLS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256 (0xc02f)

Полученный сертификат: версия. Валидность сертификата, валидность ключа, удостоверяющий центр.

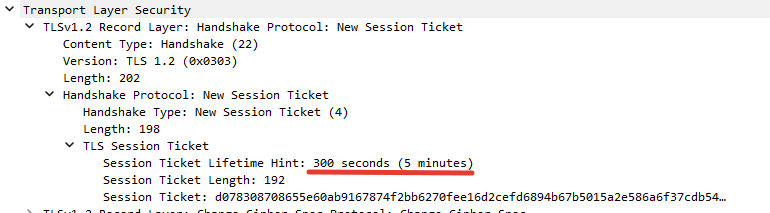


Время установки соединения



5102,642489 - 5102,614832 = 0,027657

Стоит отметить, что в отличие от сайта госуслуг в процессе подключения создается New Session Ticket, благодаря которому можно не тратить время на пересчет ключей при последующей сессии, а просто восстановить их, если новая сессия была начата в пределах указанного временного промежутка.



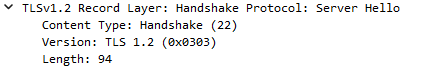
# 

# **www.mai.ru (217.9.89.254)**

Имя сервера

# 

Версия TLS



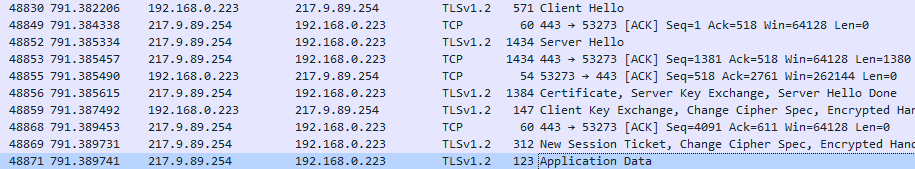
Выбранный алгоритмы шифрования

Cipher Suite: TLS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256 (0xc02f)

Полученный сертификат: версия. Валидность сертификата, валидность ключа, удостоверяющий центр.



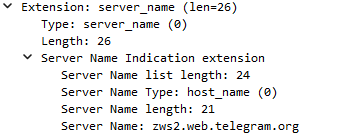
Время установки соединения



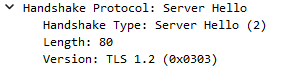
791,389741 - 791,382206 = 0,007535

# **www.telegram.org (149.154.167.99)**

Имя сервера



Версия TLS



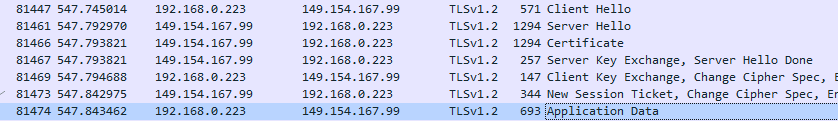
Выбранный алгоритмы шифрования

Cipher Suite: TLS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256 (0xc02f)

Полученный сертификат: версия. Валидность сертификата, валидность ключа, удостоверяющий центр.

# 

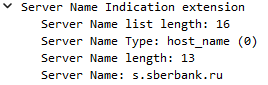
Время установки соединения



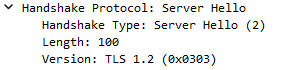
547,843462 - 547,745014 = 0,098448

# **www.sberbank.ru (194.54.14.168)**

Имя сервера



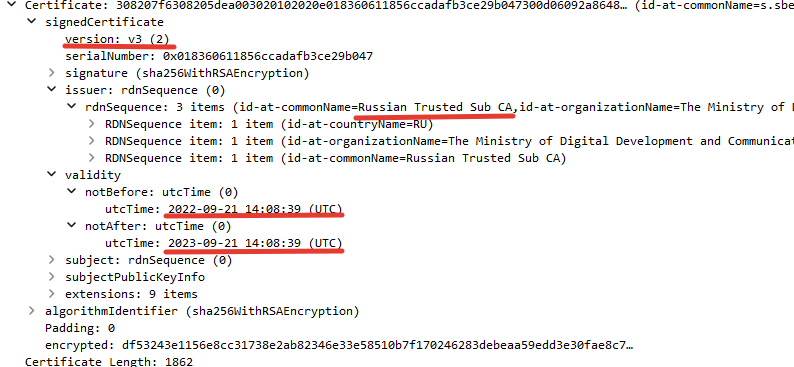
Версия TLS



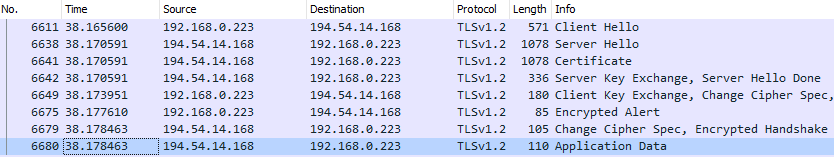
Выбранный алгоритмы шифрования

Cipher Suite: TLS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256 (0xc02f)

Полученный сертификат: версия. Валидность сертификата, валидность ключа, удостоверяющий центр.



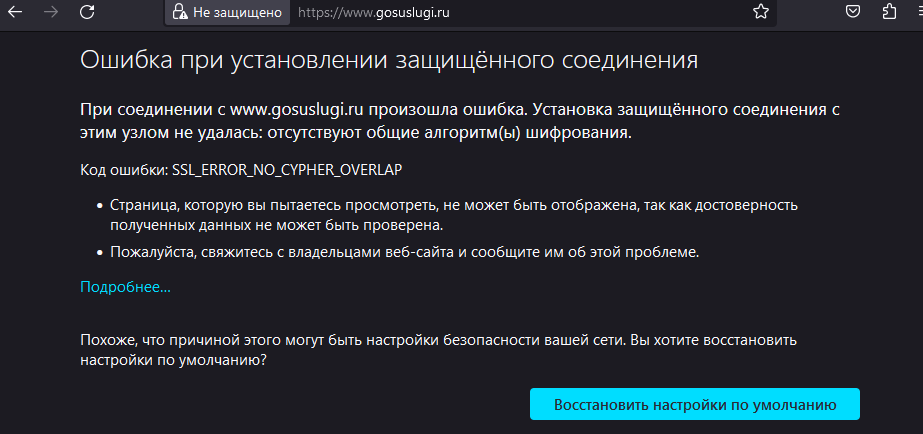
Время установки соединения



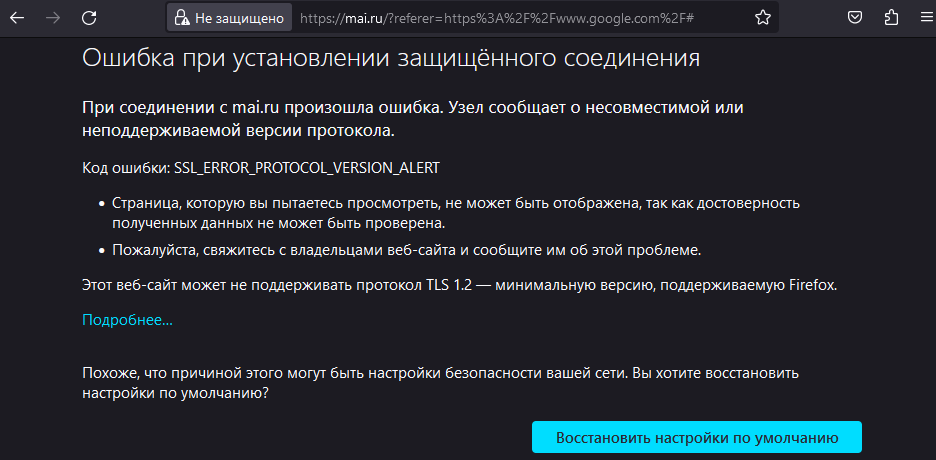
38,178463 - 38,165600 = 0,012863

# **Принудительное изменение параметров TLS соединения в Firefox на TLS 1.0**

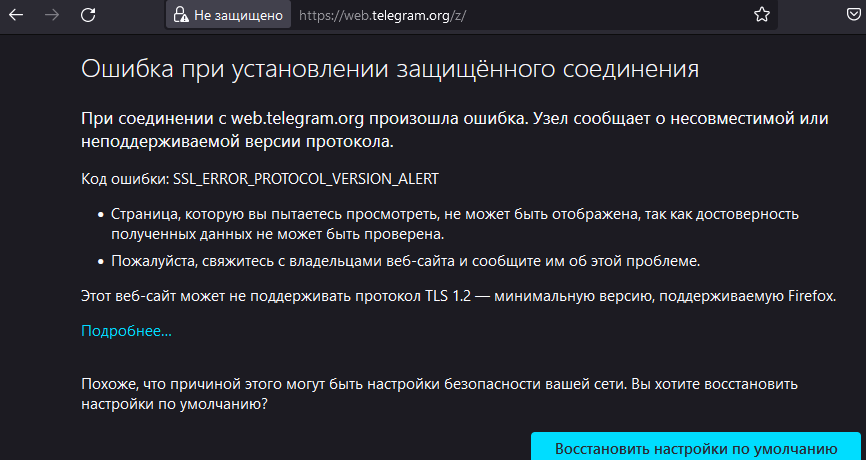
**gosuslugi.ru**

****

**mai.ru**

****

**telegram.org**



Три сервера из представленных выше не поддерживают TLS 1.0, оставшиеся *продолжили работу, но на других наборах шифров.*

## Уязвимости TLS 1.0 и TLS 1.1

В TLS 1.0 и TLS 1.1 есть целый ряд известных уязвимостей, которые могут быть эксплуатированы хакерами. К ним относятся:

* [POODLE](https://www.ibm.com/docs/en/sdk-java-technology/7.1?topic=ip-padding-oracle-downgraded-legacy-encryption-poodle-security-vulnerability) (Padding Oracle On Downgraded Legacy Encryption),
* [BEAST](https://beaglesecurity.com/blog/vulnerability/browser-exploit-against-ssl-tls.html) (Browser Exploit Against SSL/TLS),
* [CRIME](https://www.acunetix.com/vulnerabilities/web/crime-ssl-tls-attack/) (Compression Ratio Info-leak Made Easy),
* [FREAK](https://beaglesecurity.com/blog/support/vulnerability/2018/06/19/Factoring-RSA-Export-Keys-(FREAK)-Attack.html) (Factoring Attack on RSA-EXPORT Keys),
* [LOGJAM](https://en.wikipedia.org/wiki/Logjam_(computer_security)) (Diffie-Hellman Key Exchange Weakness).

Эти уязвимости позволяют выполнять атаки по типу «человек посередине», расшифровывать чувствительную информацию и перехватывать сеансы пользователей. Отключая на своём сервере TLS 1.0 и TLS 1.1, вы можете защитить себя от этих атак.

# 

# **Выводы**

В ходе этой лабораторной работы мною был изучен протокол защиты транспортного уровня - TLS с помощью утилиты Wireshark. Познакомился с различными угрозами перехвата / подмены пакетов и средствами защиты от них. Сравнительный анализ времени ответов серверов показал, что в большей мере на время ответа влияет загруженность сервера, а самый популярный криптографический протокол - алгоритм Диффи - Хеллмана.