**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: В.Р. Орусский

Преподаватель: А. В. Борисов

Оценка:

Дата: 17.05.2024

Москва, 2024

**ОГЛАВЛЕНИЕ**

[1 Тема 3](#_Toc158983147)

[2 Задание 3](#_Toc158983148)

[3 Теория 4](#_Toc158983149)

[4 Ход лабораторной работы 5](#_Toc158983150)

[5 Выводы 6](#_Toc158983151)

# **Тема**

Аутентификация с асимметричными алгоритмами шифрования. Чтение трафика с помощью WireShark.

# **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 необходимые для последующего 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. Для оставшихся серверов повторить п.п.1-4.
2. Если браузер поддерживал соединение TLS 1.2, принудительно изменить параметры TLS соединения в Firefox на TLS 1.0 (перейти по адресу “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.

# **Теория**

HTTP (HyperText Transfer Protocol) — сетевой протокол прикладного уровня (модель OSI/ISO), который изначально предназначался для получения с серверов гипертекстовых документов в формате HTML, а с течением времени стал универсальным средством взаимодействия между узлами как Всемирной паутины, так и изолированных веб-инфраструктур.

Основой HTTP является технология «клиент-сервер», то есть предполагается существование:

Потребителей (клиентов), которые инициируют соединение и посылают запрос;

Поставщиков (серверов), которые ожидают соединения для получения запроса, производят необходимые действия и возвращают обратно сообщение с результатом.

HTTPS (HyperText Transfer Protocol Secure) — расширение протокола HTTP для поддержки шифрования в целях повышения безопасности. Данные в протоколе HTTPS передаются поверх криптографических протоколов TLS или устаревшего в 2015 году SSL. В отличие от HTTP, который по стандарту пользуется портом 80, для HTTPS по умолчанию используется порт 443.

HTTPS не является отдельным протоколом, по сути это обёртка над обычным HTTP, которая позволяет работать через защищённые механизмы транспортировки информации (TLS, SSL).

TCP (Transmission Control Protocol) — один из основных протоколов передачи данных интернета. Предназначен для управления передачей данных интернета. Пакеты в TCP называются сегментами. Выполняет функции транспортного уровня.

Механизм TCP предоставляет поток данных с предварительной установкой соединения, осуществляет повторный запрос данных в случае потери данных и устраняет дублирование при получении двух копий одного пакета, гарантируя тем самым (в отличие от UDP) целостность передаваемых данных и уведомление отправителя о результатах передачи.

SSL (Secure Sockets Layer) — криптографический протокол, который подразумевает более безопасную связь. Он использует асимметричную криптографию для аутентификации ключей обмена, симметричное шифрование для сохранения конфиденциальности, коды аутентификации сообщений для целостности сообщений.

Протокол SSL предоставляет «безопасный канал», который имеет три основных свойства:

* Канал является частным. Шифрование используется для всех сообщений после простого диалога, который служит для определения секретного ключа.
* Канал аутентифицирован. Серверная сторона диалога всегда аутентифицируется, а клиентская делает это опционально.
* Канал надёжен. Транспортировка сообщений включает в себя проверку целостности.

Преимуществом SSL является то, что он независим от прикладного протокола. Протоколы приложений (HTTP, FTP, TELNET и т. д.) могут работать поверх протокола SSL совершенно прозрачно, то есть SSL может согласовывать алгоритм шифрования и ключ сессии, а также аутентифицировать сервер до того, как приложение примет или передаст первый байт сообщения.

В 2014 году был признан уязвимым и должен быть заменён на TLS.

TLS (transport layer security) — криптографические протоколы, обеспечивающие защищённую передачу данных между узлами в сети Интернет. TLS и SSL используют асимметричное шифрование для аутентификации, симметричное шифрование для конфиденциальности и коды аутентичности сообщений (хеширование) для сохранения целостности сообщений.

Данный протокол широко используется в приложениях, работающих с сетью Интернет, таких как веб-браузеры, работа с электронной почтой, обмен мгновенными сообщениями и IP-телефония (VoIP).

Основные шаги процедуры создания защищённого сеанса связи:

* клиент подключается к серверу, поддерживающему TLS, и запрашивает защищённое соединение;
* клиент предоставляет список поддерживаемых алгоритмов шифрования и хеш-функций;
* сервер выбирает из списка, предоставленного клиентом, наиболее надёжные алгоритмы среди тех, которые поддерживаются сервером, и сообщает о своём выборе клиенту;
* сервер отправляет клиенту цифровой сертификат для собственной аутентификации. Обычно цифровой сертификат содержит имя сервера, имя удостоверяющего центра сертификации и открытый ключ сервера;
* клиент, до начала передачи данных, проверяет валидность (аутентичность) полученного серверного сертификата относительно имеющихся у клиента корневых сертификатов удостоверяющих центров (центров сертификации). Клиент также может проверить, не отозван ли серверный сертификат, связавшись с сервисом доверенного удостоверяющего центра;
* для шифрования сессии используется сеансовый ключ. Получение общего секретного сеансового ключа клиентом и сервером проводится по протоколу Диффи-Хеллмана. Существует исторический метод передачи сгенерированного клиентом секрета на сервер при помощи шифрования асимметричной криптосистемой RSA (используется ключ из сертификата сервера). Данный метод не рекомендован, но иногда продолжает встречаться на практике.

Wireshark — программа-анализатор трафика для компьютерных сетей.

Функциональность, которую предоставляет Wireshark, очень схожа с возможностями программы tcpdump, однако Wireshark имеет графический пользовательский интерфейс и гораздо больше возможностей по сортировке и фильтрации информации. Программа позволяет пользователю просматривать весь проходящий по сети трафик в режиме реального времени.

# **Ход лабораторной работы**

Для решения данной работы пришлось предварительно настроить логгирование сессионных ключей от браузера в конкретный файл, дабы WireShark смог читать защищённые TLS соединения.

Для работы были взяты следующие сайты:

* [YouTube](https://www.youtube.com/); IP = 74.125.205.198
* [Avito](https://www.avito.ru/); IP = 176.114.122.24
* [Yandex](https://yandex.ru/company/); IP = 77.88.55.88
* [МВД РФ](https://мвд.рф); IP = 82.202.191.70
* [Сайт белого дома США](https://www.whitehouse.gov/); IP = 192.0.66.168

С помощью WireShark был захвачен трафик при соединении с каждым из этих сайтов, после чего была получена следующая информация:

1. YouTube

* Имя сервера: [www.youtube.com](http://www.youtube.com)
* Версия TLS: 1.2
* Алгоритм шифрования: SHA-256 with RSA Encryption
* Серийный номер сертификата:

00fa59c19729ce37bf1007d1047af1f737

* Версия сертификата: v3
* Центр сертификации:

<http://i.pki.goog/wr2.crt>

<http://o.pki.goog/wr2>

* Время установки соединения: 0.021659

1. Avito

* Имя сервера: www.avito.ru
* Версия TLS: 1.0

Не смотря на использование TLS сайтом Авито, «рукопожатие» так и не было завершено, пакет с информацией о сертификате не был передан сервером.

1. Yandex

* Имя сервера: yandex.ru
* Версия TLS: 1.2
* Алгоритм шифрования: ECDSA with SHA-384
* Серийный номер сертификата:

7097913e97c436858de28d6c

* Версия сертификата: v3
* Центр сертификации:

<http://secure.globalsign.com/cacert/gseccovsslca2018.crt>

http://ocsp.globalsign.com/gseccovsslca2018

* Время установки соединения: 0.01081

1. МВД.РФ

* Имя сервера: мвд.рф (xn--b1aew.xn--p1ai)
* Версия TLS: 1.2
* Алгоритм шифрования: SHA-256 with RSA Encryption
* Серийный номер сертификата:

59e04f84d0d859acbba3d9f9

* Версия сертификата: v3
* Центр сертификации:

http://secure.globalsign.com/cacert/gsgccr6alphasslca2023.crt

http://ocsp.globalsign.com/gsgccr6alphasslca2023

* Время установки соединения: 0.353202

1. Сайт Белого дома США

* Имя сервера: whitehouse.gov
* Версия TLS: 1.0

Не удалось установить ответный пакет с сертификатом от сервера.

Можно заметить, что используя в браузере TLS 2+, на сайтах, где поддерживается TLS 1, не получается установить обратный пакет «рукопожатия», который бы содержал в себе необходимые сертификаты. По наблюдениям, можно сказать, что скорее всего на данной версии TLS, конечный сертификат отправляется не сервером, с которым началось «рукопожатие», а каким-то вспомогательным сервером, но установить принадлежность такого сервера к нужному веб-сайту не представляется возможным.

Заменив версию TLS в браузере на 1.0, получаем невозможность посетить определённые страницы: МВД, сайт Белого дома (что очень странно, ведь этот сайт использует TLS 1.0). Во всём остальном, никаких изменений непосредственно в сертификатах не наблюдается.

С изменением версии наблюдается немного изменённая модель поведения сервера в процессе «рукопожатия», сервер не присылает конечный пакет с обговорённым сертификатом, он присылает только приветственный пакет, в котором также есть информация о сертификате, более никаких пакетов не поступает.

# **Выводы**

В процессе выполнения данной ЛР, я узнал много нового, а также повторил, что было известно о модели OSI / ISO, почитал побольше про протоколы, используемые для передачи и защиты данных в интернете. Изучил процесс работы с WireShark, научился ориентироваться в нём, применять фильтры, отслеживать пакеты, находить нужные. Косвенно изучил некоторые стандарты, по которым существует интернет (RFC).

# **Список используемой литературы**

1. TLS - <https://ru.wikipedia.org/wiki/TLS>
2. TCP - <https://ru.wikipedia.org/wiki/TCP>
3. Сетевая модель OSI - [https://ru.wikipedia.org/](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_OSI#%D0%9F%D1%80%D0%B8%D0%BA%D0%BB%D0%B0%D0%B4%D0%BD%D0%BE%D0%B9_%D1%83%D1%80%D0%BE%D0%B2%D0%B5%D0%BD%D1%8C)
4. HTTPS - <https://ru.wikipedia.org/wiki/HTTPS>
5. HTTP - <https://ru.wikipedia.org/wiki/HTTP>
6. Логгирование сессионных ключей - [https://habr.com/ru](https://habr.com/ru/articles/253521/)
7. Работа с WireShark - https://habr.com/ru/articles/735866/