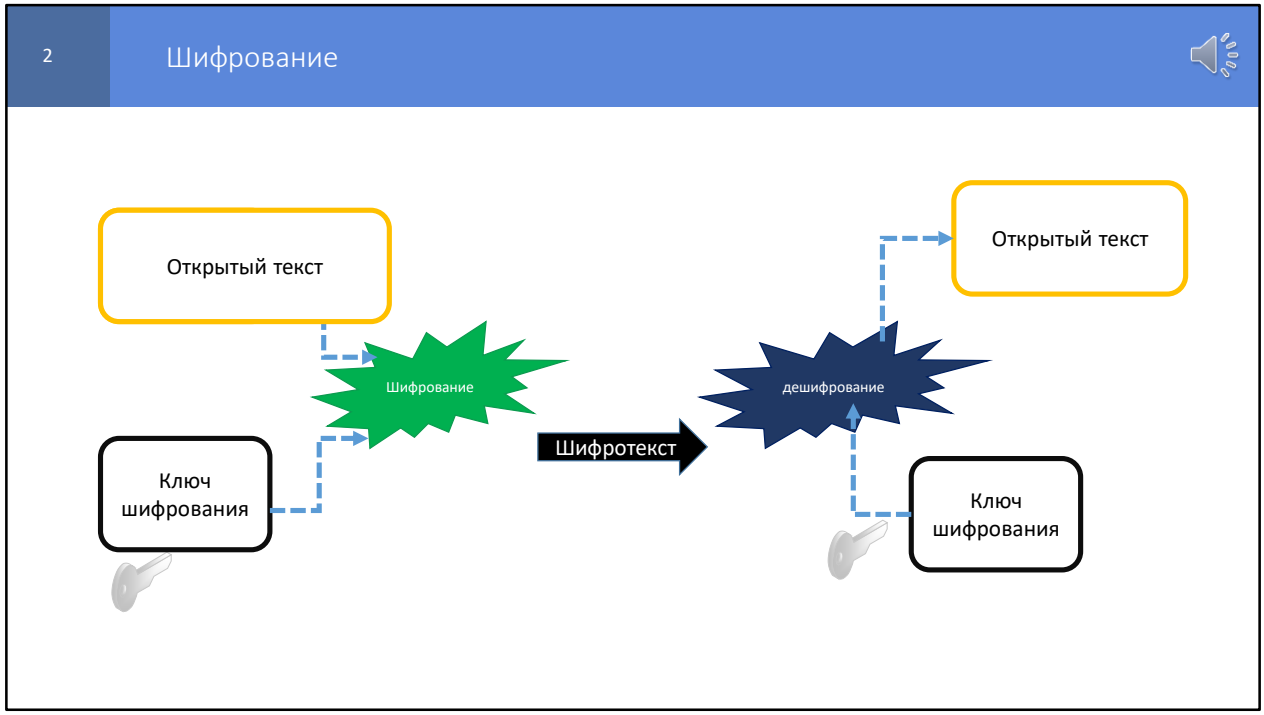
Здравствуйте,
Протокол TLS сейчас используется везде и всюду, но практика показывает, что часто его не очень понимают
Я в силу некоторых причин разобрался и постараюсь рассказать основные моменты максимально просто и доступно



Современное шифрование работает довольно просто. Есть исходные данные, называемые открытый текст, есть набор байт, называемое ключом шифрование. Всё это подаётся на вход компьютерной программе, там происходит *ВЖУХ* и на выходе получается шифротекст. Его уже можно ничего не боясь передавать по любым каналам. Принимающая сторона использует такой же алгоритм в обратную сторону, этот же ключ шифрования и после своего *ВЖУХ* получает на выходе исходный текст.

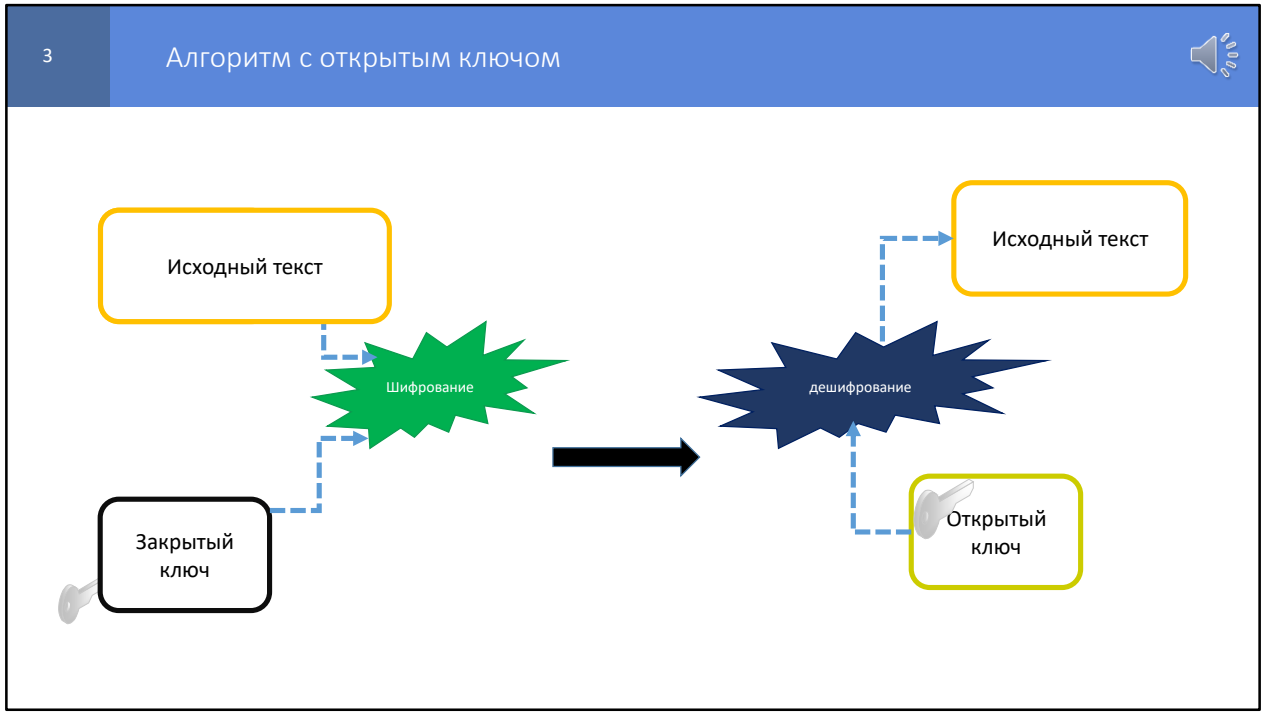
Сами шифрованные данные не зная ключа восстановить невозможно. Такая схема называется симметричным шифрованием поскольку используется один ключ для шифровки и для расшифровки. То есть ключ этот надо держать в секрете, на нём одно всё и держится

echo Hello world | openssl enc -e -aes-128-cbc -out ciphertext

openssl enc -d -aes-128-cbc

показано как на любом компьютере воспроизвести этот процесс шифрования и дешифрования. На вход openssl подаётся открытый текст, указывается алгоритм

шифрования с режимом работы, затем он спрашивает пароль, делает из него ключ, им шифрует и сохраняет в выходном файле.
Вторая команда, наоборот, спрашивает пароль и из шифрованного файла получает исходный открытый текст.
Openssl сейчас есть даже в составе Windows, так что это легко можно попробовать



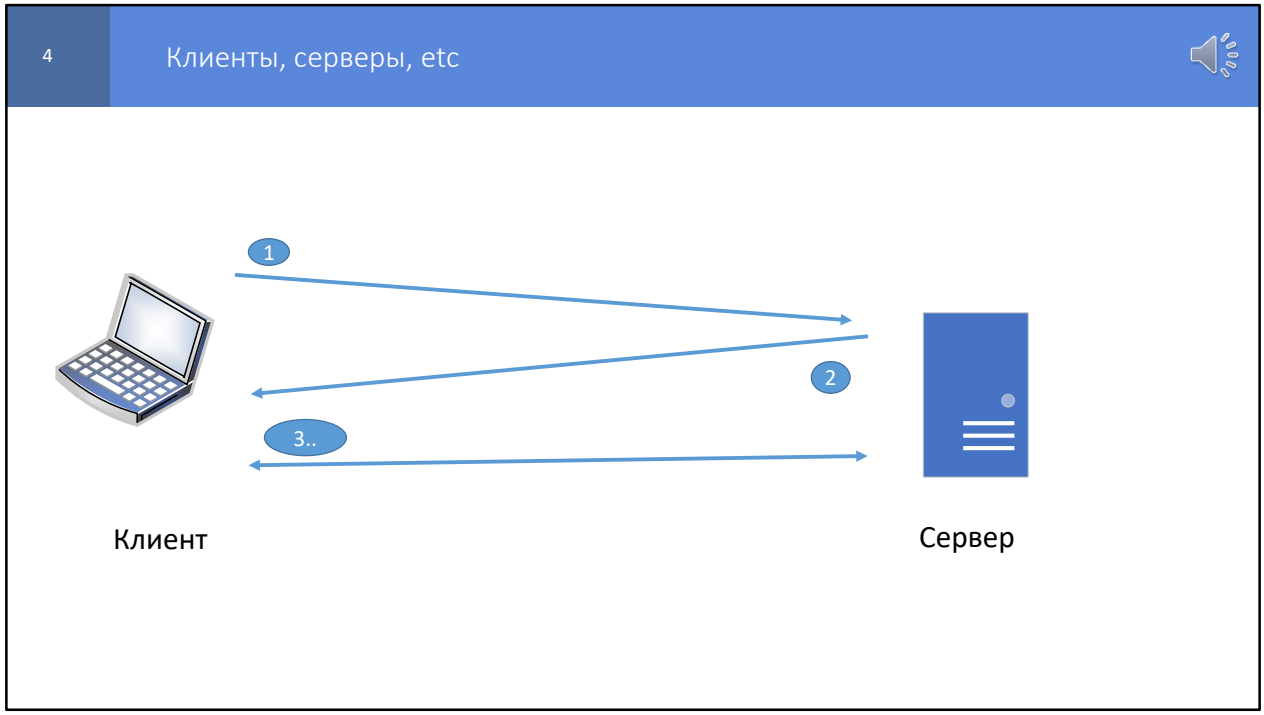
На практике, понятное дело, было бы неудобно: заходите на веб-сайт и там надо вводить какие-то ключи и тп.

Поэтому полвека назад придумали вариант с двумя ключами, т.н. схема с открытым ключом. Отправитель генерирует два длинных числа, связанных друг с другом некоторым хитрым образом, одно держит в секрете, его называют закрытым ключом, второе – называет открытым и распространяет повсеместно

При шифровании делается *ВЖУХ* с исходным текстом и закрытым ключом. При дешифровании – наоборот с открытым и получают назад исходный текст

По сравнению с симметричными алгоритмами тут есть важное преимущество – публичный ключ не надо держать в секрете. На симметричном алгоритме чтобы отправить тайные данные надо как-то передать и ключ, то тут такой необходимости нет

Вместо открытый и закрытый ключи я дальше буду использовать жаргонизмы приватный и публичный - как более понятные



Как шифрованные соединения через Интернет могли бы работать по-простому

(1) Клиент подключается к серверу
(2) Сервер ему отправляет свой публичный ключ, затем создаёт ключ симметричного шифрования, шифрует на приватном ключе и тоже отправляет клиенту
(3) Клиент расшифровывает его с помощью публичного ключа сервера,
(4) и дальше они могли бы работать по зашифрованному этим симметричным алгоритмом каналу

(Этот ключ симметричного шифрования называют сеансовым или сессионным)

5

MITM: человек посередине

Казалось бы всё хорошо: данные зашифрованными, при передаче их никто не прочитает. Зачем тогда какие-то сертификаты и прочее?

Но остаются проблемы с активными жуликами, сидящими на пути данных

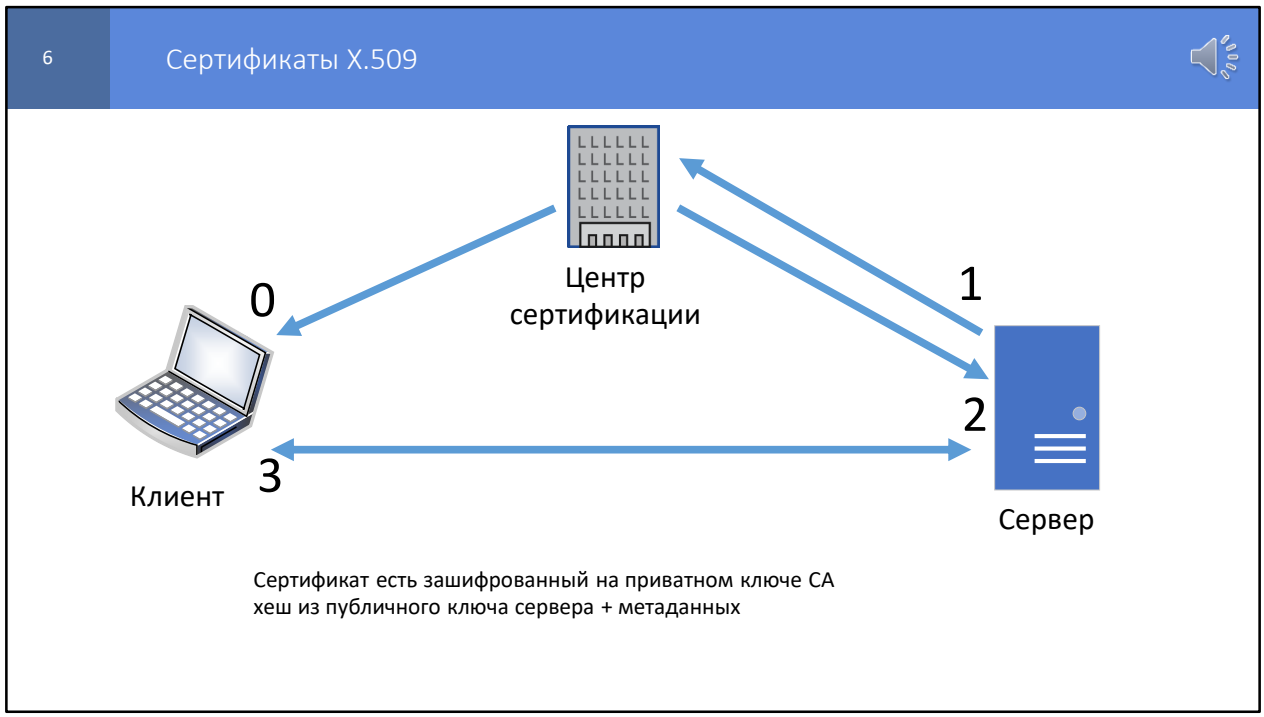
Компьютер в разрыве может прикидываться сервером для клиентов и клиентом для сервера и видеть/изменять данные в открытом виде. Это может быть прокси сервер, роутер/гейт или даже L2 ARP атака

Жулик этот чего делает – при подключении клиента берёт имя сервера из запроса, подключается к настоящему серверу, с обычным шифрованным соединением, полученные данные расшифровывает как обычно, затем шифрует их заново на своих ключах и отправляет клиенту

Ситуация называется MITM: человек посередине

Таким образом этому устройству посередине видны передаваемые данные в открытом виде и никто из серверов и клиентов об этом не подозревает

(ДЕМО)



Именно для защиты от MITM используется третья сторона, которой все доверяют: центр сертификации (CA)

(0) начинается всё с добавления клиентом сертификата центра в список доверенных
(1) Для получения сертификата сервер отправляет свой публичный ключ в центр
(2) Центр его *подписывает* и выдаёт сертификат серверу

(3) И TLS соединение работает так: сервер отправляет сертификат, внутри которого его публичный ключ, клиент проверяет его цифровую подпись и, если всё сходится, то устанавливается шифрованное соединение

>Это самый главный слайд во всей презентации. Хочется чтобы вы его понимали<

На слайде написано как работает цифровая подпись, но на практике это не важно, не заостряйте здесь внимание.
Надо знать, что есть *ВЖУХ* подпрограмма, которая на основе приватного ключа и полезных данных создаёт набор байт называемый цифровой подписью

и другая подпрограмма (тоже *ВЖУХ*), которая на основе публичного ключа и этих данных выдаёт однозначный ответ – верна подпись или нет.

Подписи эти создаются центром сертификации на своём ключе при выдаче сертификата серверу. И создаются сервером на своём ключе при подключении клиента. Клиент их проверяет тоже два раза: сначала, что сертификат верный (на публичном ключе CA) и второй раз, что сервер прислал правильный сертификат, от которого на сервере есть приватный ключ

Сертификаты нужны только для защиты от активных атак на канале между клиентом и сервером. Сертификат подтверждает, что есть приватный ключ на сервере с таким DNS-именем.

И не более того

7

Процесс получения сертификата

1. Создание ключевой пары (приватный+публичный ключи)
2. Из публичного ключа + метаданных создаётся CSR – запрос на сертификат
3. CSR отправляется в CA
4. Из CA получается сертификат
5. Приватный ключ и сертификат добавляется в конфигурацию сервера

На сервере создают приватный/публичный ключи. В публичный добавляется имя сервера и он отправляется в виде CSR в центр сертификации.
Из CA приходит в ответ сертификат и они с приватным ключом добавляется в конфигурацию сервера

После всего CSR уже не нужен

Альтернативная ситуация – повесить существующий сертификат на сервер. Он должен быть с приватным ключом, поэтому максимально аккуратно, не пересылая по email и не по tftp забрасывается на сервер и там прописывается в конфигах

8

Что значит быть доверенным

1. Сертификат подписан доверенным центром сертификации
2. Выписан на верное имя DNS (DN и SubjectAltName)
3. Время актуальное

Что значит быть доверенным?
Решает клиент
Сертификат выписан центром, от которого у клиента есть публичный ключ
Имя сервера верное, то есть DNS имя и должен работать DNS. Настоящие сертификаты большого интернета не выписывают на IP-адреса и не выписывают на имена, не видимые с интернета, типа .local и тп.
Аналогично, когда сертификат на DNS-имя, а подключаетесь к серверу по IP-адресу, то доверять не будут
DNS-имя должно быть полное, www.example.com на просто www сертификат не выпишут и при подключении по короткому имени
Время должно быть более-менее верное

9

Степени доверия сертификату

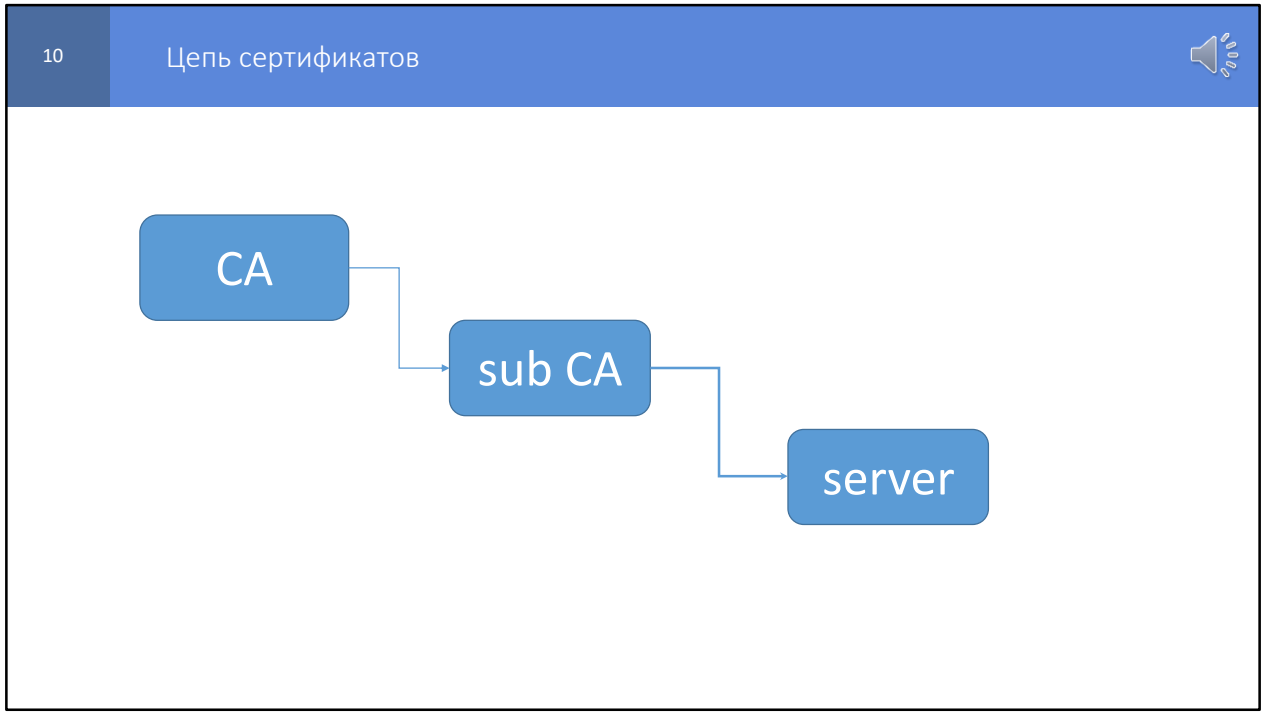
- Сертификат выписан CA из списка
- CA не из списка
- Самоподписанный / selfsigned

Часто в этом деле путаются. Отправляемый сервером сертификат выписан CA из списка доверенных у клиента, тогда всё работает без проблем вообще, самый простой случай

Сертификат может быть выписан центром не из списка, тогда при добавлении его клиентом в список снова соединению доверяют и всё работает

Сертификат у сервера может быть самоподписанный, тогда никак работать не будет – такому сертификату клиент не будет доверять никогда

(Демо: сертификаты)



Настоящие сертификаты больших центров в интернете выписывают на sub-CA. Сертификат subCA должен отправляться сервером. Один широко известный сервер этого не делает и проблему диагностировать может быть достаточно сложно, браузеры могут накешировать их от других сайтов и у части клиентов будет работать, а у других - нет

(ДЕМО)

На практике очень важно время действия сертификатов.
Очень много случаев когда срок вышел и важные системы превратились в тыкву.
Так несколько лет назад в Англии сломался их крупный сотовый оператор, позже выяснилось что где-то внутри оборудования Ericsson закончился срок сертификата.
Аналогично в прошлом году фирма Cisco поздравила свою клиентуру SDWAN с майскими праздниками

https://www.theguardian.com/business/2018/dec/06/o2-customers-unable-to-get-online
https://www.ericsson.com/en/press-releases/2018/12/update-on-software-issue-impacting-certain-customers?707191720
https://www.cisco.com/c/en/us/support/docs/routers/sd-wan/220448-identify-vedge-certificate-expired-on-ma.html

12

Тик-так

Следить за сроком действия

Это второй самый важный слайд в презентации. Как только поставили где-то сертификат, то сразу же заводите будильник с напоминанием его предстоящей замены

Исправление истёкшего сертификата может быть достаточно сложно – ездить по стране, лазить по вышкам связи и т.п

Срок действия общедоступных сертификатов сейчас не более одного года, так Google с Apple решили

Вы можете подумать такие, что проверять сертификаты очень муторно и игнорировать их. Поучительный пример, когда такой подход полностью разрушает всю систему

Есть дверные замки, смартзамки, то есть управляются через приложение со смартфона, подключаются по WiFi. Производитель решил игнорировать проверку сертификата. Это привело к тому, что можно перебить сконфигурированную точку доступа одноимённой, но ближе расположенной, подделать в ней DNS и подделать сервер бэкенда, чтобы тот отвечал положительно на любой запрос открытия дверей

Таким образом полностью разламывается этот замок. Заходи кто хочешь, бери что хочешь

https://research.nccgroup.com/2022/07/25/technical-advisory-multiple-vulnerabilities-in-nuki-smart-locks-cve-2022-32509-cve-2022-32504-cve-2022-32502-cve-2022-32507-cve-2022-32503-cve-2022-32510-cve-2022-32506-cve-2022-32508-cve-2/#CVE-2022-32509

14

Клиентские сертификаты / Mutual TLS

- Аутентифицируют клиента на сервере
- Часто применяются внутри систем когда несколько устройств взаимодействуют между собой (M2M) вместо логинов и паролей
- Проверяется аналогично: у сервера список доверенных CA и он разрешает доступ клиентам с сертификатами оттуда
- Популярно в OpenVPN, используется на сайтах ЭТП
- Тоже надо следить за сроком действия, проверять сертификаты и ограничить список доверенных CA

В дополнение к серверным сертификатам, в некоторых случаях используются клиентские. Схема работы их примерно такая же: приватный ключ внутри устройства, публичный подписывается CA и при подключении отправляется на сервер

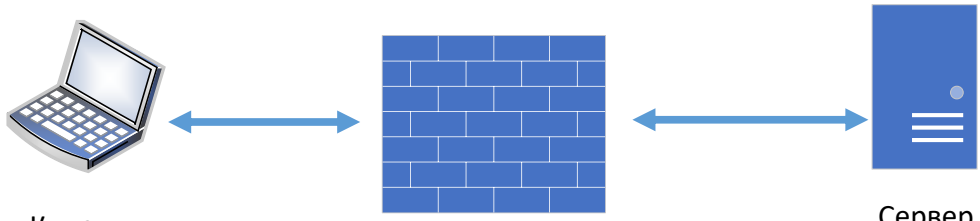
Сервер его проверяет по своему списку доверенных CA

Редко применяется в браузерах, но популярен в соединениях между машинами вместо логинов и паролей

Тоже надо следить за сроком действия и кому доверяет сервер

15

Пиннинг сертификатов



Клиент

FW SSL Decryption / SSL Visibility

Сервер

- Дополнительная защита от MITM хозяином клиентского устройства
- При подключении к серверу проверяет не только список доверенных CA, но и явно его публичный ключ
- Расстраивает софт в ситуации SSL Decryption на брандмауэрах

Зашифрованный трафик, естественно, недоступен для проверки корпоративным брандмауэром.
Поэтому производители таких устройств предлагают клиентам функционал MITM: железяка выписывает на ходу новые сертификаты со своего CA, сертификат этого CA заранее конфигурируется в клиенты. Копия слайда с жуликом, но в этот раз тут наши люди

Вы можете столкнутся с таким на практике, когда компоненты вашей системы общаются между собой по TLS. И если между ними появляется такой брандмауэр, то взаимодействие может сломаться с неочевидной диагностикой.

Далеко не всегда можно добавить нужные сертификаты во внутрь закрытой системы

Такой фаерволл сейчас могут запихнуть прямо между соседними адресами, не надо больше выносить в отдельные VLAN и т.п. Поэтому сломаться может внезапно: ничего не меняли в конфигурации, но больше не работает :(

Вторая конфигурация которая может сломаться в такой ситуации – это приложения делающие пиннинг сертификатов. То есть софт проверяет не

только подпись сертификата доверяемого центра, но и явно его публичный ключ. Файервол подделать такой ключ не может и софт не работает

Если хотите улучшить показатели в Tinder или Snapchat, то именно пинниг сертификатов будет мешать

16

Wildcard сертификаты

- *.example.com
- Только на домены одного уровня: ✔ www.example.com и lab.example.com, ✕ uc.lab.example.com
- Усиливаются риски для приватного ключа
- Особо внимательно следить за расползанием и везде за сроком действия
- Можно через Shodan и Censys

Сертификаты выписываются на определенное DNS-имя. Имя это может быть шаблоном и тогда сертификат действителен для всех имён попадающих под эту маску, но одного уровня

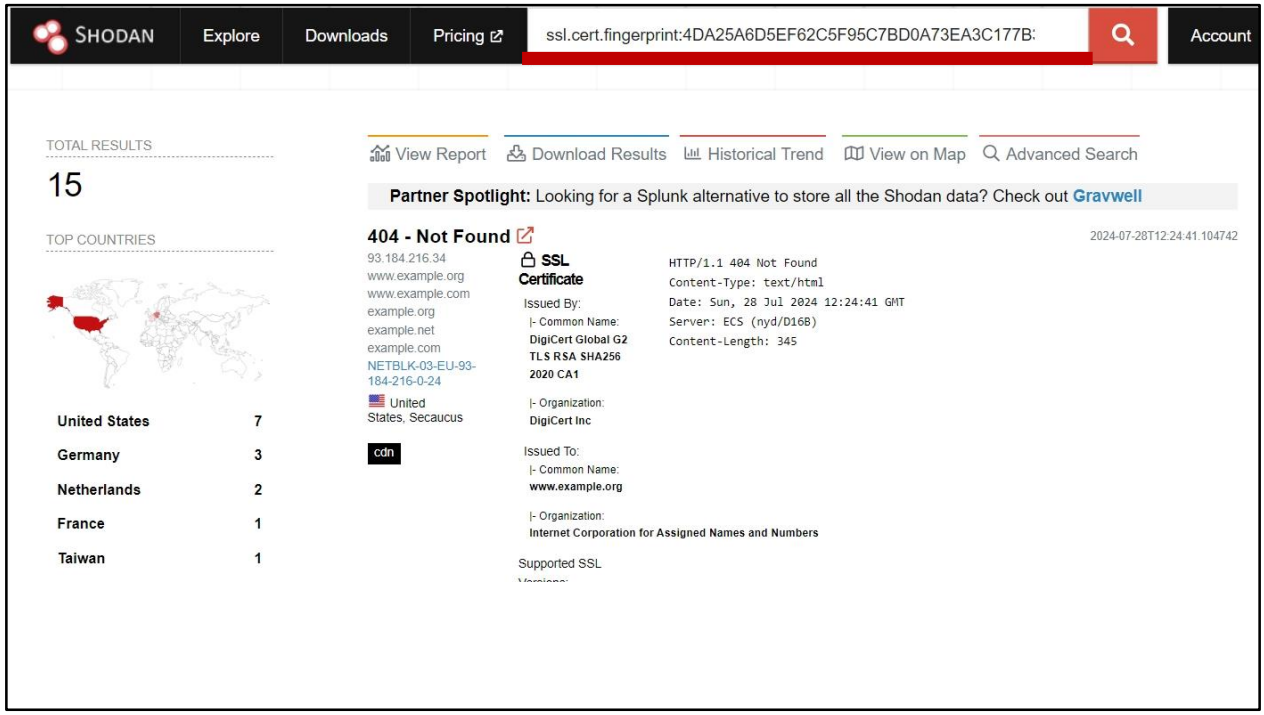
Если покупать, то за такой просят больше денег, но это дешевле чем несколько сертификатов на разные имена.

Не будет работать для поддоменов, как на слайде показано: lab.example.com верный, а вот для uc.lab.example.com – уже доверенным не будет

Усиливаются риски потери приватного ключа и надо внимательно следить куда вешаете эти сертификаты чтобы вовремя их заменять

Спецсайты по поиску могут помочь

ДЕМО shodan с ssl.cert.subject.cn:example.com и ssl.cert.fingerprint:X


SHODAN
Explore
Downloads
Pricing
ssl.cert.fingerprint:4DA25A6D5EF62C5F95C7BD0A73EA3C177B:
Account
TOTAL RESULTS
15
TOP COUNTRIES
United States 7
Germany 3
Netherlands 2
France 1
Taiwan 1
View Report
Download Results
Historical Trend
View on Map
Advanced Search
Partner Spotlight: Looking for a Splunk alternative to store all the Shodan data? Check out Gravwell
404 - Not Found
2024-07-28T12:24:41.104742
93.184.216.34
www.example.org
www.example.com
example.org
example.net
example.com
NETBLK-03-EU-93-184-216-0-24
United States, Secaucus
cdn
SSL Certificate
Issued By:
|- Common Name:
DigiCert Global G2 TLS RSA SHA256 2020 CA1
|- Organization:
DigiCert Inc
Issued To:
|- Common Name:
www.example.org
|- Organization:
Internet Corporation for Assigned Names and Numbers
Supported SSL
HTTP/1.1 404 Not Found
Content-Type: text/html
Date: Sun, 28 Jul 2024 12:24:41 GMT
Server: ECS (nyd/D16B)
Content-Length: 345

18

Бесплатные сертификаты

- Let’s Encrypt и аналоги
- Короткий срок действия, всего 90 дней
- Полный автомат
- Могут быть wildcard

Можно получать настоящие официальные большие работающие сертификаты бесплатно. Это сервис Let’s encrypt и его аналоги. Выдают они только на домены второго и третьего уровня. Только на 90 дней, поэтому необходимы автоматически работающие скрипты.

Сертификаты могут быть Wildcard

19

Форматы файлов с криптоматериалами

- DER
- PEM (*.crt,*.csr,*.key, etc)
- PFX/PKCS#12
- Java Keystore (JKS)
- и др

Когда создаёте ключ и получаете сертификаты они могут быть в разных форматах файлов
В самом исходном виде это DER, бинарный формат, состоит из наборов значений ASN.1, таких же как в SNMP.

PEM – это DER в текстовом виде через Base64. (ДЕМО) Бывает с приватным ключами, сертификатами, CSR, CRL и т.д.

PFX или PKCS12 – бинарный формат, популярен на Windows

JKS использовался с серверами на Java, утилита keytool. Последние годы переходят на PKCS12

Всякие другие бывают, от Oracle, в софте IBM и т.п.
Аналогия: примерно как архивы ZIP и RAR – внутри одно и тоже, но программы разные. Openssl и другие тулзы преобразуют эти файлы между собой. В интернете есть сайты обещающие такие конвертации – лучше их избегать поскольку это отправляет приватные ключи неизвестно куда и кому

Приватные ключи могут быть зашифрованы, но тогда их надо вводить вручную

при старте сервера

20

Хранилища приватных ключей

Приватные ключи – это основа всей этой TLS штуки и очень много чего другого. И чтобы их не стащили созданы специализированные устройства

Устройства эти не разрешают экспортировать приватные ключи, аппаратно противодействуют попыткам так сделать, ведут журналы аудита и тп.

Обычный компьютер создаёт файл с приватным ключом и его можно в дальнейшем копировать как угодно, куда угодно, шифровать на нём чего хочешь и сколько хочешь раз. Аппаратная штука умеет создавать ключи, но не даёт их из себя скопировать. Даже специальными методами: даже под электронным микроскопом не должно быть возможности вытащить приватный ключ

(1) HSM, здоровый ящик, ставится в серверный шкаф. Очень экзотичный, их сложно было завести в страну даже до санкций
(2) Мелкие ключи похожи на обычные флешки, но аппаратно реализуют криптографию: там где обычная флешка отрабатывает чтение и запись файлов, эта штука делает операции создать ключевую пару, подписать ключом и тд. Нет операции прочитать ключ
(3) Security Enclave в смартфонах, аппаратно выделенный сопроцессор,

отрабатывает криптографию, хранит пин-код и тп

(4) TPM на персональных компьютерах

и т.д.

https://support.apple.com/ru-ru/guide/security/sec59b0b31ff/web
https://www.infineon.com/dgdl/Infineon-OPTIGA+TPM+SLB+9672+FW16-DataSheet-v01_00-EN.pdf?fileId=8ac78c8c7f2a768a017f899f82094435

21

Версии TLS и криптоалгоритмы

1. Версии SSL 2.0, TLS1.0, TLS1.1, 1.2 и 1.3
2. Алгоритмы шифрования, цифровой подписи, хеш-функции, …

Проблема только для активных атак MITM

ECDHE-RSA-AES128-GCM-SHA256
TLS_AES_128_GCM_SHA256
ARIA256-GCM-SHA384
ECDHE-RSA-CHACHA20-POLY1305
[…]

Сейчас должно быть разрешены только TLS1.2 и 1.3
DES/3DES, RC4, MD5/SHA1 – точно запрещено

SSL и TLS довольно пожившие протоколы, есть несколько версий, не все из них являются безопасными.
Внутри протоколов ещё работают криптографические алгоритмы шифрования, цифровой подписи, хеширования и другого. Некоторые из них тоже недостаточно безопасны

Тут вы можете подумать, что особой беды в наличии устаревших алгоритмов нет – плохо зашифрованное соединение всяко лучше чем открытое, но работает это не так. Считается, что злоумышленники посередине канала связи могут даунгрейдить соединения между клиентом-сервером до небезопасных алгоритмов и затем эксплуатировать в них уязвимость. В целом, относительно остального, риск не выглядит великим, поэтому если явно не просят, то лучше ничего не трогать

Т.е. проблема это только для активных атак с MITM

Иногда заказчики хотят запретить небезопасные варианты алгоритмов и протоколов. Тогда залезаете в конфигурацию сервера и там вносите изменения

Вместе эти алгоритмы описываются вот такими строками в конфигурации.

Разобраться в этом неспециалисту нужды особой нет

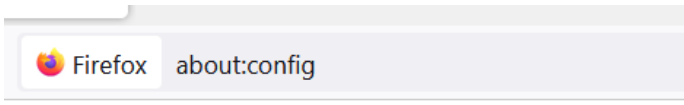
У клиента и сервера должна быть хоть одна общая реализация, иначе работать не будет

Сейчас должны быть разрешены только версии 1.2 и 1.3. Крипто-алгоритмы DES, RC4, MD5 с SHA1 точно запрещены. В принципе, современный софт так по дефолту и сделает

https://www.iana.org/assignments/tls-parameters/tls-parameters.xhtml#tls-parameters-4

22

Подключение к старым софтам



Firefox ESR

- security.tls.version.enable-deprecated: true
- security.tls.version.min: 1

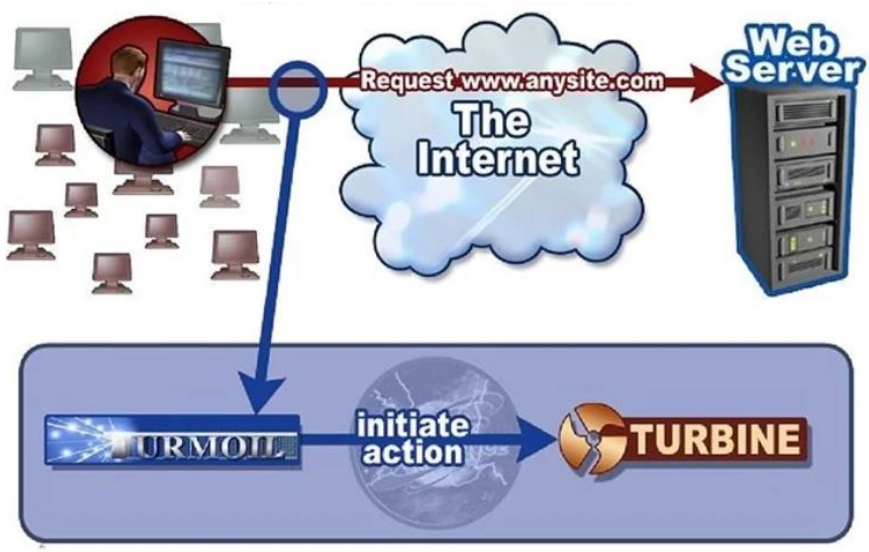
- софты типа stunnel, haproxy, nginx
- Риски старого ПО остаются

Иногда бывает надо с новых компьютеров и ПО подключится к оборудованию со старыми протоколами и шифрами. Браузеры это ограничивают по дефолту, но есть выход: Firefox, в адресной строке переходите на url по имени about:config, там два параметра меняете как на слайде написано

При необходимости подключения на постоянной основе, можно спереди поставить прокси софты типа stunnel, haproxy, nginx, но понимая, что старое ПО использовать рисково, там могут быть известные уязвимости

23

Quantum insert



https://blog.fox-it.com/2015/04/20/deep-dive-into-quantum-insert/

Почему в интернете повсеместно используется TLS для HTTP трафика? Казалось бы если зашли посмотреть погоду с новостями на веб-сайт, то риска особого нет: пароли не отправляются, важные данные не передаются.

Но существует американская система Quantum Insert. Их оборудование стояло посреди интернета, когда видело интересный трафик, подменяло ответы серверов и посылало эксплойты для взлома клиентских браузеров. Из браузеров пролезало дальше и тому подобное

https://blog.fox-it.com/2015/04/20/deep-dive-into-quantum-insert/

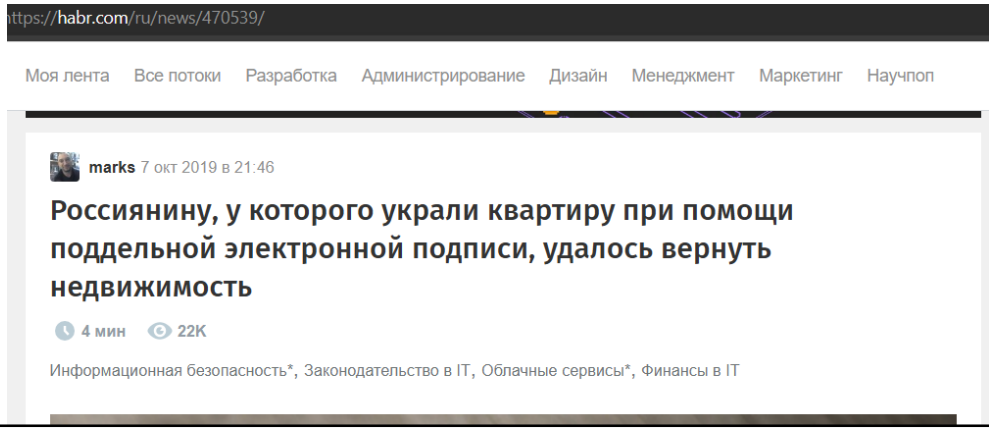
Общественности и IETF это не понравилось, они выпустили RFC8280 и пошло-поехало повсеместное шифрование в интернете

(На самом деле это конечно не так, просто Google сказал, если у кого не будет на сайте https, тех он в выдаче вниз подвинет)

24

X.509

- Сертификат X.509, не SSL
- Применяется также в подписи файлов, IPsec, электронной почте S/MIME, загрузчиках ОС, WiFi, OpenVPN и тп

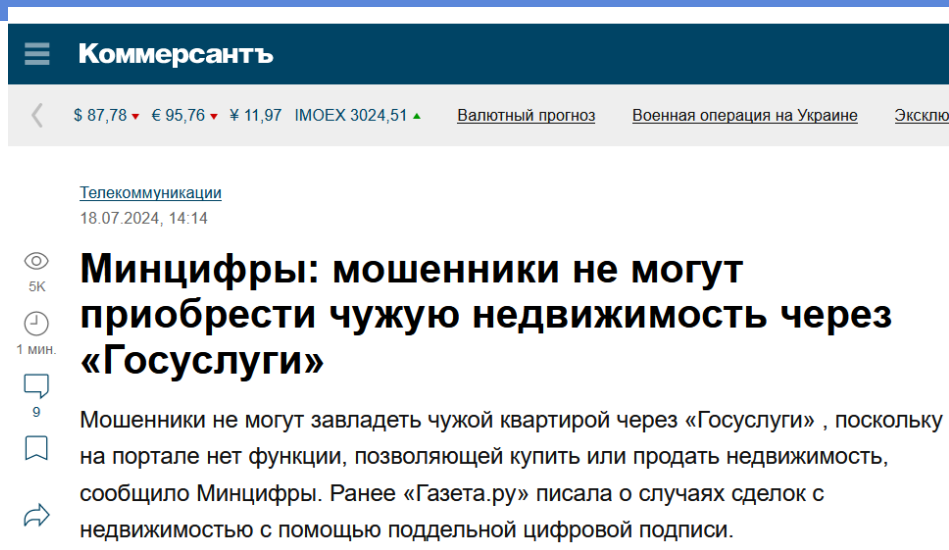


Сертификаты называются x.509 поскольку используются не только в TLS, а вообще повсеместно
В подписи файлов, электронной почте, загрузчиках операционных систем (jailbreak, SecureBoot), Wifi, openvpn и много где ещё
В том числе в России они используются в цифровых подписях для заключения сделок и за этим надо следить

Цифровая подпись на госуслугах не должна разрешить попятить наши квартиры
https://habr.com/ru/news/470539/

25

Следить за своими устройствами



Коммерсантъ

$ 87,78 € 95,76 ¥ 11,97 IMOEX 3024,51 Валютный прогноз Военная операция на Украине Эксклю

Телекоммуникации

18.07.2024, 14:14

5K

1 мин.

9

Минцифры: мошенники не могут приобрести чужую недвижимость через «Госуслуги»

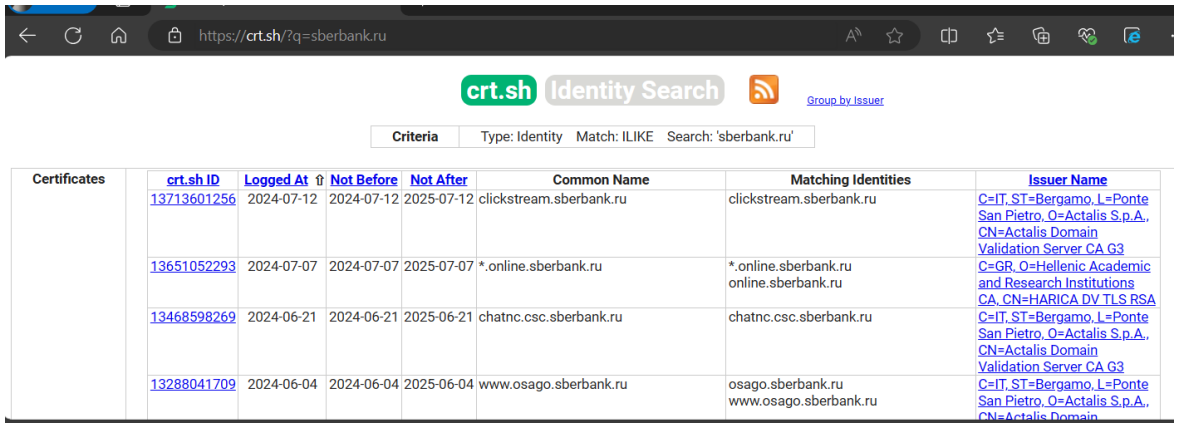
Мошенники не могут завладеть чужой квартирой через «Госуслуги» , поскольку на портале нет функции, позволяющей купить или продать недвижимость, сообщило Минцифры. Ранее «Газета.ру» писала о случаях сделок с недвижимостью с помощью поддельной цифровой подписи.

На прошлой неделе госорганы в очередной раз нам разъяснили, что такое невозможно

https://www.kommersant.ru/doc/6837954

26

Certificate transparency



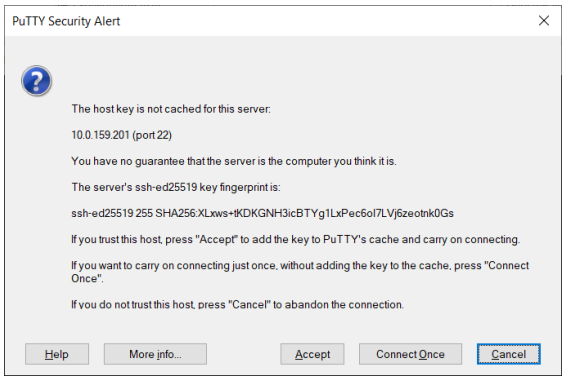
https://crt.sh/?q=sberbank.ru

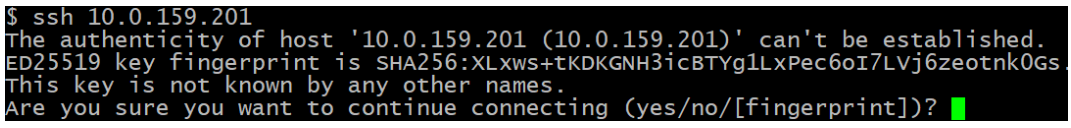
Все публичные CA отправляют выданные сертификаты в большой центральный лист и можно проверить какие выписаны для вашего домена. И других доменов тоже
Легко убедится что в sberbank.ru есть настоящие нормальные сертификаты

https://crt.sh/?q=sberbank.ru

27

SSH





Протокол SSH реализован на тех же криптопримитивах, что и TLS: криптография с открытыми ключами плюс симметричное шифрование, но не использует сертификаты. Как там реализована защита от MITM?

Очень просто: при первом подключении спрашивают у пользователя хочет ли доверять этому серверу, сохраняют публичный ключ в локальном списке и всё. Если ключ поменялся, то подключение по дефолту не разрешается

ДЕМО https://github.com/ssh-mitm/ssh-mitm

28

Контроль ключей сервера SSH

```
$ ls -l /etc/ssh/ssh_host_*_key*
-rw------- 1 root root  505 Jul  6  2022 /etc/ssh/ssh_host_ecdsa_key
-rw-r--r-- 1 root root  176 Jul  6  2022 /etc/ssh/ssh_host_ecdsa_key.pub
-rw------- 1 root root  411 Jul  6  2022 /etc/ssh/ssh_host_ed25519_key
-rw-r--r-- 1 root root   96 Jul  6  2022 /etc/ssh/ssh_host_ed25519_key.pub
-rw------- 1 root root 2602 Jul  6  2022 /etc/ssh/ssh_host_rsa_key
-rw-r--r-- 1 root root  568 Jul  6  2022 /etc/ssh/ssh_host_rsa_key.pub
```

```
$ ssh-keygen -l -f /etc/ssh/ssh_host_ed25519_key.pub
256 SHA256:wt8Hn/e+3HTLfuX/vJj9WEKQNDCZdlTYvmmNbrRKgWI root@hyperbole (ED25519)
```

```
$ ssh localhost
The authenticity of host 'localhost (::1)' can't be established.
ED25519 key fingerprint is SHA256:wt8Hn/e+3HTLfuX/vJj9WEKQNDCZdlTYvmmNbrRKgWI.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? ^C
```

1) Как проверить что подключаетесь к нужному серверу?
Его ключи лежат в файлах конфигурации
2) Команда как экране покажет значение SHA256 для публичного ключа из файла
3) При подключении к серверу сравниваете эти два значения, если они совпадают, то попали на нужное устройство

Когда заказываете виртуалки в облаке, то ключи SSH показывают в консоли управления.
Когда клонируете ВМ, то ключи лучше перегенерировать. И наоборот, если хотите пересетапить сервер, то копируете эти файлы на новый и никто ничего не заметит при следующем подключении

29

Варианты SSH

- Протокол: всегда версии 2
- Алгоритмы шифрования
- Хеш-функции, etc

Так же как и в TLS, в SSH за годы накопились разные варианты и шифронаборы, не все из которых считаются безопасными.
Точно как с TLS это проблема только для активных MITM атак с людьми посередине

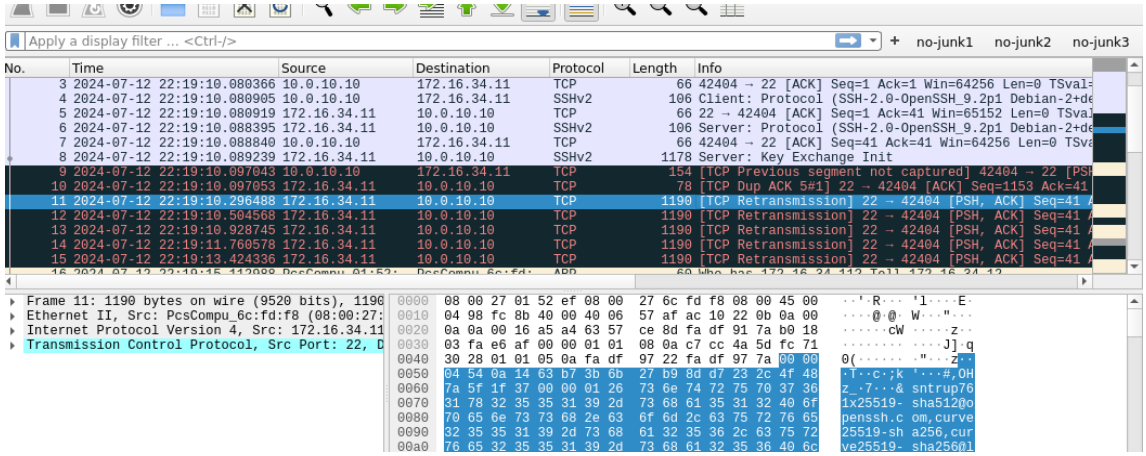
Протокол должен быть всегда 2, последние лет 20. Алгоритмы шифрования и прочее если вам укажут, то запретите лишние в конфиге

Аналогично может быть несовместимость между старыми серверами с новыми клиентами и наоборот. Тут надо будет или что-то где-то разрешить или же обновлять софт

Запрещать надо с осторожностью: можно довести до полной неработоспособности и снова придётся ездить по объектам с консольным кабелем

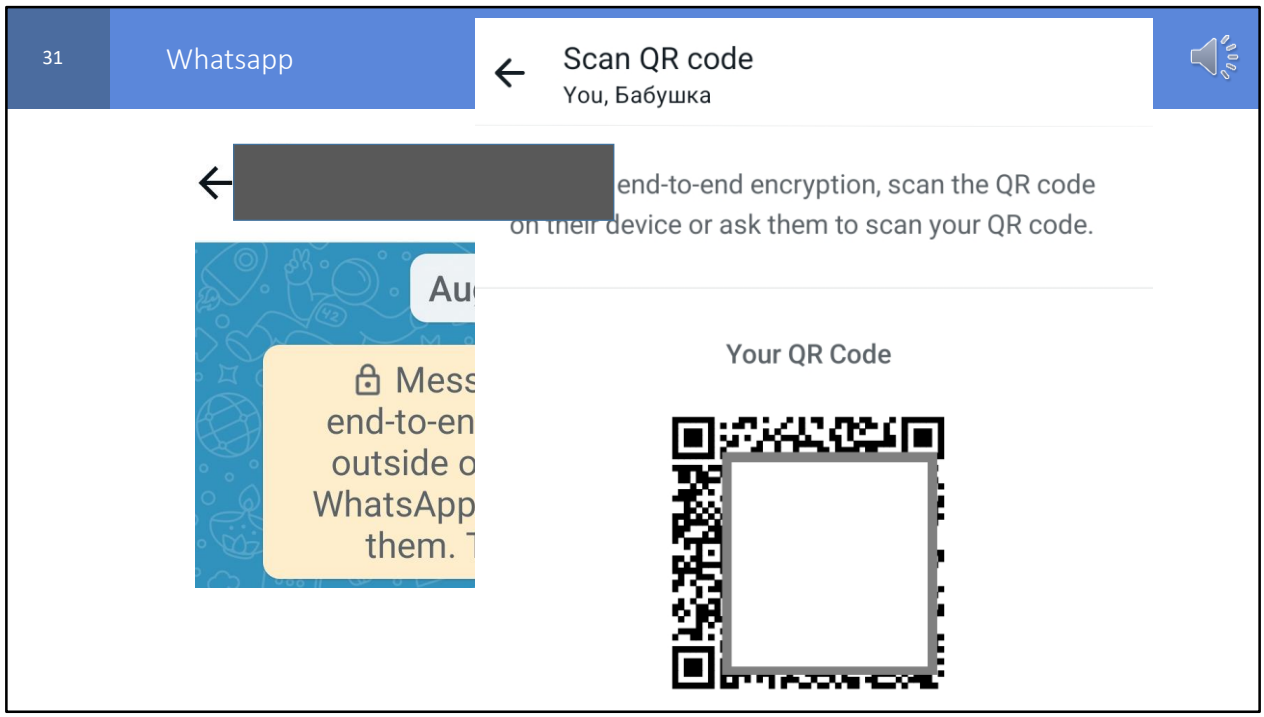
30

Path MTU discovery проблема



Иногда при подключении через VPN или ещё какой туннель может возникнуть ситуация: ping ходит, TCP-соединения устанавливаются, но подключения по SSH зависают. Это происходит из-за оверхеда туннелей и снижение максимальной длины пакетов в TCP. В такой ситуации надо или на клиенте уменьшить MTU, проверить что ICMP по пути не зафильтрован или включить функцию ограничения MSS на пограничных устройствах

С TLS может быть тоже самое

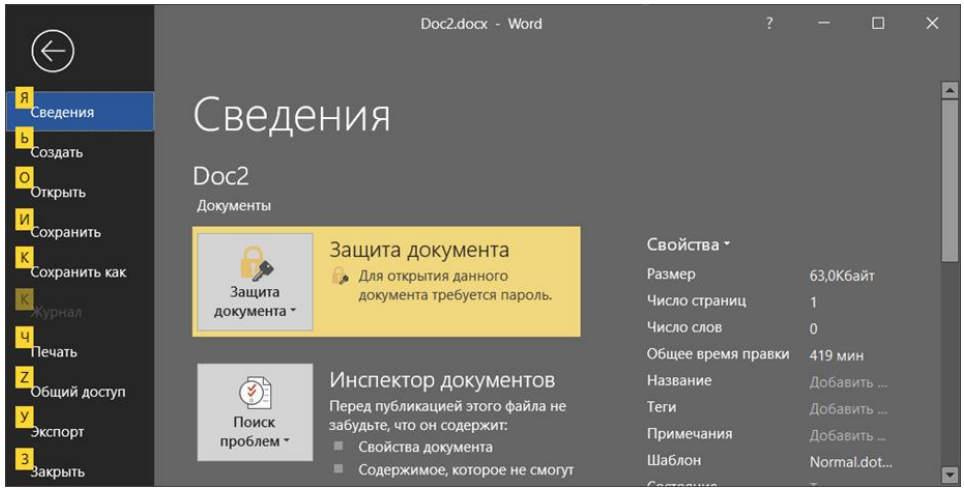


Такая же схема защиты от MITM как в SSH используется в Whatsapp. Точно также есть приватные/публичные ключи, скрытые где-то внутри устройства и при первом подключении предлагают доверять. При желании можно проверить значения ключей, но на более гламурном уровне чем в SSH: с помощью сканирования QR-кодов вместо ручного сравнения Base64

End-to-end шифрование тут значит, что все криптооперации выполняются внутри оконечных устройств, сервера видят только шифротекст и отправителей/получателей, но не имеют доступа к содержимому сообщений

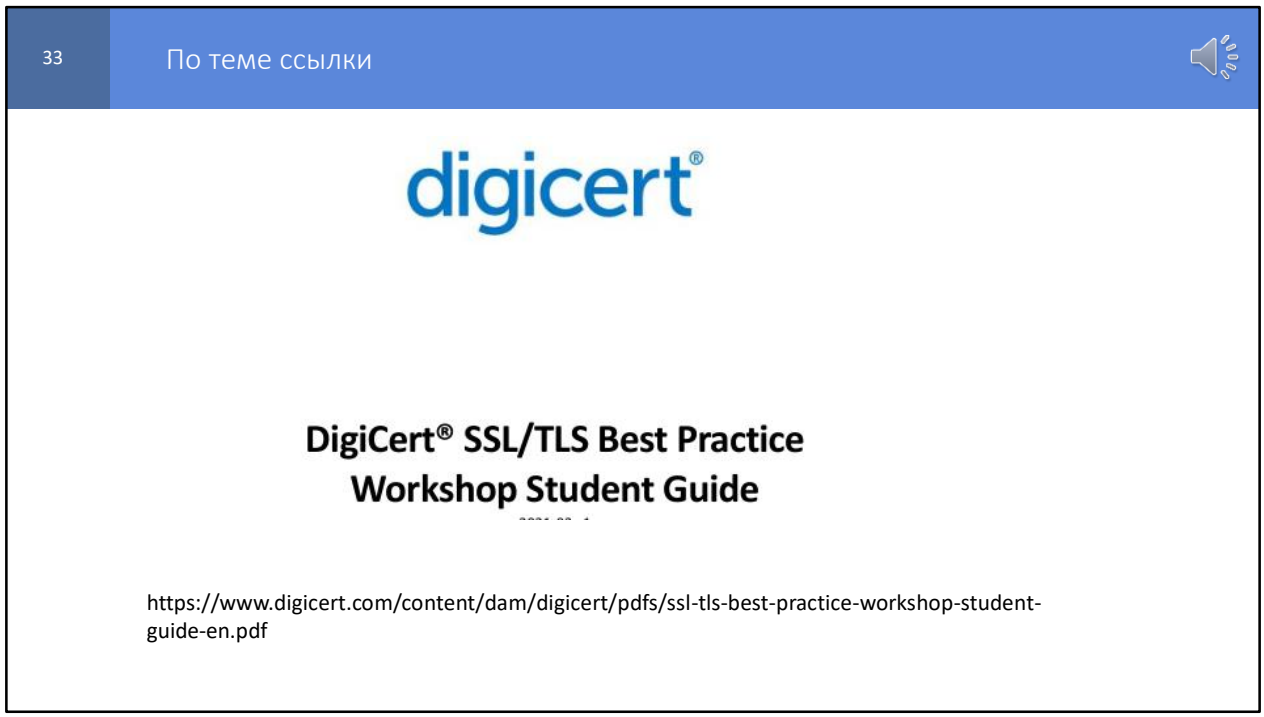
32 Отправить зашифрованные файлы

Файлы drag’n’drop в Word и сохранить с паролем



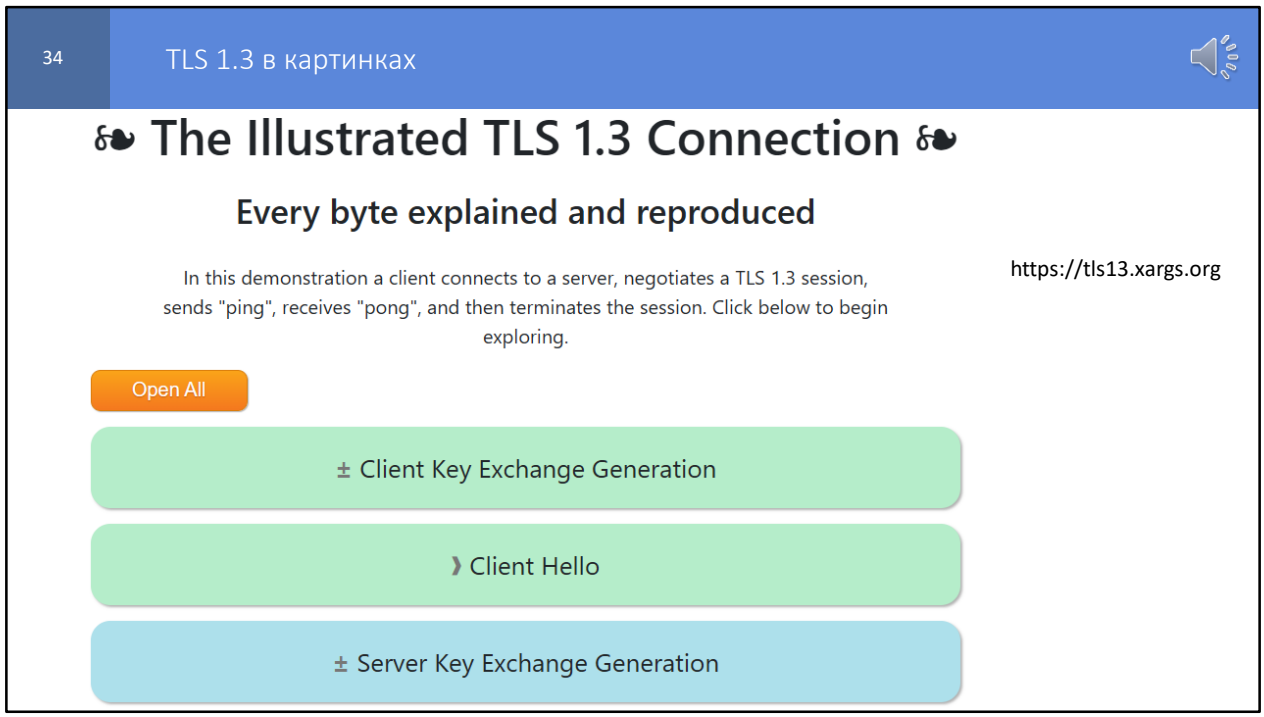
Иногда надо переправить к клиентам или от них конфиденциальные файлы, но почтовые фильтры не любят контент типа шифрованных архивов

Способ из офисного гетто: вставить файлы в MS Word, сохранить его с паролем и отправить. Шифрование в актуальных версиях достаточно надёжное, а фильтры в большинстве своём пропускают вордовые документы



https://www.digicert.com/content/dam/digicert/pdfs/ssl-tls-best-practice-workshop-student-guide-en.pdf

Учебник от центра сертификации. В больших подробностях рассказывает чем я вам. Но зато молчит про бесплатные сертификаты, хехе



https://tls13.xargs.org/
Сайт показывает протокол TLS в деталях

35

Бережно обращаться с приватными ключами

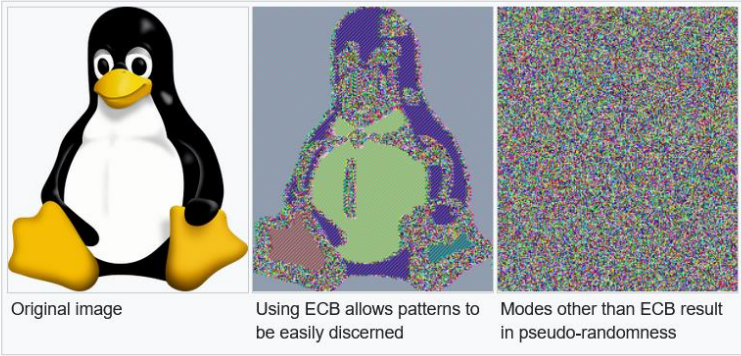
- Показал в социальной сети кусочек приватного ключа
- Многое отретушировал, но оставшегося хватило для получения полных данных

https://blog.cryptohack.org/twitter-secrets

Деятель во время пентеста выцепил приватный ключ, решил похвастаться в твиттер и запостил сильно отретушированную часть.
Общественность восстановила ключ полностью. Интересен рассказом о внутренностях ключа и текстом скриптов

36

Не использовать ECB



https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation#Electronic_codebook_(ECB)

Когда программируете чего-нибудь с шифрованием, то нельзя просто взять исходные данные, порезать на куски и пропустить через функцию симметричного шифра, типа AES или аналогов. Такая реализация раскрывает исходные данные в шифротексте и сам ключ шифрования.

На слайде очень наглядная картинка из википедии. Исходное изображение, после такого шифрования и как выглядит выход корректно реализованной криптосистемы

Надо действовать более хитро

37

Спасибо за внимание

- Вопросы
- Слайды
- Видео

Like & subscribe