**Приватность в TSL**

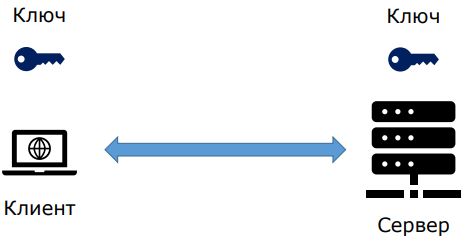
Приватность – это защита от пассивной атаки, когда злоумышленник подключается к каналу связи и читает все что передается по этому каналу. Чтобы защититься от атаки такого вида используется шифрование.

Шифрование бывает симметричное, и ассиметричное.

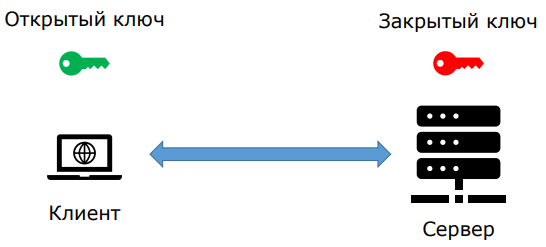
При установке соединения TLS выполняются две операции:

* Клиент и сервер договариваются о том наборе шифров TLS которые они будут использовать
* Обмен ключами для симметричного шифрования.

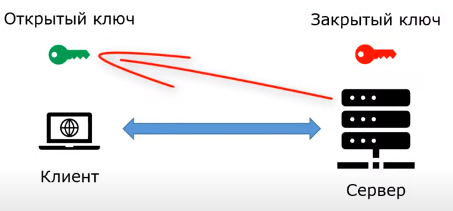
В симметричном шифровании, для того чтобы зашифровать и расшифровать данные используется один и тот же ключ. Он иногда называется разделяемый ключ. Этот ключ должен храниться в секрете. Ни клиент, ни сервер не должны никому этот ключ показывать. Если злоумышленник получит этот ключ, он сможет расшифровать все данные.



Другой подход к шифрованию используется в ассиметричном шифровании. Здесь используются 2 ключа – открытый и закрытый. Открытый ключ может свободно передаваться по каналам связи, а в тайне нужно хранить только закрытый.



В нашем случае закрытый ключ хранится на сервере, а открытый ключ передается клиенту.



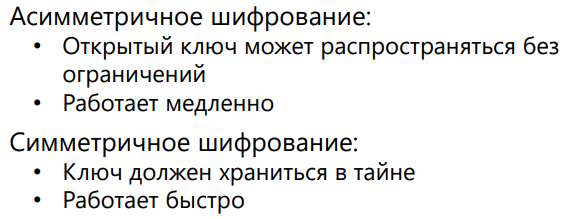
Для шифрования данных используется открытый ключ, а расшифровать данные, которые зашифрованы открытым ключом, можно только с помощью закрытого ключа.

**Симметричное vs Асимметричное шифрование**

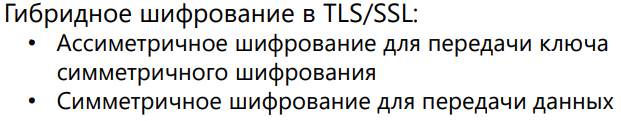
Нельзя сказать, что один вид шифрования лучше, чем другой.

Ассиметрично шифрование более продвинутый алгоритм, там используются 2 ключа, и открытый ключ спокойно можно передавать по открытым каналам, но недостаток в том, что ассиметричное шифрование работает очень медленно, в несколько раз медленнее, чем симметричное.

Симметричное шифрование работает быстрее, но ключ который используется для этого шифрования должен храниться в тайне – его нельзя передавать по открытым каналам связи, по этому есть проблема, как сделать так, чтобы клиент и сервер имели одинаковый ключ



В TLS/SSL, как и во многих других сетевых протоколах используется гибридный подход, который сочетает в себе достоинства и того, и другого типа шифрования.



Ассиметричное шифрование используется на начальном этапе, для того чтобы клиент и сервер могли передать друг другу ключи для симметричного шифрования. А после того, как клиент и сервер с помощью ассиметричного шифрования передали друг другу ключи, они могут использовать для передачи данных симметричное шифрование, которое работает гораздо быстрее, и меньше создает вычислительную нагрузку на процессоры устройств.

**Алгоритм обмена ключами RSA**

Набор шифров TLS (cipher suite):

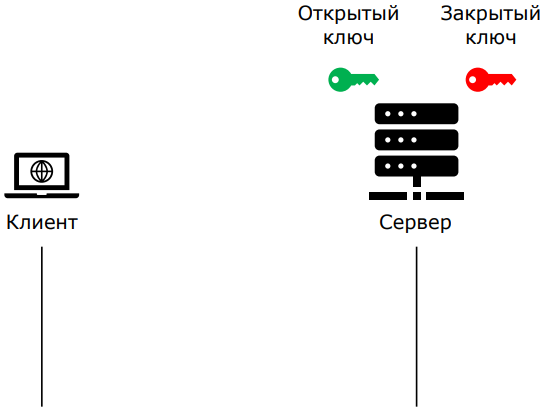
• Алгоритм обмена ключами (RSA, DH)

• Алгоритм симметричного шифрования (AES, 3DES и др.)

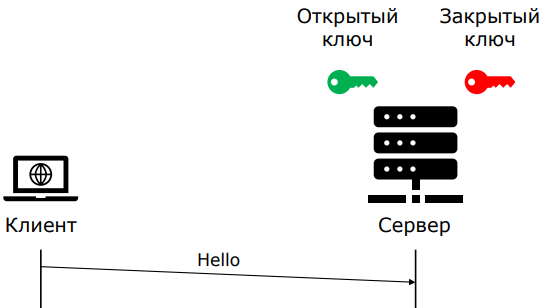
Эти алгоритмы используются в протоколе TLS при установке соединения.

Протокол RSA – протокол ассиметричного шифрования.

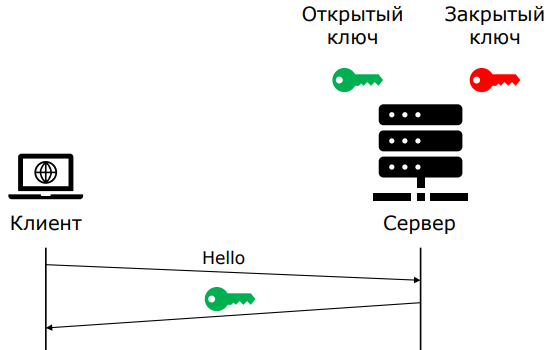
У сервера есть 2 ключа – открытый и закрытый.



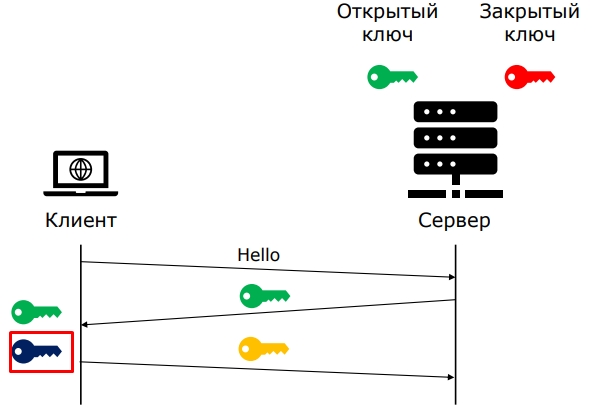
Клиент подключается к серверу, и передает сообщение «Hello», которое говорит о том, что клиент хочет установить защищенное соединение с сервером.



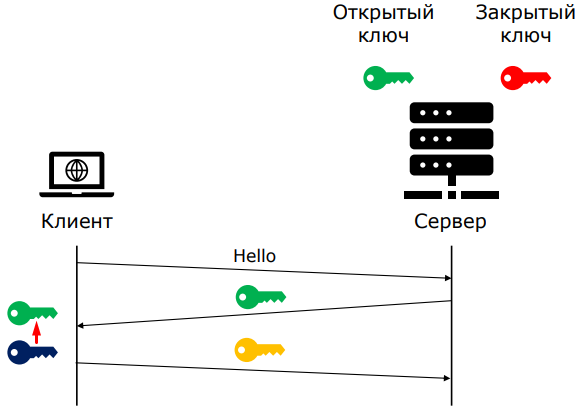
В ответ сервер берет открытый ключ и передает его клиенту. Открытый ключ можно спокойно передавать по незащищенным каналам связи.



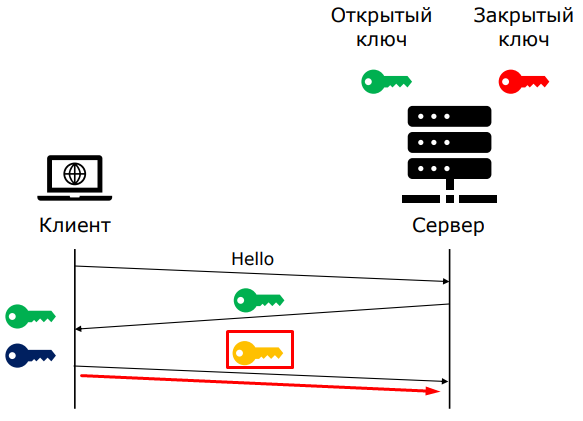
Клиент получает этот открытый ключ, генерирует ключ, который будет применяться для симметричного шифрования.



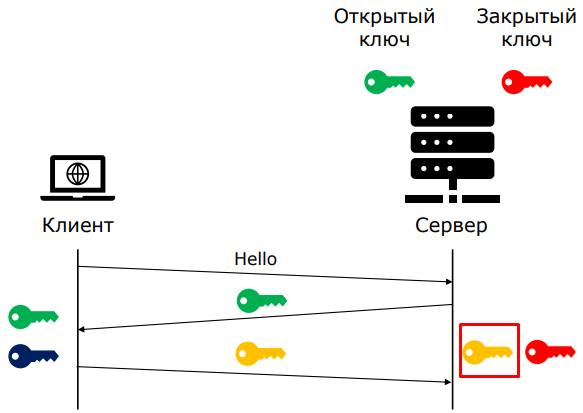
Зашифровывает это ключ с помощью открытого ключа, который он получил от сервера.



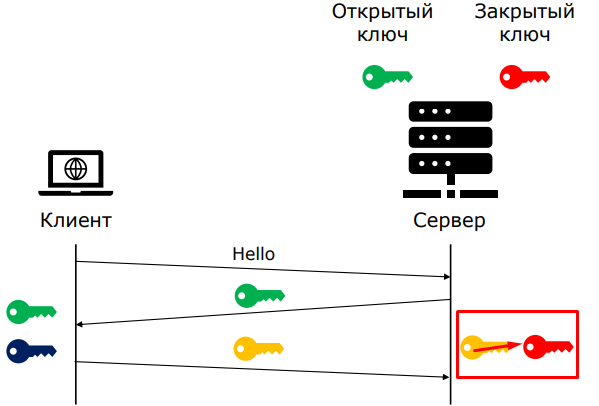
И, зашифрованным, в таком виде ключ, передает на сервер.



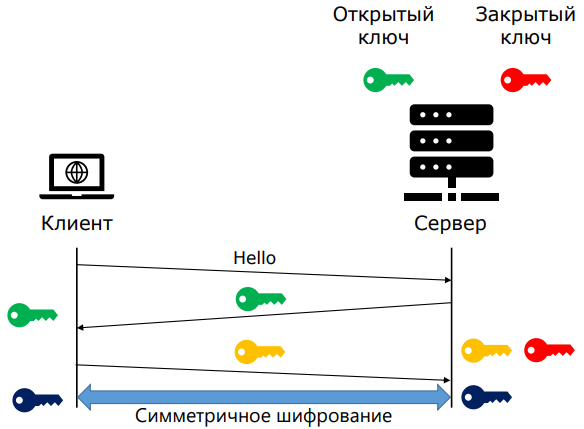
Сервер получает этот ключ.



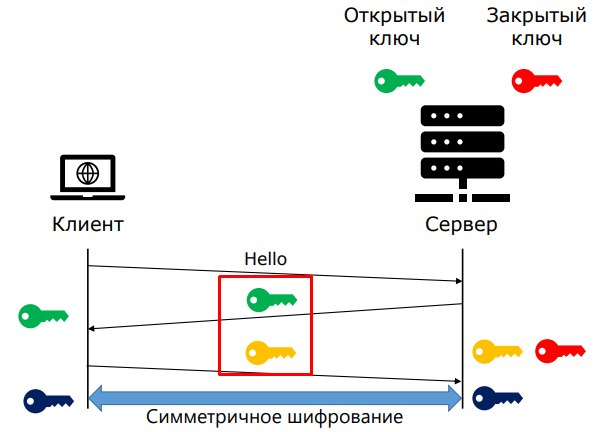
Сервер использует закрытый ключ, доступ к которому есть только у этого сервера для того, чтобы расшифровать ключ



И таким образом получает разделяемый ключ для симметричного шифрования.



После этого и клиент, и сервер имеют один и тот же ключ, и могут использовать симметричное шифрование. При этом, ключ для симметричного шифрования в открытом виде через каналы связи не передается.



Недостатки алгоритма обмена ключами RSA

Не обеспечивается совершенная прямая секретность (Perfect forward secrecy):

• Доступ к закрытому ключу сервера позволит

расшифровать все передаваемые данные

Уязвимости RSA:

• Атака миллиона сообщений (million message attack,

Bleichenbacher, 1998)

• ROBOT (Return Of Bleichenbacher's Oracle Threat, 2017)

Алгоритм обмена ключами RSA запрещено использовать начиная с TLS 1.3

Алгоритм обмена ключами Диффи-Хеллмана

Условия работы алгоритма Диффи-Хеллмана:

• p – большое простое число, минимум 1024 бита

• g – первообразный корень по модулю p, небольшое

целое число

• Невозможно вычислить ключ даже на современных

суперкомпьютерах

Совершенная прямая секретность:

• Невозможно раскодировать зашифрованные данные, даже если есть доступ к серверу

Более совершенный вариант алгоритма:

• Диффи-Хеллман на эллиптических кривых