Механизм шифрования с открытым и закрытым ключом, используемый в протоколе HTTPS с использованием сертификата SSL/TLS, основан на криптографической системе с открытым ключом.

1. Генерация ключей: Сервер, желающий установить защищенное соединение, создает пару ключей — открытый и закрытый. Закрытый ключ остается в секрете на сервере, а открытый ключ будет распространяться.

2. Получение сертификата: Сервер обращается к надежному стороннему удостоверяющему центру (CA) и запрашивает SSL-сертификат. Сертификат содержит публичный ключ сервера и информацию о его владельце. CA выпускает сертификат, подписывая его собственным закрытым ключом, что подтверждает его подлинность.

3. Распространение сертификата: Сервер отправляет сертификат клиенту вместе с открытым ключом.

4. Установление соединения: Клиент, получив сертификат, проверяет его подлинность, используя публичный ключ CA, который предварительно встроен в браузер или операционную систему. Если сертификат является доверенным, клиент генерирует случайный сеансовый ключ и зашифровывает его с помощью открытого ключа сервера.

5. Расшифровка сеансового ключа: Сервер, получив зашифрованный сеансовый ключ, расшифровывает его с помощью своего закрытого ключа.

6. Шифрование данных: Далее, сервер и клиент используют общий сеансовый ключ для шифрования и дешифрования данных, передаваемых между ними во время сеанса HTTPS. Это обеспечивает конфиденциальность и целостность передаваемой информации.

Таким образом, применение открытого и закрытого ключей в сочетании с сертификатами SSL/TLS обеспечивает безопасность соединения и защиту данных в протоколе HTTPS.

Вот как происходит процесс проверки сертификата сервера:

Когда вы пытаетесь установить защищенное соединение с сервером (через HTTPS), сервер отправляет свой сертификат клиенту.

Браузер (или другой клиентский программный продукт) проверяет подлинность сертификата. Он сравнивает данные в сертификате с корневыми сертификатами, которыми он доверяет. Если сертификат подписан корневым сертификатом, включенным в список доверенных, и данные в сертификате совпадают с данными сервера, то сертификат считается доверенным.

Если сертификат признан доверенным, браузер генерирует случайный сеансовый ключ, шифрует его с использованием публичного ключа из сертификата сервера и отправляет его серверу.

Сервер расшифровывает сеансовый ключ с помощью своего закрытого ключа и теперь у сервера и клиента есть общий сеансовый ключ, который будет

использоваться для шифрования и дешифрования данных во время текущей сессии.