Шифры с асимметричным ключом

Криптосистемы с асимметричным ключом / криптосистемы с открытым ключом (такие как RSA, криптография на эллиптической кривой (ECC), Диффи-Хеллман, Элгамал, Макэлис, NTRU и другие) используют пару математически связанных ключей: открытый ключ (ключ шифрования) и закрытый ключ (ключ дешифрования).

Криптосистемы с асимметричным ключом обеспечивают генерацию пары ключей (закрытый + открытый ключ), алгоритмы шифрования (шифры с асимметричным ключом и схемы шифрования, такие как RSA-OAEP и ECIES), алгоритмы цифровой подписи (такие как DSA, ECDSA и EdDSA) и алгоритмы обмена ключами (такие как DHKE и ECDH).

Сообщение, зашифрованное открытым ключом, позже расшифровывается с помощью закрытого ключа. Сообщение, подписанное закрытым ключом, позже проверяется с помощью открытого ключа. Открытый ключ обычно доступен всем, в то время как закрытый ключ хранится в секрете. Вычисление закрытого ключа из соответствующего ему открытого ключа по своей конструкции неосуществимо с точки зрения вычислений.

Криптосистемы с открытым ключом

Хорошо известными криптосистемами с открытым ключом являются: RSA, ECC, ElGamal, DHKE, ECDH, DSA, ECDSA, EdDSA, подписи Шнорра. Различные криптосистемы с открытым ключом могут предоставлять одну или несколько из следующих возможностей:

Генерация пары ключей: генерируйте случайные пары закрытого ключа + соответствующий открытый ключ.

Шифрование / дешифрование: шифрование даты с помощью открытого ключа и дешифрование данных с помощью закрытого ключа (часто с использованием гибридной схемы шифрования).

Цифровые подписи (аутентификация сообщений): подписывают сообщения закрытым ключом и проверяют подписи открытым ключом.

Алгоритмы обмена ключами: безопасный обмен криптографическим ключом между двумя сторонами по небезопасному каналу.

Наиболее важными и наиболее часто используемыми криптосистемами с открытым ключом являются RSA и ECC. Криптография на эллиптической кривой (ECC) является рекомендуемой и наиболее предпочтительной современной криптосистемой с открытым ключом, особенно с современными высокооптимизированными и безопасными кривыми (такими как Curve25519 и Curve448) из-за меньших размеров ключей, более коротких подписей и лучшей производительности.

Криптосистема с открытым ключом RSA основана на математической концепции модульного возведения в степень (числа, возведенные в степень по модулю), наряду с некоторыми математическими конструкциями и задачей целочисленной факторизации (которая считается вычислительно невыполнимой для достаточно больших ключей).

Криптосистема криптографии на эллиптических кривых (ECC) основана на математике алгебраической структуры эллиптических кривых над конечными полями и задаче дискретного логарифмирования на эллиптических кривых (ECDLP), которая считается вычислительно невыполнимой для больших ключей. ECC поставляется вместе с алгоритмом ECDSA (алгоритм цифровой подписи на эллиптической кривой). ECC использует меньшие ключи и подписи, чем RSA, и предпочтительнее в большинстве современных приложений. Позже мы обсудим ECC и ECDSA более подробно, наряду с примерами.

Большинство криптосистем с открытым ключом (таких как RSA, ECC, DSA, ECDSA и EdDSA) поддаются квантовому взлому (квантово-небезопасны), что означает, что (по крайней мере, теоретически) достаточно мощный квантовый компьютер сможет взломать их защиту и вычислить закрытый ключ из заданного открытого ключа за считанные секунды.

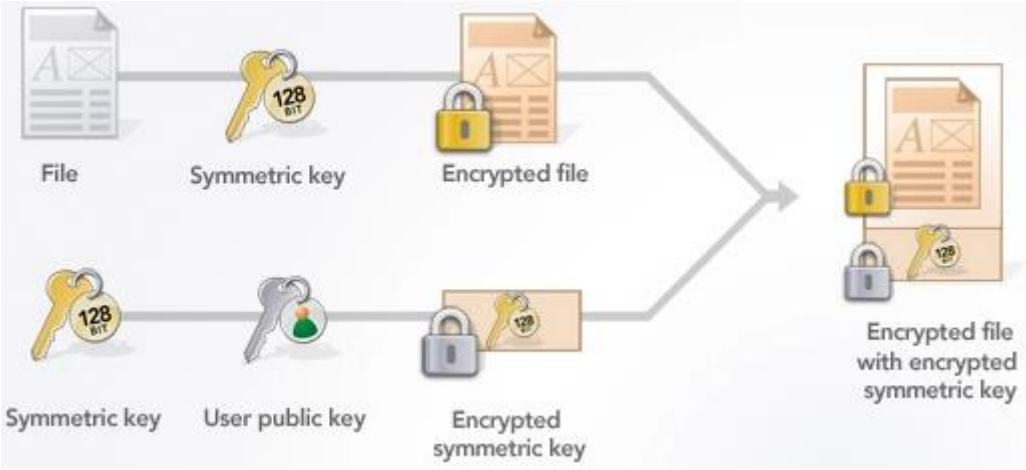
Асимметричные схемы шифрования

Асимметричное шифрование сложнее симметричного не только потому, что в нем используются открытый и закрытый ключи, но и потому, что асимметричное шифрование может шифровать / расшифровывать только небольшие сообщения, которые должны быть сопоставлены с математикой, лежащей в основе криптосистемы с открытым ключом. Некоторые криптосистемы (например, ECC) не предоставляют непосредственных примитивов шифрования, поэтому следует использовать более сложные схемы.

В системе RSA входное сообщение должно быть преобразовано в большое целое число (например, с использованием заполнения OAEP), в то время как в ECC сообщение не может быть зашифровано напрямую и используется более сложная схема шифрования, основанная на обмене ключами Диффи-Хеллмана с эллиптической кривой (ECDH). Это будет подробно объяснено позже в этой главе. Кроме того, асимметричные шифры значительно медленнее симметричных (например, шифрование RSA в 1000 раз медленнее, чем AES).

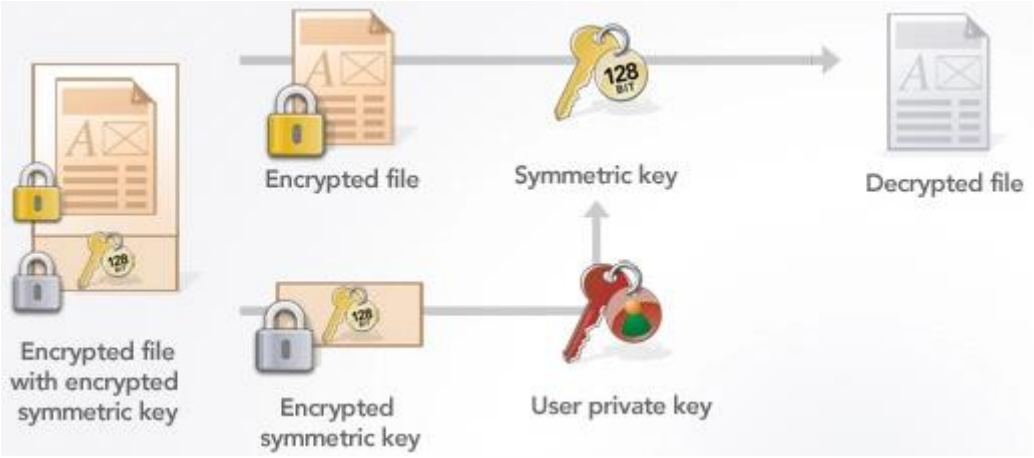
Чтобы преодолеть вышеуказанные ограничения и разрешить шифрование сообщений любого размера, современная криптография использует асимметричное шифрование схем (также известный как шифрование с открытым ключом схемы / асимметричное шифрование сооружений / гибридных схем шифрования), как ключевые механизмы инкапсуляции (Кэм) и комплексной зашифрованные схемы, которые совмещают асимметричного шифрования с симметричным ключом шифра.

Вот как можно зашифровать большой документ или файл, комбинируя криптографию с открытым ключом и симметричный криптоалгоритм:



На приведенной выше диаграмме зашифрованный симметричный ключ известен как KEM block (инкапсулированный ключ с шифрованием открытым ключом), а зашифрованный файл данных известен как DEM block (инкапсулированные данные с симметричным шифрованием). Зашифрованное сообщение состоит из этих двух блоков вместе (инкапсулированный ключ + инкапсулированные данные).

Это соответствующий процесс дешифрования (расшифруйте зашифрованный большой документ, используя криптографию с открытым ключом и симметричный криптоалгоритм):



Примерами таких асимметричных схем шифрования являются: RSA-OAEP, RSA-KEM и ECIES-KEM.

Интегрированные схемы шифрования

Интегрированные схемы шифрования - это современные схемы шифрования с открытым ключом, которые сочетают симметричные шифры, асимметричные шифры и алгоритмы получения ключа для обеспечения безопасного шифрования на основе открытого ключа (PKE). В схеме EIS асимметричные алгоритмы (такие как RSA или ECC) используются для шифрования или инкапсуляции симметричного ключа, используемого позже симметричными шифрами (такими как AES или ChaCha20) для шифрования входного сообщения. Некоторые схемы EIS обеспечивают также аутентификацию сообщений. Примерами схем EIS являются DLIES (интегрированная схема шифрования с дискретным логарифмом) и ECIES (интегрированная схема шифрования с эллиптической кривой).

Механизмы инкапсуляции ключей (KEMS)

Механизмы инкапсуляции ключей (KEM) представляют собой асимметричные криптографические методы, используемые для шифрования и инкапсуляции секретного ключа (называемого "эфемерный симметричный ключ"), который используется для шифрования входного сообщения с использованием симметричного криптографического шифра. KEM инкапсулирует эфемерный симметричный ключ шифрования как часть зашифрованного сообщения, шифруя его открытым ключом получателя. В криптографии этот процесс известен как "инкапсуляция ключа".

Выходные данные гибридной схемы шифрования на основе KEM состоят из блока KEM, содержащего инкапсулированный зашифрованный симметричный ключ (или определенные параметры, используемые для его получения), и блока DEM (механизм инкапсуляции данных), содержащего инкапсулированные симметрично зашифрованные данные (параметры шифрования + зашифрованный текст + необязательно тег аутентификации).

Механизмы инкапсуляции ключей (KEMS) используются в схемах гибридного шифрования и в схемах интегрированного шифрования, где случайный элемент генерируется в базовой криптосистеме с открытым ключом, а симметричный ключ получается из этого случайного элемента путем хеширования. Такой подход упрощает процесс комбинирования асимметричного и симметричного шифрования. Примерами современных механизмов инкапсуляции ключей являются: RSA-KEM, ECIES-KEM и PSEC-KEM.

Инкапсуляцию ключа не следует путать с переносом ключа.

инкапсуляция ключа (KEM) относится к шифрованию с открытым ключом другого ключа (симметричного или асимметричного). Используется для создания доказуемо безопасных гибридных схем шифрования, например, для шифрования секретного ключа AES с помощью заданного открытого ключа ECC.

Перенос ключа относится к шифрованию с использованием симметричного ключа другого ключа (который может быть как симметричным, так и асимметричным ключом). Используется для шифрования, защиты целостности и транспортировки криптографических ключей. Упаковка ключей обеспечивает конфиденциальность и защиту целостности специализированных данных, таких как криптографические ключи, без использования одноразовых номеров. Подробности см. в RFC 3394.

Цифровые подписи

В криптографии цифровые подписи обеспечивают аутентификацию сообщений, целостность и неотказуемость цифровых документов. Цифровые подписи работают в криптосистемах с открытым ключом и используют пары открытый / закрытый ключи. Подписание сообщения выполняется с помощью закрытого ключа, а проверка сообщения выполняется с помощью соответствующего открытого ключа.

Подпись сообщения математически гарантирует, что определенное сообщение было подписано определенным (секретным) закрытым ключом, который соответствует определенному (несекретному) открытому ключу. После подписания сообщения сообщение и подпись не могут быть изменены, и, таким образом, обеспечивается аутентификация сообщения и целостность. Любой, кто знает открытый ключ подписавшего сообщение, может проверить подпись. После подписания автор подписи не может отказаться от акта подписания (это известно как не отрицание).

Цифровые подписи сегодня широко используются для подписания цифровых контрактов, авторизации банковских платежей и подписания транзакций в общедоступных системах блокчейн для передачи цифровых активов.

Большинство криптосистем с открытым ключом, таких как RSA и ECC, обеспечивают защищенные схемы цифровой подписи, такие как DSA, ECDSA и EdDSA. Мы обсудим цифровые подписи более подробно позже в этом разделе.

Алгоритмы обмена ключами

В криптографии алгоритмы обмена ключами (протоколы согласования ключей / схемы согласования ключей) позволяют обмениваться криптографическими ключами между двумя сторонами, позволяя использовать криптографический алгоритм, в большинстве случаев симметричный шифр шифрования. Например, когда ноутбук подключается к домашнему маршрутизатору Wi-Fi, обе стороны договариваются о сеансовом ключе, используемом для симметричного шифрования сетевого трафика между ними.

Большинство алгоритмов обмена ключами основаны на криптографии с открытым ключом и математике, лежащей в основе этой системы: дискретных логарифмах, эллиптических кривых или других.

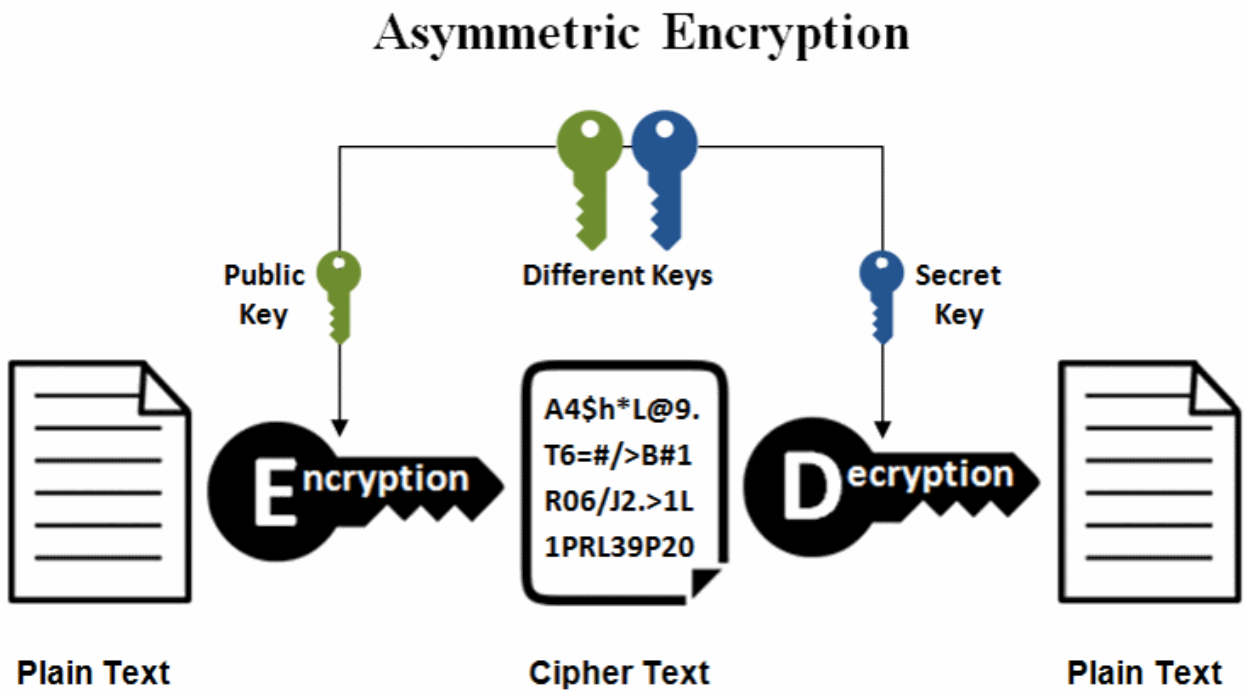
Анонимный обмен ключами, подобный обмену Диффи–Хеллмана (DHKE и ECDH), не обеспечивает аутентификацию сторон и, таким образом, уязвим для атак "человек посередине", но безопасен от атак "перехвата трафика" (прослушивания)...."....."....."....."

Схемы соглашения о аутентифицированном ключе аутентифицируют личности сторон, участвующих в обмене ключами, и, таким образом, предотвращают атаки "человек посередине" с использованием ключей с цифровой подписью (например, сертификата PKI), соглашения о ключе с аутентификацией паролем или другого метода.

Шифрование / дешифрование ECC

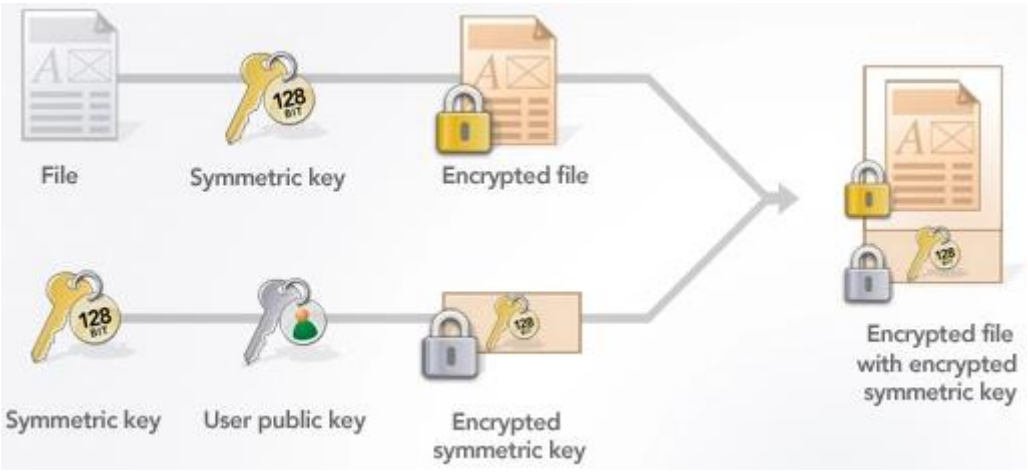
В этом разделе мы объясним, как реализовать шифрование / дешифрование с открытым ключом на основе эллиптической кривой (асимметричная схема шифрования на основе ECC). Это нетривиально и обычно включает в себя разработку гибридной схемы шифрования, включающей криптографию ECC, обмен ключами ECDH и алгоритм симметричного шифрования.

Предположим, у нас есть пара ECC private-public key. Мы хотим зашифровать и расшифровать данные, используя эти ключи. По определению, асимметричное шифрование работает следующим образом: если мы зашифруем данные с помощью закрытого ключа, мы сможем позже расшифровать зашифрованный текст с помощью соответствующего открытого ключа:

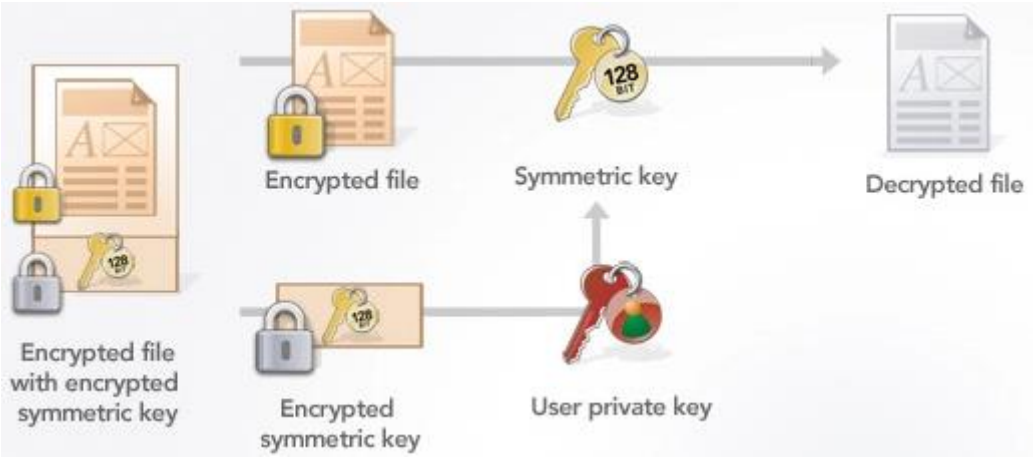


Описанный выше процесс может быть непосредственно применен к криптосистеме RSA, но не к ECC. Криптография на эллиптической кривой (ECC) не предоставляет метод шифрования напрямую. Вместо этого мы можем разработать гибридную схему шифрования, используя схему обмена ключами ECDH (эллиптическая кривая Диффи–Хеллмана) для получения общего секретного ключа для симметричного шифрования и дешифрования данных.

Так работает большинство гибридных схем шифрования (процесс шифрования).:



Так работает большинство гибридных схем шифрования (процесс дешифрования):



Давайте подробнее рассмотрим, как разработать и реализовать схему гибридного шифрования на основе ECC.

Вывод секретного ключа на основе ECC (с использованием ECDH)

Предположим, что у нас есть криптографическая эллиптическая кривая над конечным полем вместе с ее образующей точкой G. Мы можем использовать следующие две функции для вычисления совместно используемого секретного ключа для шифрования и дешифрования (полученные из схемы ECDH):

calculateEncryptionKey(pubKey) --> (sharedECCKey, ciphertextPubKey)

Сгенерируйте ciphertextPrivKey = новый случайный закрытый ключ.

Вычислить ciphertextPubKey = ciphertextPrivKey * G.

Вычислите общий секрет ECDH: sharedECCKey = pubKey * ciphertextPrivKey.

Верните оба sharedECCKey + ciphertextPubKey. Используйте sharedECCKey для симметричного шифрования. Используйте случайно сгенерированный ciphertextPubKey для последующего вычисления ключа дешифрования.

Вычисляемый ключ шифрования(privKey, ciphertextPubKey) --> sharedecckey's

Вычислите общий секрет ECDH: sharedECCKey = ciphertextPubKey * privKey.

Верните sharedECCKey и используйте его для расшифровки.

В приведенных выше вычислениях используется та же математика, что и в алгоритме ECDH (см. предыдущий раздел). Напомним, что точки EC обладают следующим свойством:

(a * G) * b = (b * G) * a

Теперь предположим, что a = privKey, a * G = pubKey, b = ciphertextPrivKey, b * G = ciphertextPubKey.

Приведенное выше уравнение принимает следующий вид:

Общедоступный ключ * ciphertextPrivKey = ciphertextPubKey * privKey = sharedECCKey

Именно это вычисляют две вышеупомянутые функции, непосредственно следуя схеме соглашения о ключах ECDH. В гибридных схемах шифрования инкапсулированный ciphertextPubKey также известен как "эфемерный ключ", поскольку он используется временно для получения симметричного ключа шифрования с использованием схемы согласования ключей ECDH.

Обмен ключами ECDH

ECDH (обмен ключами Диффи–Хеллмана с эллиптической кривой) - это схема анонимного соглашения о ключах, которая позволяет двум сторонам, каждая из которых имеет пару открытых и закрытых ключей с эллиптической кривой, устанавливать общий секрет по небезопасному каналу. ECDH очень похож на классический алгоритм DHKE (обмен ключами Диффи–Хеллмана), но он использует точечное умножение ECC вместо модульного возведения в степень. ECDH основан на следующем свойстве точек EC:

(a * G) * b = (b * G) * a

Если у нас есть два секретных номера a и b (два закрытых ключа, принадлежащих Алисе и Бобу) и эллиптическая кривая ECC с точкой генератора G, мы можем обмениваться по небезопасному каналу значениями (a * G) и (b * G) (открытые ключи Алисы и Боба), а затем мы можем получить общий секрет: секрет = (a * G) * b = (b * G) * a. Довольно просто. Приведенное выше уравнение принимает следующий вид:

alicePubKey * bobPrivKey = bobPubKey * alicePrivKey = секрет

Алгоритм ECDH (обмен ключами Диффи–Хеллмана с эллиптической кривой) тривиален:

Алиса генерирует случайную пару ключей ECC: {alicePrivKey, alicePubKey = alicePrivKey * G}

Боб генерирует случайную пару ключей ECC: {bobPrivKey, bobPubKey = bobPrivKey * G}

Алиса и Боб обмениваются своими открытыми ключами по небезопасному каналу (например, через Интернет)

Алиса вычисляет общий ключ = bobPubKey * alicePrivKey

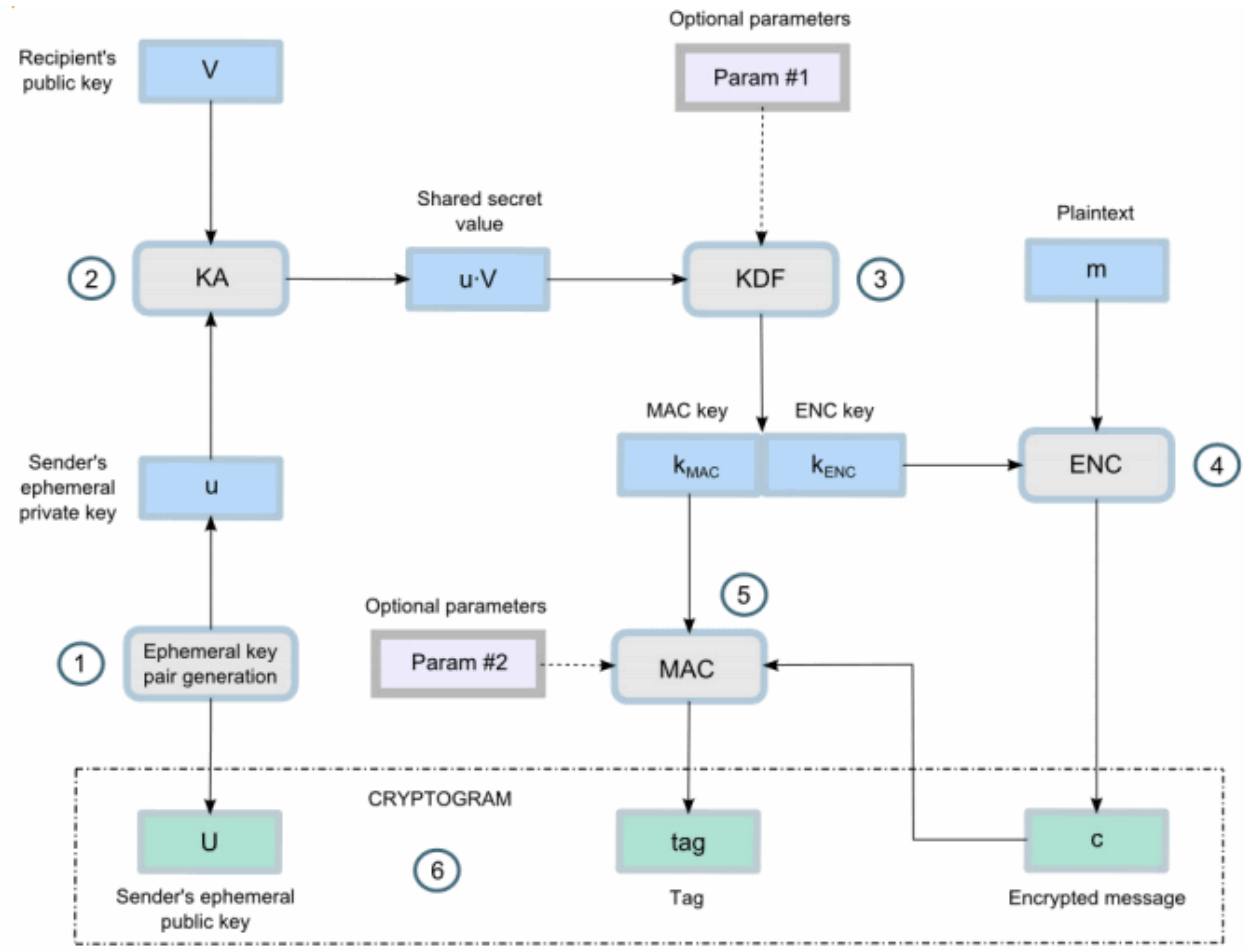
Боб вычисляет SharedKey = alicePubKey * bobPrivKey

Теперь у Алисы и Боба один и тот же общий ключ == bobPubKey * alicePrivKey == alicePubKey * bobPrivKey

Гибридная схема шифрования ECIES

Гибридная схема шифрования, аналогичная ранее продемонстрированному коду, стандартизирована под названием Elliptic Curve Integrated Encryption Scheme (ECIES) во многих стандартах шифрования, таких как SECG SEC-1, ISO/IEC 18033-2, IEEE 1363a и ANSI X9.63. ECIES - это схема шифрования с аутентификацией с открытым ключом, которая работает аналогично приведенным выше примерам кода, но использует KDF (функцию получения ключа) для получения отдельного MAC-ключа и симметричного ключа шифрования из общего секрета ECDH. Она имеет много вариантов.

Стандарт ECIES сочетает в себе асимметричную криптографию на основе ECC с симметричными шифрами для обеспечения шифрования данных с помощью закрытого ключа EC и дешифрования с помощью соответствующего открытого ключа EC. Схема шифрования ECIES использует криптографию ECC (криптосистема с открытым ключом) + функцию получения ключа (KDF) + алгоритм симметричного шифрования + алгоритм MAC, объединенные вместе, как показано на рисунке ниже:



Ввод для шифрования ECIES состоит из открытого ключа получателя + простого текстового сообщения. Выходные данные состоят из эфемерного открытого ключа отправителя (открытый ключ зашифрованного текста) + зашифрованного сообщения (зашифрованный текст + параметры симметричного алгоритма) + тега аутентификации (MAC-код):

ECIES-encrypt(recipientPublicKey, plaintextMessage) ➔ { cipherTextPublicKey, encryptedMessage, authTag }

Расшифровка ECIES использует выходные данные шифрования + закрытый ключ получателя и выдает исходное текстовое сообщение или обнаруживает проблему (например, ошибку целостности / аутентификации):

ECIES-decrypt(cipherTextPublicKey, encryptedMessage, authTag, recipientPrivateKey,) ➔ plaintextMessage

Схема шифрования ECIES - это структура, а не конкретный алгоритм. Это может быть реализовано путем подключения различных алгоритмов, например, эллиптической кривой secp256k1 или P-521 для вычислений с открытым ключом + PBKDF2 или Scrypt для функции KDF + AES-CTR или AES-GCM или ChaCha20-Poly1305 для симметричного шифрования и тега аутентификации + HMAC-SHA512 для алгоритма MAC (в случае шифрования без проверки подлинности).

Криптография на эллиптической кривой (ECC)

Криптография на эллиптических кривых (ECC) — это современное семейство криптосистем с открытым ключом, основанное на алгебраических структурах эллиптических кривых над конечными полями и на сложности задачи дискретного логарифмирования на эллиптических кривых (ECDLP).

ECC реализует все основные возможности асимметричных криптосистем: шифрование, подписи и обмен ключами.

ECC-криптография считается естественным современным преемником криптосистемы RSA, поскольку ECC использует меньшие ключи и подписи, чем RSA, при том же уровне безопасности и обеспечивает очень быструю генерацию ключей, быстрое согласование ключей и быстрые подписи.

ECC-ключи

Закрытые ключи в ECC являются целыми числами (в диапазоне размера поля кривой, обычно это 256-битные целые числа). Примером 256-битного закрытого ключа ECC (в шестнадцатеричном кодировании, 32 байта, 64 шестнадцатеричных цифры) является: 0x51897b64e85c3f714bba707e867914295a1377a7463a9dae8ea6a8b914246319.

Генерация ключа в ECC-криптографии так же проста, как безопасная генерация случайного целого числа в определенном диапазоне, поэтому выполняется чрезвычайно быстро. Любое число в пределах диапазона является действительным закрытым ключом ECC.

Открытыми ключами в ECC являются EC точки - пары целых координат {x, y}, лежащие на кривой. Благодаря своим особым свойствам точки EC могут быть сжаты всего до одной координаты + 1 бит (четный или нечетный). Таким образом, сжатый открытый ключ, соответствующий 256-битному закрытому ключу ECC, представляет собой 257-битное целое число. Примером открытого ключа ECC (соответствующего вышеупомянутому закрытому ключу, закодированного в формате Ethereum в виде hex с префиксом 02 или 03) является: 0x02f54ba86dc1ccb5bed0224d23f01ed87e4a443c47fc690d7797a13d41d2340e1a. В этом формате открытый ключ фактически занимает 33 байта (66 шестнадцатеричных цифр), которые могут быть оптимизированы ровно до 257 бит.

Кривые и длина ключа

Криптоалгоритмы ECC могут использовать различные лежащие в их основе эллиптические кривые. Разные кривые обеспечивают разный уровень безопасности (криптографической надежности), разную производительность (скорость) и разную длину ключа, а также могут включать разные алгоритмы.

Кривые ECC, принятые в популярных криптографических библиотеках и стандартах безопасности, имеют название (именованные кривые, например, secp256k1 или Curve25519), размер поля (который определяет длину ключа, например, 256-битный), надежность безопасности (обычно размер поля / 2 или меньше), производительность (операций в секунду) и многие другие параметры.

Ключи ECC имеют длину, которая напрямую зависит от базовой кривой. В большинстве приложений (таких как OpenSSL, OpenSSH и Биткойн) длина ключа по умолчанию для закрытых ключей ECC составляет 256 бит, но в зависимости от кривой возможно множество различных размеров ключа ECC: 192-битный (кривая secp192r1), 233-битный (кривая sect233k1), 224-битный (кривая secp224k1), 256-битный (кривые secp256k1 и Curve25519), 283-битный (кривая sect283k1), 384-битный (кривая бит (кривые p384 и secp384r1), 409-бит (кривая sect409r1), 414-бит (кривая Curve41417), 448-бит (кривая Curve448-Goldilocks), 511-бит (кривая M-511), 521-бит (кривая P-521), 571-бит (кривая sect571k1) и многие другие.

Алгоритмы ECC

Криптография на эллиптических кривых (ECC) предоставляет несколько групп алгоритмов, основанных на математике эллиптических кривых над конечными полями:

Алгоритмы цифровой подписи ECC, такие как ECDSA (для классических кривых) и EdDSA (для скрученных кривых Эдвардса).

Алгоритмы шифрования ECC и гибридные схемы шифрования, такие как интегрированная схема шифрования ECIES и EEECC (ElGamal на базе EC).

Алгоритмы согласования ключей ECC, такие как ECDH, X25519 и FHMQV.

Все эти алгоритмы используют кривую позади (например, , или secp256k1) для вычислений и зависят от сложности ECDLP (задача дискретного логарифмирования по эллиптической кривой). Все эти алгоритмы используют пары открытого и закрытого ключей, где закрытый ключ является целым числом, а открытый ключ - точкой на эллиптической кривой (EC point). Давайте подробнее рассмотрим эллиптические кривые над конечными полями.curve25519p521

Эллиптические кривые

В математике эллиптические кривые - это плоские алгебраические кривые, состоящие из всех точек {x, y}, описываемые уравнением:

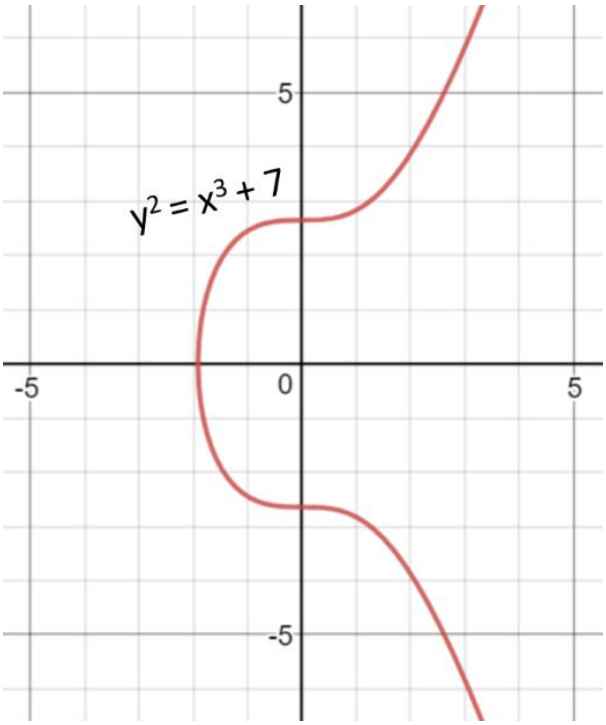
В криптографии используются эллиптические кривые в упрощенной форме (форма Вейерштрасса), которая определяется как:

y2 = x3 + _a _x + b

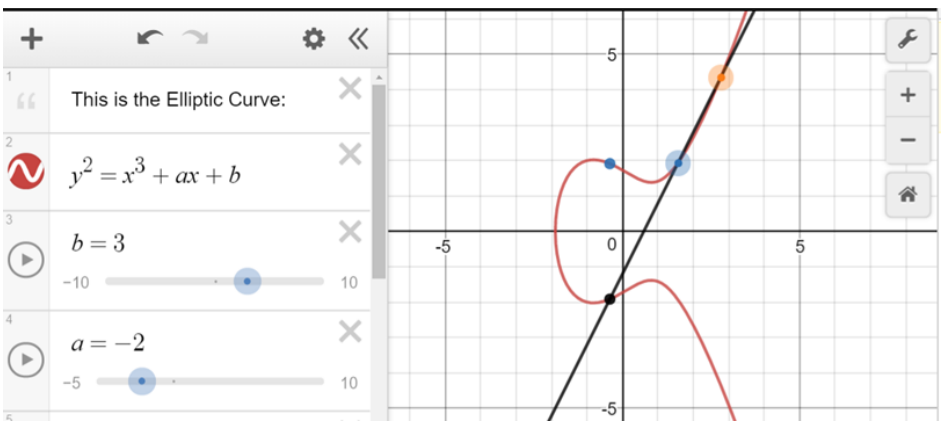
For example, the NIST curve secp256k1 (used in Bitcoin) is based on an elliptic curve in the form:

y2 = x3 + 7 (the above elliptic curve equation, where a = 0 and b = 7)

Это визуализация вышеупомянутой эллиптической кривой:



Чтобы узнать больше об уравнениях эллиптических кривых и о том, как они выглядят, немного поиграйте с этим онлайн-инструментом визуализации эллиптических кривых:



Эллиптические кривые над конечными полями

В криптографии на эллиптических кривых (ECC) используются эллиптические кривые над конечным полем $\mathbb{F}p$ (где p простое число и p > 3) или m (где размер полей p = 2_m__). Это означает, что поле представляет собой квадратную матрицу размером p x p, а точки на кривой ограничены только целочисленными координатами внутри поля. Все алгебраические операции внутри поля (такие как сложение и умножение точек) приводят к другой точке внутри поля. Уравнение эллиптической кривой над конечным полем $\mathbb{F}p$ принимает следующую модульную форму:

$y2 \equiv x3 + \_a\_x + b \pmod p$

Соответственно, "кривая биткойна" secp256k1 принимает вид:

$y2 \equiv x3 + 7 \pmod p$

В отличие от RSA, который использует для своего пространства ключей целые числа в диапазоне [0...p-1] (поле $\mathbb{Z}P$), ECC использует точки {x, y} в поле Галуа $\mathbb{F}p$ (где x и y - целые числа в диапазоне [0...p-1]).

Эллиптическая кривая над конечным полем $\mathbb{F}p$ состоит из:

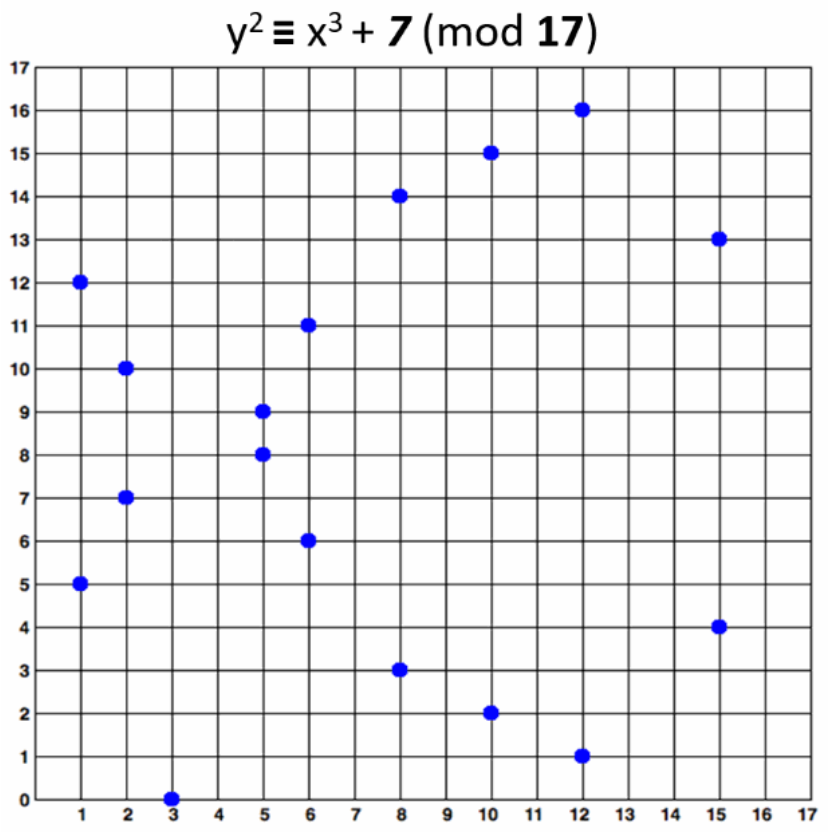
набор целых координат {x, y}, таких, что $0 \leq x, y < p$

оставаясь на эллиптической кривой: $\_y\_2 \equiv x3 + \_a\_x + b \pmod p$

Пример эллиптической кривой над конечным полем ≈17:

$y2 \equiv x3 + 7$ (версия 17)

Эта эллиптическая кривая над $\mathbb{F}17$ выглядит следующим образом:



Обратите внимание, что эллиптическая кривая над конечным полем y2 ≡ x3 + 7 (mod 17) состоит из синих точек на приведенном выше рисунке, т. е. на практике "эллиптические кривые", используемые в криптографии, являются "наборами точек в квадратной матрице", а не классическими "кривыми".

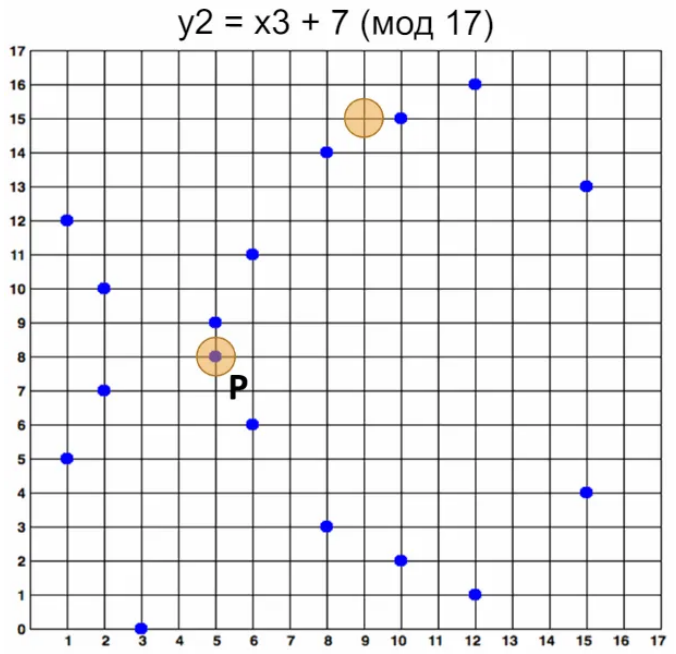
Приведенная выше кривая является "образовательной". Она обеспечивает очень малую длину ключа (4-5 бит). В реальном мире разработчики обычно используют кривые с 256 битами или более.

Эллиптические кривые над конечными полями: вычисления

Довольно легко вычислить, принадлежит ли определенная точка определенной эллиптической кривой на конечном поле. Например, точка {x, y} принадлежит кривой y2 ≡ x3 + 7 (mod 17) тогда и только тогда, когда:

x3 + 7 - y2 ≡ 0 (мод 17)

Точка P {5, 8} принадлежит кривой, потому что (5**3 + 7 - 8**2) % 17 == 0. Точка {9, 15} не принадлежит кривой, потому что (9**3 + 7 - 15**2) % 17 != 0. Эти вычисления выполнены в стиле Python. Вышеупомянутая эллиптическая кривая и точки {5, 8} и {9, 15} визуализированы ниже:



Умножение точки ECC на целое число

Можно добавить две точки над эллиптической кривой (EC points), и результатом будет еще одна точка. Эта операция известна как добавление точки EC. Если мы добавим точку G к самой себе, результатом будет G + G = 2 * G. Если мы снова добавим к результату G, то получим 3 * G и так далее. Вот как определяется умножение точек EC.

Точку G на эллиптической кривой над конечным полем (EC point) можно умножить на целое число k, и результатом будет другая EC точка P на той же кривой, и эта операция выполняется быстро:

P = k * G

Описанная выше операция включает в себя некоторые формулы и преобразования, но для простоты мы их пропустим. Важно знать, что умножение точки EC на целое число возвращает другую точку EC на той же кривой, и эта операция выполняется быстро. Умножение точки EC на 0 возвращает специальную точку EC, называемую "бесконечность".

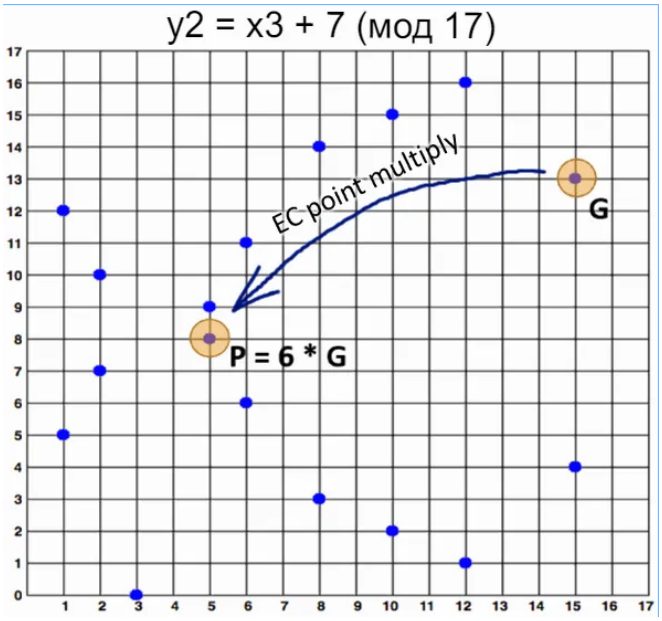
Пример: Умножьте точку EC на целое число

Формулы для умножения EC различаются для разных форм представления кривой. В этом примере мы будем использовать эллиптическую кривую в классической форме Вейерштрасса.

Для примера давайте возьмем точку EC G = {15, 13} на эллиптической кривой над конечным полем y2 ≡ x3 + 7 (mod 17) и умножим ее на k = 6. Мы получим точку EC P = {5, 8}:

P = k * G = 6 * {15, 13} = {5, 8}

На рисунке ниже показан этот пример умножения точек EC:



Порядок и кофактор эллиптической кривой

Эллиптическая кривая над конечным полем может образовывать конечную циклическую алгебраическую группу, которая состоит из всех точек кривой. В циклической группе, если добавить две точки EC или умножить точку EC на целое число, результатом будет другая точка EC из той же циклической группы (и на той же кривой). Порядок кривой — это общее количество всех точек EC на кривой. Это общее количество точек включает также специальную точку, называемую "точка в бесконечности", которая получается, когда точка умножается на 0.

Некоторые кривые образуют единственную циклическую группу (содержащую все их EC-точки), в то время как другие образуют несколько неперекрывающихся циклических подгрупп (каждая из которых содержит подмножество EC-точек кривой). Во втором сценарии точки на кривой разбиваются на h циклических подгрупп (разделов), каждая порядка r (каждая подгруппа содержит равное количество точек). Порядок всей группы равен n = h * r (количество подгрупп, умноженное на количество точек в каждой подгруппе). Количество подгрупп h, содержащих точки EC, называется кофактором.

Кофактор обычно выражается следующей формулой:

h = n / r где

n - порядок кривой (количество всех ее точек).

h - кофактор кривой (количество неперекрывающихся подгрупп точек, которые вместе содержат все точки кривой).

r - порядок подгрупп (количество точек в каждой подгруппе, включая точку бесконечности для каждой подгруппы)

Другими словами, точки над эллиптической кривой остаются в одном или нескольких неперекрывающихся подмножествах, называемых циклическими подгруппами. Количество подгрупп называется "кофактором". Общее количество точек во всех подгруппах называется "порядком" кривой и обычно обозначается n. Если кривая состоит из только одной циклической подгруппы, ее кофактор h = 1. Если кривая состоит из нескольких подгрупп, ее кофактор > 1.

Примером эллиптической кривой, имеющей кофактор = 1, является secp256k1.

Примером эллиптической кривой, имеющей кофактор = 8, является Curve25519.

Примером эллиптической кривой, имеющей кофактор = 4, является Curve448.

Точка "Генератора" в ECC

Для эллиптических кривых над конечными полями криптосистемы ECC определяют специальную предопределенную (постоянную) точку EC, называемую точкой генератора G (базовая точка), которая может сгенерировать любую другую точку в своей подгруппе на эллиптической кривой путем умножения G на некоторое целое число в диапазоне [0...r]. Число r называется "порядком" циклической подгруппы (общее количество всех точек в подгруппе).

Для кривых с кофактором = 1 существует только одна подгруппа, а порядок n кривой (общее количество различных точек на кривой, включая бесконечность) равен числу r.

Когда G и n тщательно выбраны, а кофактор = 1, все возможные точки EC на кривой (включая специальную точку бесконечности) могут быть сгенерированы из генератора G путем умножения его на целое число в диапазоне [1...n]. Это целое число n известно как "порядок кривой".

Важно знать, что порядок r подгруппы, полученный из определенной точки генератора EC G (которая может отличаться от порядка кривой) определяет общее количество всех возможных закрытых ключей для этой кривой: r = n / h (порядок кривой, деленный на коэффициент кривой). Криптографы тщательно выбирают параметры области эллиптической кривой (уравнение кривой, точка генератора, кофактор и т.д.), Чтобы гарантировать, что пространство ключей достаточно велико для обеспечения определенной криптографической надежности.

Подводя итог, в криптографии ECC точки EC вместе с точкой генератора G образуют циклические группы (или циклические подгруппы), что означает, что существует число r (r > 1), такое, что r * G = 0 * G = бесконечность и все точки в подгруппе могут быть получены путем умножения G на целое число в диапазоне [1 ... r]. Число r называется порядком в группе (или подгруппе).

Подгруппы на эллиптической кривой обычно имеют много точек генерации, но криптографы тщательно выбирают одну из них, которая генерирует всю группу (или подгруппу) и подходит для оптимизации производительности вычислений. Это генератор, известный как "G".

Известно, что для некоторых кривых разные точки генератора генерируют подгруппы разного порядка. Точнее, если порядок групп равен n, то для каждого простого числа, d делящего n, существует точка Q, такая, что d * Q = бесконечности. Это означает, что некоторые точки, используемые в качестве генераторов для одной и той же кривой, будут генерировать меньшие подгруппы, чем другие. если группа маленькая, безопасность слабая. Это известно как атаки "малой подгруппы". Именно по этой причине криптографы обычно выбирают порядок подгрупп r в качестве простого числа.

Для эллиптических кривых с коэффициентом h > 1 разные базовые точки могут генерировать разные подгруппы EC-точек на кривой. Выбирая определенную точку генератора, мы решаем работать над определенной подгруппой точек кривой, и большинство операций с точками EC и криптоалгоритмов ECC будут работать хорошо. Тем не менее, в некоторых случаях следует уделять особое внимание, поэтому рекомендуется использовать только проверенные реализации ECC, алгоритмы и пакеты программного обеспечения.

Точка генератора - пример

В приведенном выше примере (EC над конечным полем y2 ≈ x3 + 7 mod 17), если мы возьмем точку G = {15, 13} в качестве генератора, любая другая точка кривой может быть получена путем умножения G на некоторое целое число в диапазоне [1...18]. Таким образом, порядок этого EC равен n = 18, а его сомножитель h = 1.

Обратите внимание, что кривая имеет 17 нормальных точек EC (показаны на рисунках выше) + одну специальную "точку на бесконечности", все они находятся в одной подгруппе, а порядок расположения кривой равен 18 (а не 17).

Обратите также внимание, что если мы возьмем точку {5, 9} в качестве генератора, она сгенерирует всего 3 точки EC : {5, 8}, {5, 9} и бесконечность. Поскольку порядок кривых не является простым числом, разные генераторы могут генерировать подгруппы разного порядка. Это хороший пример, почему мы не должны "изобретать" наши собственные эллиптические кривые для криптографических целей, и мы должны использовать проверенные кривые.

Закрытый ключ, открытый ключ и точка генератора в ECC

В ECC, когда мы умножаем фиксированную точку EC G (точку генератора) на определенное целое число k (k можно рассматривать как закрытый ключ), мы получаем точку EC P (соответствующий ей открытый ключ).

Следовательно, в ECC мы имеем:

Эллиптическая кривая (EC) над конечным полем $\mathbb{F}p$

G == точка генератора (фиксированная константа, базовая точка на EC)

k == закрытый ключ (целое число)

P == открытый ключ (точка)

Очень быстро вычисляется P = k * G с использованием хорошо известных алгоритмов умножения ECC за время log_2(k_), например "алгоритм удвоения и суммирования". Для 256-битных кривых потребуется всего несколько сотен простых операций EC.

Вычисление k = P / G выполняется чрезвычайно медленно (считается неосуществимым при больших k).

Эта асимметрия (быстрое умножение и неосуществимо медленная обратная операция) является основой надежности криптографии ECC, также известной как проблема ECDLP.

Задача дискретного логарифмирования на эллиптической кривой (ECDLP)

Задача дискретного логарифмирования на эллиптической кривой (ECDLP) в информатике определяется следующим образом:

По заданной эллиптической кривой над конечным полем $\mathbb{F}p$ и образующей точке G на кривой и точке P на кривой найдите целое число k (если оно существует), такое, что P = k * G

Для тщательно подобранных (криптографами) конечных полей и эллиптических кривых проблема ECDLP не имеет эффективного решения.

Умножение точек эллиптической кривой в группе $\mathbb{F}p$ аналогично возведению в степень целых чисел в группе $\mathbb{Z}P$ (это известно как мультипликативная нотация), и именно этим задача ECDLP похожа на задачу DLP (задача дискретного логарифмирования).

В криптографии ECC многие алгоритмы зависят от вычислительной сложности задачи ECDLP над тщательно выбранным полем $\mathbb{F}p$ и эллиптической кривой, для которых не существует эффективного алгоритма.

Надежность ECC и кривой защиты

Потому что самый быстрый известный алгоритм для решения ECDLP для ключа размера k нуждается $\sqrt{k}$

шаги, это означает, что для достижения k-разрядной надежности защиты требуется, по крайней мере, 2* k-разрядной кривой. Таким образом, 256-битные эллиптические кривые (где размер поля p равен 256-битному числу) обычно обеспечивают почти 128-битный уровень безопасности.

На самом деле, надежность немного меньше, потому что порядок кривой (n) обычно меньше размера полей (p) и потому, что кривая может иметь кофактор h > 1 (и порядок подгрупп r = n / h, меньше, чем n) и потому, что количество шагов не совсем $\sqrt{k}$

, но является $0.886 * \sqrt{k}$

Например, кривая secp256k1 (p = 256) обеспечивает ~ 128-битную защиту (127,8 бита, если быть точным), а Curve448 (p = 448) обеспечивает ~ 224-битную безопасность (222,8 бита, если быть точным).

RSA или ECC? Что лучше?

Спорно, что лучше - ECC или RSA в пространстве криптосистем с открытым ключом, поэтому мы представим их сильные и слабые стороны. Обе криптосистемы (RSA и криптография с эллиптической кривой) работают с закрытыми и открытыми ключами и предоставляют схожие возможности, такие как генерация ключей, цифровые подписи, схемы согласования ключей и схемы шифрования.

Обычно считается, что ECC является современной и более предпочтительной криптосистемой с открытым ключом из-за меньших ключей, более коротких подписей и лучшей производительности, но некоторые люди не согласны. Сегодня широко используются обе криптосистемы. Например (по состоянию на ноябрь 2018 года) Facebook и Google защищают свои основные веб-сайты 256-разрядными закрытыми ключами ECC, в то время как Amazon и Apple защищают свои основные веб-сайты 2048-разрядными закрытыми ключами RSA.

Преимущества ECC

Криптография с эллиптической кривой (ECC) имеет следующие преимущества перед RSA:

Меньшие ключи, зашифрованные тексты и подписи. Обычно для 128-битного шифрования используется 256-битный закрытый ключ EC. Длина подписей обычно в 2 раза превышает длину закрытого ключа или больше (зависит от схемы кодирования). По контракту в RSA для обеспечения 128-битной безопасности требуется 3072-битный закрытый ключ. Длина зашифрованного текста обычно равна длине незашифрованного обычного текста.

Очень быстрая генерация ключей. В криптосистемах ECC генерация ключей состоит из генерации случайных чисел + умножения точек EC и выполняется чрезвычайно быстро, даже для самых сложных кривых, используемых на практике. В отличие от этого, в RSA генерация ключей может быть медленной, особенно для длинных ключей (например, 4096-битных и более длинных ключей), из-за процесса генерации простых чисел.

Быстрые подписи. Подписание сообщений и проверка подписей ECC выполняются очень быстро: всего несколько вычислений точки EC.

Быстрое согласование ключа. Согласование ключа (ECDH) так же просто, как умножение точки EC на целое число, и выполняется очень быстро.

Быстрое шифрование и дешифрование. Фактически шифрование выполняется так же быстро, как базовое шифрование с симметричным ключом (благодаря схеме шифрования ECIES) + небольшое замедление из-за соглашения о ключе ECDH.

Недостатки ECC

Криптография с эллиптической кривой (ECC) имеет некоторые недостатки:

ECC сложнее и сложнее в безопасной реализации, чем RSA. Это не обязательно проблема, если вы используете проверенные криптографические библиотеки от надежных поставщиков.

Подписание с помощью неисправного генератора случайных чисел компрометирует закрытый ключ подписывающего. Двукратное подписание одним и тем

же случайным числом напрямую раскрывает закрытый ключ подписывающего. Некоторые схемы подписи избегают случайности и не имеют такой потенциальной проблемы (например, детерминированный ECDSA).

Не все стандартизированные кривые считаются безопасными. Вы должны знать, как выбрать надежную кривую для ваших вычислений ECC.

Преимущества RSA

Криптосистема RSA обладает следующими преимуществами:

RSA проще реализовать, чем ECC. Если вы не реализуете алгоритмы самостоятельно, это может быть не так важно.

RSA легче понять, чем ECC. Подписание и дешифрование аналогичны; шифрование и проверка аналогичны. Это упрощает понимание. Простота важнее, если вы реализуете алгоритмы самостоятельно, а не используете хорошо известную библиотеку.

RSA предоставляет очень **быстрые ** и простые **алгоритмы шифрования **и **подписи **. Это связано с небольшим общедоступным показателем.

Некоторые криптографы считают, что RSA с очень большими ключами (например, 16384 бита) обычно сильнее ECC.

Недостатки RSA

Криптосистема RSA имеет следующие недостатки:

В криптосистеме RSA генерация ключей происходит очень медленно (особенно при большой длине ключа). На современном ноутбуке может потребоваться несколько минут, чтобы сгенерировать 16384-битную пару ключей RSA!

В криптосистеме RSA подписание и дешифрование выполняются медленно (для больших закрытых ключей), что немного сложно реализовать безопасно. Обычно в криптосистеме RSA общедоступный показатель невелик (например, 65537 или 3), поэтому вычисления для шифрования сообщения или проверки подписи выполняются быстро. В то же время частный показатель обычно большой (например, 4096-битное целое число), поэтому вычисления для подписи или дешифрования сообщения выполняются медленно.

RSA создает большие подписи (той же длины, что и закрытый ключ). Это неприемлемо для многих систем, которые хранят много подписей, например, общедоступных блокчейнов.