Задания СРС:

**Цель: у**глубление теоретических знаний и развитие практических навыков по обеспечению кибербезопасности, формирование компетенций в области защиты информации, освоение принципов функционирования систем защиты, а также анализ современных киберугроз и методов противодействия им.

**Лекция №5: Криптографические алгоритмы с открытым ключом и их использование**

### ****Цель:**** Познакомиться с основными алгоритмами криптографии с открытым ключом и понять их принципы, применение и область использования.

### ****Задание:****

#### ****1. Теоретическая часть: изучить и изложить в конспекте****

* Принципы криптографии с открытым ключом
* Отличие от симметричной криптографии
* Структура ключей: открытый и закрытый ключ
* Основные алгоритмы:
  + **RSA**: принцип работы, ключевая генерация, шифрование/дешифрование
  + **ElGamal**: идея алгоритма, плюсы/минусы
  + **ECC (эллиптические кривые)**: особенности и преимущества
  + Использование алгоритмов для **цифровой подписи**

#### ****2. Практическая часть:****

* Составить таблицу сравнения RSA, ElGamal и ECC по критериям:
  + Безопасность
  + Скорость
  + Размер ключей
  + Область применения
* Рассчитать простой пример шифрования и расшифровки с помощью RSA (на небольших числах)
* Привести примеры реального использования (например, HTTPS, PGP, цифровая подпись)

#### ****3. Подготовить:****

* Презентацию (10–12 слайдов) с кратким описанием алгоритмов и их применения
* Краткий письменный отчёт (1–2 страницы) по выполненным задачам

**Цель лекции**: изучить принципы работы некоторых криптографических алгоритмов с открытым ключом.

### Алгоритм RSA

#### Основные сведения

Алгоритм шифрования с открытым ключом **RSA** был предложен одним из первых в конце 70-х годов ХХ века. Его название составлено из первых букв фамилий авторов: Р.Райвеста (R.Rivest), А.Шамира (A.Shamir) и Л.Адлемана (L.Adleman). Алгоритм RSA является, наверно, наиболее популярным и широко применяемым *асимметричным алгоритмом* в криптографических системах.

Алгоритм основан на использовании того факта, что задача разложения большого числа на простые сомножители является трудной. Криптографическая система RSA базируется на следующих двух фактах из теории чисел:

1. задача проверки числа на простоту является сравнительно легкой;
2. задача разложения чисел вида n = pq ( р и q — простые числа); на множители является очень трудной, если мы знаем только n, а р и q — большие числа (это так называемая задача факторизации, подробнее о ней см. ["Основные положения теории чисел, используемые в криптографии с открытым ключом"](https://www.intuit.ru/studies/courses/691/547/lecture/12389)).

Алгоритм RSA представляет собой блочный алгоритм шифрования, где зашифрованные и незашифрованные данные должны быть представлены в виде целых чисел между 0 и n -1 для некоторого n.

#### Шифрование

Итак, рассмотрим сам алгоритм. Пусть абонент А хочет передать зашифрованное сообщение абоненту Б. В этом случае абонент Б должен подготовить пару (открытый ключ; закрытый ключ) и отправить свой открытый ключ пользователю А.

Первым этапом является генерация открытого и закрытого ключей. Для этого вначале выбираются два больших простых числа Р и Q. Затем вычисляется произведение N:

N = PQ.

После этого определяется вспомогательное число f:

f = (Р - l)(Q - 1).

Затем случайным образом выбирается число d < f и взаимно простое с f.

Далее необходимо найти число е, такое, что

еd mod f = 1.

Числа d и N будут открытым ключом пользователя, а значение е – закрытым ключом.

Таким образом, на этом этапе у пользователя должна быть информация, указанная в следующей таблице:

|  |  |  |
| --- | --- | --- |
|  | Открытый ключ | Закрытый ключ |
| Пользователь системы | N, d | e |

Так как пользователь Б хочет получить зашифрованное сообщение от пользователя А, значит пользователь Б должен отправить свой открытый ключ (d, N) пользователю А. Числа Р и Q больше не нужны, однако их нельзя никому сообщать; лучше всего их вообще забыть.

На этом этап подготовки ключей закончен и можно использовать основной протокол RSA для шифрования данных.

**Второй этап – шифрование данных**. Если абонент А хочет передать некоторые данные абоненту Б, он должен представить свое сообщение в цифровом виде и разбить его на блоки m1, m2, m3, ... , где mi < N. Зашифрованное сообщение будет состоять из блоков сi.

Абонент А шифрует каждый блок своего сообщения по формуле

ci = mid mod N

используя *открытые параметры* пользователя Б, и пересылает зашифрованное сообщение С=(с1, с2, с3, ...) по открытой линии.

Абонент Б, получивший зашифрованное сообщение, расшифровывает все блоки полученного сообщения по формуле

mi = ce mod N

Все расшифрованные блоки будут точно такими же, как и исходящие от пользователя А.

Злоумышленник, перехватывающий все сообщения и знающий всю открытую информацию, не сможет найти исходное сообщение при больших значениях Р и Q.

#### Пример вычислений по алгоритму

Пусть пользователь А хочет передать пользователю Б сообщение. В этом случае вначале пользователь Б должен подготовить открытый и закрытый ключи. Пусть им выбраны, например, следующие параметры:

Р = 3, Q = 11, N = 3x11 = 33.

Тогда f = (Р - l)(Q - 1) = (3-1)(11-1) = 20.

Затем пользователь Б выбирает любое число d, не имеющее общих делителей с f (это необходимо для того, чтобы зашифрованное сообщение можно было потом однозначно восстановить). Пусть d = 13. Это число будет одним из компонентов открытого ключа.

Далее необходимо найти число е, которое можно будет использовать в качестве закрытого ключа для расшифрования сообщения. Значение е должно удовлетворять соотношению

еd mod f = 1.

Для малых значений f число е можно найти подбором. В общем случае для поиска е можно использовать обобщенный алгоритм Евклида, приведенный в ["Основные положения теории чисел, используемые в криптографии с открытым ключом"](https://www.intuit.ru/studies/courses/691/547/lecture/12389). В нашем случае подходит е=17. (Проверяем: 13\*17 mod 20 = 221 mod 20 = 1.)

Теперь пользователь Б должен запомнить свой закрытый ключ 17, отправить открытый ключ (13, 33) пользователю А и уничтожить числа Р = 3 и Q = 11.

Пользователь А, получивший открытый ключ (13, 33), увидев, что N=33, разбивает исходное сообщение на три блока, причем значение каждого меньше N. Например, пусть имеется три блока m1=8, m2=27, m3,=5. Затем пользователь А шифрует каждый блок:

c1=813 mod 33 = 17

c2 = 2713 mod 33 = 15

c3 = 513 mod 33 = 26

Зашифрованное сообщение, состоящее из трех блоков (17, 15, 26), передается пользователю Б, который, используя свой закрытый ключ е = 17 и N=33, расшифровывает сообщение:

m1 = 1717 mod 33 = 8

m2 = 1517 mod 33 = 27

m3 = 2617 mod 33 = 5

Таким образом, абонент Б расшифровал сообщение от абонента А.

#### Вопросы практического использования алгоритма RSA

На протяжении многих лет алгоритм RSA активно используется как в виде самостоятельных криптографических продуктов, так и в качестве встроенных средств в популярных приложениях. Открытое шифрование на базе алгоритма RSA применяется в популярном пакете шифрования PGP, операционной системе Windows, различных Интернет-браузерах, банковских компьютерных системах. Кроме того, различные международные стандарты шифрования с открытым ключом и формирования цифровой подписи используют RSA в качестве основного алгоритма.

Для обеспечения высокой надежности шифрования необходимо, чтобы выступающее в качестве модуля число N было очень большим – несколько сотен или тысяч бит. Только в этом случае будет практически невозможно по *открытым параметрам* определить закрытый ключ. Так, известно, что в конце 1995 года удалось практически реализовать раскрытие шифра RSA для 500-значного модуля. Для этого с помощью сети Интернет было задействовано более тысячи компьютеров.

Сами авторы RSA рекомендовали использовать следующие размеры модуля N: 768 бит - для частных лиц; 1024 бит - для *коммерческой информации*; 2048 бит - для особо секретной информации. С момента получения их рекомендаций прошло какое-то время, поэтому современные пользователи должны делать поправки в сторону увеличения размера ключей. Однако, чем больше размер ключей, тем медленнее работает система. Поэтому увеличивать размер ключа без необходимости не имеет смысла.

С размером ключей связан и другой аспект реализации RSA - *вычислительный*. При использовании алгоритма вычисления необходимы как при создании ключей, так и при шифровании/расшифровании, при этом, чем больше размер ключей, тем труднее производить расчеты. Для работы с громадными числами приходится использовать аппарат *длинной арифметики*. Числа, состоящие из многих сотен бит, не умещаются в регистры большинства микропроцессоров и их приходится обрабатывать по частям. При этом как шифрование, так и расшифрование включают возведение большого целого числа в целую степень по модулю N. При прямых расчетах промежуточные значения были бы невообразимыми. Чтобы упростить процесс вычислений используют специальные алгоритмы для работы с большими числами, основанные на свойствах модульной арифметики, а также оптимизацию при возведении в степень.

Алгоритм RSA реализуется как программным, так и аппаратным путем. Многие мировые фирмы выпускают специализированные микросхемы, производящие шифрование алгоритмом RSA. Программные реализации значительные медленнее, чем аппаратные. К достоинствам программного шифрования RSA относится возможность гибкой настройки параметров, возможность интеграции в различные программные пакеты. В целом, и программная, и аппаратная реализации RSA требуют для выполнения примерно в тысячи раз большего времени по сравнению с *симметричными алгоритмами*, например ГОСТ 28147-89.

Алгоритм RSA может использоваться для формирования электронной цифровой подписи, а также и для обмена ключами. Возможность применения алгоритма RSA для получения электронной подписи связана с тем, что секретный и открытый ключи в этой системе равноправны. Каждый из ключей, d или e, могут использоваться как для шифрования, так и для расшифрования. Это свойство выполняется не во всех криптосистемах с открытым ключом. Использование алгоритма RSA для формирования ЭЦП рассматривается в ["Электронная цифровая подпись"](https://www.intuit.ru/studies/courses/691/547/lecture/12393).

### Алгоритм Диффи-Хеллмана

#### Основные сведения

Первая публикация данного алгоритма появилась в 70-х годах ХХ века в статье Диффи и Хеллмана, в которой вводились основные понятия криптографии с открытым ключом. *Алгоритм Диффи-Хеллмана* не применяется для шифрования сообщений или формирования электронной подписи. Его назначение – в распределении ключей. Он позволяет двум или более пользователям обменяться без посредников ключом, который может быть использован затем для симметричного шифрования. Это была первая криптосистема, которая позволяла защищать информацию без использования секретных ключей, передаваемых по защищенным каналам. Схема открытого распределения ключей, предложенная Диффи и Хеллманом, произвела настоящую революцию в мире шифрования, так как снимала основную проблему классической криптографии – проблему распределения ключей.

Алгоритм основан на трудности вычислений *дискретных логарифмов*. Попробуем разобраться, что это такое. В этом алгоритме, как и во многих других алгоритмах с открытым ключом, вычисления производятся по модулю некоторого большого простого числа Р. Вначале специальным образом подбирается некоторое натуральное число А, меньшее Р. Если мы хотим зашифровать значение X, то вычисляем

Y = AX mod P.

Причем, имея Х, вычислить Y легко. Обратная задача вычисления X из Y является достаточно сложной. Экспонента X как раз и называется *дискретным логарифмом Y*. Таким образом, зная о сложности вычисления *дискретного логарифма*, число Y можно открыто передавать по любому каналу связи, так как при большом модуле P исходное значение Х подобрать будет практически невозможно. На этом математическом факте основан *алгоритм Диффи-Хеллмана* для формирования ключа.

#### Формирование общего ключа

Пусть два пользователя, которых условно назовем пользователь 1 и пользователь 2, желают сформировать общий ключ для алгоритма симметричного шифрования. Вначале они должны выбрать большое простое число Р и некоторое специальное число А, 1 < A < P-1, такое, что все числа из интервала [1, 2, ..., Р-1] могут быть представлены как различные степени А mod Р. Эти числа должны быть известны всем абонентам системы и могут выбираться открыто. Это будут так называемые *общие параметры*.

Затем первый пользователь выбирает число Х1 (X1<P), которое желательно формировать с помощью датчика случайных чисел. Это будет закрытый ключ первого пользователя, и он должен держаться в секрете. На основе закрытого ключа пользователь 1 вычисляет число

Описание: Y_1 = A^{X_1}\: mod \: P

которое он посылает второму абоненту.

Аналогично поступает и второй пользователь, генерируя Х2 и вычисляя

Описание: Y_2 = A^{X_2}\: mod \: P

Это значение пользователь 2 отправляет первому пользователю.

После этого у пользователей должна быть информация, указанная в следующей таблице:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Общие параметры | Открытый ключ | Закрытый ключ |
| Пользователь 1 | Р, А | Y1 | Х1 |
| Пользователь 2 | Y2 | Х2 |

Из чисел Y1 и Y2, а также своих закрытых ключей каждый из абонентов может сформировать общий секретный ключ Z для сеанса симметричного шифрования. Вот как это должен сделать первый пользователь:

Описание: Z = (Y_2)^{X_1}\: mod \: P

Никто другой кроме пользователя 1 этого сделать не может, так как число Х1 секретно. Второй пользователь может получить то же самое число Z, используя свой закрытый ключ и открытый ключ своего абонента следующим образом:

Описание: Z = (Y_1)^{X_2}\: mod \: P

Если весь протокол формирования общего секретного ключа выполнен верно, значения Z у одного и второго абонента должны получиться одинаковыми. Причем, что самое важное, противник, не зная секретных чисел Х1 и Х2, не сможет вычислить число Z. Не зная Х1 и Х2, злоумышленник может попытаться вычислить Z, используя только передаваемые открыто Р, А, Y1 и Y2. Безопасность формирования общего ключа в алгоритме Диффи-Хеллмана вытекает из того факта, что, хотя относительно легко вычислить экспоненты по модулю простого числа, очень трудно вычислить дискретные логарифмы. Для больших простых чисел размером сотни и тысячи бит задача считается неразрешимой, так как требует колоссальных затрат вычислительных ресурсов.

Пользователи 1 и 2 могут использовать значение Z в качестве секретного ключа для шифрования и расшифрования данных. Таким же образом любая пара абонентов может вычислить секретный ключ, известный только им.

#### Пример вычислений по алгоритму

Пусть два абонента, желающие обмениваться через Интернет зашифрованными сообщениями, решили сформировать секретный ключ для очередного сеанса связи. Пусть они имеют следующие общие параметры:

Р = 11, А = 7.

Каждый абонент выбирает секретное число Х и вычисляет соответствующее ему открытое число Y. Пусть выбраны

Х1 = 3, Х2= 9.

Вычисляем

Y1 = 73 mod 11 = 2,

Y2= 79 mod 11 = 8.

Затем пользователи обмениваются открытыми ключами Y1 и Y2. После этого каждый из пользователей может вычислить общий секретный ключ:

пользователь 1: Z = 83 mod 11 = 6.

пользователь 2: Z = 29 mod 11 = 6.

Теперь они имеют общий ключ 6, который не передавался по каналу связи.

#### Вопросы практического использования алгоритма Диффи-Хеллмана

Для того, чтобы *алгоритм Диффи-Хеллмана* работал правильно, то есть оба пользователя, участвующих в протоколе, получали одно и то же число Z, необходимо правильным образом выбрать число А, используемое в вычислениях. Число А должно обладать следующим свойством: все числа вида

A mod P, A2 mod P, A3 mod P,... , AP-1 mod P

должны быть различными и состоять из целых положительных значений в диапазоне от 1 до Р-1 с некоторыми перестановками. Только в этом случае для любого целого Y < Р и значения A можно найти единственную экспоненту Х, такую, что

Y = AХmod P, где 0 <= X <= (P - 1)

При произвольно заданном Р задача выбора параметра А может оказаться трудной задачей, связанной с разложением на простые множители числа Р-1. На практике можно использовать следующий подход, рекомендуемый специалистами. Простое число Р выбирается таким, чтобы выполнялось равенство Р = 2q + l, где q — также простое число. Тогда в качестве А можно взять любое число, для которого справедливы неравенства

1<A<P-1 и Aq mod P ≠ 1

На подбор подходящих параметров А и Р необходимо некоторое время, однако это обычно не критично для системы связи и не замедляет ее работу. Эти параметры являются общими для целой группы пользователей. Они обычно выбираются один раз при создании сообщества пользователей, желающих использовать *протокол Диффи-Хеллмана*, и не меняются в процессе работы. А вот значения закрытых ключей рекомендуется каждый раз менять и выбирать их с помощью генераторов псевдослучайных чисел.

Следует заметить, что данный алгоритм, как и все *алгоритмы асимметричного шифрования*, уязвим для атак типа "man-in-the-middle" ("человек в середине"). Если противник имеет возможность не только перехватывать сообщения, но и заменять их другими, он может перехватить открытые ключи участников, создать свою пару открытого и закрытого ключа и послать каждому из участников свой открытый ключ. После этого каждый участник вычислит ключ, который будет общим с противником, а не с другим участником. Способы предотвращения такой атаки и некоторых других рассмотрены в конце этой лекции.

### Алгоритм Эль-Гамаля

#### Основные сведения

*Асимметричный алгоритм*, предложенный в 1985 году Эль-Гамалем (T. ElGamal), универсален. Он может быть использован для решения всех трех основных задач: для шифрования данных, для формирования цифровой подписи и для согласования общего ключа. Кроме того, возможны модификации алгоритма для схем проверки пароля, доказательства идентичности сообщения и другие варианты. Безопасность этого алгоритма, так же как и *алгоритма Диффи-Хеллмана*, основана на трудности вычисления *дискретных логарифмов*. Этот алгоритм фактически использует схему Диффи-Хеллмана, чтобы сформировать общий секретный ключ для абонентов, передающих друг другу сообщение, и затем сообщение шифруется путем умножения его на этот ключ.

И в случае шифрования, и в случае формирования цифровой подписи каждому пользователю необходимо сгенерировать пару ключей. Для этого, так же как и в схеме Диффи-Хеллмана, выбираются некоторое большое простое число Р и число А, такие, что различные степени А представляют собой различные числа по модулю Р. Числа Р и А могут передаваться в открытом виде и быть общими для всех абонентов сети.

Затем каждый абонент группы выбирает свое секретное число Хi, 1 < Хi < Р-1, и вычисляет соответствующее ему открытое число Описание: Y_i : Y_i = A^{X_i}\: mod \: P. Таким образом, каждый пользователь может сгенерировать закрытый ключ Хi и открытый ключ Yi.

Информация о необходимых параметрах системы сведена в следующую таблицу.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Общие параметры | Открытый ключ | Закрытый ключ |
| Пользователь 1 | Р, А | Y1 | Х1 |
| … | … | … |
| Пользователь i | Yi | Хi |

#### Шифрование

Теперь рассмотрим, каким образом производится шифрование данных. Сообщение, предназначенное для шифрования, должно быть представлено в виде одного числа или набора чисел, каждое из которых меньше Р. Пусть пользователь 1 хочет передать пользователю 2 сообщение m. В этом случае последовательность действий следующая.

1. Первый пользователь выбирает случайное число k, взаимно простое с Р-1, и вычисляет числа

Описание: r=A^k\: mod \: P, \qquad e=m \times Y_2^k \: mod \:P

где Y2 – открытый ключ пользователя 2. Число k держится в секрете.

1. Пара чисел (r, е), являющаяся шифротекстом, передается второму пользователю.
2. Второй пользователь, получив (r,e), для расшифрования сообщения вычисляет

Описание: m=e \times r^{P-1-X_2} \: mod \:P

где Х2 – закрытый ключ пользователя 2. В результате он получает исходное сообщение m.

Если злоумышленник узнает или перехватит Р, А, Y2, r, e, то он не сможет по ним раскрыть m. Это связано с тем, что противник не знает параметр k, выбранный первым пользователем для шифрования сообщения m. Вычислить каким-либо образом число k практически невозможно, так как это задача дискретного логарифмирования. Следовательно, злоумышленник не может вычислить и значение m, так как m было умножено на неизвестное ему число. Противник также не может воспроизвести действия законного получателя сообщения (второго абонента), так как ему не известен закрытый ключ Х2 (вычисление Х2 на основании Y2 — также задача дискретного логарифмирования).

По аналогичному алгоритму может производиться и согласование ключа, используемого для симметричного шифрования больших объемов данных. Более того, алгоритм Эль-Гамаля на практике целесообразно использовать именно для согласования общего *ключа сессии*, а не прямого шифрования больших сообщений. Это связано с тем, что в алгоритме используются операции возведения в степень и умножения по большому модулю. Так же как и в алгоритмах RSA и Диффи-Хеллмана, операции производятся над большими, состоящими из нескольких сотен или тысяч бит, числами. Поэтому шифрование больших сообщений производится крайне медленно.

#### Пример шифрования

Пусть два абонента, обменивающиеся через Интернет зашифрованными сообщениями, имеют следующие общие параметры:

Р = 11, А = 7.

Кроме того, пользователи 1 и 2 имеют пары закрытых и открытых ключей, вычисляемые также, как в п. 5.3.3:

Пользователь 1: закрытый ключ Х1= 3, открытый ключ Y1 = 73 mod 11 = 2,

Пользователь 2: закрытый ключ Х2 = 9, открытый ключ Y2 = 79mod 11 = 8.

Первый абонент желает передать второму сообщение. Для этого первый абонент запрашивает из *центра распределения ключей* открытый ключ второго абонента Y2 = 8. Теперь он может зашифровать свое сообщение, которое в числовом виде пусть имеет значение m=9.

Первый абонент выбирает случайно число k, например k = 7. Число k должно быть взаимно простым с Р-1. Значение k = 7 не имеет общих делителей с Р-1=10, значит, оно нам подходит. Первый абонент шифрует свое сообщение по формулам:

r = Ak mod P = 77 mod 11=6

e = m \* Y2k mod P = 9 \* 87 mod 11 = 7

Пара чисел (6, 7) будет представлять собой шифротекст и передается второму пользователю. Второй пользователь, получив (6,7) и используя свой закрытый ключ Х2 = 9 для расшифрования сообщения, вычисляет

Описание: m=e \times r^{P-1-X_2} \: mod \: P = 7 \times 6^{11-1-9} \: mod \: 11 = 7 \times 6^1 \: mod \: 11 = 9

В результате он действительно получает исходное сообщение m.

### Криптографические системы на эллиптических кривых

В 1985 году американские ученые Н. Коблиц (Neal Koblitz) и В. Миллер (Victor Miller) предложили использовать для криптосистем с открытым ключом теорию эллиптических кривых. Дальнейшие исследования подтвердили наличие подходящих свойств у этих математических функций и привели к созданию реальных криптографических систем, использующих математический аппарат эллиптических кривых. С 1998 года использование эллиптических кривых для решения криптографических задач, таких, как *цифровая подпись*, было закреплено в стандартах США *ANSI* X9.62 и *FIPS* 186-2, а в 2001 году аналогичный стандарт, ГОСТ Р34.10-2001, был принят и в России.

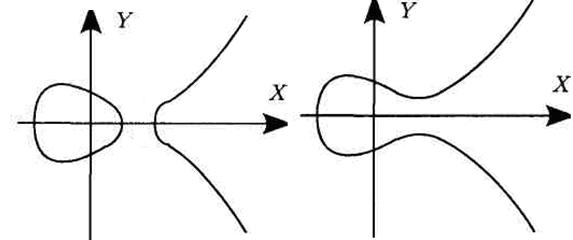
Основное достоинство **криптосистем на эллиптических кривых** состоит в том, что по сравнению с другими асимметричными криптосистемами, рассмотренными нами ранее, они обеспечивают существенно более высокую криптостойкость при равных затратах на обработку и вычисления. Это объясняется тем, что *вычисление* обратных функций на эллиптических кривых значительно сложнее, чем, например, *вычисление* *дискретных логарифмов* (алгоритмы Диффи-Хеллмана и Эль-Гамаля) или решение задачи факторизации (*алгоритм* *RSA*). В результате тот уровень стойкости, который достигается, скажем, в *RSA* при использовании 1024-битовых модулей, в системах на эллиптических кривых реализуется при размере модуля 160 *бит*, что обеспечивает более простую как программную, так и аппаратную реализацию.

*Криптография* эллиптических кривых использует достаточно сложный аппарат высшей алгебры, поэтому мы, в рамках данного учебного пособия, не сможем подробно рассмотреть используемые на практике алгоритмы и выполнить соответствующие примеры вычислений. Сформулируем основные принципы построения криптографических систем с использованием эллиптических кривых.

В криптографии используются *эллиптические кривые* на плоскости, определяемые уравнениями вида

Y2= X3+ аХ + b mod р,

где р – некоторое большое *простое число*, а a и b – *константы*. *График* эллиптической кривой при разных значениях параметров а и b имеет вид, как на [рис. 11.1](https://www.intuit.ru/studies/courses/691/547/lecture/12391?page=4" \l "image.11.1).



**Рис. 11.1.**Варианты графиков эллиптических кривых

Принцип использования эллиптических кривых следующий. Для группы пользователей выбирается общая *эллиптическая кривая* Е и некоторая точка G на ней. Закрытым ключом пользователя выступает некоторое *целое число* с, а открытым – точка D на кривой Е, полученная в результате специального преобразования *композиции* с использованием числа с. Параметры кривой и *список* открытых ключей абонентов, как и обычно, передаются всем пользователям сети. Открытые и закрытые ключи пользователей используются для выполнения операций шифрования и расшифрования в зависимости от назначения алгоритма.

С помощью эллиптических кривых могут быть реализованы многие известные протоколы с открытым ключом. Любая *криптосистема*, основанная на дискретном логарифмировании, легко может быть перенесена на *эллиптические кривые*. Например, можно заменить математические *операции* вида у = gхmod р на *операции* математического аппарата эллиптических кривых (*операции* вычисления композиции точек) в алгоритмах формирования ключа Диффи-Хеллмана или вычисления цифровой подписи Эль-Гамаля. В результате получатся те же алгоритмы, но с другими математическими операциями.

Несмотря на сложность математического аппарата эллиптических кривых, существуют эффективные вычислительные методы, позволяющие достаточно быстро реализовывать необходимые расчеты. За счет использования модуля меньшей длины *операции* генерации ключей и шифрования выполняются быстрее, чем, скажем, в алгоритме *RSA* или классическом алгоритме Диффи-Хеллмана. Криптографические методы на эллиптических кривых считаются перспективными и, закрепленные в различных стандартах, находят применение в современных системах защиты информации.

### Возможные атаки при использовании алгоритмов асимметричного шифрования

#### Атака "человек-в-середине"

Попробуем проанализировать простейший протокол шифрования с открытым ключом, рассмотренный в ["Введение в криптографию с открытым ключом"](https://www.intuit.ru/studies/courses/691/547/lecture/12387), с точки зрения возможности проведения злоумышленником различных атак. Вспомним, что этот протокол предусматривал следующие действия пользователей. Если пользователь А желает передать секретное сообщение пользователю Б так, чтобы никто другой не смог его прочитать, он должен получить от пользователя Б открытый ключ UБ и зашифровать свое сообщение этим открытым ключом. Зашифрованное сообщение может пересылаться по любому каналу связи, например, по электронной почте. Получив сообщение от пользователя А, пользователь Б может расшифровать его своим закрытым ключом RБ. Такая процедура обмена зашифрованными сообщениями с использованием *асимметричного алгоритма* не позволит противнику, контролирующему открытый канал связи, по перехваченным открытым ключам и зашифрованным сообщениям восстановить исходные сообщения. Это обеспечивается свойствами *односторонней функции*, а именно, сложностью вычисления обратной функции.

Однако такая схема уязвима для атак типа "man-in-the-middle" ( **"человек-в-середине"** ). Эта атака заключается в следующем. Допустим, злоумышленник может не только перехватывать сообщения, но и заменять их другими, т.е. имеет возможность осуществлять *активную атаку*. Это вполне возможно в современных сетях передачи данных, например в Интернете, где информация от одного пользователя передается другому через множество промежуточных узлов, не контролируемых этими пользователями. Злоумышленником может быть, например, системный администратор сети. Такой нарушитель может не только перехватывать сообщения пользователей, но изменять, удалять или заменять их своими. Он может выдавать себя за одного из участников сеанса связи. Вот как может производиться атака "man-in-the-middle":

1. Пользователь Б посылает пользователю А свой открытый ключ UБ. Противник перехватывает этот ключ, сохраняет его и заменяет его своим открытым ключом UП.
2. Пользователь А шифрует свое сообщение М полученным открытым ключом UП, предполагая, что использует открытый ключ абонента Б, и пересылает зашифрованное сообщение пользователю Б.
3. Злоумышленник перехватывает это сообщение, расшифровывает его своим закрытым ключом RП, читает или меняет, а затем зашифровывает открытым ключом пользователя Б и посылает пользователю Б.

Аналогично взломщик перехватит и открытый ключ пользователя А, чтобы читать ответы пользователя Б. В результате нарушитель сможет читать (а, возможно, и изменять) всю корреспонденцию абонентов. Пользователи А и Б, скорее всего, ничего не заподозрят, так как у них нет способа проверить, действительно ли они общаются друг с другом.

На практике разработано несколько способов предотвращения атаки "man-in-the-middle". Один из способов заключается в разделении каждого зашифрованного сообщения на две части, каждая из которых бесполезна без другой. Части сообщения пересылаются по очереди и не могут быть расшифрованы по отдельности. Вот как может выглядеть этот протокол для обмена сообщениями между двумя пользователями А и Б:

1. Пользователи А и Б обмениваются открытыми ключами.
2. Пользователь А шифрует свое сообщение открытым ключом пользователя Б и пересылает половину зашифрованного сообщения пользователю Б.
3. Пользователь Б шифрует свое сообщение открытым ключом пользователя А и пересылает половину зашифрованного сообщения пользователю А.
4. Пользователь А пересылает вторую половину зашифрованного сообщения пользователю Б.
5. Пользователь Б соединяет обе полученные половины сообщения от пользователя А и расшифровывает его своим закрытым ключом. Затем посылает вторую половину своего зашифрованного сообщения пользователю А.
6. Пользователь А складывает полученные от пользователя Б половины сообщения и расшифровывает его своим закрытым ключом.

Этот усовершенствованный протокол не позволит злоумышленнику читать или изменять корреспонденцию пользователей А и Б. Нарушитель, как и раньше, может подменить открытые ключи абонентов, а также перехватить передаваемые между ними данные. Однако, получив на шаге 2 протокола в свое распоряжение первую половину зашифрованного сообщения от А к Б, он не сможет расшифровать ее своим закрытым ключом и снова зашифровать открытым ключом абонента Б. Абоненты А и Б тоже не смогут прочитать сообщения до окончания протокола (шагов 5 и 6), но в этом нет ничего плохого, так как в результате они получат корректную корреспонденцию. Для осуществления протокола процесс разделения сообщения на две части может производиться разными способами, например, каждый нечетный байт помещается в первое сообщение, а каждый четный – во второе или как-то иначе.

Атаки "человек-в-середине" можно избежать и другими способами, например, добавляя к передаваемым открытым ключам цифровые подписи специального удостоверяющего центра.

#### Атака на основе выбранного открытого текста

Алгоритмы с открытым ключом чувствительны к атакам по выбранному открытому тексту. Как известно, такая атака имеет место, если криптоаналитик имеет возможность не только использовать предоставленные ему пары "текст-шифротекст", но и сам формировать нужные ему тексты и шифровать их.

Факт возможности проведения атаки по выбранному открытому тексту объясняется следующим образом. Предположим, мы используем *асимметричный алгоритм* F для согласования общего секретного ключа. Пусть один из абонентов отправил другому 64-битовый сеансовый ключ K, зашифрованный открытым ключом y другого абонента C=F(K, y). Злоумышленник, перехватив зашифрованное сообщение С, не сможет его, конечно, дешифровать, так как не имеет закрытого ключа x. Однако нарушитель может поступить по-другому, а именно, попытаться подобрать подходящее значение К. Для этого нужно зашифровать все возможные 64-битовые комбинации открытых текстов открытым ключом y и сравнить результаты с С. Это возможно, так значения y и C передавались в открытом виде. Особенно актуальна угроза такой атаки, если число возможных исходных сообщений не очень велико, например, если длина исходного сообщения мала или если не все исходные тексты допустимы на практике.

Для того, чтобы избежать возможности такой атаки, используют рандомизированные (или вероятностные) алгоритмы шифрования и формирования ЭЦП с открытым ключом. Такие алгоритмы шифруют одно и то же сообщение при наличии одинакового ключа каждый раз по-разному, так как используют некоторый случайный элемент. Примерами рандомизированных алгоритмов с открытым ключом могут служить алгоритмы Эль-Гамаля и алгоритмы формирования ЭЦП по ГОСТ Р34.10.

Другим вариантом предотвращения атаки на основе выбранного открытого текста является добавление в шифруемое сообщение некоторой дополнительной "случайной" информации, например, метки даты времени.

### Ключевые термины

**Алгоритм RSA** – *алгоритм* шифрования с открытым ключом. Название алгоритма составлено из первых букв фамилий авторов: Р.Райвеста (R.Rivest), А.Шамира (A.Shamir) и Л.Адлемана (L.Adleman). *Алгоритм* *RSA* основан на сложности задачи факторизации больших чисел. Данный *алгоритм* является, возможно, наиболее популярным и широко применяемым *асимметричным алгоритмом* в криптографических системах.

**Алгоритм Диффи-Хеллмана** – *алгоритм* шифрования с открытым ключом. Этот *алгоритм* основан на трудности вычислений *дискретных логарифмов*. *Алгоритм Диффи-Хеллмана* может использоваться для распределения ключей, которые могут быть использованы для симметричного шифрования.

**Алгоритм Эль-Гамаля** – *алгоритм* шифрования с открытым ключом, основанный на трудности вычислений *дискретных логарифмов*. *Алгоритм* Эль-Гамаля может быть использован для шифрования данных, для формирования цифровой подписи и для согласования общего ключа. Этот *алгоритм* фактически использует схему Диффи-Хеллмана, чтобы сформировать общий секретный *ключ* для абонентов, передающих друг другу сообщение, и затем сообщение шифруется путем умножения его на этот *ключ*.

**Атака "человек-в-середине"** (англ. "man-in-the-middle") – термин в криптографии, обозначающий ситуацию, когда атакующий способен читать и видоизменять по своей воле сообщения, которыми обмениваются корреспонденты, причём ни один из последних не может догадаться о его присутствии в канале связи.

**Криптосистемы на эллиптических кривых** – *группа* алгоритмов с открытым ключом, использующих в качестве математического аппарата свойства эллиптических кривых на плоскости.

### Краткие итоги

*Алгоритм* *RSA* – *алгоритм* шифрования с открытым ключом. *Алгоритм* *RSA* основан на сложности задачи факторизации больших чисел. Математические основы алгоритма *RSA* следующие. Выбираются два больших простых числа Р и Q и вычисляется *произведение* N = PQ. После этого определяется вспомогательное число f = (Р - l)(Q - 1). Затем случайным образом выбирается число d < f и взаимно простое с f. Далее необходимо найти число е, такое, что еd mod f = 1. Числа d и N будут открытым ключом пользователя, а *значение* е – закрытым ключом. Шифруемое сообщение должно быть представлено в цифровом виде и разбито на блоки m1, m2, m3, ... , где mi < N. Зашифрованное сообщение будет состоять из блоков ci = mid mod N. Расшифровывание производится по формуле mi = ce mod N. *Алгоритм* *RSA* может использоваться для шифрования данных небольшого размера, формирования электронной цифровой подписи, а также и в протоколах обмена ключами для симметричных систем шифрования.

*Алгоритм Диффи-Хеллмана* – *алгоритм* шифрования с открытым ключом. Этот *алгоритм* основан на трудности вычислений *дискретных логарифмов*. *Алгоритм Диффи-Хеллмана* может использоваться для распределения ключей. Принцип работы алгоритма следующий. Вначале выбираются большое *простое число* Р и число А, 1 < A < P-1, такое, что все числа из интервала [1, 2, ..., Р-l] могут быть представлены как различные степени А mod Р. Это – общие параметры. Затем два пользователя выбирают себе закрытые ключи Х1 и Х2 (Xi<P). На основе закрытых ключей пользователи вычисляют открытые ключи Описание: Y_i=A^{X_i} \: mod \: P, которыми они обмениваются. Из чисел Y1 и Y2, а также своих закрытых ключей каждый из абонентов может сформировать общий секретный *ключ* Z для сеанса симметричного шифрования: Описание: Z=(Y_2)^{X_1} \: mod \: P==(Y_1)^{X_2} \: mod \: P .

*Алгоритм* Эль-Гамаля может быть использован для шифрования данных, для формирования цифровой подписи и для согласования общего ключа. Этот *алгоритм* фактически использует схему Диффи-Хеллмана, чтобы сформировать общий секретный *ключ* для абонентов, передающих друг другу сообщение, и затем сообщение шифруется путем умножения его на этот *ключ*. Общими открытыми параметрами в криптосистеме системе Эль-Гамаля являются числа Р (большое *простое число*) и А (А< P). Закрытыми ключами абонентов являются числа Хi, 1 < Х i < Р-1, открытыми ключами – значения Описание: Y_i:Y_i=A^{X_i} \: mod \: P. *Пользователь*, передающий сообщение, выбирает случайное число k, взаимно простое с Р-1, и вычисляет числа Описание: r=A^k \: mod \: P, e=m\times Y_2^k \: mod \: P. Пара чисел (r, е), являющаяся шифротекстом, передается другому пользователю. Для расшифрования сообщения необходимо вычислить Описание: m=e \times r^{P-1-X_2}\: mod \: P. В результате получается исходное сообщение m.

Криптосистемы на эллиптических кривых – самая молодая *группа* алгоритмов с открытым ключом, использующих в качестве математического аппарата свойства эллиптических кривых на плоскости. Основное отличие таких систем состоит в том, что по сравнению с асимметричными криптосистемами, предложенными ранее, они обеспечивают существенно более высокую криптостойкость при равных затратах на обработку и вычисления. Это объясняется тем, что *вычисление* обратных функций на эллиптических кривых значительно сложнее, чем, например, *вычисление* *дискретных логарифмов* или решение задачи факторизации. В результате тот уровень стойкости, который достигается, скажем, в *RSA* при использовании 1024-битовых модулей, в системах на эллиптических кривых реализуется при размере модуля 160 *бит*, что обеспечивает более простую как программную, так и аппаратную реализацию.

Несмотря на достаточную *надежность* алгоритмов шифрования с открытым ключом, существует возможность проведения атак в системах, использующих асимметричное *шифрование*. Это связано с тем, что *атака* может быть направлена не на сам *алгоритм* шифрования, а на протокол, использующий этот *алгоритм*. Для исключения возможностей проведения различных атак в системах шифрования с открытым ключом применяют специальные меры, например, заверяют открытые ключи пользователей цифровыми подписями удостоверяющего центра или добавляют к шифруемым сообщениям некоторую случайную информацию.

#### Вопросы для самопроверки

1. Для каких целей может применяться алгоритм RSA?
2. Опишите процесс шифрования с использованием алгоритма RSA.
3. Для каких целей может применяться *алгоритм Диффи-Хеллмана*?
4. Опишите последовательность действий при использовании *алгоритма Диффи-Хеллмана*.
5. Для каких целей может применяться алгоритм Эль-Гамаля?.
6. Опишите последовательность действий при использовании алгоритма Эль-Гамаля.
7. Какие атаки возможны при использовании алгоритмов шифрования с открытым ключом?