ТЕМА 3. Введение в криптографию

3.1. Криптографические алгоритмы

Криптография – совокупность методов преобразования данных, направленных на, то чтобы сделать эти данные бесполезными для противника, для обеспечения:

- конфиденциальности,
- аутентификации,
- неотказуемости,
- целостности.

Механизмы реализации функций криптографии

Для реализации указанных функций используются криптографические технологии шифрования, цифровой подписи и аутентификации.

Конфиденциальность - алгоритмы и методы симметричного и асимметричного шифрования, взаимная аутентификация абонентов на основе многоразовых и одноразовых паролей, цифровых сертификатов, смарт-карт и т. и.

Целостность и подлинность - технологии электронной подписи, основанные на односторонних функциях и асимметричных методах шифрования.

Аутентификация - разрешает устанавливать соединения только между легальными пользователями и предотвращает доступ к средствам сети нежелательных лиц. Абонентам, доказавшим свою легитимность (аутентичность), предоставляются разрешенные виды сетевого обслуживания.

Фундаментальные принципы криптографии

Основой большинства криптографических средств защиты информации является шифрование данных.

Под шифром понимают совокупность процедур и правил криптографических преобразований, используемых для зашифровывания и расшифровывания информации по ключу шифрования.

Под зашифрованием информации понимается процесс преобразования открытой информации (исходного текста) в зашифрованный текст (шифртекст).

Процесс восстановления исходного текста по криптограмме с использованием ключа шифрования называют расшифровыванием (дешифрованием).

Алгоритмы шифрования общедоступны; секретны только ключи (секретность обеспечивается длиной!). 64-бит – простейшие применения, 128-битный код – коммерческие организации, 256 бит – госсекреты.

Принципы:

– Избыточность (для обеспечения подлинности, например, код Хэмминга или Рида-Соломона для определения и коррекции ошибок.);

– Предотвращение повторной отправки посланных ранее сообщений (например, включение в каждое сообщение временного штампа).

Криптографические алгоритмы

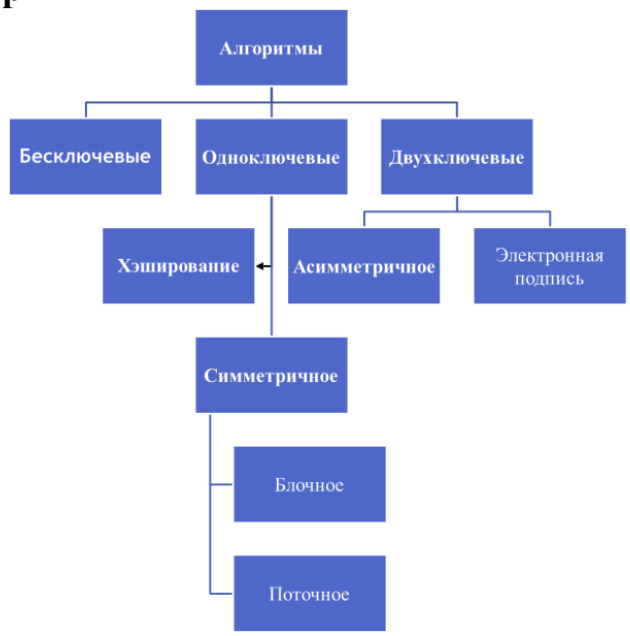

Рисунок 3.1 Криптографические алгоритмы

Хеширование

Хеширование – это метод криптозащиты, представляющий собой контрольное преобразование информации: из данных любого размера путем выполнения криптографических преобразований вычисляется хеш-значение фиксированной длины, однозначно соответствующее исходным данным.

Хеширование может выполняться как с использованием некоторого секретного ключа, так и без него.

Такое криптографическое контрольное суммирование широко используется в различных методах защиты информации, в частности, для подтверждения целостности данных, если использование электронной подписи невозможно (например, из-за большой ресурсоемкости) или избыточно.

Кроме того, данный метод применяется в схемах электронной подписи («подписывается» обычно хеш- значение данных, а не все данные целиком), а также в схемах аутентификации пользователей (при проверке, действительно ли пользователь является тем, за кого себя выдает).

3.2. Симметричная криптография.

Симметричные одноключевые криптосистемы.

Симметричное шифрование использует один и тот же ключ как для зашифрования, так и для расшифрования информации. Фактически оба ключа (зашифрования и расшифрования) могут и различаться, но, если в каком-либо криптоалгоритме их легко вычислить один из другого в обе стороны, такой

алгоритм однозначно относится к симметричному шифрованию.

Симметричное шифрование подразделяется на два вида: блочное и поточное.

Рисунок 3.2 Симметричные криптосистемы

Иногда полагают, что поточное шифрование – это шифрование блоков единичной длины.

Блочное шифрование

Блочное шифрование характеризуется тем, что информация предварительно разбивается на блоки фиксированной длины (например, 64 или 128 бит).

При этом в различных КА или даже в разных режимах работы одного и того же алгоритма блоки могут шифроваться как независимо друг от друга, так и «со сцеплением» – когда результат шифрования текущего блока данных зависит от значения предыдущего блока или от результата шифрования предыдущего блока.

Поточное шифрование

Поточное шифрование применяется прежде всего тогда, когда информацию невозможно разбить на блоки, – скажем, есть некий поток данных.

Алгоритмы поточного шифрования шифруют данные побитно.

В идеале длина ключа потокового шифра равна длине передаваемого сообщения (одноразовый блокнот).

Если биты ключа выбраны случайно – вскрыть шифр невозможно.

Рисунок 3.3. Достоинства и недостатки блочных и поточных шифров

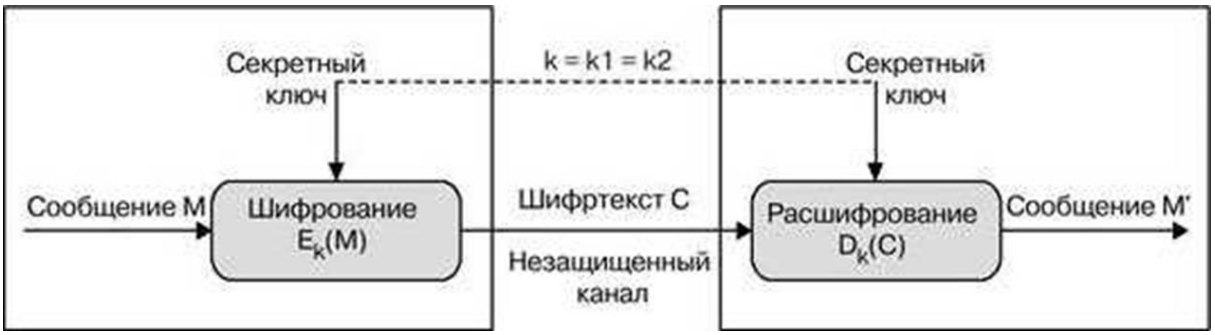

Рисунок 3.4. Общая схема симметричной криптосистемы шифрования

Симметричный алгоритм шифрования DES (Data Encryption Standard).

Алгоритм шифрования данных DES (Data Encryption Standard) был опубликован в 1977 году.

Блочный симметричный алгоритм DES до сих нор остается распространенным алгоритмом, используемым в системах защиты коммерческой информации.

Алгоритм DES построен в соответствии с методологией сети Фейстеля и состоит из чередующейся последовательности перестановок и подстановок.

Алгоритм DES осуществляет шифрование 64-битных блоков данных с помощью 64-битного ключа, в котором значащими являются 56 бит (остальные 8 бит – проверочные биты для контроля на четность).

Обобщенная схема процесса шифрования в блочном алгоритме DES

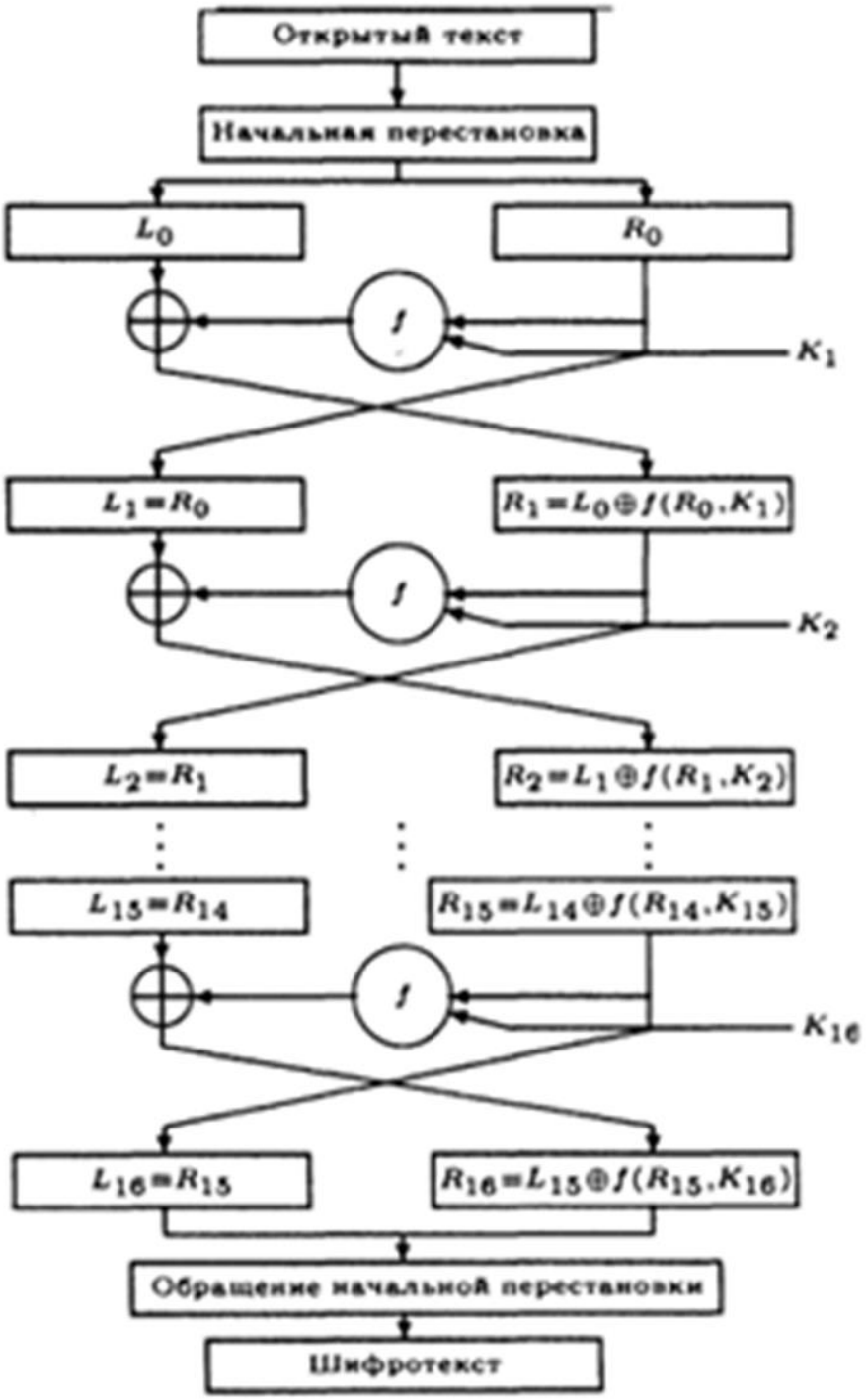

Рисунок 3.5. Структура алгоритма DES

Общий вид цикла преобразования

Если Li и Ri — левая и правая половины, полученные в результате i-й итерации, Ki — 48-битный ключ для цикла i, а f — функция, выполняющая все

подстановки, перестановки и XOR с ключом, то один цикл преобразования можно представить как:

(Li, Ri) = (Ri−1, (Li−1⊕ f(Ri−1, K))

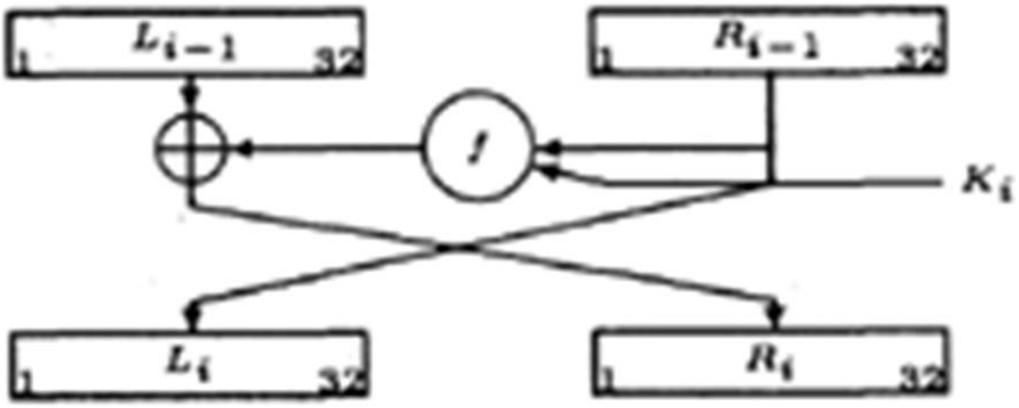

Рисунок 3.6 Структура одного раунда DES

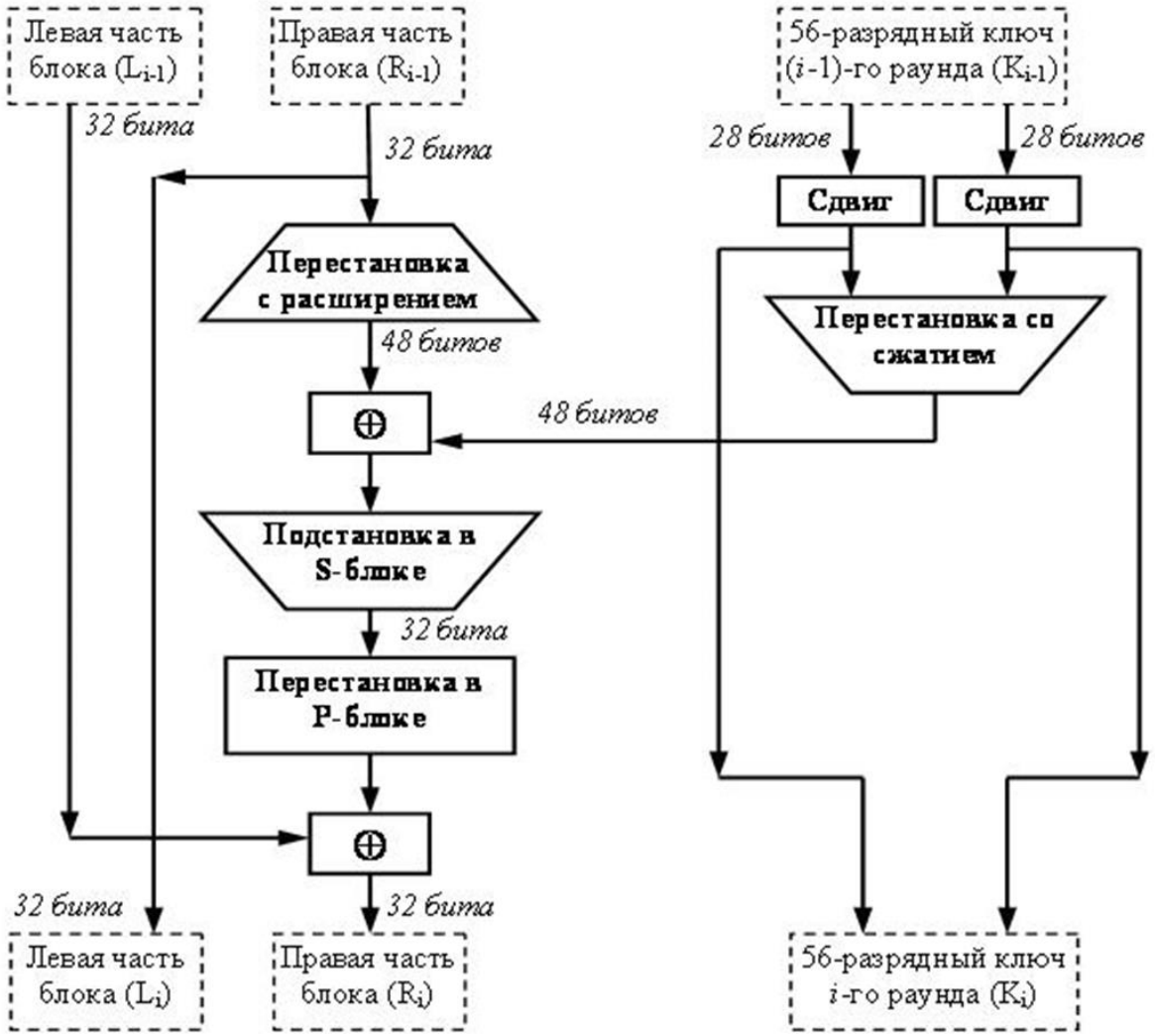

Рисунок 3.7 Подробная структура одного раунда DES

Описание работы алгоритма DES

Алгоритм использует только стандартную арифметику 64-битовых чисел и логические операции, поэтому легко реализуется на аппаратном уровне.

Если при шифровании использовались ключи K1, K2, K3, …, K16, то ключами дешифрования будут K16, K15, K14, …, K1

DES работает с 64-битовым блоком открытого текста.

После первоначальной перестановки блок разбивается на правую и левую половины длиной по 32 бита.

Затем выполняется 16 преобразований (функция f), в которых данные правой половины объединяются с ключом.

Четыре операции выполняются функцией f над правой половиной данных:

– увеличивается до 48 битов с помощью перестановки с расширением;

– объединяется посредством XOR с 48 битами смещенного и переставленного ключа;

– проходит через 8 S-блоков, образуя 32 новых бита;

– переставляется снова.

Описание работы алгоритма DES

На каждом цикле (см. рис.) биты ключа сдвигаются и затем из 56 битов ключа выбираются 48 битов.

Затем результат функции f объединяется с левой половиной с помощью другого XOR. В итоге этих действий появляется новая правая половина, а старая правая становится новой левой половиной.

Эти действия повторяются 16 раз, образуя 16 циклов DES.

После шестнадцатого цикла правая и левая половины объединяются, и алгоритм завершается заключительной перестановкой (обратной по отношению к первоначальной).

S_5	Значения 2-го, 3-го, 4-го и 5-го бит на входе															
	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Значения 1-го и 6-го бит на входе 00	0010	1100	0100	0001	0111	1010	1011	0110	1000	0101	0011	1111	1101	0000	1110	1001
01	1110	1011	0010	1100	0100	0111	1101	0001	0101	0000	1111	1010	0011	1001	1000	0110
10	0100	0010	0001	1011	1010	1101	0111	1000	1111	1001	1100	0101	0110	0011	0000	1110
11	1011	1000	1100	0111	0001	1110	0010	1101	0110	1111	0000	1001	1010	0100	0101	0011

Рисунок 3.8. Программная реализация s-блока с сужением

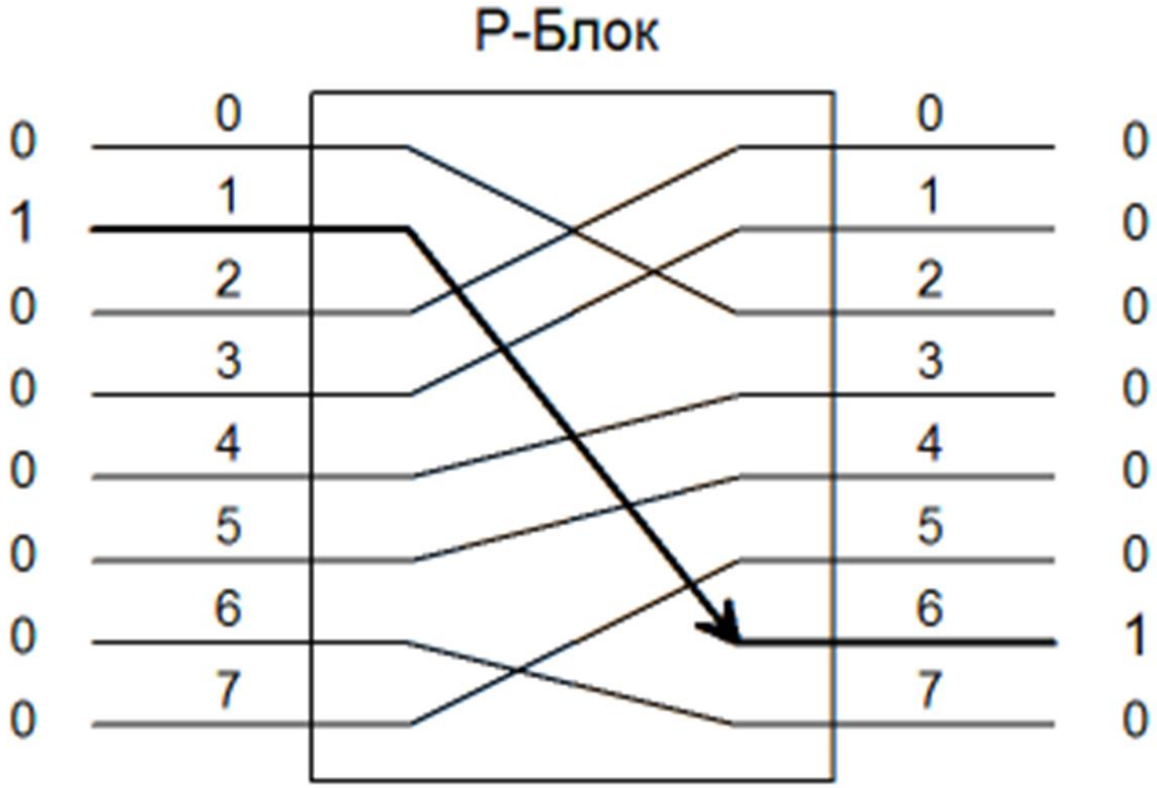

Рисунок 3.9. Программная реализация р-блока

Стандарт шифрования ГОСТ 28147-89

ГОСТ 28147-89 представляет собой симметричный 64-битовый блочный алгоритм с 256-битовым ключом.

Этот алгоритм криптографического преобразования данных предназначен для аппаратной и программной реализации, удовлетворяет криптографическим требованиям и не накладывает ограничений на степень секретности защищаемой информации.

Схема алгоритма ГОСТ 28147-89

Данные, подлежащие зашифровке, разбивают на 64-разрядные блоки.

Эти блоки разбиваются на два субблока N1 и N2 по 32 бит

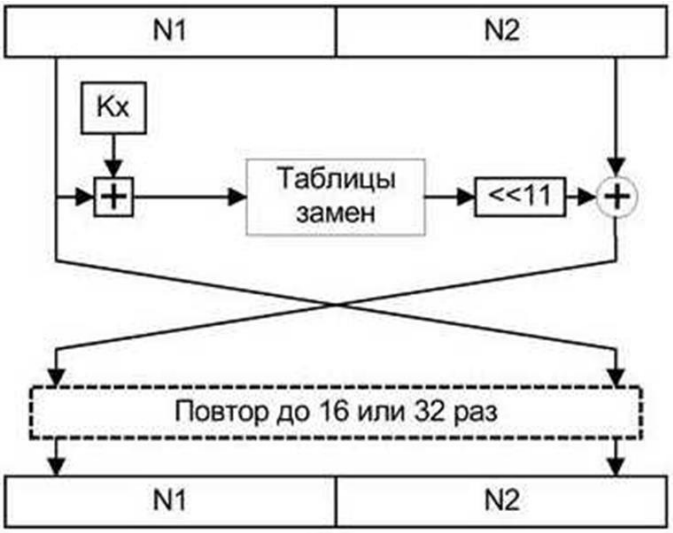

Рисунок 3.10. Схема алгоритма ГОСТ 28147-89

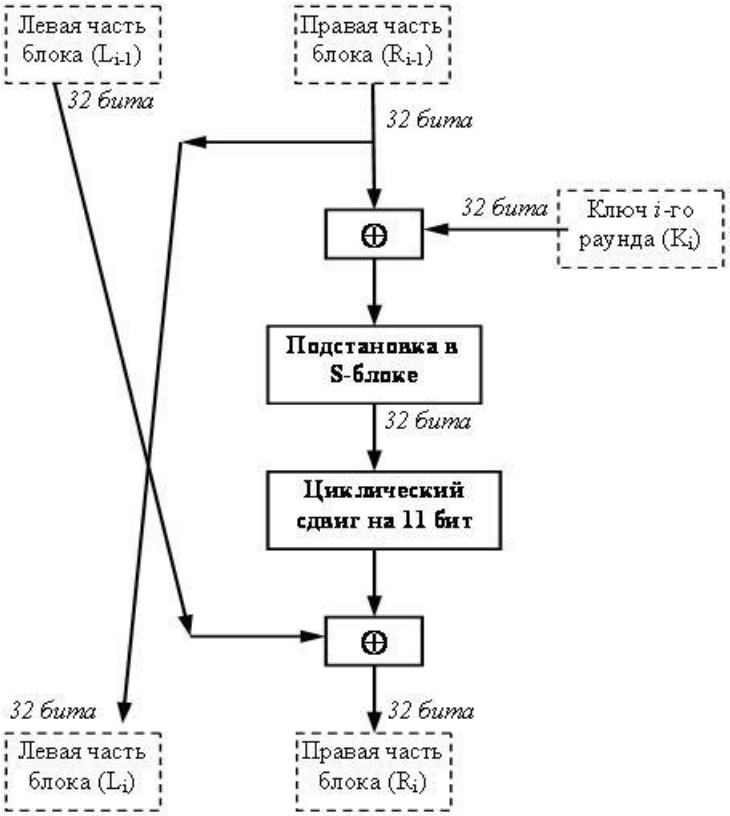

Рисунок 3.11. Структура одного раунда ГОСТ 28147-89

Описание работы ГОСТ 28147-89.

Шифруемый блок данных разбивается на две части, которые затем обрабатываются как отдельные 32-битовые целые числа без знака.

Сначала правая половина блока и подключ раунда складываются по модулю 2^{32}.

Затем производится поблочная подстановка.

32-битовое значение, полученное на предыдущем шаге (обозначим его S), интерпретируется как массив из восьми 4-битовых блоков кода: $S=(S_0,S_1,S_2,S_3,S_4,S_5,S_6,S_7)$.

Далее значение каждого из восьми блоков заменяется на новое, которое выбирается **по таблице замен**

Номер S-блока \ x_{ij}	Значение y_{ij}															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	4	10	9	2	13	8	0	14	6	11	1	12	7	15	5	3
2	14	11	4	12	6	13	15	10	2	3	8	1	0	7	5	9
3	5	8	1	13	10	3	4	2	14	15	12	7	6	0	9	11
4	7	13	10	1	0	8	9	15	14	4	6	12	11	2	5	3
5	6	12	7	1	5	15	13	8	4	10	9	14	0	3	11	2
6	4	11	10	0	7	2	1	13	3	6	8	5	9	12	15	14
7	13	11	4	1	3	15	5	9	0	10	14	7	6	8	2	12
8	1	15	13	0	5	7	10	4	9	2	3	14	6	11	8	12

Рисунок 3.12 Таблица замен ГОСТ 28147-89

В каждой строке таблицы замен записаны числа от 0 до 15 в произвольном порядке без повторений. Значения элементов таблицы замен взяты от 0 до 15, так как в четырех битах, которые подвергаются подстановке, может быть записано целое число без знака в диапазоне от 0 до 15. Значение блока S1 (четыре младших бита 32-разрядного числа S) заменится на число, стоящее на позиции, номер которой равен значению заменяемого блока. Например, в этом случае S1=0 заменится на 4, если S1=1, то оно заменится на 10 и т.д.

Описание ГОСТ 28147-89.

После выполнения подстановки все 4-битовые блоки снова объединяются в единое 32-битное слово, которое затем циклически сдвигается на 11 битов влево.

Наконец, с помощью побитовой операции "сумма по модулю 2" результат объединяется с левой половиной, вследствие чего получается новая правая половина Ri.

Новая левая часть Li берется равной младшей части преобразуемого блока: Li= Ri-1.

Полученное значение преобразуемого блока рассматривается как результат выполнения одного раунда алгоритма шифрования.

Процедуры шифрования и расшифрования.

ГОСТ 28147-89 является блочным шифром, поэтому преобразование данных осуществляется блоками в так называемых базовых циклах.

Базовые циклы заключаются в многократном выполнении для блока данных основного раунда, рассмотренного нами ранее, с использованием

разных элементов ключа и отличаются друг от друга порядком использования ключевых элементов. В каждом раунде используется один из восьми возможных 32-разрядных подключей.

Рассмотрим процесс создания подключей раундов. В ГОСТ эта процедура очень проста, особенно по сравнению с DES. 256-битный ключ K разбивается на восемь 32-битных подключей, обозначаемых K0, K1, K2,K3, K4, K5, K6, K7. Алгоритм включает 32 раунда, поэтому каждый подключ при шифровании используется в четырех раундах в последовательности, представленной в табл. 3.1.

Таблица 3.1. Последовательность использования подключей при шифровании

Раунд	1	2	3	4	5	6	7	8
Подключ	K0	K1	K2	K3	K4	K5	K6	K7
Раунд	9	10	11	12	13	14	15	16
Подключ	K0	K1	K2	K3	K4	K5	K6	K7
Раунд	17	18	19	20	21	22	23	24
Подключ	K0	K1	K2	K3	K4	K5	K6	K7
Раунд	25	26	27	28	29	30	31	32
Подключ	K7	K6	K5	K4	K3	K2	K1	K0

Процесс расшифрования производится по тому же алгоритму, что и шифрование. Единственное отличие заключается в порядке использования подключей Ki. При расшифровании подключи должны быть использованы в обратном порядке, а именно, как указано в табл. 3.2.

Таблица 3.2. Последовательность использования подключей при расшифровании

Раунд	1	2	3	4	5	6	7	8
Подключ	K0	K1	K2	K3	K4	K5	K6	K7
Раунд	9	10	11	12	13	14	15	16
Подключ	K7	K6	K5	K4	K3	K2	K1	K0
Раунд	17	18	19	20	21	22	23	24
Подключ	K7	K6	K5	K4	K3	K2	K1	K0
Раунд	25	26	27	28	29	30	31	32
Подключ	K7	K6	K5	K4	K3	K2	K1	K0

ГОСТ 28147-89 Режимы работы

Режим простой замены: все блоки шифруются независимо друг от друга с разными подключами в разных раундах. Для одинаковых блоков сообщения M

блоки шифртекста будут одинаковыми.

Режим гаммирования: В регистры N1 и N2 записывается 64-битовая синхропосылка (вектор инициализации) и шифруется с использованием СК. Результат подается на вход регистров и снова шифруется с использованием ключа. Получается «одноразовый блокнот».

В режиме гаммирования с обратной связью для заполнения регистров N1 и N2, начиная со 2-го блока, используется результат зашифрования предыдущего блока открытого текста.

Рисунок 3.13 Работа криптосистемы в режиме гаммирования

Симметричные криптоалгоритмы применяются для:

– Абонентского шифрования данных (информации, предназначенной для отправки кому-либо, например, через Интернет).

– В качестве национальных стандартов шифрования: AES, 3DES (США), ГОСТ 28147-89 (Россия, Беларусь).

– При обработке финансовой информации (эмиссия и обработка карт VISA), шифровании потоков информации (канальные шифраторы), шифровании информации, передаваемой по стандарту GSM (A8).

– Для защиты соединений в TLS (протокол используется в приложениях, работающих с сетью Интернет, таких как веб-браузеры, работа с электронной почтой, обмен мгновенными сообщениями и IP-телефония (VoIP)).

Порядок использования систем с симметричными ключами:

1) Симметричный секретный ключ должен создаваться, распространяться и сохраняться безопасным образом.

2) Для получения зашифрованного текста отправитель применяет к исходному сообщению симметричный алгоритм шифрования вместе с секретным симметричным ключом. Таким образом неявно подготавливается аутентификация отправителя и получателя. Только получатель знает симметричный секретный ключ и может расшифровать этот текст.

3) Отправитель передает зашифрованный текст. Симметричный секретный

ключ никогда не передается в открытой форме по незащищенным каналам связи.

4) Получатель применяет к зашифрованному тексту тот же самый симметричный алгоритм шифрования/расшифрования вместе с тем же самым симметричным ключом (который уже есть у получателя) для восстановления исходного текста. Но успешное восстановление аутентифицирует того, кто знает секретный ключ.

Всем системам симметричного шифрования присущи следующие недостатки:

– Принципиальным является требование защищенности и надежности канала передачи секретного ключа для каждой нары участников информационного обмена;

– Предъявляются повышенные требования к службе генерации и распределения ключей, обусловленные тем, что для п абонентов при схеме взаимодействия «каждый с каждым» требуется п×(п-1)/2 ключей, то есть зависимость числа ключей от числа абонентов является квадратичной;

– (например для п = 1000 абонентов требуемое количество ключей будет равно п(п-1)/2 = 499 500 ключей)

ВЫВОД: Без эффективной организации защищенного распределения ключей широкое использование обычной системы симметричного шифрования в больших сетях, и в частности в глобальных сетях, практически невозможно.

3.3. Криптография с открытым ключом.

Концепция криптосистемы с открытым ключом

Эффективными системами криптографической защиты данных являются асимметричные криптосистемы, называемые также криптосистемами с открытым ключом. В таких системах для шифрования данных используется один ключ, а для расшифрования – другой. Первый ключ является открытым и может быть опубликован для использования всеми пользователями системы, которые зашифровывают данные. Расшифрование данных с помощью открытого ключа невозможно.

Для расшифровывания данных получатель шифрованной информации использует второй ключ, который является секретным.

Ключ расшифровывания не может быть определен из ключа шифрования.

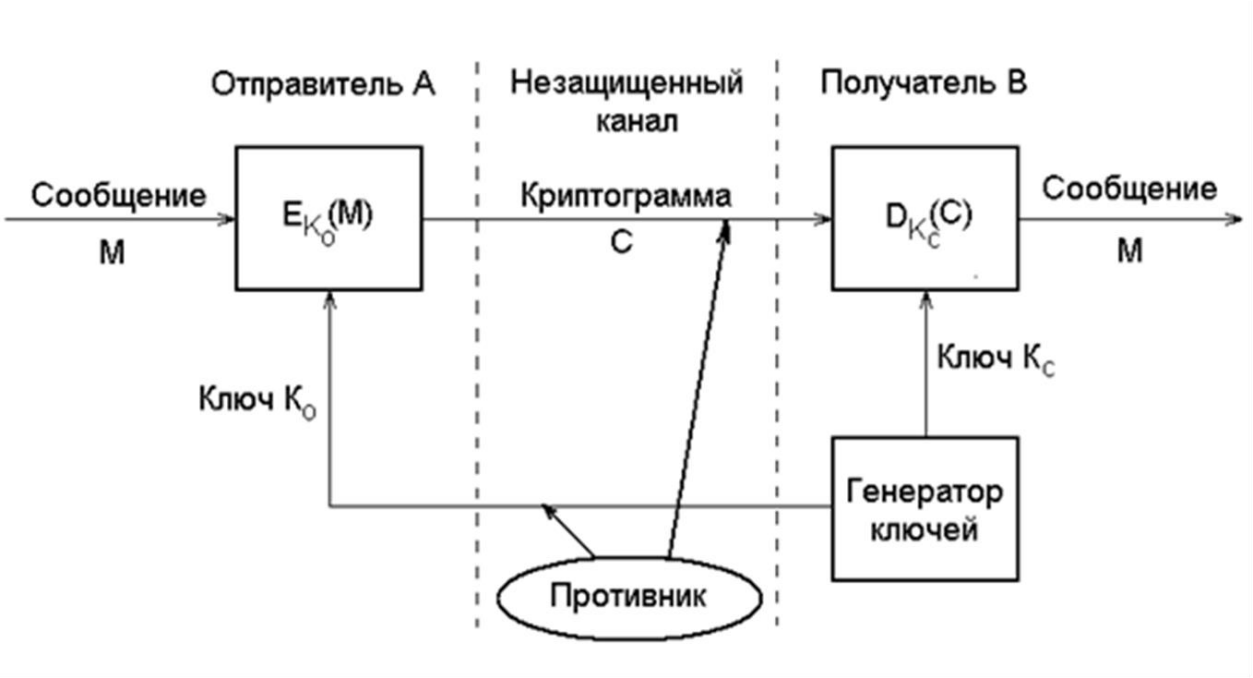

Рисунок 3.14. Обобщенная схема асимметричной криптосистемы с открытым ключом

Описание асимметричной криптосистемы с открытым ключом.

Генератор ключа целесообразно располагать на стороне получателя В (чтобы не пересылать секретный ключ по незащищенному каналу). Значения ключей Ко, Кс – зависят от начального состояния генератора ключей. Раскрытие секретного ключа Кс по известному ключу Ко должно быть вычислительно неразрешимой задачей.

Характерные особенности асимметричных криптосистем:

1) Открытый ключ Ко и криптограмма С могут быть отправлены по незащищенному каналу, т.е. могут быть известны противнику.

2) Алгоритмы шифрования и расшифрования являются открытыми.

3) Защита информации в асимметричной криптосистеме основана на секретности ключа Кс .

Требования к безопасности асимметричной криптосистемы

1) Вычисление пары ключей **Ко** и **Кс** получателем В на основе начального условия должно быть простым.

2) Отправитель А, зная открытый ключ Ко и сообщение М, может легко вычислить криптограмму С=Ек0(М).

3) В, используя секретный ключ Кс и криптограмму С, может легко восстановить исходное сообщение.

4) Противник, зная открытый ключ Ко, при попытке вычислить секретный ключ **Кс** наталкивается на непреодолимую вычислительную проблему.

5) Противник, зная пару (Ко,С), при попытке вычислить исходное

сообщение **M**, наталкивается на непреодолимую вычислительную проблему.

Алгоритм шифрования RSA.

Генерация ключей в RSA осуществляется следующим образом:

1) Выбираются два простых числа p и q (такие, что $p \neq q$).
2) Вычисляется модуль $N=p*q$.
3) Вычисляется значение функции Эйлера от модуля N: $\phi(N)=(p-1)(q-1)$.
4) Выбирается число e, называемое открытой экспонентой, число e должно лежать в интервале $1<e<\phi(N)$, а так же быть взаимно простым со значением функции $\phi(N)$.
5) Вычисляется число d, называемое секретной экспонентой, такое, что $d*e=1(\text{mod } \phi(N))$, то есть является мультипликативно обратное к числу e по модулю $\phi(N)$.

Шифрование и расшифровывание в RSA.

Шифрование: Отправитель A шифрует сообщение m, используя открытый ключ получателя B (e,N):

$C=E(M)=(M\hat{}e)mod(N)$, и отправляет его получателю B.

Расшифровывание: B принимает зашифрованное сообщение C.

Используя закрытый ключ (d,N), получатель B расшифровывает сообщение:

$M=D(C)=(C\hat{}d)mod(N)$

Пример шифрования RSA.

1) Выбираем простые числа (небольшие, чтобы упростить вычисления): $p=3$ и $q=11$
2) Вычисляем модуль $n=p*q=3*11=33$
3) Вычисляем функцию Эйлера от модуля N : $\phi(N)=(p-1)*(q-1)=2*10=20$.
4) Выбираем открытую экспоненту $e=7$
5) Определяем закрытую экспоненту d: $d*e = 1(mod\ \phi(N))$ => $d=(20+1)/7=3$

Пример шифрования RSA

Будем шифровать сообщение «RSA», пусть букве A соответствует цифра 1, B - 2, C - 3 и т.д. (Подобное соответствие вносим для простоты), тогда:
R=18; S=19; A=1.
Открытый ключ: (e,N)=(7,33)

C1=(18^7)mod33=6
C2=(19^7)mod33=13
C3=(1^7)mod33=1
C("RSA")=6131

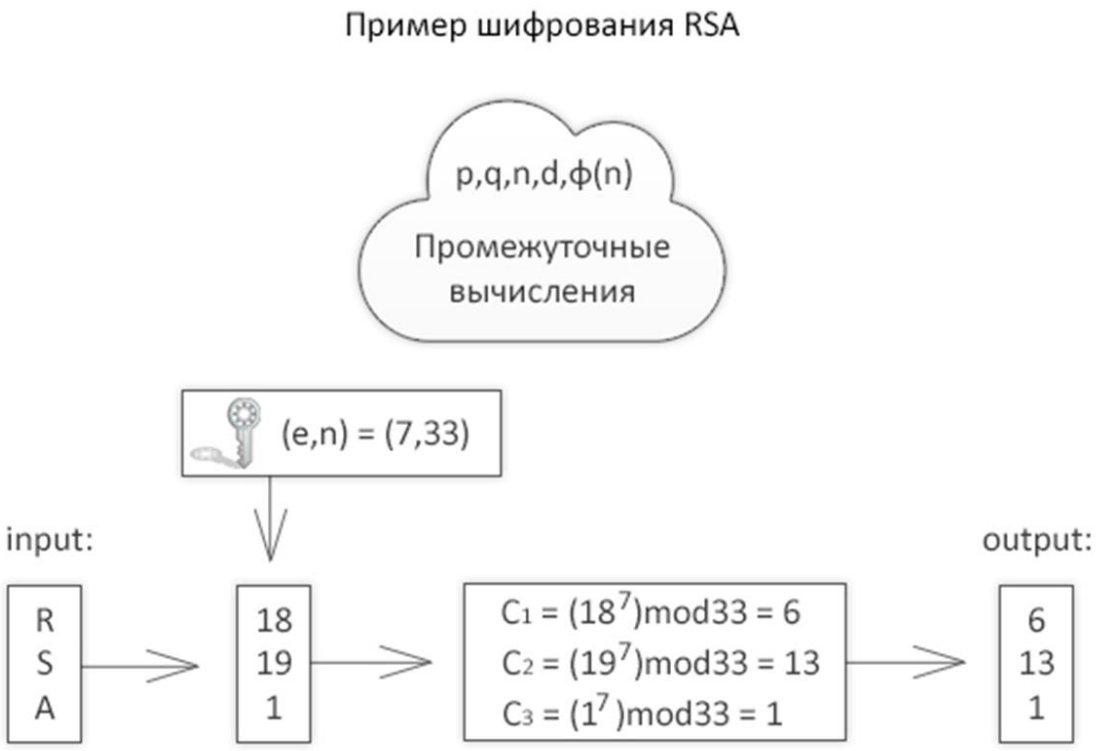

Рисунок 3.15. Пример шифрования RSA

Пример расшифрования RSA.

Используем закрытый ключ: (d,N)=(3,33)

M1=(6^3) mod 33=18

M2=(13^3) mod 33=19

M3=(1^3) mod 33=1

18=R; 19=S; 1=A

Получаем исходное сообщение - RSA.

Безопасность схемы RSA.

Допустим, что у Противника есть открытый ключ Получателя (e,N).

Для того, чтобы расшифровать сообщение C, необходимо знать закрытый ключ (d,N).

Мы знаем, что $d*e=1(mod\phi(N))$, однако Противник не знает $\phi(N)=(p-1)*(q-1)$.

Таким образом задача в общем случае сводится к нахождению простых чисел p и q, которые связаны с известным N следующим образом $N=p*q$.

Для повышения криптостойкости алгоритма рекомендуется:

1) Выбрать два больших простых случайных числа p и q (к примеру, >= 1024 бита каждое), должны быть не слишком различными и быть не слишком близкими

2) Наибольший общий делитель $(p-1)$ и $(q-1)$ должен быть небольшим, в лучшем случае равен двум.

3) Выбрать большое значение открытой экспоненты *e*, как правило, выбирают простые числа Ферма: 17, 257, 65537...

4) Сохранение в секрете закрытого ключа.

Стойкость алгоритма шифрования RSA.

Ассиметричный алгоритм шифрования является стойким, если атакующий, имея два открытых текста *M*1 и *M*2, а так же зашифрованный текст *Ci*, не может с вероятностью большей, чем 1/2 определить к какому из сообщений *M*1 или *M*2 относится *Ci*

На практике к сообщению добавляют некоторую случайную величину.

Например, если ответ подразумевает «Да» «Нет»: то C1=E("ДаA1B2"), C2=E("Нет"), C3=E("Да"), то C1, C2 и C3 при анализе не совпадут.

Для проверки расшифрованных данных на подлинность также используются хеш-функции.

Алгоритм Эль-Гамаля.

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;
- ✓ 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.

Схема Диффи-Хеллмана формирования секретного ключа

Вычисления производятся по модулю некоторого большого простого числа P.

Специальным образом подбирается некоторое натуральное число A<P.

Если мы хотим зашифровать значение X, то вычисляем

$$Y = A^X \bmod P$$

Экспонента X как раз и называется *дискретным логарифмом Y, который* сложно вычислить.

Число Y можно открыто передавать по любому каналу связи, так как при большом модуле P исходное значение X подобрать будет практически невозможно.

На этом математическом факте основан алгоритм Диффи-Хеллмана для формирования ключа.

Формирование общего ключа

1) Пусть два пользователя (пользователь 1 и пользователь 2), должны выбрать большое простое число P и некоторое специальное число A, $1 < A < P-1$, такое, что все числа из интервала [1, 2, ..., P-1] могут быть представлены как различные степени A mod P.

2) Эти числа должны быть известны всем абонентам системы и могут выбираться открыто (общие параметры).

3) Пользователь 1 выбирает свой закрытый ключ, который держит в секрете - число X1 (X1<P) (желательно формировать с помощью датчика случайных чисел).

4) На основе закрытого ключа пользователь 1 вычисляет число:

$$Y1 = A^{x1} mod\ P$$

которое он посылает второму абоненту.

5) Аналогично поступает Пользователь 2, генерируя X2 и вычисляя:

$$Y2 = A^{x2} mod\ P$$

6) Это значение пользователь 2 отправляет первому пользователю.

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

Таблица 3.2. Формирование общего ключа

	Общие параметры	Открытый ключ	Закрытый ключ
Пользователь 1	P, A	Y_1	X_1
Пользователь 2		Y_2	X_2

Общий секретный ключ Z

8) Из чисел Y1 и Y2, а также своих закрытых ключей каждый из абонентов может сформировать общий секретный ключ Z для сеанса симметричного шифрования.

9) Вот как это должен сделать первый пользователь:

$$Z = Y2^{X1} mod\ P$$

10) Никто другой кроме пользователя 1 этого сделать не может, так как число X1 секретно.

11) Второй пользователь может получить то же самое число Z, используя свой закрытый ключ и открытый ключ своего абонента следующим образом:

$$Z = Y1^{X2} mod\ P$$

12) Если весь протокол формирования общего секретного ключа выполнен верно, значения **Z** у одного и второго абонента должны получиться одинаковыми.

13) Пользователи 1 и 2 могут использовать значение **Z** в качестве секретного ключа для шифрования и расшифрования данных.

14) Таким же образом любая пара абонентов может вычислить

секретный ключ, известный только им.

Безопасность в схеме Диффи-Хеллмана

Противник, не зная секретных чисел X1 и X2, не сможет вычислить число Z.

Не зная X1 и X2, злоумышленник может попытаться вычислить Z, используя только передаваемые открыто P, A, 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.

Пример вычислений по схеме

Пусть два абонента, желающие обмениваться через Интернет зашифрованными сообщениями, решили сформировать секретный ключ для очередного сеанса связи.

Пусть они имеют следующие общие параметры:

P = 11, A = 7.

Каждый абонент выбирает секретное число X и вычисляет соответствующее ему открытое число Y.

Пусть выбраны

$X_1 = 3$, $X_2 = 9$.

Вычисляем

Y1 = 7^3 mod 11 = 2, Y2= 7^9 mod 11 = 8.

Затем пользователи обмениваются открытыми ключами Y1 и Y2.

После этого каждый из пользователей может вычислить общий секретный ключ:

пользователь 1: Z = 8^3 mod 11 = 6.

пользователь 2: Z = 2^9 mod 11 = 6.

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

Вернемся к алгоритму Эль-Гамаля.

В случае шифрования и в случае формирования цифровой подписи каждому пользователю необходимо сгенерировать пару ключей.

Для этого, выбираются некоторое большое простое число P и число A, такие, что различные степени A представляют собой различные числа по модулю P.

Числа P и A могут передаваться в открытом виде и быть общими для всех абонентов сети.

Затем каждый абонент группы выбирает свое секретное число

X_i, $1 < X_i < P-1$, и вычисляет соответствующее ему открытое число

$$Yi: Yi = A^{Xi} mod\ P$$

Таким образом, каждый пользователь может сгенерировать закрытый ключ X_i и открытый ключ Y_i.

Информация о необходимых параметрах системы сведена в следующую таблицу.

Таблица 3.3. Общие параметры системы

	Общие параметры	Открытый ключ	Закрытый ключ
Пользователь 1	P, A	Y_1	X_1
…		…	…
Пользователь i		Y_i	X_i

Шифрование по алгоритму Эль-Гамаля

Сообщение, предназначенное для шифрования, должно быть представлено в виде одного числа или набора чисел, каждое из которых меньше P.

Пусть пользователь 1 хочет передать пользователю 2 сообщение **m**.

В этом случае последовательность действий следующая:

Первый пользователь выбирает случайное число **k**, взаимно простое с **P-1**, и вычисляет числа:

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

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

Пара чисел (r, e), являющаяся шифротекстом, передается второму пользователю.

Второй пользователь, получив (r,e), для расшифрования сообщения вычисляет:

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

где X2 – закрытый ключ пользователя 2. В результате он получает исходное сообщение m.

Безопасность алгоритма Эль-Гамаля

Если злоумышленник узнает или перехватит (P, A, Y_2, 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 X_2 (вычисление X_2 на основании Y_2 — также задача дискретного логарифмирования).

Пример шифрования

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

P = 11, A = 7.

Кроме того, пользователи 1 и 2 имеют пары закрытых и открытых ключей, вычисляемые также, как в Диффи-Хеллмана.

Пользователь 1:

закрытый ключ X_1= 3,

открытый ключ $Y_1 = 7^3 \bmod 11 = 2$,

Пользователь 2:

закрытый ключ X_2 = 9,

открытый ключ $Y_2 = 7^9 \bmod 11 = 8$.

Для передачи Пользователю 2 сообщения, Пользователь 1 запрашивает из центра распределения ключей открытый ключ второго абонента Y_2 = 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 P-1.

Значение k = 7 не имеет общих делителей с P-1=10, значит, оно нам подходит.

Первый абонент шифрует свое сообщение по формулам:

r = A^k mod P = 7^7 mod 11=6

e = m * Y2^k mod P = 9 * 8^7 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 X2 = 9 для расшифрования сообщения, вычисляет

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

В результате он действительно получает исходное сообщение m.

Выводы алгоритм Эль-Гамаля

Алгоритм Эль-Гамаля на практике целесообразно использовать именно для согласования общего ключа сессии, а не прямого шифрования больших сообщений. (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. Шифрование больших сообщений производится крайне медленно.