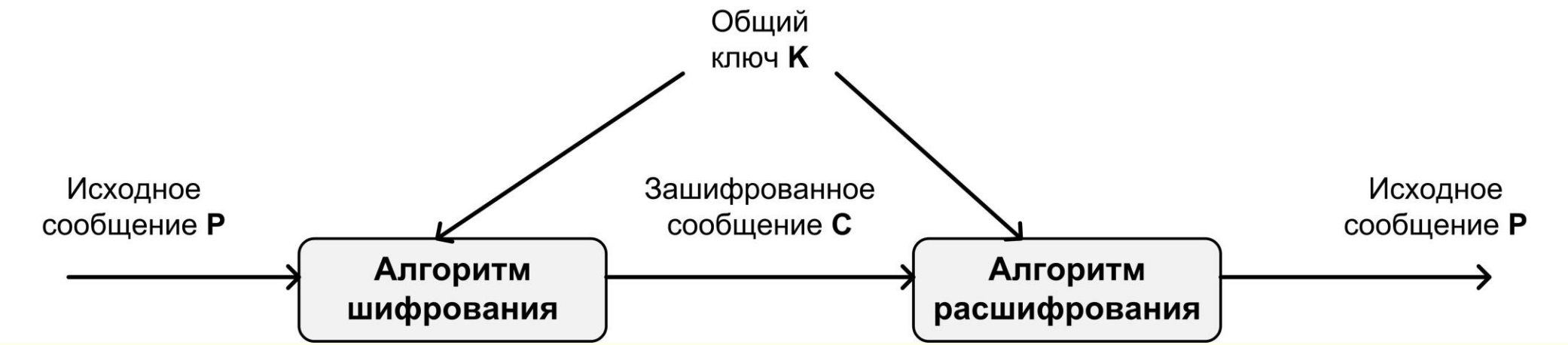
Алгоритмы симметричного шифрования

- Основные понятия
- Криптоанализ
- Области применения
- Блочные алгоритмы шифрования
  - Сеть Фейстеля
  - SP-сеть
  - Алгоритм DES
  - Алгоритм тройной DES
  - Алгоритм Blowfish
  - Алгоритм IDEA
  - Алгоритм ГОСТ 28147-89
  - Алгоритм AES (Rijndael)
  - Алгоритм ГОСТ 34.12-2015
- Поточные алгоритмы шифрования
  - Алгоритм Salsa20
  - Алгоритм ChaCha
- Режимы выполнения алгоритмов симметричного шифрования
  - Режим ECB
  - Режим CBC
  - Режим PCBC
  - Режим CFB
  - Режим OFB
  - Режим CTR
  - Режим RD
  - Режим RD с проверкой целостности
  - Режим GCM
- Создание случайных чисел

Основные понятия



Основные понятия

- Процесс шифрования состоит в использовании определенного алгоритма, на вход которому подаются исходное незашифрованное сообщение, называемое также **plaintext**, и **ключ**. Выходом алгоритма является зашифрованное сообщение, называемое также **ciphertext.** Ключ является значением, не зависящим от шифруемого сообщения. Изменение ключа должно приводить к изменению зашифрованного сообщения.
- Зашифрованное сообщение передается получателю. Получатель преобразует зашифрованное сообщение в исходное незашифрованное сообщение с помощью алгоритма расшифрования и **того же самого ключа, который использовался при шифровании**.
- Незашифрованное сообщение будем обозначать **P** или **M**, от слов **plaintext** и **message.** Зашифрованное сообщение будем обозначать **C**, от слова **ciphertext**.

Основные понятия

- Алгоритмы симметричного шифрования различаются **способом**, которым обрабатывается исходный текст: шифрование **блоками** или шифрование всего **потока**.

Основные понятия

- Безопасность, обеспечиваемая традиционной криптографией, зависит от нескольких факторов.
- Во-первых, криптографический алгоритм должен быть достаточно **сильным**, чтобы передаваемое зашифрованное сообщение **невозможно было расшифровать без ключа**, используя только различные статистические закономерности зашифрованного сообщения или какие-либо другие способы его анализа.
- Во-вторых, **безопасность передаваемого сообщения должна зависеть от секретности ключа, но не от секретности алгоритма**. Алгоритм должен быть проанализирован специалистами, чтобы исключить наличие слабых мест, при которых плохо скрыта взаимосвязь между незашифрованным и зашифрованным сообщениями. К тому же при выполнении этого условия производители могут создавать дешевые аппаратные чипы и свободно распространяемые программы, реализующие данный алгоритм шифрования.
- В-третьих, алгоритм должен быть таким, чтобы **нельзя было узнать ключ, даже зная достаточно много пар (зашифрованное сообщение, незашифрованное сообщение)**, полученных при шифровании с использованием данного ключа.

Криптоанализ

- Процесс, при котором предпринимается попытка узнать **P**, **K** или и то, и другое, называется **криптоанализом**. Одной из возможных атак на алгоритм шифрования является «**лобовая атака**», называемая также «**атакой грубой силы**» - «**brute force атака**». Данная атака состоит в простом переборе всех возможных ключей. Если множество ключей достаточно большое, то подобрать ключ нереально. При длине ключа **n бит** количество возможных ключей равно $\mathbf{2^n}$. Таким образом, чем длиннее ключ, тем более стойким считается алгоритм для лобовой атаки.
- Существуют различные типы атак, основанные на том, что противнику известно определенное количество пар незашифрованное сообщение – зашифрованное сообщение. При анализе зашифрованного сообщения противник часто применяет статистические методы анализа текста. При этом он может иметь общее представление о типе сообщения, например, английский или русский текст, выполняемый файл конкретной ОС, исходный текст на некотором языке программирования и т.д.

Криптоанализ

Во многих случаях криптоаналитик имеет достаточно много информации об исходном тексте. Криптоаналитик может иметь возможность перехвата одного или нескольких незашифрованных сообщений вместе с их зашифрованным видом. Или криптоаналитик может знать основной формат или основные характеристики сообщения. Говорят, что криптографическая схема **абсолютно** безопасна, если зашифрованное сообщение не содержит никакой информации об исходном сообщении.

Говорят, что криптографическая схема **вычислительно** безопасна, если:

- ***Цена** расшифровки сообщения больше цены самого сообщения.*
- ***Время**, необходимое для расшифровки сообщения, больше срока жизни сообщения.*

Криптоанализ

- Диффузия
- Конфузия
- Лавинный эффект – несколько критериев.

1. Лавинный критерий – изменение одного бита входной последовательности приводит к изменению в среднем половины выходных битов для функции f: $\{0,1\}^n \rightarrow \{0,1\}^n$
2. Строгий лавинный критерий – булева функция удовлетворяет строгому лавинному критерию, если при изменении одного из n входных битов каждый выходной бит меняется с вероятностью ровно ½.
3. Критерий независимости битов – при изменении одного входного бита любые два выходных бита меняются независимо друг от друга.
4. Гарантированный лавинный эффект **порядка ɣ** выполняется, если при изменении одного входного бита **меняется как минимум ɣ выходных битов**.

Области применения

Необходимо, чтобы алгоритм симметричного шифрования мог применяться в следующих областях:

- **Шифрование данных.** Алгоритм должен быть эффективен при шифровании как небольших файлов и блоков данных, так и большого потока данных.
- **Создание случайных чисел.** Алгоритм должен быть эффективен для создания любого количества случайных битов.
- **Хеширование.** Алгоритм должен эффективно преобразовываться в одностороннюю хеш-функцию.

Области применения

Платформы

Стандартный алгоритм шифрования должен быть реализован на различных платформах, которые, имеют разные характеристики.

- **Специальная аппаратура.** Алгоритм должен эффективно реализовываться на специализированной аппаратуре, предназначенной для выполнения шифрования / расшифрования.
- **Большие процессоры.** Хотя в приложениях, требующих максимальной скорости, всегда используется специальная аппаратура, программные реализации применяются чаще. Алгоритм должен допускать эффективную программную реализацию на 32-битных процессорах.
- **Процессоры среднего размера.** Алгоритм должен работать на микроконтроллерах и других процессорах среднего размера.
- **Малые процессоры.** Должна существовать возможность реализации алгоритма на смарт-картах, с учетом жестких ограничений на используемую память.

Области применения

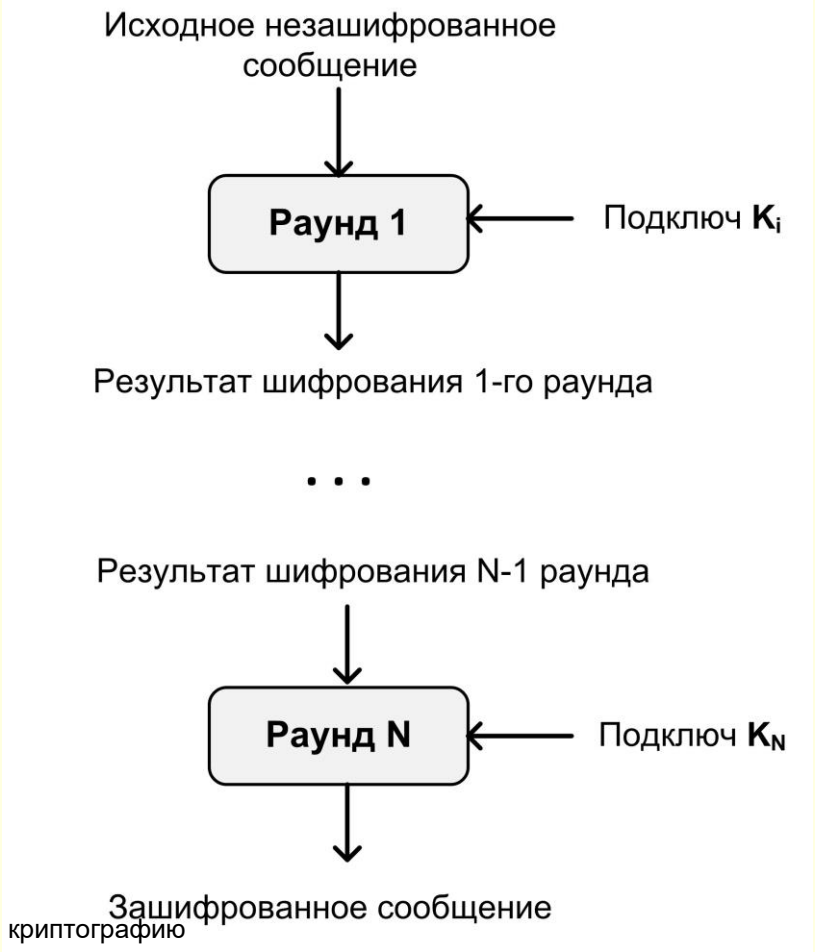
Алгоритм шифрования должен, по возможности, удовлетворять некоторым **дополнительным требованиям**.

- Алгоритм должен быть **простым** с точки зрения написания кода, чтобы минимизировать вероятность программных ошибок.
- Алгоритм должен иметь **плоское пространство ключей**, т.е. допускать любую случайную строку битов нужной длины в качестве возможного ключа. Наличие слабых ключей нежелательно.
- Алгоритм должен **легко модифицироваться для различных уровней безопасности** и удовлетворять как минимальным, так и максимальным требованиям.
- Все операции с данными должны осуществляться над блоками, кратными байту или **32-битному слову**.

Блочные алгоритмы шифрования

$$Y = E_K [X]$$
$$X = D_K [Y]$$

Структура блочного алгоритма симметричного шифрования



Блочные алгоритмы шифрования

Используемые критерии при разработке алгоритмов

- Иметь **размер блока** 64 или 128 бит.
- Иметь **масштабируемый ключ** до 256 бит.
- Использовать **простые операции**, которые эффективны на микропроцессорах, т.е. исключающее или, сложение, табличные подстановки, умножение по модулю. Не должно использоваться сдвигов переменной длины, побитовых перестановок или условных переходов.
- Должна быть возможность реализации алгоритма на **8-битном процессоре с минимальными требованиями к памяти**.

Блочные алгоритмы шифрования

- Использовать **заранее вычисленные подключи**. На системах с большим количеством памяти эти подключи могут быть заранее вычислены для ускорения работы. В случае невозможности заблаговременного вычисления подключей должно произойти только замедление выполнения. Всегда должна быть возможность шифрования данных без каких-либо предварительных вычислений.
- **Число итераций может варьироваться.** Для приложений с маленькой длиной ключа нецелесообразно применять большое число итераций для противостояния дифференциальным и другим атакам. Следовательно, должна быть возможность уменьшить число итераций без существенной потери безопасности (не более чем на уменьшенный размер ключа).

Блочные алгоритмы шифрования

- **По возможности не иметь слабых ключей.** Если это невозможно, то количество слабых ключей должно быть минимальным, чтобы уменьшить вероятность случайного выбора одного из них. Все слабые ключи должны быть заранее известны, чтобы их можно было отбраковать в процессе создания ключа.

Блочные алгоритмы шифрования

Основные операции, используемые в алгоритмах

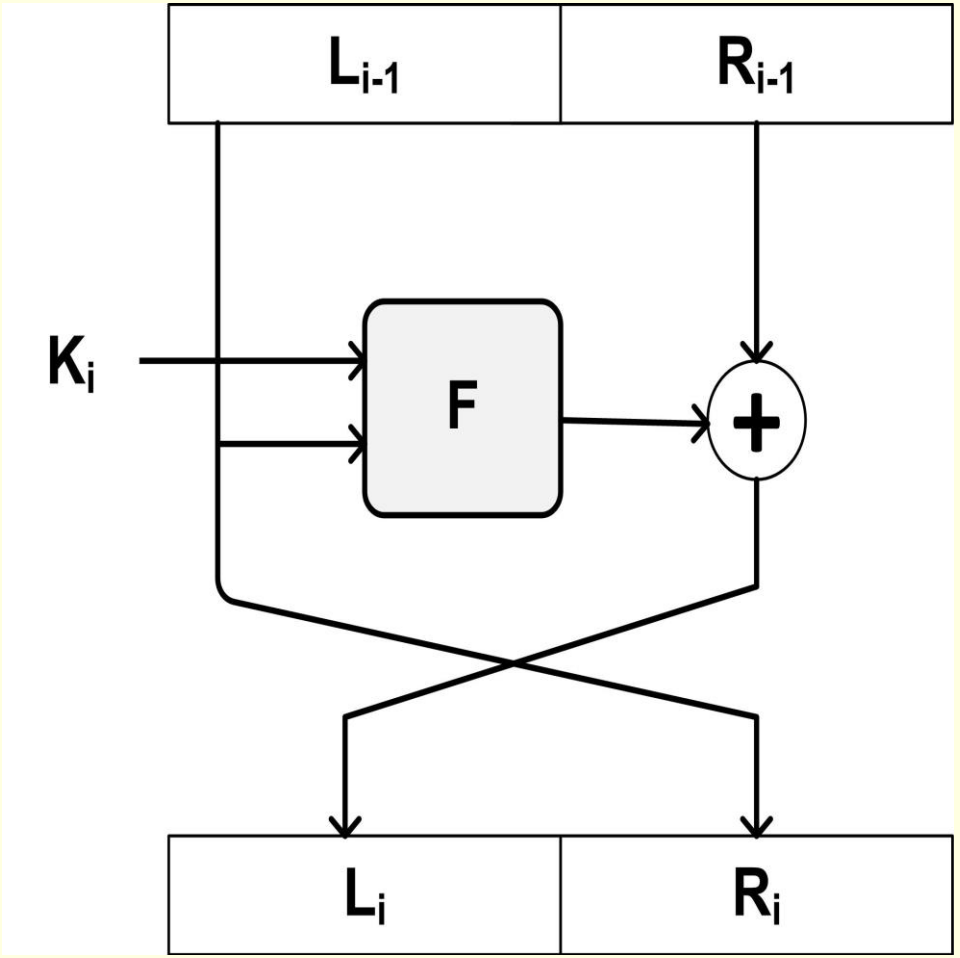
- **Табличная подстановка**, при которой группа битов отображается в другую группу битов. Это так называемые S-box.
- **Перемещение**, с помощью которого биты сообщения переупорядочиваются.
- Операция сложения **по модулю 2**, обозначаемая **XOR** или $\oplus$.
- Операция сложения **по модулю 2^{32}** или по модулю 2^{16}.
- **Циклический сдвиг** на некоторое число битов.

Сеть Фейстеля

- **Входной блок делится на несколько равной длины подблоков, называемых ветвями.** В случае, если блок имеет длину 64 бита, используются две ветви по 32 бита каждая. Каждая ветвь обрабатывается независимо от другой, после чего осуществляется циклический сдвиг всех ветвей влево. Такое преобразование выполняется циклически.

Сеть Фейстеля

Функция **F** называется образующей. Каждый раунд состоит из вычисления функции **F** для одной ветви и побитового выполнения операции **XOR** результата **F** с другой ветвью.



Сеть Фейстеля

- После этого ветви меняются местами. Считается, что оптимальное число раундов должно быть от 8 до 32. Важно то, что увеличение количества раундов значительно увеличивает криптостойкость алгоритма. Возможно, эта особенность и повлияла на столь активное распространение сети Фейстеля, так как для большей криптостойкости достаточно просто увеличить количество раундов, не изменяя сам алгоритм. В последнее время количество раундов не фиксируется, а лишь указываются допустимые пределы.

- **Сеть Фейстеля является обратимой** даже в том случае, если функция **F** не является таковой, **так как для расшифрования не требуется вычислять F^{-1}**. Для расшифрования используется тот же алгоритм, но на вход подается зашифрованное сообщение, и ключи используются в обратном порядке.

Сеть Фейстеля

- В настоящее время все чаще используются различные разновидности сети Фейстеля для 128-битного блока с четырьмя ветвями. Увеличение количества ветвей, а не размерности каждой ветви связано с тем, что наиболее популярными до сих пор остаются процессоры с **32-разрядными словами**, следовательно, оперировать 32-разрядными словами эффективнее, чем с 64-разрядными.
- Основной характеристикой алгоритма, построенного на основе сети Фейстеля, является функция F. Различные варианты касаются также **начального и конечного преобразований**. Подобные преобразования, называемые забеливанием (whitening), осуществляются для того, чтобы выполнить начальную рандомизацию входного текста.

SP-сеть

- SP-сеть (Substitution-Permutation network, подстановочно-перестановочная сеть) — разновидность блочного шифра, предложенная в 1971 году Хорстом Фейстелем. В простейшем варианте представляет собой **«сэндвич» из слоёв двух типов**, используемых многократно по очереди. Первый тип слоя — **P-слой**, состоящий из P-блока большой разрядности, за ним идёт второй тип слоя — **S-слой**, представляющий собой большое количество S-box малой разрядности, потом опять P-слой и т. д. Первым криптографическим алгоритмом на основе SP-сети был «Люцифер» (1971). В настоящее время из алгоритмов на основе SP-сетей широко используется **AES (Rijndael)**.
- В современных алгоритмах вместо S- и P-блоков используются различные математические или логические функции. Любая двоичная функция может быть сведена к S-box, некоторые функции — к P-блоку. Например, к P-блоку сводится циклический сдвиг, сам P-блок является частным случаем S-box. Такие функции, как правило, легко реализуются в аппаратуре, обеспечивая при этом хорошую криптостойкость.

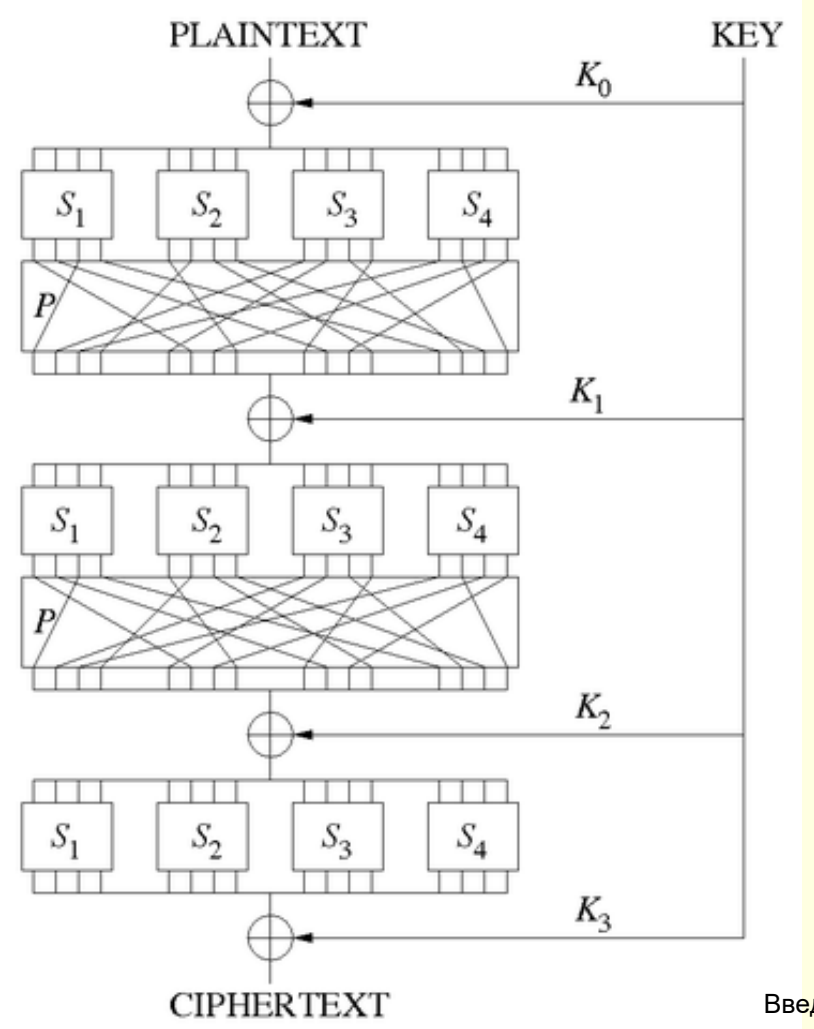
SP-сеть

- Шифр на основе SP-сети получает на вход блок и ключ и совершает несколько чередующихся раундов, состоящих из чередующихся стадий подстановки (**substitution stage**) и стадий перестановки (**permutation stage**).
- Для достижения безопасности достаточно одного S-блока, но такой блок будет требовать большого объёма памяти. Поэтому используются маленькие S-блоки, смешанные с P-блоками.
- Нелинейная стадия подстановки перемешивает биты ключа с битами открытого текста, создавая конфузию Шеннона. Линейная стадия перестановки распределяет избыточность по всей структуре данных, порождая диффузию.

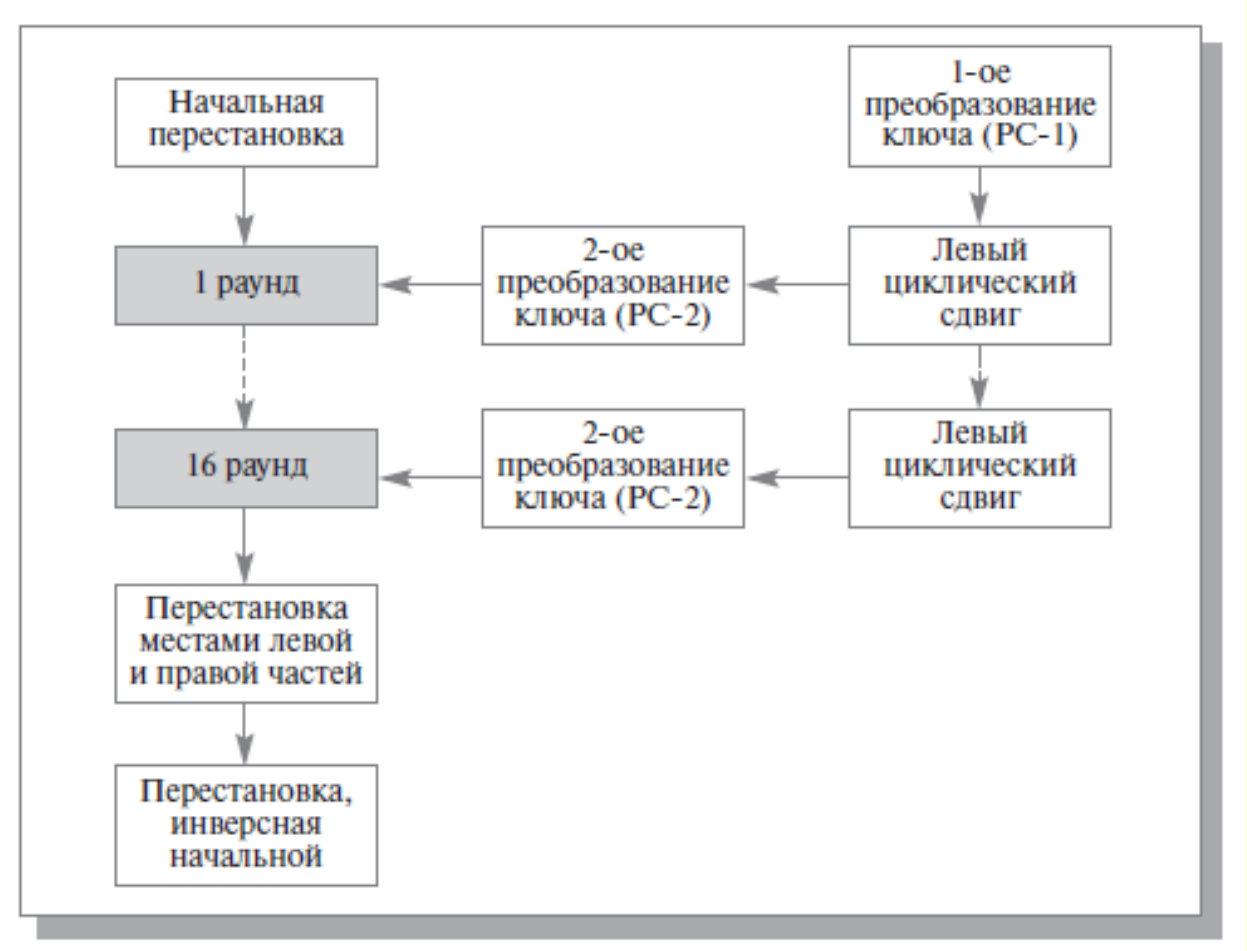
SP-сеть

- **S-box (substitution box)** замещает маленький блок входных бит на другой блок выходных бит. Эта замена должна быть взаимно однозначной, чтобы гарантировать обратимость. Назначение S-box заключается в нелинейном преобразовании, что препятствует проведению линейного криптоанализа. Одним из свойств S-box является лавинный эффект, то есть изменение одного бита на входе приводит к изменению всех бит на выходе.
- **P-блок (permutation box)** — перестановка всех бит: блок получает на вход выход S-box, меняет местами все биты и подает результат S-box следующего раунда. Важным качеством P-блока является возможность распределить выход одного S-box между входами как можно больших S-box.
- Для каждого раунда используется свой, получаемый из первоначального, ключ. Подобный ключ называется раундовым. Он может быть получен как делением первоначального ключа на равные части, так и каким-либо преобразованием всего ключа.

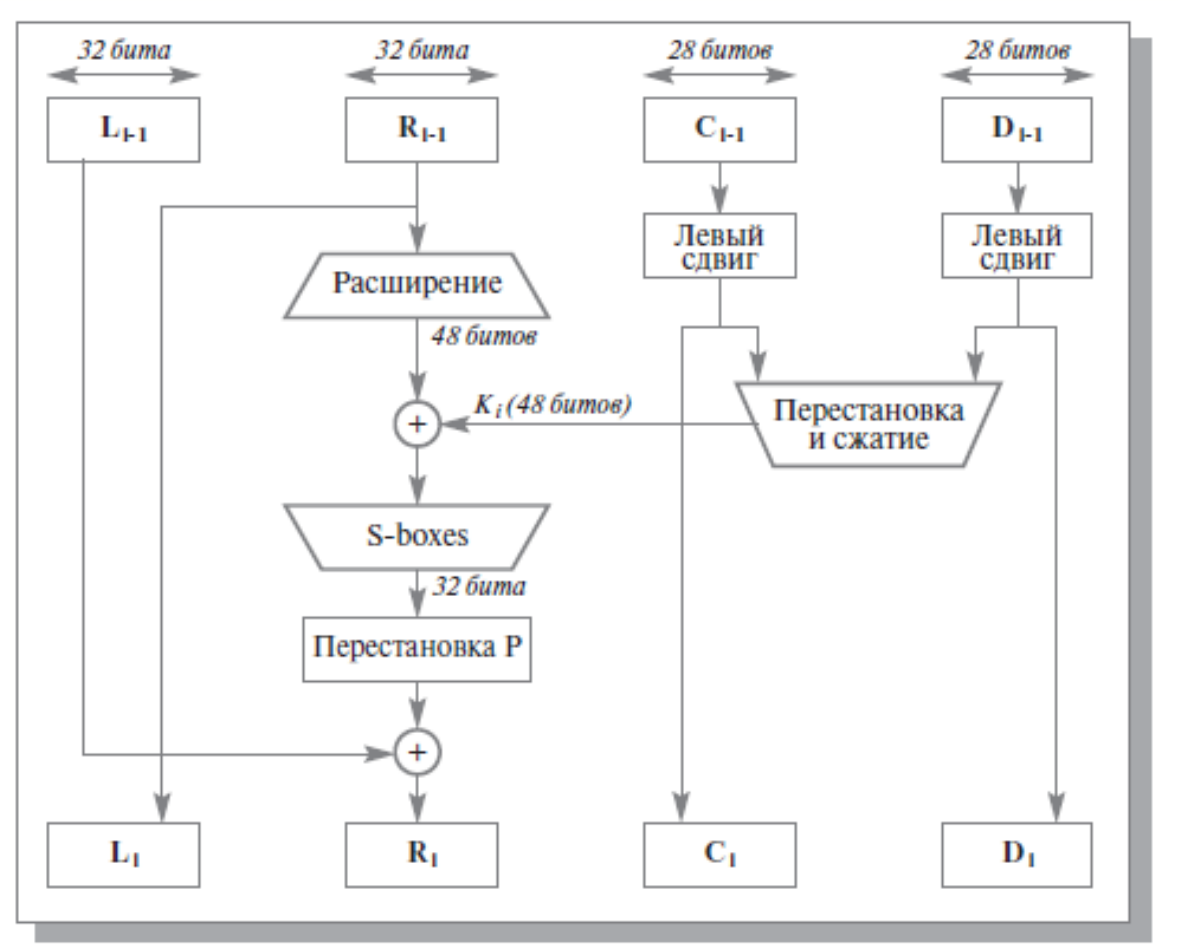
SP-сеть


PLAINTEXT
KEY
K_0
S_1 S_2 S_3 S_4
P
K_1
S_1 S_2 S_3 S_4
P
K_2
S_1 S_2 S_3 S_4
K_3
CIPHERTEXT

Алгоритм DES



Алгоритм DES



Алгоритм DES

- Шифрование
  - $L_i = R_{i-1}$
  - $R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$
- Расшифрование
  - $L^d_0 || R^d_0 = IP$ (зашифрованный текст)
  - Зашифрованный текст = $IP^{-1}(R_{16} || L_{16})$
  - $L^d_0 || R^d_0 = IP(IP^{-1}(R_{16} || L_{16})) = R_{16} || L_{16}$

Алгоритм DES

Шифрование:

- $L_{16} = R_{15}$
- $R_{16} = L_{15} \oplus F\,(R_{15}, K_{16})$

Алгоритм DES

Расшифрование:

- $L^d_1 = R^d_0 = L_{16} = R_{15}$
- $R^d_1 = L^d_0 \oplus F(R^d_0, K_{16}) = R_{16} \oplus F(R^d_0, K_{16}) = (L_{15} \oplus F(R_{15}, K_{16})) \oplus F(R_{15}, K_{16}) = L_{15}$

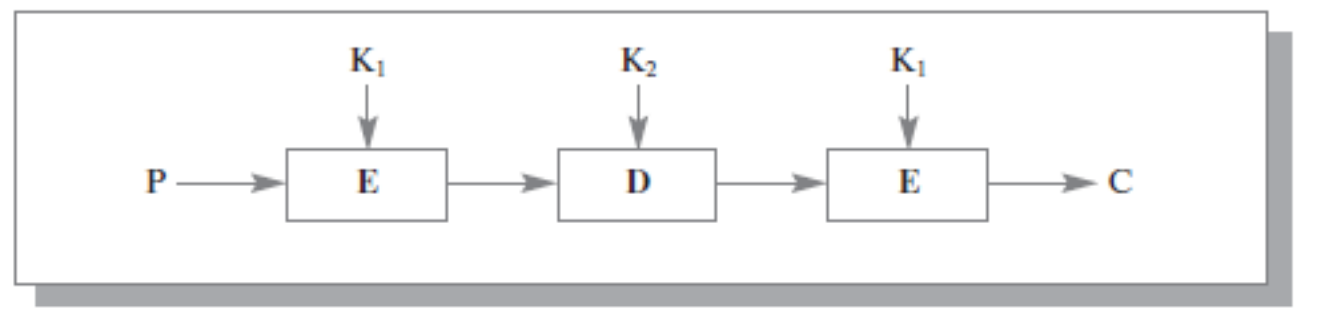
Алгоритм тройной DES

- Недостатки двойного DES
  - $C = E_{k2}[E_{k1}[P]]$
  - $P = D_{k1}[D_{k2}[C]]$

Атака «встреча посередине»:

- $X = E_{k1}[P] = D_{k2}[C]$

Тройной DES с двумя ключами:



Алгоритм Blowfish

- Ключи:

  $K_1, K_2, ..., K_{16}$

- Четыре 32-битных S-boxes с 256 входами каждый:

  $S_{1,0}, S_{1,1}, \ldots S_{1,255}$;

  $S_{2,0}, S_{2,1}, \ldots S_{2,255}$;

  $S_{3,0}, S_{3,1}, \ldots S_{3,255}$;

  $S_{4,0}, S_{4,1}, \ldots S_{4,255}$;

Алгоритм Blowfish

Шифрование:

$X_L = X_{L-1} \oplus K_i$

$X_R = F(X_{L-1}) \oplus X_{R-1}$

Swap X_L и X_R

Алгоритм Blowfish

Функция F:

- Разделить X_L на четыре 8-битных элемента A, B, C, D.

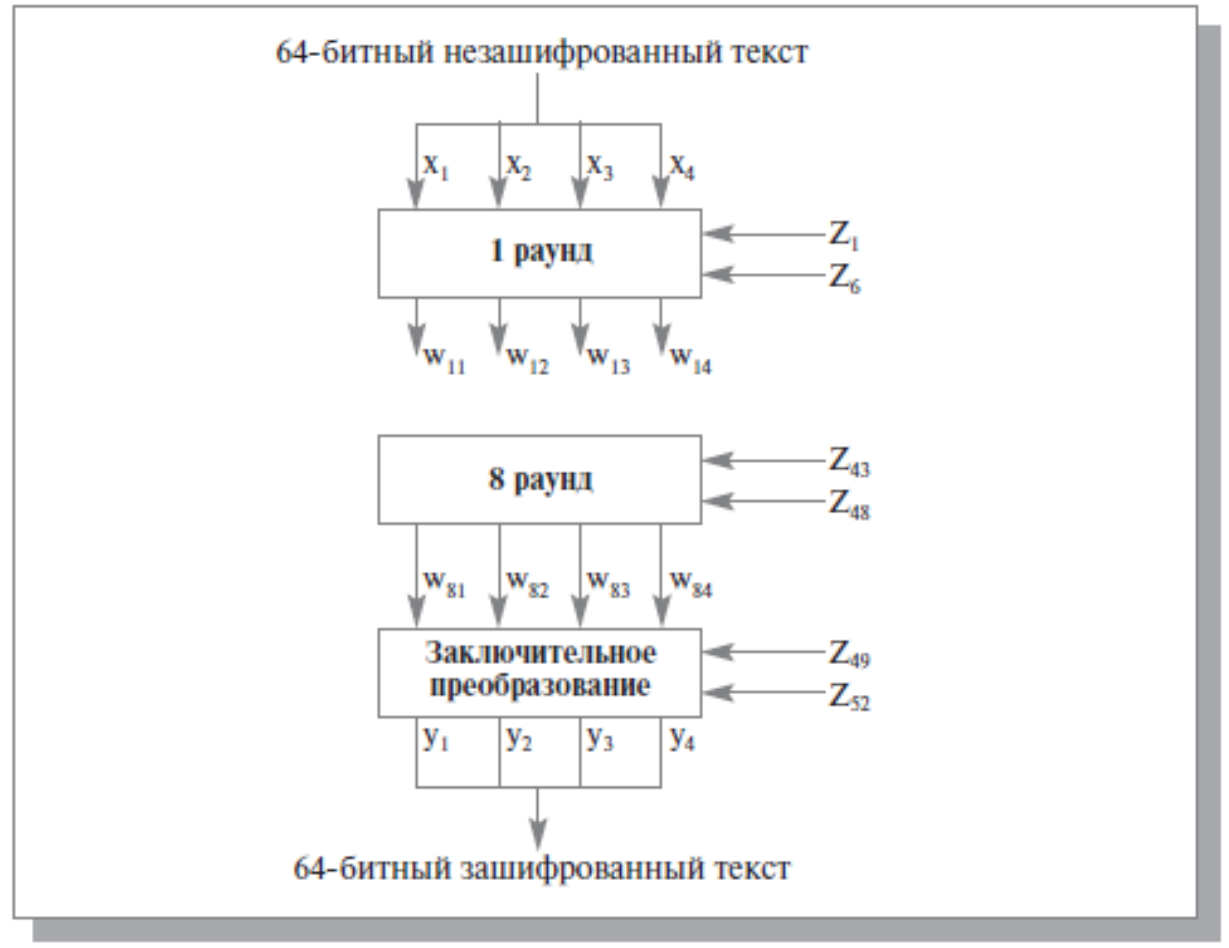
- $F(X_L) = (((S_{1,A} + S_{2,B}) \bmod 2^{32} \oplus S_{3,C}) + S_{4,D}) \bmod 2^{32}$

Алгоритм IDEA

Операции:

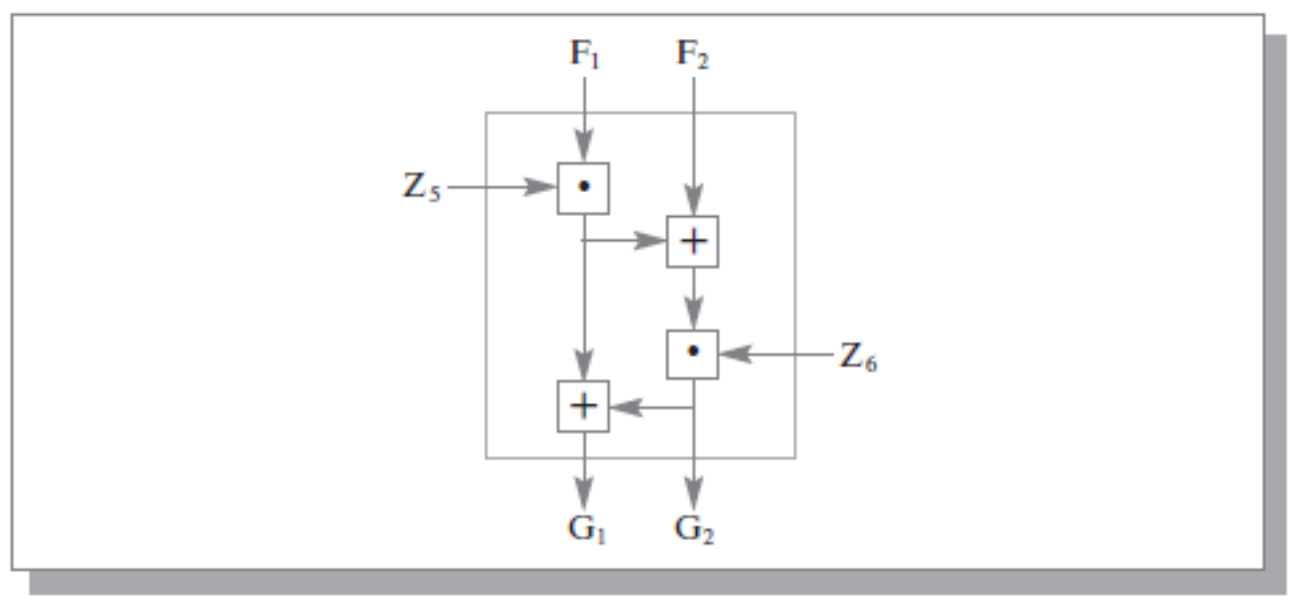
- Побитовое исключающее OR, обозначаемое как $\oplus$.
- Сумма целых по модулю 2^{16} (по модулю 65536), при этом входы и выходы трактуются как беззнаковые 16-битные целые. Эту операцию обозначим как +.
- Умножение целых по модулю $2^{16} + 1$ (по модулю 65537), при этом входы и выходы трактуются как беззнаковые 16-битные целые, за исключением того, что блок из одних нулей трактуется как 2^{16}. Эту операцию обозначим как •.

Алгоритм IDEA



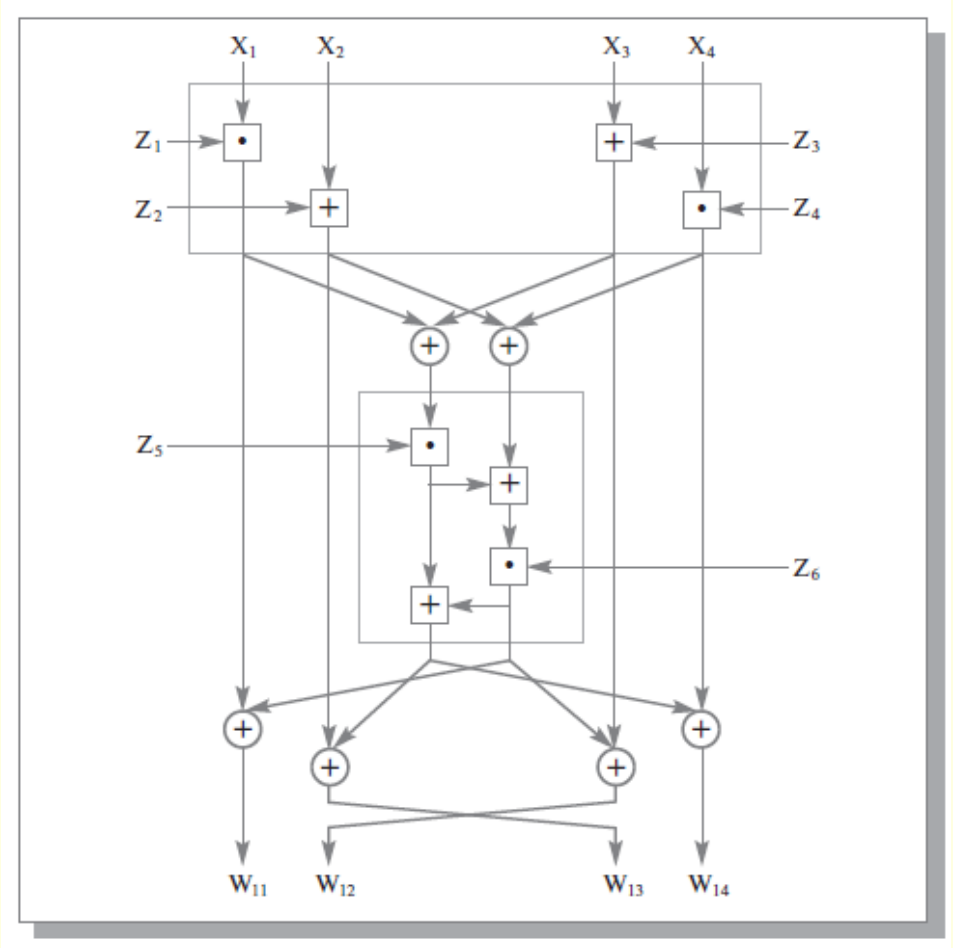
Алгоритм IDEA

Структура МА (умножение/сложение)



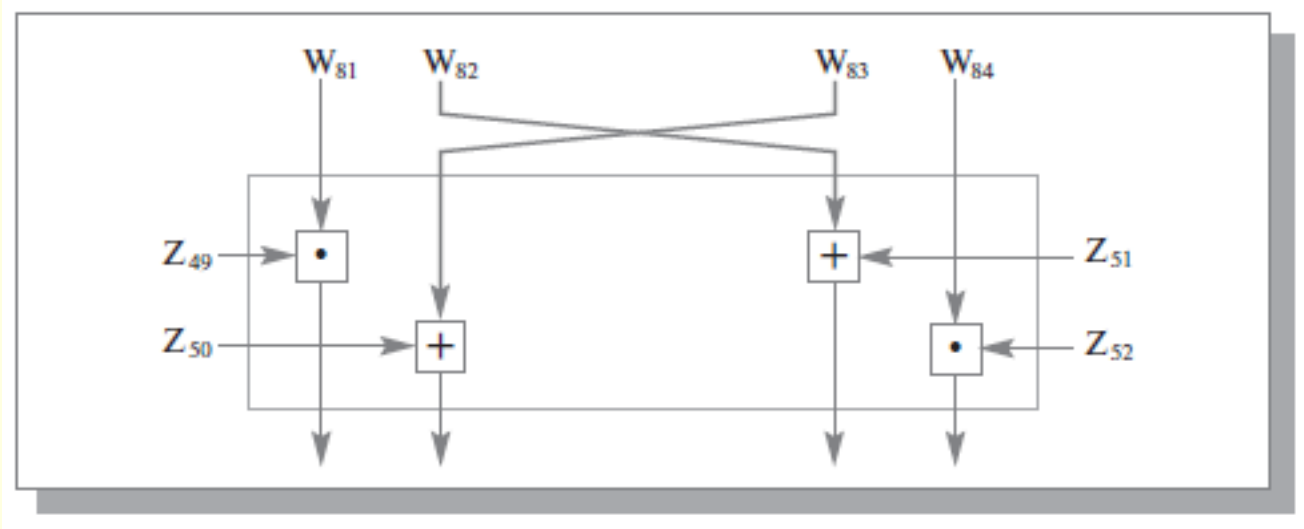
Алгоритм IDEA

i-ый раунд IDEA



Алгоритм IDEA

Заключительное преобразование



Алгоритм ГОСТ 28147-89