УГАТУ

Лекция №2

ОСНОВЫ СИММЕТРИЧНОГО ШИФРОВАНИЯ

Дисциплина: Криптографическая защита информации
Преподаватель: Миронов Константин Валерьевич
Поток: БПС-3
Учебный год: 2020/21

Содержание лекции

- **Базовые операции**
  - **Побитное сложение по модулю 2**
  - **Арифметические операции**
  - **Перестановка**
  - **Подстановка**
  - **Дополнение**
- Способы построения алгоритмов
- Режимы блочного шифрования

УГАТУ
Уфимский государственный авиационный технический университет

Побитное сложение по модулю 2

$$y = x \oplus k$$

x	k	y
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**0**

- Для криптографии важно свойство: $x \oplus k \oplus k = x$
- При этом по $y = x \oplus k$ нельзя определить ни x, ни k
- Шифр Вернама: $\text{ШТ} = \text{ОТ} \oplus \text{К}$ $\text{ОТ} = \text{ШТ} \oplus \text{К}$
- Длина ключа равна длине сообщения – практически бесполезен
- Разбить сообщение на блоки и складывать их с ключом меньшей длины **нельзя**

$$\text{ШТ1} = \text{ОТ1} \oplus \text{К}$$

$$\text{ШТ2} = \text{ОТ2} \oplus \text{К}$$

$$\text{ШТ1} \oplus \text{ШТ2} = \text{ОТ1} \oplus \text{К} \oplus \text{ОТ2} \oplus \text{К} = \text{ОТ1} \oplus \text{ОТ2}$$

- Если в ОТ изменить 1 бит, в ШТ изменится тот же по порядку бит

Арифметические операции

- Применяются сложение (a ⊞ b), вычитание (a ⊟ b), умножение
- При выполнении операций игнорируются разряды, вылезающие за границы обрабатываемого слова
  - т.е. операции выполняются по модулю 2^n, где n – число бит в слове
- Чтобы обратить сложение применяется вычитание и наоборот
- Сложение/вычитание – альтернатива XOR
- Умножение применяется в операциях, которые не придется обращать
  - Время выполнения умножения часто зависит от значения умножаемых данных, что создает потенциальную уязвимость к тайминг-атакам

УГАТУ
Уфимский государственный авиационный технический университет

Перестановка [permutation]

$$y = P(x)$$

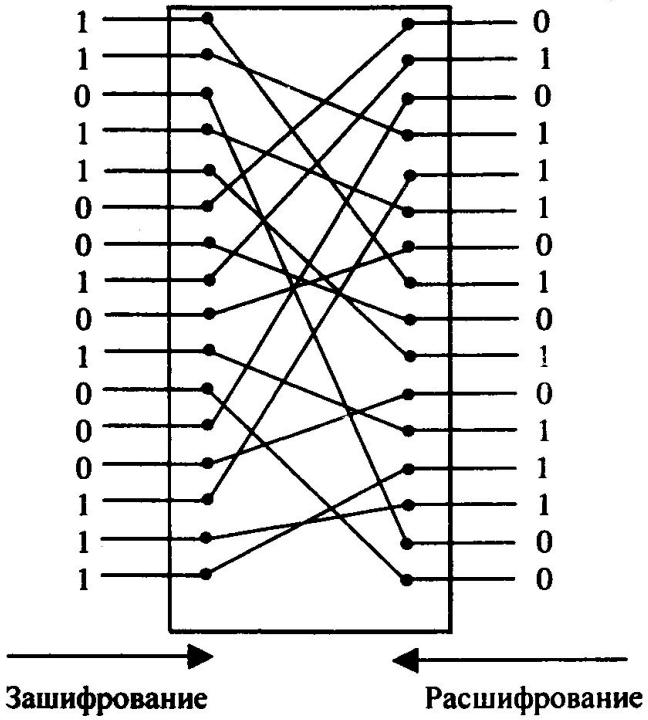
- Биты меняются местами
- Количество возможных перестановок =n! где n – число разрядов
- Если изменить 1 бит x, изменится 1 бит y, но расположенный в другом месте
- Простейшая аппаратная реализация – перепутывание линий в нужном порядке

Программная реализация:

- Таблица перестановок
- Умножение матрицы на вектор данных
- Некоторые математические функции могут реализовывать подстановку

58	50	42	34	26	18	10	2	60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6	64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1	59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5	63	55	47	39	31	23	15	7

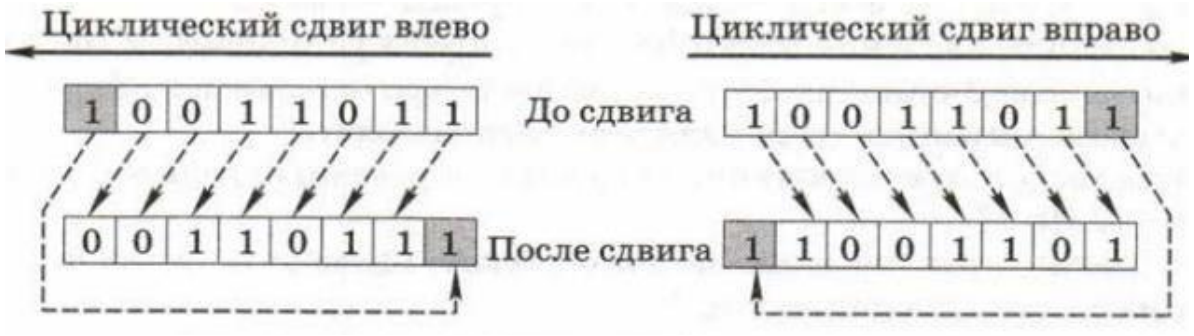
Таблица начальной перестановки DES.
В 1-й бит y записывается 58-й бит x.
Во 2-й бит y записывается 50-й бит x и т. д.



Перестановка [permutation]

Простейший вариант перестановки – **циклический сдвиг**

$y = x \lll 1$



$y = x \ggg 1$

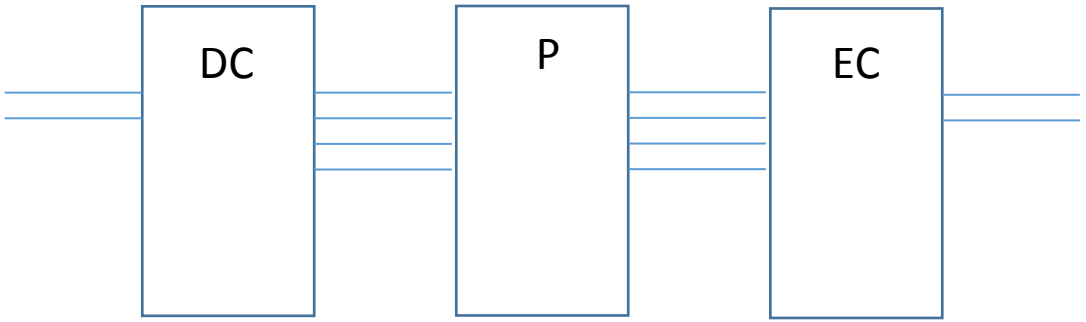
$y = x \lll 7$

- Встречается циклический сдвиг не на постоянное число, а на переменное: $y = x \lll z$
  - Если длина регистра x составляет 2^n, то сдвиг x на z – это тоже самое, что сдвиг x на число, записанное в младших n битами z
  - Время выполнения такого сдвига часто зависит от значения данных в регистре z, что создает потенциальную уязвимость к тайминг-атакам

Подстановка [substitution]

$$y = S(x)$$

- Замена блока бит на некоторый другой блок бит той же величины
- Подстановка должна:
  - Существовать для любого значения блока
  - Быть обратимой
- Простейшая аппаратная реализация – перестановка в унарном коде
- Если в x изменить 1 бит, y изменится целиком – **лавинный эффект**



УГАТУ
Уфимский государственный авиационный технический университет

Подстановка [substitution]

Программная реализация:

- Таблица замен
- Любая обратимая функция y=f(x), где y и x имеют одинаковый размер, является подстановкой

Таблица замен AES.
Байт со значением 00 заменяется на байт со значением 63
Байт со значением 01 заменяется на байт со значением 7C
и т. д.

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Подстановки и перестановки

- Существуют подобные операции, при которых меняется размер блока
  - **Расширение** (E) – подстановка/перестановка с увеличением размера
  - **Сжатие** - подстановка/перестановка с уменьшением размера (необратимо)
- Таблицы замен/перестановок гипотетически могут оставлять лазейку, позволяющую создателю алгоритма взламывать шифр
- Поэтому они как правило составляются на основе некоторых общеизвестных псевдослучайных данных (например, двоичное представление числа п)

Дополнение [padding]

- Алгоритмы часто оперируют блоками данных определенной длины
- Величина текста может не быть кратной длине блока
- В таком случае текст может дополняться до нужной величины (нулями, единицами, случайными значениями и т.д.)
- Желательно, чтобы исходный текст можно было однозначно восстановить
  - Может быть непонятно, где кончается блок и начинается дополнение
- Примеры:
  - В последний байт записывается количество значимых бит в последнем блоке, промежуток между ними и концом сообщения заполняется нулями
  - В самое начало сообщения записывается его длина

Содержание лекции

- Базовые операции
- **Способы построения алгоритмов**
  - **Гаммирование**
  - **SP-сеть**
  - **Сеть Фейстеля**
- Режимы блочного шифрования

Способы построения алгоритмов

- Симметричные алгоритмы бывают **блочные** и **поточные** (асимметричные – только блочные)
- И те и другие могут оперировать блоками данных фиксированной длины (в поточном алгоритме блок может состоять из одного бита)
- Результат зашифровывания блока зависит
  - В блочных алгоритмах – от своего исходного значения и от ключа
  - В поточных алгоритмах - <…> и от порядкового номера блока
- Блочный алгоритм можно использовать в поточном режиме

	Затраты ресурсов на зашифровку блока	Затраты ресурсов на инициализацию
Блочные алгоритмы	Высокие	Низкие
Поточные алгоритмы	Низкие	Высокие
Блочные алгоритмы в поточных режимах	Высокие	Низкие

Гаммирование

- Основной способ построения поточных алгоритмов
- Развитие идеи шифра Вернама
- Гамма Г=f(K) – **псевдо**случайная последовательность, вычисленная на основе секретного ключа К и равная по длине шифруемому тексту
- Если ключ есть у Боба и Алисы, оба могут рассчитать Г
- Зашифровывание $\text{ШТ} = \text{ОТ} \oplus \Gamma$
- Расшифровывание $\text{ОТ} = \text{ШТ} \oplus \Gamma$

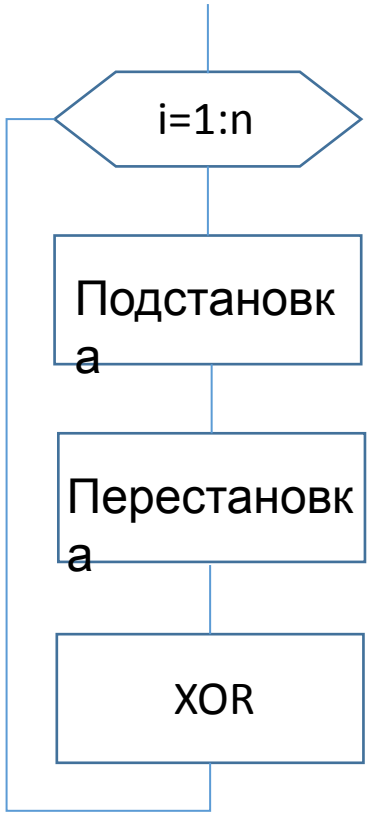
+ Два одинаковых фрагмента ОТ при зашифровывании дадут два разных фрагмента ШТ

- Гаммирование не обеспечивает лавинного эффекта: если в ОТ изменить 1 бит, в ШТ изменится тот же самый бит

Подстановочно-перестановочная сеть

SP-сеть

- Способ построения блочных алгоритмов
- Операция XOR выполняется с **раундовым ключом**
- Раундовые ключи для каждого раунда генерируются на основе исходного ключа с помощью процедуры **расширения ключа**
- Расшифровка представляет собой выполнение обратных операций в обратном порядке



УГАТУ
Уфимский государственный авиационный технический университет

Сеть Фейстеля

Сбалансированная

- Способ построения блочных алгоритмов
- Блок данных делится на два полублока L и R
- На каждом раунде выполняется:

  $L(i+0)=L(i) \oplus f(R(i),K(i))$

  $R(i+0)=R(i)$

  $L(i+1) = R(i+0)$

  $R(i+1) = L(i+0)$

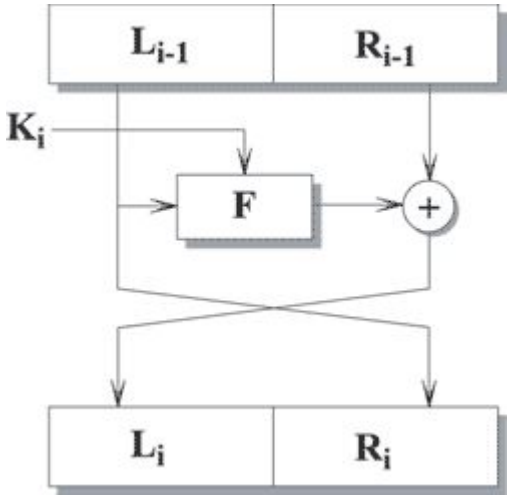
  где f – функция от полублока и раундового ключа
- Даже если функция f односторонняя, раунд будет обратим:

  $L0= R(i+1)= L(i+0)= L(i) \oplus f(R(i),K(i))$

  $R0=L(i+1)= R(i+0)=R(i)$

  $L= L0 \oplus f(R0,K(i))= L(i) \oplus \cancel{f(R(i),K(i)) \oplus f(R(i),K(i))} = L(i)$

  $R= R0 = R(i)$



Сеть Фейстеля

Сбалансированная

- При расшифровывании выполняются те же операции, что при зашифровывании, но:
  - раундовые ключи берутся в обратном порядке,
  - полублоки меняются местами в начале раунда, а не в конце

Несбалансированная

На каждом раунде:

- блок разбивается на части A и B
- выполняется A=f1(A, f2(B,Ki))

  где f1 обратима

  f2 может быть односторонней

За все раунды любой бит блока должен поменяться несколько раз

Блочные алгоритмы

Сравнение SP-сети с сетью Фейстеля

- При аппаратной реализации удобнее сеть Фейстеля, поскольку одну и ту же схему можно применять и для зашифровки, и для зашифровки
- При программной реализации особой разницы нет, однако SP-сеть позволяет обрабатывать весь блок за один раунд, а сеть Фейстеля в лучшем случае за 2
  - На процессорах с высокой разрядностью SP-сеть предпочтительна
- Алгоритмы строят разными способами, однако в разное время были популярны разные подходы:
  - До 1977 - гаммирование и SP-сеть (сеть Фейстеля еще не была изобретена)
  - 1977-1987 - сбалансированная сеть Фейстеля
  - 1987-2000 - несбалансированная сеть Фейстеля
  - После 2000 - SP-сеть

Содержание лекции

- Базовые операции
- Способы построения алгоритмов
- **Режимы блочного шифрования**
  - **ECB**
  - **CBC**
  - **PCBC**
  - **CFB**
  - **OFB**
  - **CTR**
  - **Синхропосылки**

Режимы блочного шифрования

- Блочный алгоритм задает 2 операции:

  ШТ = Ш(ОТ,К)

  ОТ = ДШ(ШТ,К)

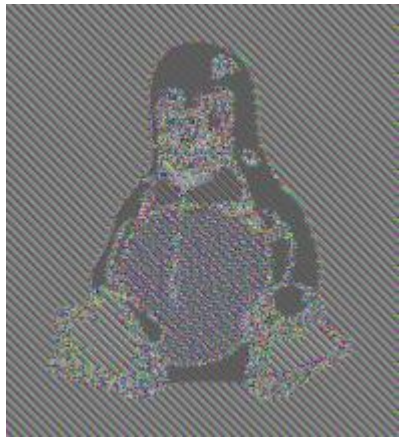
  где ОТ и ШТ – блоки фиксированной одинаковой длины
- Эти операции могут выполняться в различных **режимах**
- Исходный алгоритм блочный, но все нижеописанные режимы кроме ECB являются поточными

Режим ECB

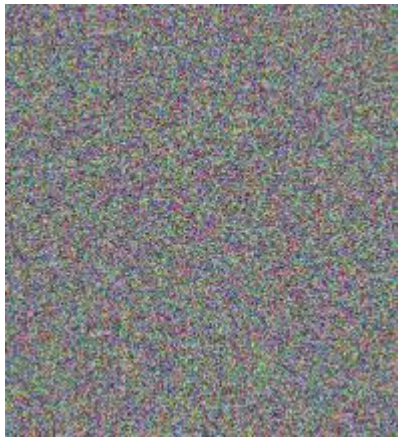
- Режим простой замены, режим электронной кодовой книги [Electronic Code Book]
- Блоки обрабатываются независимо

$$\text{ШТ}_i = \text{Ш}(\text{ОТ}_i, K)$$
$$\text{ОТ}_i = \text{ДШ}(\text{ШТ}_i, K)$$

- Одинаковые блоки ОТ соответствуют одинаковым блокам ШТ
+ Возможность распараллеливания



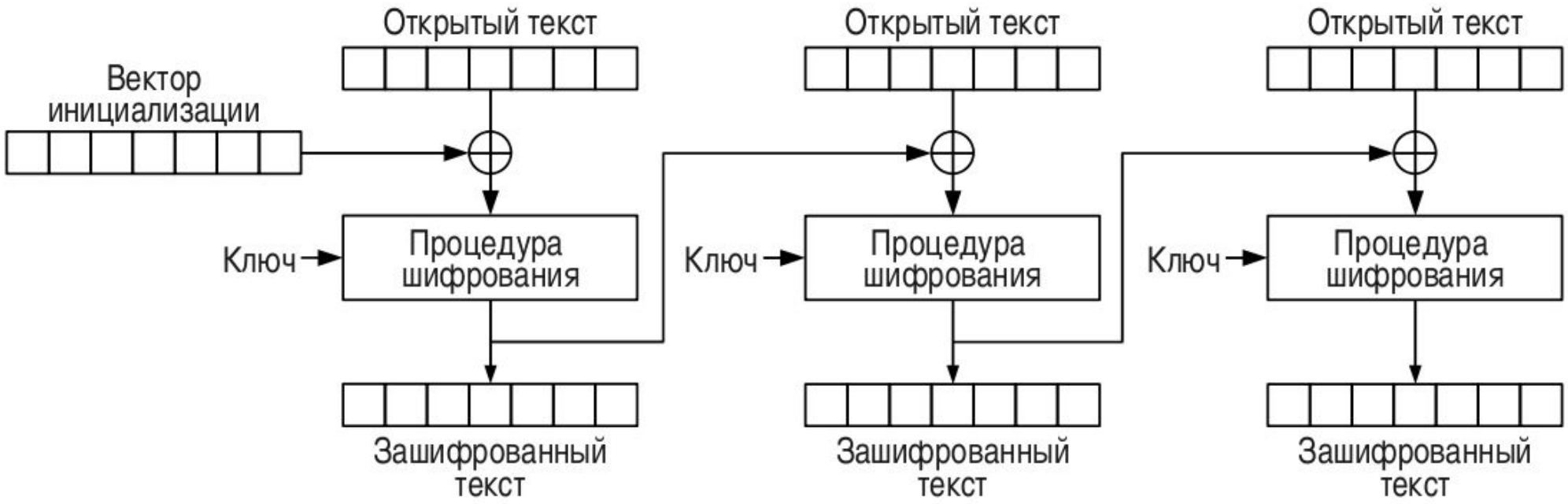
ECB



CBC

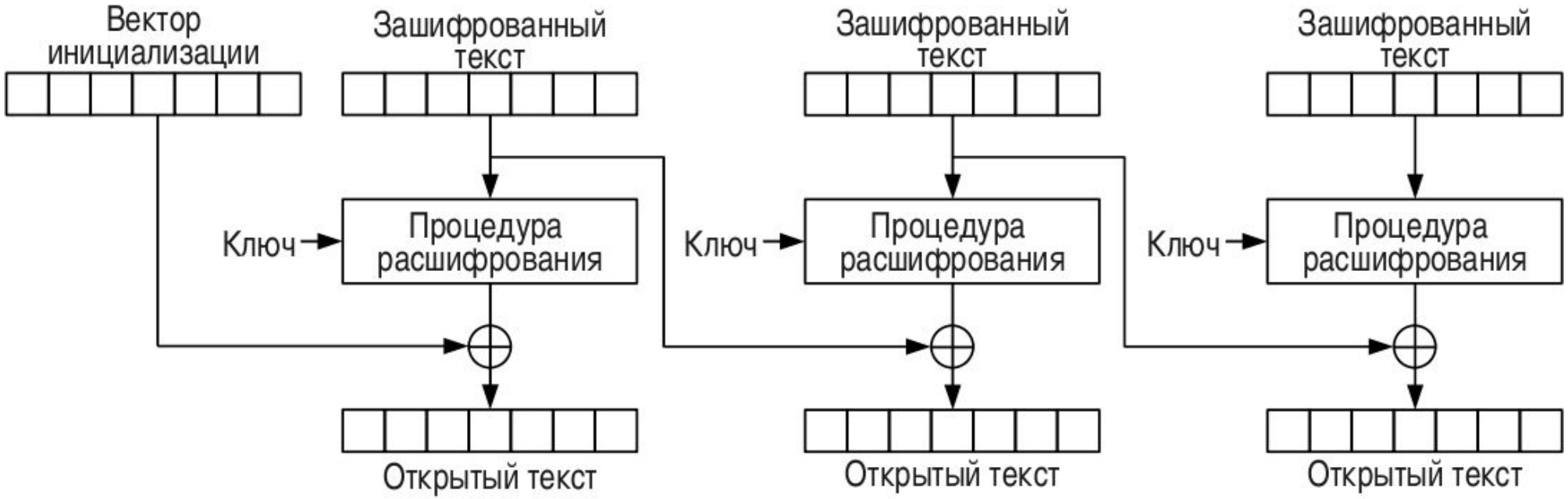
Режим CBC

$$ШТ_1 = Ш(ОТ_1 \oplus IV, K)$$
$$ШТ_i = Ш(ОТ_i \oplus ШТ_{i-1}, K)$$



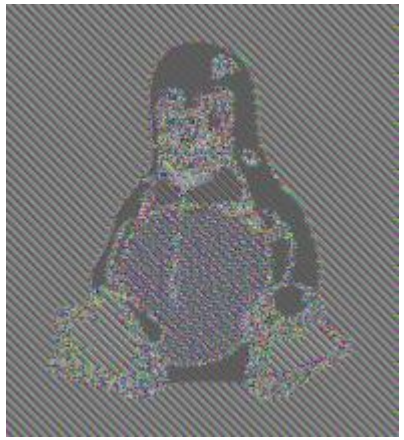
Режим CBC

$$\text{ОТ}_1 = \text{ДШ}(\text{ШТ}_1, K) \oplus IV$$
$$\text{ОТ}_i = \text{ДШ}(\text{ШТ}_i, K) \oplus \text{ШТ}_{i-1}$$

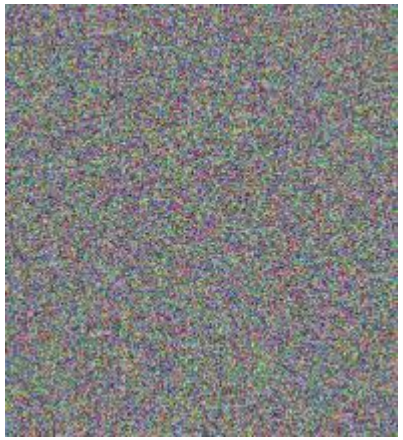


Режим CBC

- Режим зацепления блоков шифротекста [Cipher Block Chaining]
- Шифрование распараллелить нельзя, расшифровывание - можно
- Самый часто используемый режим
- IV – синхропосылка, вектор инициализации [Initialization Vector]
- Синхропосылка не является секретной, ее знание ничего не дает злоумышленнику



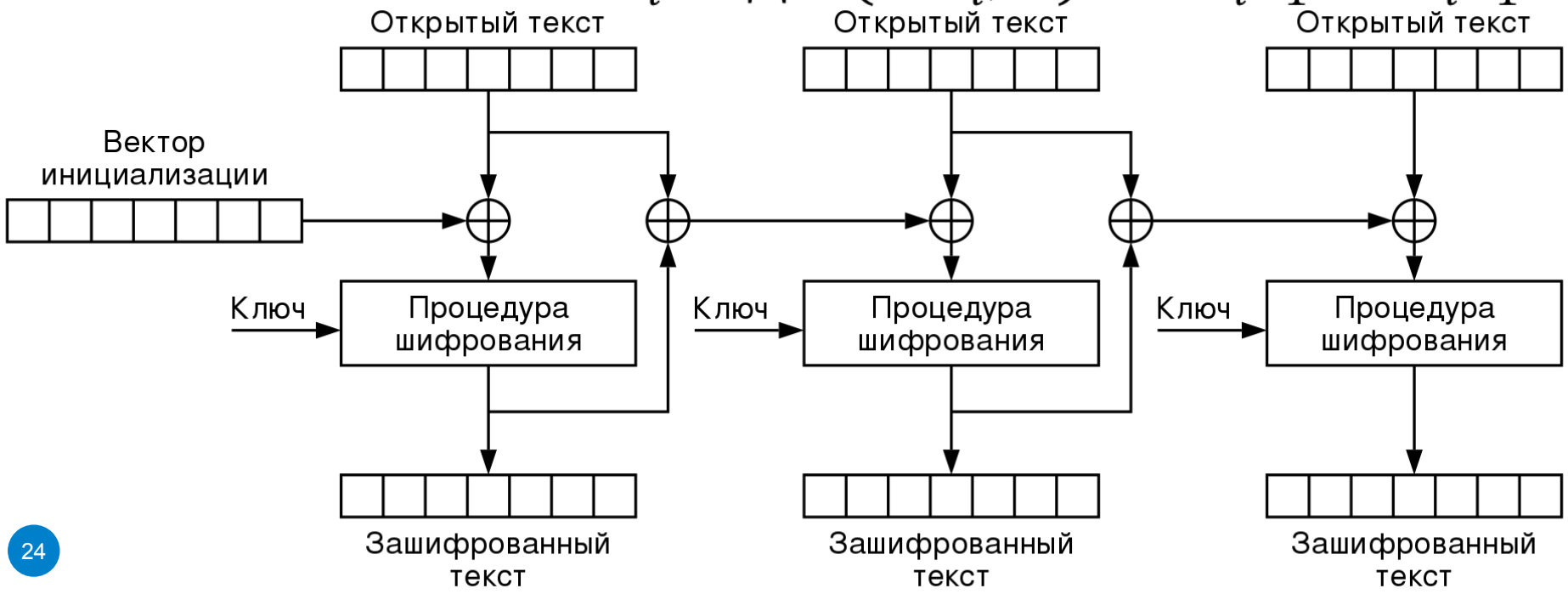
ECB



CBC

Режим PCBC

$$ШТ_1 = Ш(ОТ_1 \oplus IV, K)$$
$$ШТ_i = Ш(ОТ_i \oplus ШТ_{i-1} \oplus ОТ_{i-1}, K)$$
$$ОТ_1 = ДШ(ШТ_1, K) \oplus IV$$
$$ОТ_i = ДШ(ШТ_i, K) \oplus ШТ_{i-1} \oplus ОТ_{i-1}$$



Режим PCBC

- Режим распространяющегося зацепления блоков шифротекста [Propagating CBC]
- Если поменять значение одного из блоков, все остальные будут зашифрованы некорректно
- Расшифровывание можно распараллелить частично:
  - Параллельно рассчитать промежуточный текст:

$$\text{ПТ}_i = \text{ДШ}(\text{ШТ}_i, K)$$

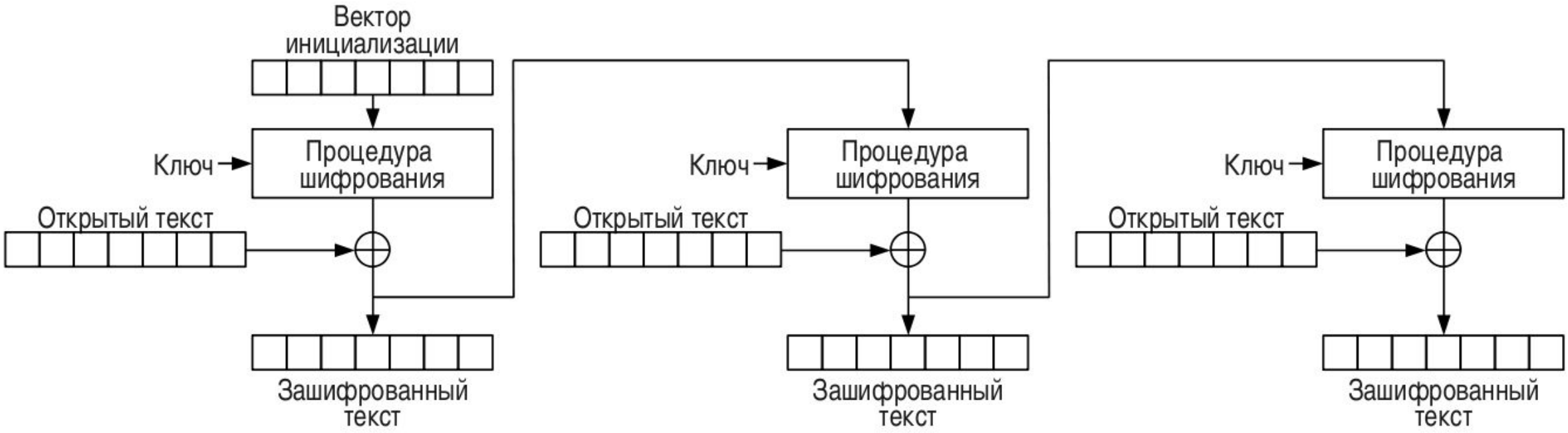
  - Последовательно расшифровать данные:

$$\text{ОТ}_1 = \text{ПТ}_1 \oplus IV$$
$$\text{ОТ}_i = \text{ПТ}_i \oplus \text{ШТ}_{i-1} \oplus \text{ОТ}_{i-1}$$

- Режим не обрел большой популярности

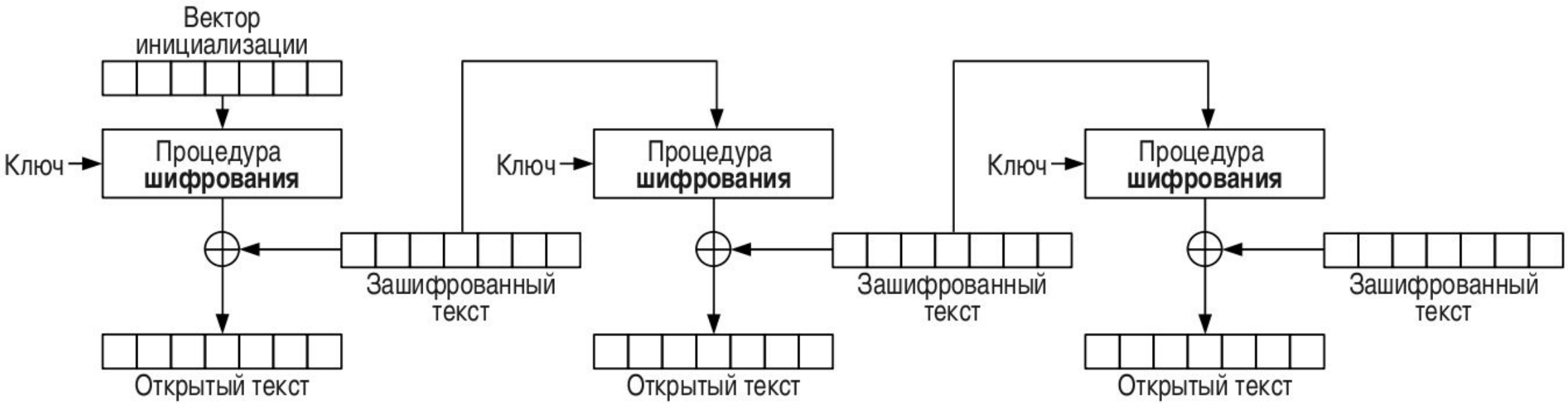
Режим CFB

$$ШТ_1 = ОТ_1 \oplus Ш(IV, K)$$
$$ШТ_i = ОТ_i \oplus Ш(ШТ_{i-1}, K)$$



Режим CFB

$$\text{ОТ}_1 = \text{ШТ}_1 \oplus \text{Ш}(IV, K)$$
$$\text{ОТ}_i = \text{ШТ}_i \oplus \text{Ш}(\text{ШТ}_{i-1}, K)$$

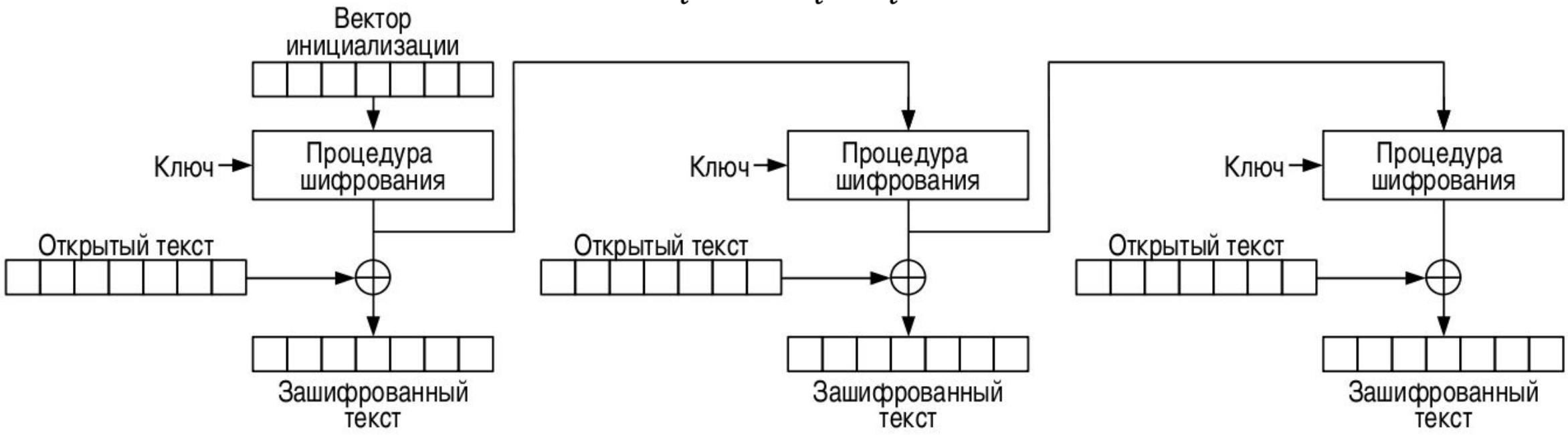


Режим CFB

- Режим гаммирования с обратной связью, режим обратной связи по шифротексту [Cipher FeedBack]
- Операция ДШ(ШТ,К) в этом режиме не применяется
- Теоретически функция Ш(ОТ,К) может быть односторонней
- Шифрование распараллелить нельзя, расшифровывание – можно
- Режим не обрел особой популярности за рубежом, однако в отечественной криптографии используется вместо CBC

Режим OFB

$$\Gamma_1 = Ш(IV, K)$$
$$\Gamma_i = Ш(\Gamma_{i-1}, K)$$
$$ШТ_i = ОТ_i \oplus \Gamma_i$$
$$ОТ_i = ШТ_i \oplus \Gamma_i$$



Режим OFB

- Режим обратной связи по выходу [Output FeedBack]
- Использование блочного алгоритма в режиме OFB является разновидностью гаммирования
- Распараллелить алгоритм нельзя, но можно рассчитать гамму заранее
- Расшифрование полностью идентично шифрованию
- Дополнение блоков необязательно
- Синхропосылка должна быть уникальна для каждого шифруемого текста, иначе

$$ШТ1 = ОТ1 \oplus Г$$

$$ШТ2 = ОТ2 \oplus Г$$

$$ШТ1 \oplus ШТ2 = ОТ1 \oplus Г \oplus ОТ2 \oplus Г = ОТ1 \oplus ОТ2$$

Режим CTR

- Счетный [CounTeR]
- Как и OFB, это разновидность гаммирования, но гамма генерируется по-другому:

$$\Gamma_i = Ш\,(\,IV||i, К\,)$$

- Длина синхропосылки равна длине блока минус число байт, отводимых на номер блока
- Как и в OFB синхропосылка должна быть уникальна для каждого шифруемого текста
- В отличие от OFB, этот режим позволяет распараллелить шифрование и расшифрование
- Режим появился поздно и изначально не был очень популярен, однако в силу вышеуказанного преимущества постепенно вытеснил OFB

* Здесь и далее в рамках курса оператор || обозначает конкатенацию, т.е. склейку строк

Синхропосылки

- Фиксированная синхропосылка – каждый раз используется одно и то же значение
  - Если в двух ОТ первые блоки совпадают, то будут совпадать и первые блоки ШТ
  - В режимах OFB и CTR неприемлемо
- Случайная синхропосылка – генерируется случайным образом
  - В таком случае ее нужно послать получателю в открытом виде («нулевой блок ШТ»)
  - Недостаток – увеличивается объем пересылаемой информации
- **Nonce** «оказия» [Number used ONCE] – в разных областях криптографии так называется двоичный код, каждое значение которого встречается один раз в данном контексте
- При использовании в качестве IV нонс имеет особенности:
  - Может вычисляться на основе порядкового номера сообщения, метки времени, идентификаторов отправителя и получателя и т.д.
  - Статистически должен выглядеть как случайные данные (кроме CTR и OFB)
  - При небольшом изменении номера сообщения нонс меняется радикально
  - Алиса посылает Бобу информацию, необходимую для вычисления нонса