Математические основы информационной безопасности

Груздев Дмитрий Николаевич

Хеш-функции

Оригинальное сообщение

С	о	в	е	щ	а	н	и
е		с	о	с	т	о	и
т	с	я		1	2		д
е	к	а	б	р	я		в
	9		ч	а	с	о	в
.		И	в	а	н	о	в
	А	л	е	к	с	е	й
.							

D1	CE	E2	E5	F9	E0	ED	E8
E5	20	F1	EE	F1	F2	EE	E8
F2	F1	FF	20	31	32	20	E4
E5	EA	E0	E1	F0	FF	20	E2
20	39	20	F7	E0	F1	EE	E2
2E	20	C8	E2	E0	ED	EE	E2
20	C0	EB	E5	EA	F1	E5	E9
2E	0	0	0	0	0	0	0

Сумма: 0x29C8

Отправлено: 0xC8

Поддельное сообщение

П	р	о	ш	у		п	е
р	е	д	а	т	ь		с
	к	у	р	ь	е	р	о
м		1	0	0	0	0	0
	р	у	б	л	е	й	.
	И	в	а	н	о	в	
А	л	е	к	с	е	й	.

CF	F0	EE	F8	F3	20	EF	E5
F0	E5	E4	E0	F2	FC	20	F1
20	EA	F3	F0	FC	E5	F0	EE
EC	20	31	30	30	30	30	30
20	F0	F3	E1	EB	E5	E9	2E
20	C8	E2	E0	ED	EE	E2	20
C0	EB	E5	EA	F1	E5	E9	2E

Подделка: 0x27B6 - 0xB6
Оригинал: 0xC8

Подходящая сумма

tab	П	р	о	ш	у		п
е	р	е	д	а	т	ь	
с		к	у	р	ь	е	р
о	м		1	0	0	0	0
0		р	у	б	л	е	й
.		tab	И	в	а	н	о
в		А	л	е	к	с	е
й	.						

09	CF	F0	EE	F8	F3	20	EF
E5	F0	E5	E4	E0	F2	FC	20
F1	20	EA	F3	F0	FC	E5	F0
EE	EC	20	31	30	30	30	30
30	20	09	F0	F3	E1	EB	E5
E9	2E	20	C8	E2	E0	ED	EE
E2	20	C0	EB	E5	EA	F1	E5
E9	2E						

Оригинальное сообщение

С	о	в	е	щ	а	н	и
е		с	о	с	т	о	и
т	с	я		1	2		д
е	к	а	б	р	я		в
	9		ч	а	с	о	в
.		И	в	а	н	о	в
	А	л	е	к	с	е	й
.							

D1	CE	E2	E5	F9	E0	ED	E8
E5	20	F1	EE	F1	F2	EE	E8
F2	F1	FF	20	31	32	20	E4
E5	EA	E0	E1	F0	FF	20	E2
20	39	20	F7	E0	F1	EE	E2
2E	20	C8	E2	E0	ED	EE	E2
20	C0	EB	E5	EA	F1	E5	E9
2E	0	0	0	0	0	0	0

429	3E2	585	592	5B5	5D2	4DC	643

Отправлено: (0x29,0xE2,0x85,0xB5,0xD2,0xDC,0x43)

Поддельное сообщение

П	р	о	ш	у		п	е
р	е	д	а	т	ь		с
	к	у	р	ь	е	р	о
м		1	0	0	0	0	0
	р	у	б	л	е	й	.
	И	в	а	н	о	в	
А	л	е	к	с	е	й	.

CF	F0	EE	F8	F3	20	EF	E5
F0	E5	E4	E0	F2	FC	20	F1
20	EA	F3	F0	FC	E5	F0	EE
EC	20	31	30	30	30	30	30
20	F0	F3	E1	EB	E5	E9	2E
20	C8	E2	E0	ED	EE	E2	20
C0	EB	E5	EA	F1	E5	E9	2E

Подделка:	3CB	582	5B0	5A3	5DA	4E9	4E3	370
Оригинал:	429	3E2	585	592	5B5	5D2	4DC	643

Хеш-функции

Хеш-функция – легко вычислимая функция, преобразующая исходное сообщение произвольной длины (прообраз) в сообщение фиксированной длины (хеш-образ).

Коллизией для хеш-функции h называется пара значений x, y, $x \neq y$, такая, что $h(x) = h(y)$.

Применение хеш-функций

- ускорение поиска данных в БД;
- проверка целостности и подлинности сообщений
- для создания сжатого образа, применяемого в процедурах ЭЦП
- защита пароля в процедурах аутентификации

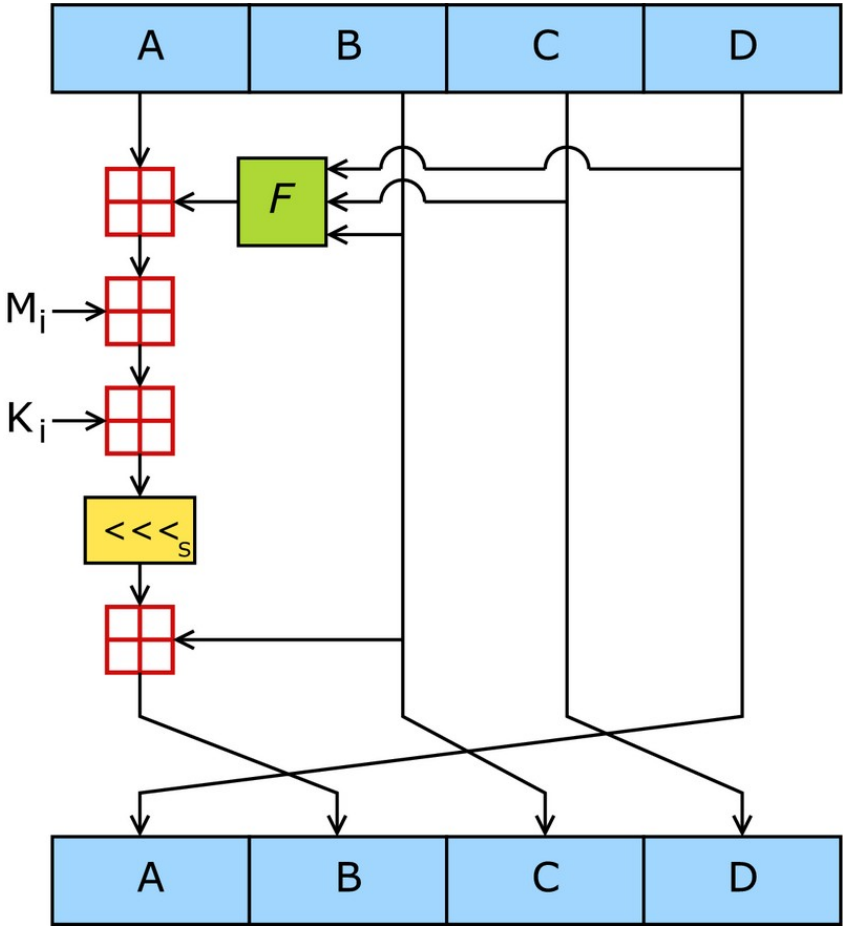
Требования к хеш-функциям

- Для данного значения h(x) невозможно найти значение аргумента x.
- Для данного аргумента x невозможно найти другой аргумент y такой, что h(x) = h(y).

Структура алгоритмов хеширования

1. Выравнивание сообщения по длине блока.
2. Разбиение сообщения на блоки.
3. Многократное применение простых преобразований к блоку.
4. Функция хеширования блока зависит от значения хеша предыдущего блока.
5. Хеш-образом сообщения является результат процедуры хеширования последнего блока.

MD5

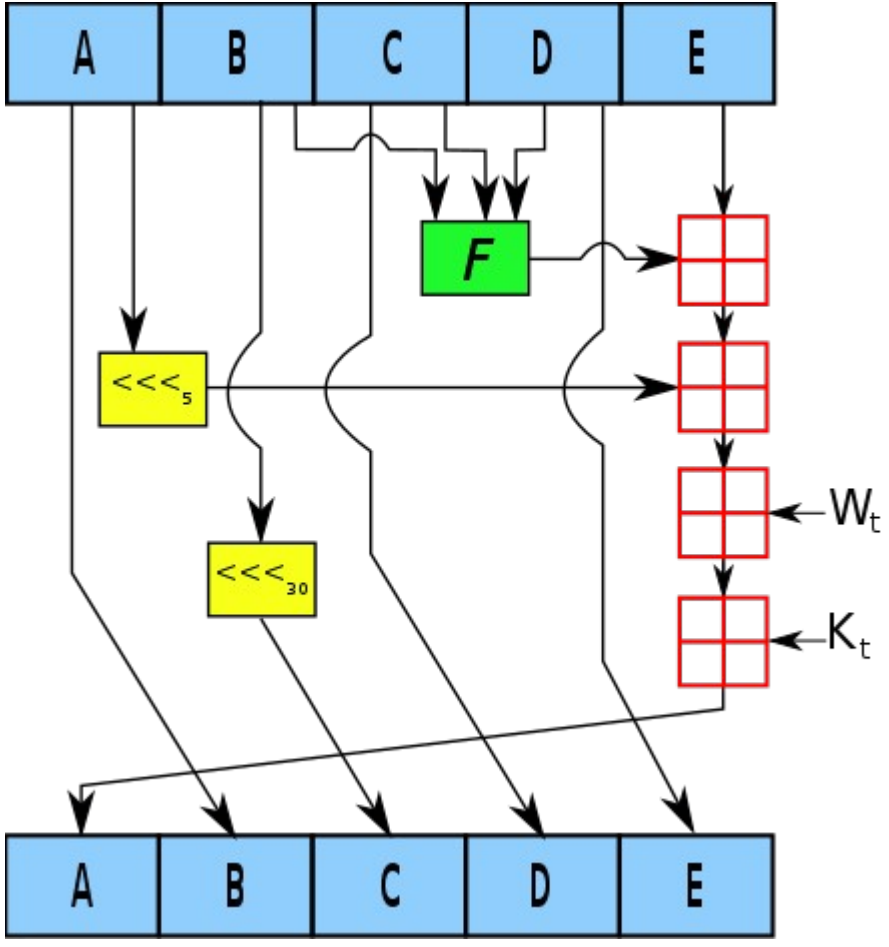


Создан в 1991 г.

Размер хеша – 128 бит.

Число раундов шифрования для одного блока – 64.

SHA-1

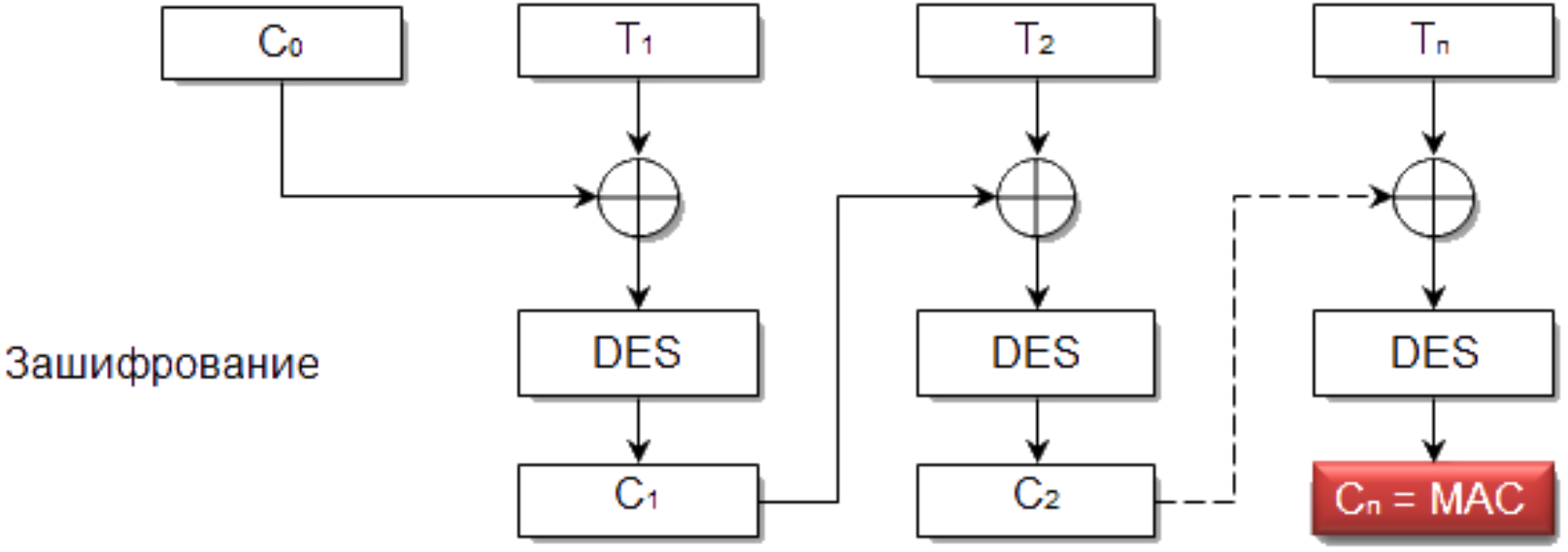


Создан в 1995 г.

Размер хеша – 160 бит.

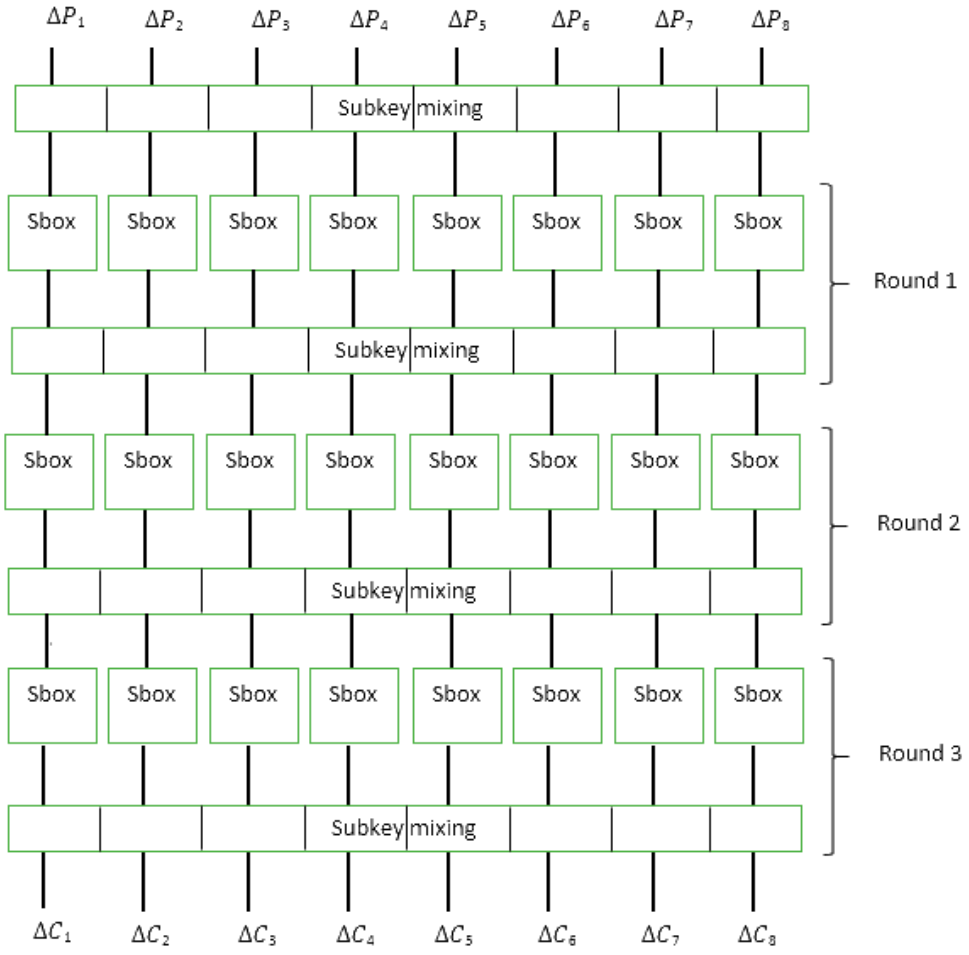
Число раундов шифрования для одного блока – 80.

Использование функции шифрования в режиме CBC



Дифференциальный криптоанализ

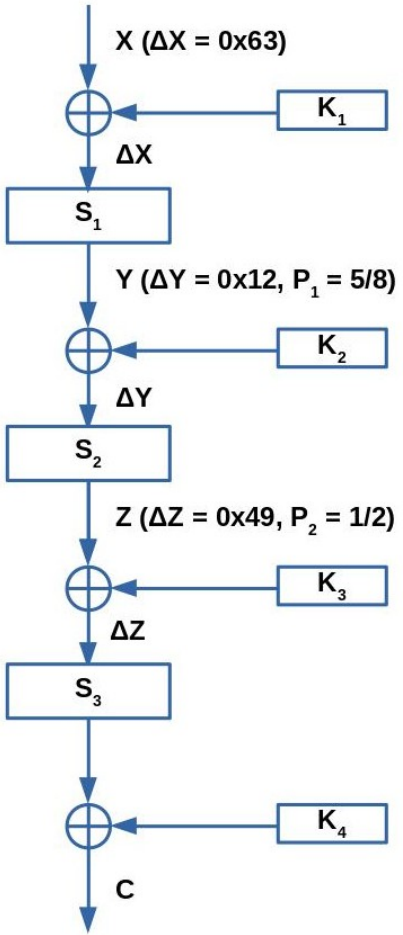
Дифференциальный Криптоанализ



ДК – атака с подобранным открытым текстом (злоумышленник имеет возможность шифровать произвольные тексты).

Цель атакующего – получить некоторые знания о ключе (полностью восстановить или сократить перебор).

Вероятности дифференциалов



X_1 и X_2 – подобранные шифротексты.

$\Delta X = X_1 \oplus X_2$ - дифференциал

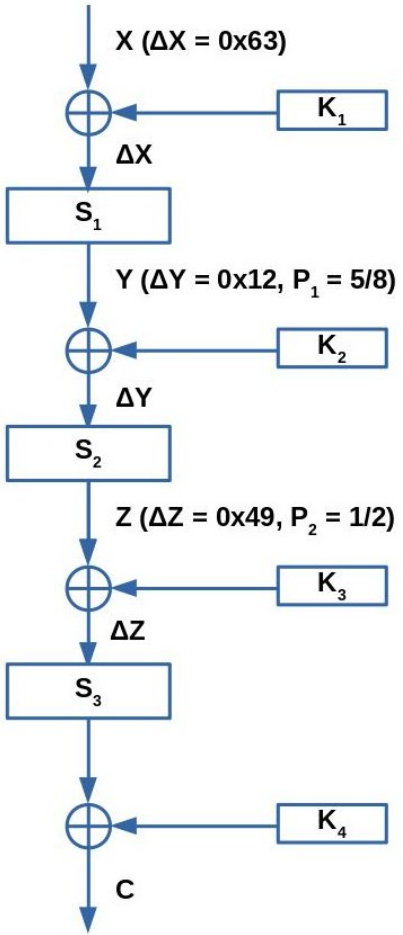
Разность текстов поступающих на S-блок: $(X_1 \oplus K_1) \oplus (X_2 \oplus K_1) = X_1 \oplus X_2$

При $X_1 \oplus X_2 = 0x63$, $Y_1 \oplus Y_2 = 0x12$ возникает в 160 случаях из 256 (с вероятностью $P_1 = 5/8$).

При $Y_1 \oplus Y_2 = 0x12$, $Z_1 \oplus Z_2 = 0x49$ возникает с вероятностью $P_2 = 1/2$.

При $X_1 \oplus X_2 = 0x63$, $Z_1 \oplus Z_2 = 0x49$ возникает с вероятностью $P = P_1 * P_2 = 5/16$.

Восстановление ключа



$D = \{ (X_1,X_2) \mid X_1 \oplus X_2 = 0x63 \}$ - множество рассматриваемых пар шифртекстов.

Предположить $K_4 = k$.

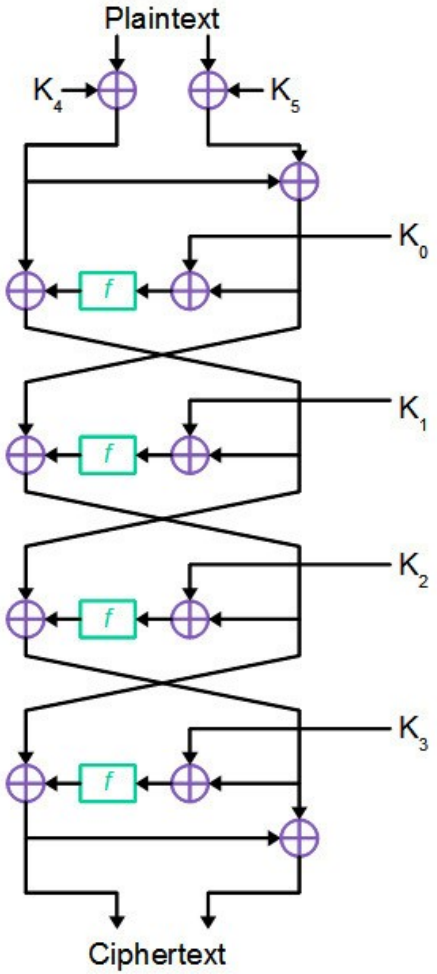
Для каждой пары $(X_1,X_2) \in D$

- Вычислить $F(X_1) = C_1$, $F(X_2) = C_2$.
- Вычислить $\Delta Z = S_3^{-1}(C_1 \oplus k) \oplus S_3^{-1}(C_2 \oplus k)$.

Если доля ΔZ=0x49 примерно равна 5/16 от общего числа, то k – кандидат для K_4.

Проверить для всех возможных значений K_4.

FEAL-4



Fast data Encipherment Algorithm

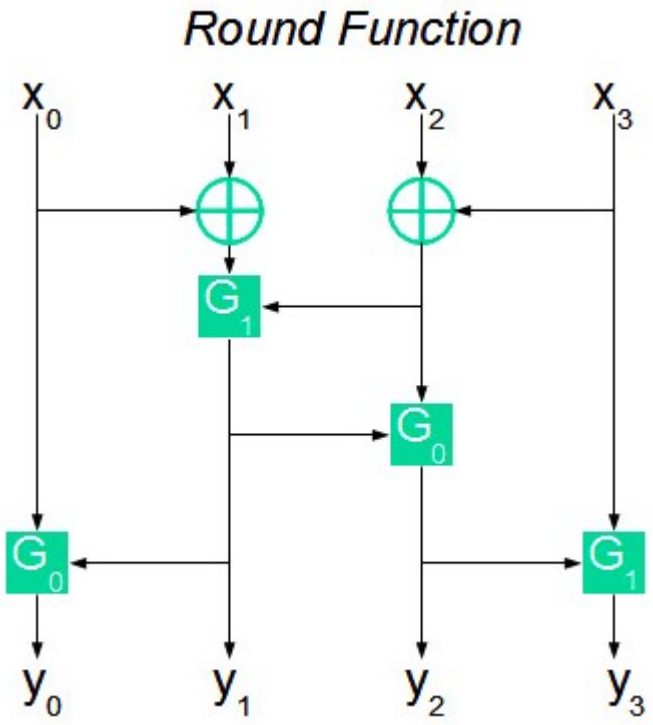
Опубликован в 1987 г.

Размер ключа – 64 бита.

Размер блока – 64 бита.

Количество раундов шифрования – 4.

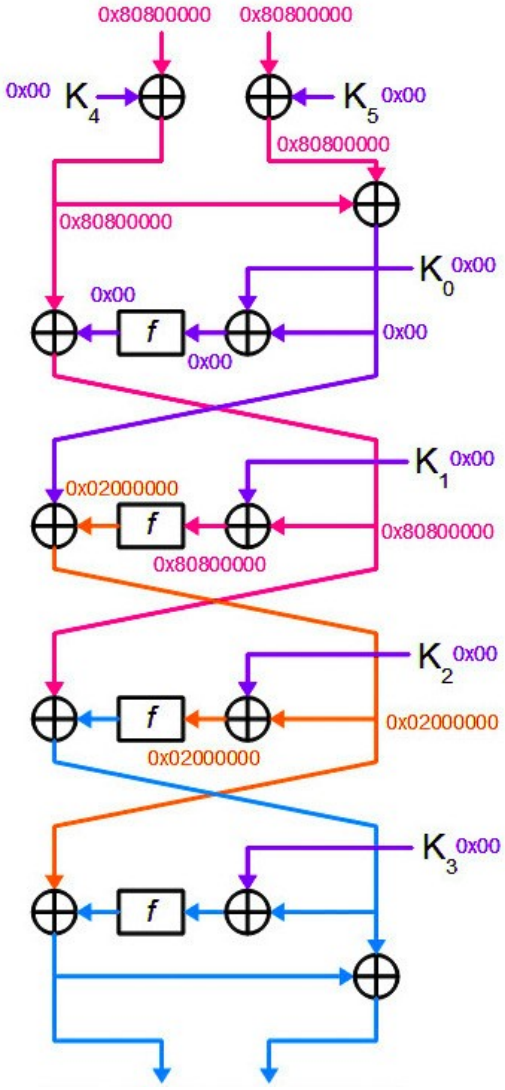
Функция Фейстеля



f – 32-битная функция.

$G_0(a,b) = (a+b \pmod{256}) <<< 2;$

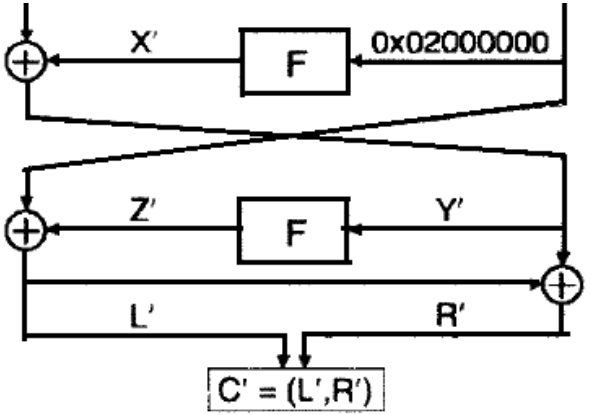
$G_1(a,b) = (a+b+1 \pmod{256}) <<< 2;$

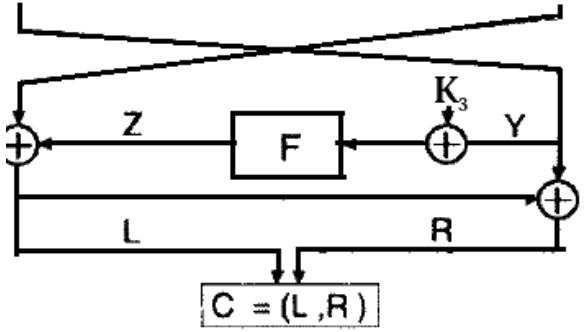


Дифференциалы

Для f при ΔX = 0x80800000
ΔY = 0x02000000.

Заключительный раунд





$L' = 0x0200000000 \oplus Z'$

$Y = L \oplus R$

$P_1 \oplus P_2 = 0x8080000080800000$

C_1, C_2 – известны, то

Y_1, Y_2, Z' - известны

Шифр определяется ключом K_3 – 2^{32} варианта.

Time-memory trade off

Вскрытие ключа шифрования

Задача:

Заранее известен открытый текст и алгоритм шифрования. Необходимо восстановить ключ шифрования по шифртексту.

Варианты решений:

- Полный перебор – слишком долго.
- Заранее просчитать и сохранить пары шифротекст-ключ – требуется много памяти.

Метод Хеллмана

В 1980 г Мартин Хеллман предложил метод Time-memory trade off. Метод позволяет вычислить ключ за $N^{2/3}$ операций, используя $N^{2/3}$ слов памяти, но требует произвести предварительные вычисления.

Обозначения

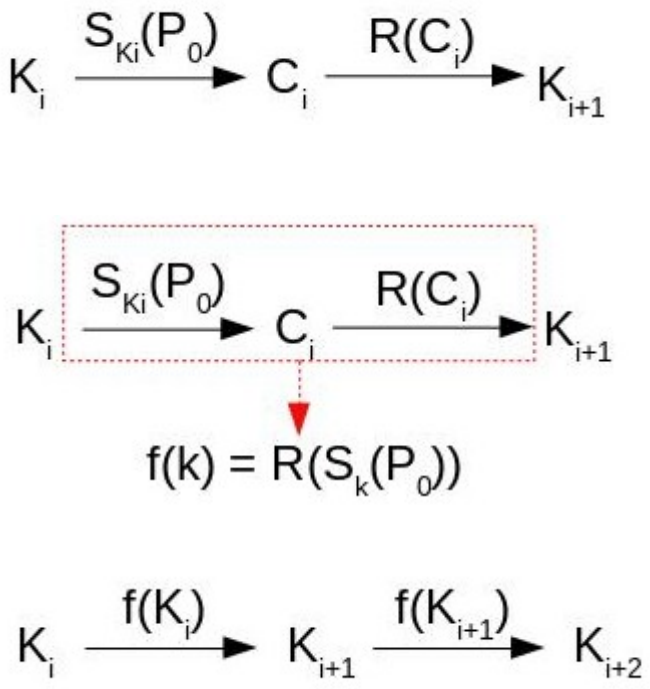
$C = S_k(P_0)$

P_0 — известный открытый текст (стандартные поля документа, шаблонная фраза в заголовке).

S_k — шифрование открытого текста на ключе k.

$R:C \to K$ — «редуцирующая» функция, переводящая любой шифротекст в некоторый ключ (шифротекст длиннее ключа).

Цепочки ключей



Заранее вычисляются m цепочек длины t.

В таблице сохраняются первые и последние элементы цепочек.

Одинаковые ключи могут появляться разных цепочках, т.к. f не является изоморфизмом. В этом случае последующие ключи в цепочках также совпадут.

Восстановление ключа

Пусть надо найти ключ шифрования K для шифротекста C.

Ищем среди сохраненных последних элементов цепочек занчения $R(C)$, $f(R(C))$, $f^2(R(C))$, ..., $f^{t-1}(R(C))$.

Если $f^p(R(C))$ совпадет с окончанием цепочки, то восстанавливаем K_{t-p-1} ключ этой цепочки.

Если $S_{Kt-p-1}(P_0) = C$, то K_{t-p-1} – искомый ключ, иначе продолжаем поиск.

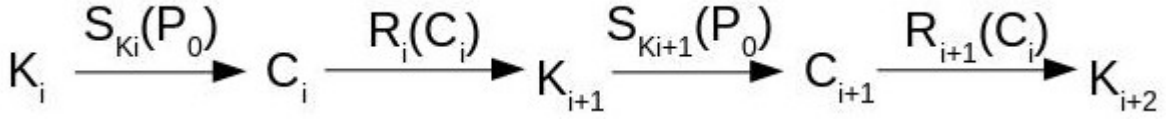
Если ключ не найден, то его нет среди ключей в цепочках.

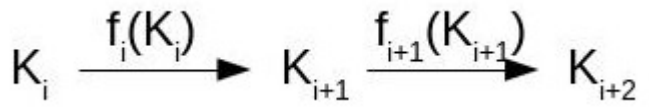
Радужные цепочки

Метод предложен в 2003 г. Филиппом Оечслин

Для каждой функции перехода следует применять новую редуцирующую функцию.

Окончания цепочек будут совпадать, только в случае, если коллизия произойдет на одной функции перехода.





Восстановление ключа

Восстановление ключа:

Дан шифротекст C, найти K – ключ шифрования.

Ищем среди сохраненных последних элементов цепочек занчения $R_{t-1}(C)$, $f_{t-1}(R_{t-2}(C))$, $f_{t-1}(f_{t-2}(R_{t-3}(C)))$, ..., $f_{t-1}(f_{t-2}(...(f_2((R_1(C)))..))$.

Затраты времени и памяти:

m – количество цепочек, t – длина цепочек, $t * m \approx N$.

$2*m*K_{LEN}$ – объем памяти для хранения цепочек.

$(t-1)*t/2$ – наибольшее число преобразований.

https://sesc-infosec.github.io/