## *SAFER*

$SAFER$ [1] (Secure And Fast Encryption Routine – стойкая и быстрая программа шифрования) – семейство итеративных блочных шифров. Далее рассматриваются шифры $SAFER\ K\text{-}64$ и $SAFER++$[2]. Алгоритмы оперируют только байтами и используют следующие операции над байтами (далее $\mathbb{B} = \{0,1,\dots,255\}$ – множество байтов):

$x \oplus y$ – побитовое сложение по модулю 2;

$x + y$ – сложение по модулю 256;

$x - y$ – вычитание по модулю 256.

Функция $exp$: $\mathbb{B} \longrightarrow \mathbb{B}$ определена как

$$exp(x) = \begin{cases} 45^x \bmod 257, & \text{если } x \neq 128; \\ 0, & \text{если } x = 128. \end{cases}$$

Отображение $x \longmapsto exp(x)$ взаимно однозначно на $\mathbb{B}$. Обратная функция обозначается как $log$. Обе функции следует задать таблицами.

**SAFER K–64.** Алгоритм шифрует 64-битовые блоки открытых данных под управлением 64-битового секретного ключа. Блок данных $X$ и ключ $K$ представляются в виде 8 байтовых массивов: $X = (x_1,\dots,x_8)$, $K = (k_1,\dots,k_8)$. Алгоритм состоит из $r$ раундов (рекомендуемое значение $r = 6$), выполняемых под управлением раундовых подключей $RK_i = (rk_{i1},\dots,rk_{i8})$, $i = 1,2,\dots,12$ (по два на каждый раунд), и заключительного преобразования под управлением раундового подключа $RK_{13} = (rk_{13,1},\dots,rk_{13,8})$.

### Генерация раундовых подключей

В качестве $RK_1$ берётся исходный секретный ключ $SK = (sk_1,\dots,sk_8)$. Остальные подключи вычисляются по формуле:

$$RK_{i+1} := rolbyte_{3i}(RK_i) + C_i\ ,\ i = 1,2,\dots,2r,$$

где $rolbyte_s(RK)$ – циклический сдвиг 8-байтового блока $RK$ на $s$ байтов влево, операция + выполняется над соответствующими байтами, а $C_i = (c_{i1}, c_{i2}, \dots, c_{i8})$ – 8-байтовая константа с компонентами

$$c_{ij} = exp(exp(9i + j) \bmod 256).$$

### Раундовые преобразования

Основу алгоритма составляет раундовое преобразование $RT[K_1, K_2]$: $\mathbb{B}^8 \longrightarrow \mathbb{B}^8$ над блоком $X = (x_1,\dots,x_8)$ под управлением подключей $K_1 = (k_{1,1}, k_{1,2}, \dots, k_{1,8})$, $K_2 = (k_{2,1}, k_{2,2}, \dots, k_{2,8}) \in \mathbb{B}^8$:

$RT[K_1, K_2](X) \equiv \{$
  **for** $i := 1,4,5,8$ **do** $x_i := exp(x_i \oplus k_{1i}) + k_{2i}$
  **for** $i := 2,3,6,7$ **do** $x_i := log(x_i + k_{1i}) \oplus k_{2i}$;
  $PHT_2(x_1, x_2)$;
  $PHT_2(x_3, x_4)$;
  $PHT_2(x_5, x_6)$;
  $PHT_2(x_7, x_8)$;
  **for** $i := 1,2$ **do** {
    $X := (x_1, x_3, x_5, x_7, x_2, x_4, x_6, x_8)$; // перестановка байтов в $X$
    $PHT_2(x_1, x_2)$;
    $PHT_2(x_3, x_4)$;
    $PHT_2(x_5, x_6)$;
    $PHT_2(x_7, x_8)$
  }
}.

---

[1] Автор шифров: *James L. Massey* (Швейцария)

[2] Авторами SAFER++ являются также *Gurgen Khachatrian* (США) и *Melsik L. Kuregian* (Армения)

Здесь $PHT_2$ – *псевдо-адамарово* преобразование (*Pseudo-Hadamard Transform*), выполняемое над 2-байтовыми словами $(a, b)$:

$$PHT_2(a, b) \equiv \{b := b + a;\ a := a + b\}.$$

Обратным к $RT[K_1, K_2]$ является преобразование

$$\begin{aligned}
&RT^{-1}[K_1, K_2](X) \equiv \{\\
&\quad PHT_2^{-1}(x_1, x_2);\\
&\quad PHT_2^{-1}(x_3, x_4);\\
&\quad PHT_2^{-1}(x_5, x_6);\ PHT_2^{-1}(x_7, x_8);\\
&\quad \textbf{for}\ i := 1,2\ \textbf{do}\ \{\\
&\qquad X := (x_1, x_3, x_5, x_7, x_2, x_4, x_6, x_8);\\
&\qquad PHT_2^{-1}(x_1, x_2);\\
&\qquad PHT_2^{-1}(x_3, x_4);\\
&\qquad PHT_2^{-1}(x_5, x_6);\\
&\qquad PHT_2^{-1}(x_7, x_8)\\
&\quad \};\\
&\quad \textbf{for}\ i := 2, 3, 6, 7\ \textbf{do}\ x_i := exp(x_i \oplus k_{2i}) - k_{1i};\\
&\quad \textbf{for}\ i := 1, 4, 5, 8\ \textbf{do}\ x_i := log(x_i - k_{2i}) \oplus k_{1i}\\
&\}.
\end{aligned}$$

Здесь $PHT_2^{-1}$ – преобразование, обратное к $PHT_2$:

$$PHT_2^{-1}(a, b) \equiv \{a := a - b;\ b := b - a\}.$$

### Алгоритм зашифрования *SAFER*

*Вход*: $M = (m_1, m_2, \dots, m_8)$ – 8-байтовый блок открытых данных.

1. ($r$ раундов зашифрования.)
   $\textbf{for}\ i := 1\ \textbf{to}\ r\ \textbf{do}\ RT[RK_{2i-1}, RK_{2i}](M);$
2. (Заключительное забеливание.)
   $\textbf{for}\ i := 1, 4, 5, 8\ \textbf{do}\ m_i := m_i \oplus rk_{2r+1,i};$
   $\textbf{for}\ i := 2, 3, 6, 7\ \textbf{do}\ m_i := m_i + rk_{2r+1,i}.$

*Выход*: $M$ – 8-байтовый блок шифртекста.

### Алгоритм расшифрования *SAFER*

*Вход*: $\mathrm{C} = (\mathrm{c1}, \mathrm{c2}, \dots, \mathrm{c8})$ – 8-байтовый блок шифртекста.

1. (Начальное отбеливание.)
   $\textbf{for}\ i := 2, 3, 6, 7\ \textbf{do}\ c_i := c_i - rk_{2r+1,i};$
   $\textbf{for}\ i := 1, 4, 5, 8\ \textbf{do}\ c_i := c_i \oplus rk_{2r+1,i};$
2. ($r$ раундов расшифрования.)
   $\textbf{for}\ i := r\ \textbf{downto}\ 1\ \textbf{do}\ RT^{-1}[RK_{2i}, RK_{2i+1}](C).$

*Выход*: C – 8-байтовый блок открытых данных.

***SAFER++.*** Алгоритм шифрует 128-битовые блоки открытых данных под управлением секретного ключа, длина которого может составлять 128 или 256 битов (16 или 32 байта). В -раундовом алгоритме используются $2r + 1$ 16-байтовых раундовых подключей $RK_1, \dots, RK_{2r+1}$ – по 2 подключа на каждый раунд и один ключ для заключительного преобразования. Число раундов $r = 7$ или 10 соответственно для 16- и 32-байтовых секретных ключей.

### Генерация раундовых подключей

Сначала вычисляются вспомогательные (“возмущающие”) 16-байтовые константы $B_i = (b_{i,1}, \dots, b_{i,16})$, $i = 2,3, \dots,21$, компоненты которых определяются как

$$b_{ij} = \begin{cases} exp\left(exp\left(byte(17i+j)\right)\right) & \text{для } 2 \le i \le 15, \\ exp\left(exp\left(byte(17i+j)\right)\right) & \text{для } 16 \le i \le 21, \end{cases}$$

где $byte(s)$ – байт со значением $s \; mod \; 256$. В формировании раундовых подключей участвует расширенный 17-байтовый подключ $EK = (ek_1, \dots, ek_{17})$. Обозначим через $rol(EK, s, t)$ преобразование, заключающееся в циклическом сдвиге $EK$ на $s$ байтов влево с одновременным циклическим сдвигом каждого байта на $t$ битов влево, а через $Select(X)$ – 16-байтовый блок, образованный первыми 16 байтами блока $X = (x_1, x_2, \dots, x_{16}, \dots)$; операцию $+$ будем трактовать как побайтовое сложение подблоков по модулю 256, т.е.

$$(x_1, x_2, \dots, x_{16}) + (y_1, y_2, \dots, y_{16}) = ((x_1 + y_1) \; mod \; 256, \dots, (x_{16}, \dots, y_{16}) \; mod \; 256).$$

Пусть $SK = (sk_1, \dots, sk_m)$ – $m$-байтовый ($m = 16$ или 32) секретный ключ. Раундовые подключи $RK_i$ с нечетными индексами формируются следующим образом:

$RK_1 := Select(K);$
$EK := (sk_1, sk_2, \dots, sk_{16}, sk_1 \oplus sk_2 \oplus \dots \oplus sk_{16});$
$\boldsymbol{for}\ i := 1\ \boldsymbol{to}\ 10\ \boldsymbol{do}\ \{$
    $rol(EK, 2, 6);$
    $RK_{2i+1} := Select(EK) + B_{2i+1}$
$\}.$

Раундовые подключи $RK_i$ с четными индексами формируются аналогично:

$\boldsymbol{if}\ m = 16\ \boldsymbol{then}\ EK := (sk_1, sk_2, \dots, sk_{16}, sk_1 \oplus sk_2 \oplus \dots \oplus sk_{16})$
    $\boldsymbol{else}\ EK := (sk_{17}, sk_{18}, \dots, sk_{32}, sk_{17} \oplus sk_{18} \oplus \dots \oplus sk_{32});$
$rol(EK, 1, 3);$
$RK_2 := Select(EK) + B_2;$
$\boldsymbol{for}\ i := 2\ \boldsymbol{to}\ 10\ \boldsymbol{do}\ \{$
    $rol(EK, 2, 6);$
    $RK_{2i} := Select(EK) + B_{2i}$
$\}.$

## Раундовые преобразования

Кроме операций над байтами, введенных выше, в алгоритме используются следующие преобразования над 16-байтовым блоком $X = (x_1, x_2, \dots, x_{16})$:

Преобразования $S_1[K]$ и $S_2[K]$ под управлением 16-байтового подключа K=($k_1$,…,$k_{16}$) определяются как

$S_1[K](X) \equiv \{$
$\boldsymbol{for}\ i := 1, 4, 5, 8, 9, 12, 13, 16\ \boldsymbol{do}\ x_i := x_i \oplus k;$
$\boldsymbol{for}\ i := 2, 3, 6, 7, 10, 11, 14, 15\ \boldsymbol{do}\ x_i := x_i + k$
$\}.$
$S_2[K](X) \equiv \{$
$\boldsymbol{for}\ i := 1, 4, 5, 8, 9, 12, 13, 16\ \boldsymbol{do}\ x_i := x_i + k;$
$\boldsymbol{for}\ i := 2, 3, 6, 7, 10, 11, 14, 15\ \boldsymbol{do}\ x_i := x_i \oplus k$
$\}.$

Обратные преобразования $S_1^{-1}[K](X)$ и $S_2^{-1}[K](X)$, возвращающие $X$ к исходному значению, получаются соответственно из $S_1[K](X)$ и $S_2[K](X)$ заменой операции + на – (по модулю 256).

Пусть $\pi$ – подстановка на множестве байтов, а $\pi^{-1}$ – обратная к ней. Преобразование $T[\pi]$ определяется как

$T[\pi](X) \equiv \{$
$\boldsymbol{for}\ i := 1, 4, 5, 8, 9, 12, 13, 16\ \boldsymbol{do}\ x_i := \pi[x_i];$
$\boldsymbol{for}\ i := 2, 3, 6, 7, 10, 11, 14, 15\ \boldsymbol{do}\ x_i := \pi^{-1}[x_i]$

$\}$.

Обратным к $T[\pi]$ является, очевидно, преобразование $T[\pi^{-1}]$. Например, обратным к $T[exp]$ будет $T[log]$, и наоборот.

Перестановка $P$ байтов в блоке $X = (x_1, x_2, \dots, x_{16})$ определяется как

$P(X) \equiv \{X := (x_9, x_6, x_3, x_{16}, x_1, x_{14}, x_{11}, x_8, x_5, x_2, x_{15}, x_{12}, x_{13}, x_{10}, x_7, x_4)\}$.

Другими словами, в позицию 1 перемещается байт из позиции 9, в позицию 2 – из позиции 6 и т.д. Обратная перестановка $P^{-1}$ имеет вид:

$$(5, 10, 3, 16, 9, 2, 15, 8, 1, 14, 7, 12, 13, 6, 11, 4),$$

т.е. в позицию 1 перемещается байт из позиции 5 и т.д.

Преобразование 16-$PHT$ определяется как

16-$PHT$ (X) $\equiv \{$
$4\text{-}PHT(x_1, x_2, x_3, x_4)$;
$4\text{-}PHT(x_5, x_6, x_7, x_8)$;
$4\text{-}PHT(x_9, x_{10}, x_{11}, x_{12})$;
$4\text{-}PHT(x_{13}, x_{14}, x_{15}, x_{16})$
$\}$,

где 4-$PHT(a, b, c, d)$ – следующее преобразование над 4-байтовым блоком $(a, b, c, d)$:

4-$PHT(a, b, c, d) \equiv \{$
$d := a + b + c + d$;
$a := a + d$;
$b := b + d$;
$c := c + d$
$\}$.

Преобразование 16- $PHT^{-1}$, обратное 16-$PHT$, получается заменой 4-$PHT$ на преобразование 4-$PHT^{-1}$, определяемое как

4-$PHT^{-1}(a, b, c, d) \equiv \{a := a - d;\ b := b - d;\ c := c - d;\ d := d - (a + b + c)\}$.

Раундовое преобразование $RT[K_1, K_2]$ под управлением 16-байтовых раундовых подключей $K_1$ и $K_2$ определяется как

$RT[K_1, K_2](X) \equiv \{$
$S_1[K_1](X)$;
$T[exp](X)$;
$S_2[K_2](X)$;
$P(X)$;
$16\text{-}PHT(X)$;
$P(X)$;
$16\text{-}PHT(X)$
$\}$.

Обратным к $RT[K_1, K_2]$ является преобразование

$RT^{-1}[K_1, K_2](X) \equiv \{$
16-$PHT^{-1}$ $(X)$;
$P^{-1}(X)$;
16-$PHT^{-1}$ $(X)$;
$P^{-1}(X)$;
$S_2^{-1}[K_2](X)$;
$T[log](X)$;
$S_1^{-1}[K_1](X)$
$\}$.

### Алгоритм зашифрования *SAFER++*

*Вход*: $M$ – 16-байтовый блок открытых данных.

$C := M$;

$\boldsymbol{for}\ i := 1\ \boldsymbol{to}\ r\ \boldsymbol{do}\ RT[RK_{2i-1}, RK_{2i}](C)$;

$S_1[RK_{2r+1}](C)$.

*Выход*: $C$ – 16-байтовый блок шифртекста.

### Алгоритм расшифрования *SAFER++*

*Вход*: $C$ – 16-байтовый блок шифртекста.

$M := C$;

$S_1^{-1}[RK_{2r+1}](M)$;

$\boldsymbol{for}\ i := r\ \boldsymbol{downto}\ 1\ \boldsymbol{do}\ RT^{-1}[RK_{2i-1}, RK_{2i}](M)$.

*Выход*: $M$ – 16-байтовый блок открытых данных.