УДК 681.3.06 : 519.248.681

И. Д. ГОРБЕНКО, д-р техн. наук, М. С. МИХАЙЛЕНКО, Т. А. ГРИНЕНКО

CAMELLIA – 128-БИТНЫЙ ПЕРСПЕКТИВНЫЙ БЛОЧНЫЙ СИММЕТРИЧНЫЙ ШИФР: МЕТОДЫ ПРЕОБРАЗОВАНИЙ, СВОЙСТВА И ОБЛАСТИ ПРИМЕНЕНИЯ

27 февраля 2003 года в европейском проекте NESSIE был представлен финальный список криптоалгоритмов, [1 – 3]. Алгоритм Camellia был включен в этот список как рекомендованный криптографический примитив.

Под аббревиатурой NESSIE скрывается название международного криптографического проекта New European Schemes for Signatures, Integrity and Encryption («Новые европейские схемы для электронных подписей, обеспечения целостности информации и шифрования»). Этот рассчитанный на три года проект запущен в январе 2000 года под эгидой Европейской комиссии. Его цель – создать «строительные блоки для будущих стандартных протоколов информационного сообщества», главными участниками являются научные и индустриальные институты семи стран: Бельгии, Великобритании, Германии, Израиля, Италии, Норвегии и Франции.

Задачи перед проектом были поставлены весьма широкие: отобрать десяток основных криптографических примитивов. В этот набор, в частности, входят алгоритмы блочного и поточного шифрования, генераторы случайных чисел, схемы быстрой аутентификации пакетов данных, хэш-функции и алгоритмы цифровой подписи. В качестве основных критериев отбора претендентов названы безопасность, производительность, гибкость и требования рынка.

Camellia это следующее поколение 128-битного блочного криптографического алгоритма, разработанного в Японии специалистами Телеграфной и Телефонной Корпорации Nippon и Электрической Корпорации Mitsubishi, поддерживающего три размера ключей: l_k = 128, 192 и 256 бит. Блочный симметричный алгоритм Camellia был разработан не только как высоко защищенный криптографический шифр, но также как алгоритм, легко переносимый на различные аппаратные платформы.

Целью настоящей статьи является ознакомление специалистов в области информационной безопасности с алгоритмом блочного симметричного шифрования, рекомендованного ведущими специалистами в данной области в качестве криптографического примитива.

1 Общая характеристика криптоалгоритма

Алгоритм Camellia является блочным симметричным криптоалгоритмом. Длина блока равна l_u = 128 бит. Криптографические преобразования блоков информации осуществляются с использованием ключевых данных. Ключ, вводимый в средства реализации Camellia, называют исходным ключом K. Разрешенными длинами исходного ключа есть l_k = 128, 192 и 256 бит. Ключи, используемые в циклах преобразования, формируются из исходного K и называются цикловыми ключами. Эти ключи формируются специальным криптографическим алгоритмом.

Криптоалгоритм Camellia может применяться в пяти режимах:

1) блочного шифрования;
2) поточного шифрования;
3) поточного шифрования с обратной связью;
4) выработки ключевой хэш-функции (кода-аутентификации);
5) генератора псевдослучайных последовательностей.

2 Представление преобразуемой информации и ключей

В алгоритме Camellia преобразования выполняются при длине блоков информации l_u, = = 128 бит и длинах исходных ключей l_k. = 128, 192 и 256 бит. Необходимая стойкость в алгоритме обеспечивается за счет многоциклового преобразования, причем в каждом из циклов применяются ключевые преобразования, то есть преобразования по развернутым ключам.

Пусть необходимо зашифровать M_i блок информации длиной 128 бит. Блок представляется в виде одномерного массива, выходные данные C_i и развернутый цикловой ключ K_i также представляются одномерными массивами. В алгоритме Camellia число циклов преобразования зависит от длины исходного ключа l_k . В табл. 1 приведено значение количества циклов как функции l_k .

Таблица 1

Длина ключа l_k, бит	128	192	256
Число циклов	18	24	24

При l_k = 128 после 6-го и 12-го циклов находятся два FL/FL^{-1} функциональных уровня (при l_k = 192, 256 функции FL/FL^{-1} расположены после 6-го, 12-го и 18-го циклов). Кроме того, перед первым и после последнего циклов выполняются криптографические преобразования – сложение по модулю 2 с цикловыми ключами. Сам цикл имеет структуру Фейстеля. Ключ преобразования имеет вид конкатенации двух 128-битных последовательностей $K_{(128)} = K_{L(128)} \| K_{R(128)}$, где $K_{R(128)}$ используется только на длинах l_k = 192, 256 бит.

3 Табличные и криптографические преобразования

На рис. 1 показана процедура зашифрования для 128-битового ключа [5]. Блок зашифрования данных имеет 18-цикловую структуру Фейстеля с двумя FL/FL^{-1}-функциональными уровнями после 6-го и 12-го циклов и 128-битовыми XOR операциями рандомизации перед первым циклом и после последнего цикла. Блок развертывания ключей генерирует подключи $kw_{t(64)}$ (t = 1, 2, 3, 4), $ku(64)$ (u = 1,2, … 18) и $kl_{v(64)}$ (v = 1,2,3,4) из секретного исходного ключа K.

В блоке рандомизации данных сначала выполняется операция XOR (сумма по модулю 2) открытого текста $M_{(128)}$ с $kw_{1(64)} \| kw_{2(64)}$ и выделение $L_{0(64)}$ и $R_{0(64)}$ равной длины, то есть:

$$M_{(128)} \oplus (kw_{1(64)} \| kw_{2(64)}) = L_{0(64)} \| R_{0(64)} ;$$

где $\oplus$ побитовая операция «Исключающее ИЛИ», а $\|$ конкатенация двух операндов.
В циклах для r = от 1 до 18 выполняется операции (кроме r = 6 и 12):

$$L_r = R_{r-1} \oplus F(L_{r-1}, k_t),$$
$$R_r = L_{r-1}.$$

Для r = 6 и 12 выполняется следующее:

$$L_{\bar{r}} = R_{r-1} \oplus F(L_{r-1}, k_r),$$
$$R_{\bar{r}} = L_{r-1},$$
$$L_r = FL(L_{\bar{r}}, kl_{2r/6-1}),$$
$$R_r = FL^{-1}(R_{\bar{r}}, kl_{2r/6}).$$

И, наконец, $R_{18(64)}$ и $L_{18(64)}$ конкатенируются и складываются по модулю 2 с $kw_{3(64)} \| kw_{4(64)}$. В результате в первом режиме формируется блок-криптограмма длиной 128 бит:

$$C_{(128)} = (R_{18(64)} \| L_{18(64)}) \oplus (kw_{3(64)} \| kw_{4(64)}) .$$

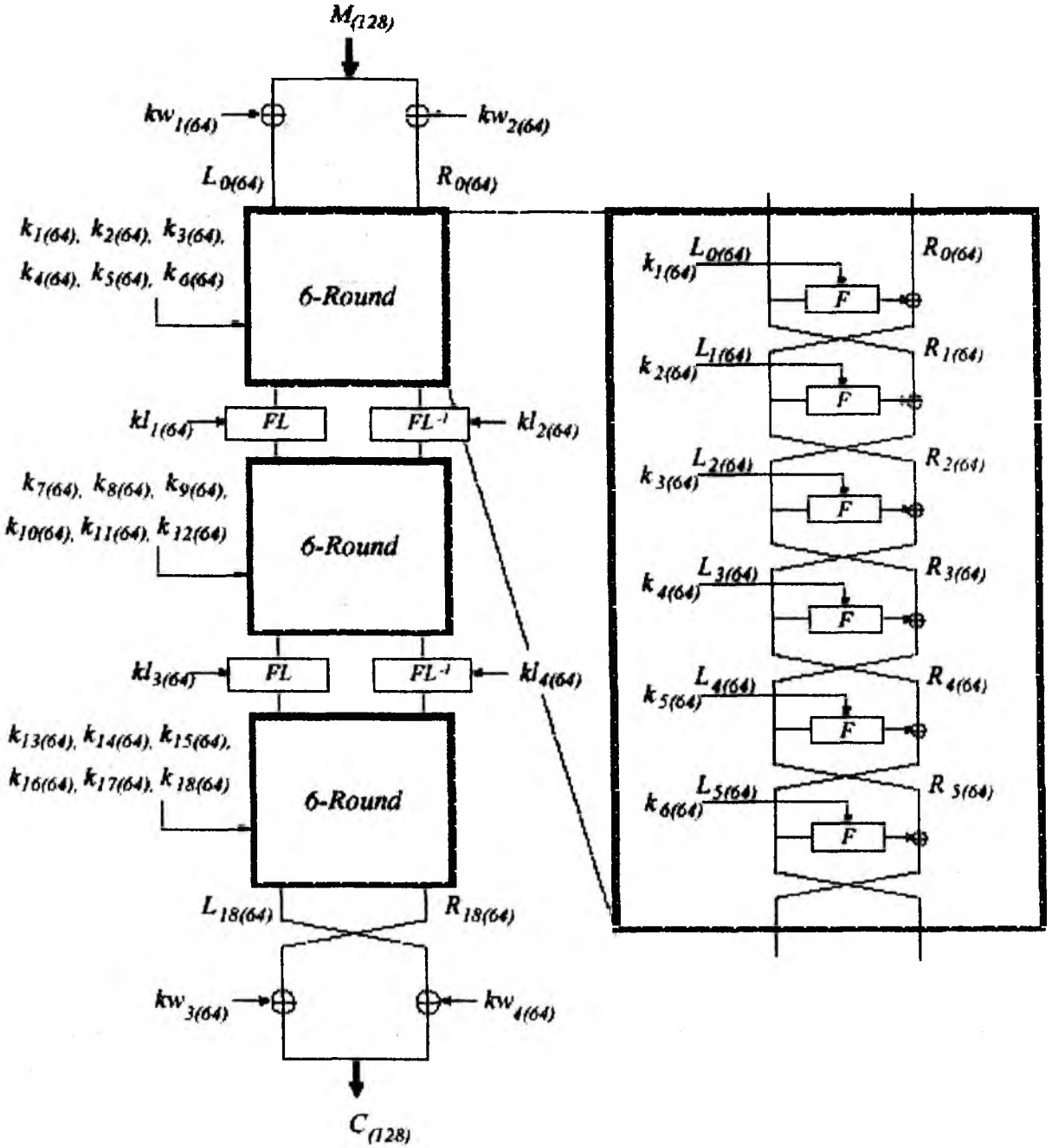

Рис. 1

При осуществлении криптопреобразований в алгоритме Camellia используется некоторый набор функций. Рассмотрим кратко функции F, FL и FL^{-1}, которые используются в алгоритме Camellia.

F-функция

На рис. 2 показана F-функция, которая определяется следующим образом:

$F : L * L \to L$,

L обозначает векторное пространство 64-битовых элементов

$$(X_{(64)}, k_{(64)}) \to Y_{(64)} = P(S(X_{(64)} \oplus k_{(64)})).$$

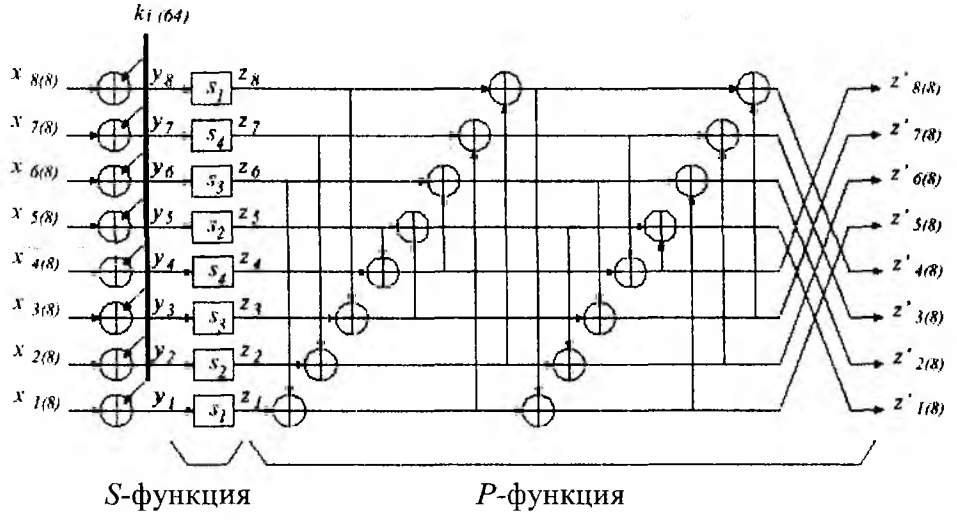

Рис. 2

S-функция

S-функция является частью *F*-функции, которая определяется следующим образом: $S: L \to L$.

$$l_{1(8)} \| l_{2(8)} \| l_{3(8)} \| l_{4(8)} \| l_{5(8)} \| l_{6(8)} \| l_{7(8)} \| l_{8(8)} \to$$
$$\to l_{\bar{1}(8)} \| l_{\bar{2}(8)} \| l_{\bar{3}(8)} \| l_{\bar{4}(8)} \| l_{\bar{5}(8)} \| l_{\bar{6}(8)} \| l_{\bar{7}(8)} \| l_{\bar{8}(8)},$$

где

$l_{\bar{1}(8)} = s_1(l_{1(8)})$; $l_{\bar{2}(8)} = s_2(l_{2(8)})$; $l_{\bar{3}(8)} = s_3(l_{3(8)})$; $l_{\bar{4}(8)} = s_4(l_{4(8)})$; $l_{\bar{5}(8)} = s_2(l_{5(8)})$; $l_{\bar{6}(8)} = s_3(l_{6(8)})$;
$l_{\bar{7}(8)} = s_4(l_{7(8)})$; $l_{\bar{8}(8)} = s_1(l_{8(8)})$.

s-блоки

Четыре *s*-блока шифра Camellia являются аффинным эквивалентом функции инверсии на $GF(2^8)$, что представлено в табл. 2,3,4 и 5. Ниже показано алгебраическое представление *s*-блоков:

$$s_1 : B \to B,$$
$$x_{(8)} \to h(g(f(0xc5 \oplus x_{(8)}))) \oplus 0x6e,$$
$$s_2 : B \to B,$$
$$x_{(8)} \to s_1(x_{(8)}) <<< 1,$$
$$s_3 : B \to B,$$
$$x_{(8)} \to s_1(x_{(8)}) >>> 1,$$
$$s_4 : B \to B,$$
$$x_{(8)} \to s_1(x_{(8)} <<< 1),$$

здесь <<<n (>>>n) – циклический сдвиг влево (вправо) операнда на n бит.
B обозначает векторное пространство 8-битовых элементов (байтов), функции *f*, *g* и *h* задаются следующим образом:
$f : B \to B$

$$a_{1(1)} \| a_{2(1)} \| a_{3(1)} \| a_{4(1)} \| a_{5(1)} \| a_{6(1)} \| a_{7(1)} \| a_{8(1)} \rightarrow$$

$$\rightarrow b_{1(1)} \| b_{2(1)} \| b_{3(1)} \| b_{4(1)} \| b_{5(1)} \| b_{6(1)} \| b_{7(1)} \| b_{8(1)},$$

где

$b_1 = a_6 \oplus a_2$; $b_2 = a_7 \oplus a_1$; $b_3 = a_8 \oplus a_5 \oplus a_3$; $b_4 = a_8 \oplus a_3$; $b_5 = a_7 \oplus a_4$; $b_6 = a_5 \oplus a_2$;
$b_7 = a_8 \oplus a_1$; $b_8 = a_6 \oplus a_4$.

$g : B \rightarrow B$

$$a_{1(1)} \| a_{2(1)} \| a_{3(1)} \| a_{4(1)} \| a_{5(1)} \| a_{6(1)} \| a_{7(1)} \| a_{8(1)} \rightarrow$$

$$\rightarrow b_{1(1)} \| b_{2(1)} \| b_{3(1)} \| b_{4(1)} \| b_{5(1)} \| b_{6(1)} \| b_{7(1)} \| b_{8(1)},$$

где

$(b_8 + b_7 a + b_6 a^2 + b_5 a^3) + (b_4 + b_3 a + b_2 a^2 + b_1 a^3)\beta =$
$= 1/((a_8 + a_7 a + a_6 a^2 + a_5 a^3) + (a_4 + a_3 a + a_2 a^2 + a_1 a^3)\beta)$;

Эта инверсия выполняется в $GF(2^8)$, где β является элементом $GF(2^8)$, который удовлетворяет условию $\beta^8 + \beta^6 + \beta^5 + \beta^3 + 1 = 0$, и $a = \beta^{238} = \beta^6 + \beta^5 + \beta^3 + \beta^2$ является элементом $GF(2^4)$, который удовлетворяет условию: $a^4 + a + {}^1 = 0$.

$h : B \rightarrow B$

$$a_{1(1)} \| a_{2(1)} \| a_{3(1)} \| a_{4(1)} \| a_{5(1)} \| a_{6(1)} \| a_{7(1)} \| a_{8(1)} \rightarrow$$

$$\rightarrow b_{1(1)} \| b_{2(1)} \| b_{3(1)} \| b_{4(1)} \| b_{5(1)} \| b_{6(1)} \| b_{7(1)} \| b_{8(1)},$$

где

$b_1 = a_5 \oplus a_6 \oplus a_2$; $b_2 = a_6 \oplus a_2$; $b_3 = a_7 \oplus a_4$; $b_4 = a_8 \oplus a_2$; $b_5 = a_7 \oplus a_3$; $b_6 = a_8 \oplus a_1$;
$b_7 = a_5 \oplus a_1$; $b_8 = a_6 \oplus a_3$.

В табл. 2, 3, 4, 5 приведены s-блоки подстановок.

Таблица 2

112	130	44	236	179	39	192	229	228	134	87	53	234	12	174	65
35	239	107	147	69	25	165	33	237	14	79	78	29	101	146	189
134	184	175	143	124	235	31	206	62	48	220	95	94	197	11	26
166	225	57	202	213	71	93	61	217	1	90	214	81	86	108	77
139	13	154	102	251	204	176	45	116	18	43	32	240	177	132	153
223	76	203	194	52	126	118	5	109	183	169	49	209	23	4	215
20	88	58	97	222	27	17	28	50	15	156	22	83	24	242	34
254	68	207	178	195	181	122	145	36	8	232	168	96	252	105	80
170	208	160	125	161	137	98	151	84	91	30	149	224	255	100	210
16	196	0	72	163	247	117	219	138	3	230	218	9	63	221	148
135	92	131	2	205	74	144	51	115	103	246	243	157	127	191	226
82	155	216	38	200	55	198	59	129	150	111	75	19	190	99	46
233	121	167	140	159	110	188	142	41	245	249	182	47	253	180	89
120	152	6	106	231	70	113	186	212	37	171	66	136	162	141	250
114	7	185	85	248	238	172	10	54	73	42	104	60	56	241	164
64	40	211	123	187	201	67	193	21	227	173	244	119	199	128	158

Таблица 3

224	5	88	217	103	78	129	203	201	11	174	106	213	24	93	130
70	223	214	39	138	50	75	66	219	28	158	156	58	202	37	123
13	113	95	31	248	215	62	157	124	96	185	190	188	139	22	52
77	195	114	149	171	142	186	122	179	2	180	173	162	172	216	154
23	26	53	204	247	153	97	60	232	33	86	64	225	99	9	51
191	152	151	133	104	252	238	10	218	111	83	98	163	46	8	175
40	176	116	194	189	54	34	56	100	30	57	44	166	48	229	68
253	136	159	101	135	107	244	35	72	16	209	81	192	249	210	160
85	161	65	250	67	19	196	47	168	182	60	43	193	255	200	165
32	137	0	144	71	239	234	183	21	6	205	181	18	126	187	41
15	184	7	4	155	148	33	102	230	206	237	231	59	254	127	197
164	55	177	76	145	110	141	118	3	45	222	150	38	125	198	92
211	242	79	25	63	220	121	29	82	235	243	109	94	251	105	178
240	49	12	212	207	140	226	117	169	74	87	132	17	69	27	245
228	14	115	170	241	221	89	20	108	146	84	208	120	112	227	73
128	80	167	246	119	147	134	131	42	199	91	233	238	143	1	61

Таблица 4

56	65	22	118	217	147	96	242	114	194	171	154	117	6	87	160
145	247	181	201	162	140	210	144	246	7	167	39	142	178	73	222
67	92	215	199	62	245	143	103	31	24	110	175	47	226	133	13
83	240	156	101	234	163	174	158	236	129	45	107	168	43	54	166
197	134	77	51	253	102	88	150	58	9	149	16	120	216	66	204
239	38	229	97	26	63	59	130	182	219	212	152	232	139	2	235
10	44	29	176	111	141	136	14	25	135	78	11	169	12	121	17
127	34	231	89	225	218	61	200	18	4	116	84	48	126	180	40
85	104	80	190	208	196	49	203	42	173	15	202	112	255	50	105
8	98	0	36	209	251	188	237	69	129	115	109	132	159	238	74
195	46	193	1	230	37	72	153	185	179	123	249	206	191	223	113
41	205	108	19	100	155	99	157	192	75	183	165	137	95	177	23
244	187	211	70	207	55	94	71	148	250	252	91	151	254	90	172
60	76	3	53	243	35	184	93	106	146	213	33	68	81	198	125
57	131	220	170	124	119	86	5	27	164	21	52	30	28	208	82
32	30	233	189	221	228	161	224	138	241	214	122	187	227	64	79

Таблица 5

112	44	179	192	228	87	234	174	35	107	69	165	237	79	29	146
134	175	124	31	62	220	94	11	166	57	213	93	217	90	81	108
139	154	251	176	116	43	240	182	223	203	53	118	109	169	209	4
20	58	222	17	50	156	83	242	254	207	195	122	36	232	96	105
170	160	161	98	84	30	224	100	16	0	163	117	138	230	9	221
135	131	205	144	115	246	157	191	82	216	200	198	129	111	19	99
233	167	159	188	61	249	47	180	120	6	251	113	212	171	136	141
114	185	248	172	54	42	60	241	64	211	187	67	21	173	119	128
130	236	39	229	133	53	12	65	239	147	25	33	14	78	101	189
134	143	235	206	48	85	197	26	225	202	71	61	1	214	86	77
13	102	204	45	16	32	177	153	76	194	126	5	183	49	23	215
88	97	27	28	15	22	24	34	68	178	181	145	8	168	252	80
208	125	137	151	91	149	255	210	196	72	247	219	3	218	63	148
92	2	74	51	103	243	127	226	155	38	55	59	150	75	190	46
121	140	110	142	246	132	253	89	152	106	70	136	37	66	162	250
7	85	238	10	73	104	56	164	40	123	201	193	227	244	199	158

P-функция

P-функция является частью *F*-функции, которая определяется следующим образом: $P: L \to L$,

$$z_{1(8)} \| z_{2(8)} \| z_{3(8)} \| z_{4(8)} \| z_{5(8)} \| z_{6(8)} \| z_{7(8)} \| z_{8(8)} \to$$
$$\to z_{\bar{1}(8)} \| z_{\bar{2}(8)} \| z_{\bar{3}(8)} \| z_{\bar{4}(8)} \| z_{\bar{5}(8)} \| z_{\bar{6}(8)} \| z_{\bar{7}(8)} \| z_{\bar{8}(8)},$$

где

$z_{\bar{1}} = z_1 \oplus z_3 \oplus z_4 \oplus z_6 \oplus z_7 \oplus z_8$; $z_{\bar{2}} = z_1 \oplus z_2 \oplus z_4 \oplus z_5 \oplus z_7 \oplus z_8$; $z_{\bar{3}} = z_1 \oplus z_2 \oplus z_3 \oplus z_5 \oplus z_6 \oplus z_8$;

$z_{\bar{4}} = z_2 \oplus z_3 \oplus z_4 \oplus z_5 \oplus z_6 \oplus z_7$; $z_{\bar{5}} = z_1 \oplus z_2 \oplus z_6 \oplus z_7 \oplus z_8$; $z_{\bar{6}} = z_2 \oplus z_3 \oplus z_5 \oplus z_7 \oplus z_8$;

$z_{\bar{7}} = z_3 \oplus z_4 \oplus z_5 \oplus z_6 \oplus z_8$; $z_{\bar{8}} = z_1 \oplus z_4 \oplus z_5 \oplus z_6 \oplus z_7$.

Это преобразование может быть представлено так же в следующем виде:

$$\begin{pmatrix} z_8 \\ z_7 \\ \vdots \\ z_1 \end{pmatrix} \to \begin{pmatrix} z_{\bar{8}} \\ z_{\bar{7}} \\ \vdots \\ z_{\bar{1}} \end{pmatrix} = P \begin{pmatrix} z_8 \\ z_7 \\ \vdots \\ z_1 \end{pmatrix},$$

где

$$P = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}.$$

FL-функция

На рис. 3 показана *FL*-функция, которая определяется так, как показано ниже: $FL: L*L \to L$,

$$(X_{L(32)} \| X_{R(32)}, kl_{L(32)} \| kl_{R(32)}) \to Y_{L(32)} \| Y_{R(32)},$$

где

$Y_{R(32)} = ((X_{L(32)} \cap kl_{L(32)}) <<< 1) \oplus X_{R(32)}$ ($\cap$ – побитовая операция И);

$Y_{L(32)} = (Y_{R(32)} \cup kl_{R(32)}) \oplus X_{L(32)}$ (U – Побитовая операция ИЛИ).

FL^{-1}-функция

На рис. 4 показана FL^{-1}-функция, которая определяется следующим образом: $FL^{-1}: LxL \to L$,

$$(Y_{L(32)} \| Y_{R(32)}, kl_{L(32)} \| kl_{R(32)}) \to X_{L(32)} \| X_{R(32)},$$

где

$$X_{L(32)} = (Y_{R(32)} \cup kl_{R(32)}) \oplus Y_{L(32)};$$
$$X_{R(32)} = ((X_{L(32)} \cap kl_{L(32)}) <<< 1) \oplus Y_{R(32)}.$$

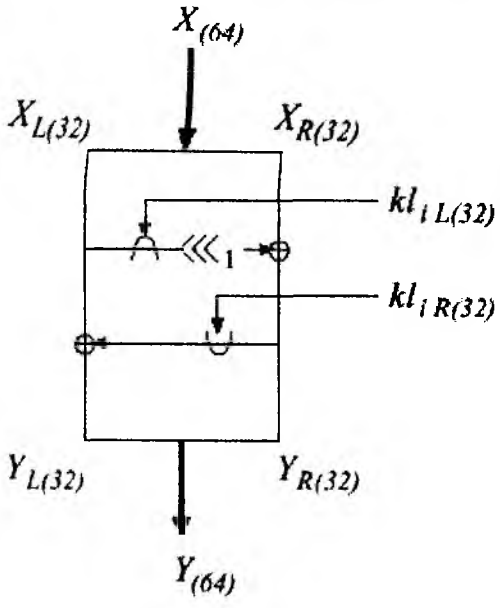

Рис. 3

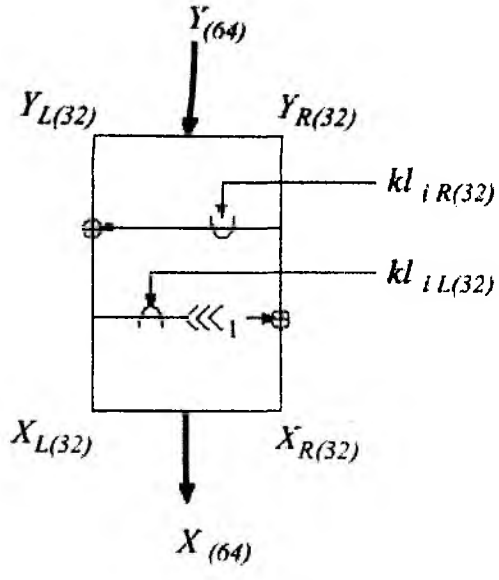

Рис. 4

Мы рассмотрели схему шифрования на длине l_k = 128 бит, шифрование на длинах l_k = 192, 256 бит выполняется аналогично за исключением того, что в схему добавляется блок из шести циклов Фейстеля и после 18-го цикла выполняются FL / FL^{-1} преобразования.

4 Выработка цикловых ключей

В алгоритме Camellia применяется принцип разворачивания цикловых ключей [5]. Сущность его заключается в том, что необходимое число цикловых ключей вырабатывается из исходного ключа K. Схема разворачивания ключей представлена на рис. 5. Как исходные так и цикловые ключи представляются в виде одномерных массивов. В блоке таблицы ключей шифра Camellia вводятся две 128-битовые переменные $K_{L(128)}$, $K_{R(128)}$ и четыре 64-битовые переменные $K_{LL(64)}$, $K_{LR(64)}$, $K_{RL(64)}$ и $K_{RR(64)}$, которые заданы так, чтобы удовлетворялись следующие отношения:

$K_{(128)} = K_{L(128)}$, $K_{R(128)} = 0$ для 128-битового ключа,

$K_{(192)} = K_{L(128)} \| K_{RL(64)}$, $K_{RR(64)} = \overline{K}_{RL(64)}$ для 192-битового ключа,

$K_{(256)} = K_{L(128)} \| K_{R(128)}$ для 256-битового ключа.

С помощью этих переменных генерируются две 128-битовых переменных $K_{A(128)}$ и $K_{B(128)}$, как показано на рис. 5, где $K_{B(128)}$ используется только при длине секретного ключа 192 или 256 бит. Сначала выполняется XOR операция $K = K_{L(128)}$ с $K_{R(128)}$, и результат «шифруется» за два цикла с использованием постоянных значений $\sum_{1(64)}$ и $\sum_{2(64)}$ как «ключей»; результат XOR-ся с $K_{L(128)}$ и зашифровуется еще на двух циклах с использованием $\sum_{3(64)}$ и $\sum_{4(64)}$. В итоге результирующим значением является $K_{A(128)}$. И наконец, выполняется XOR-операция $K_{A(128)}$ с $K_{R(128)}$, результат шифруется за два цикла с использованием постоянных значений $\sum_{5(64)}$ и $\sum_{6(64)}$. Результирующим значением является $K_{B(128)}$. Постоянные значения $\sum_i$ приведены в табл. 6.

Подключи $kw_{t(64)}$, $k_{u(64)}$ и $kl_{v(64)}$ генерируются из (левой или правой половины) циклически сдвинутых значений $K_{L(128)}$, $K_{R(128)}$, $K_{A(128)}$ и $K_{B(128)}$.

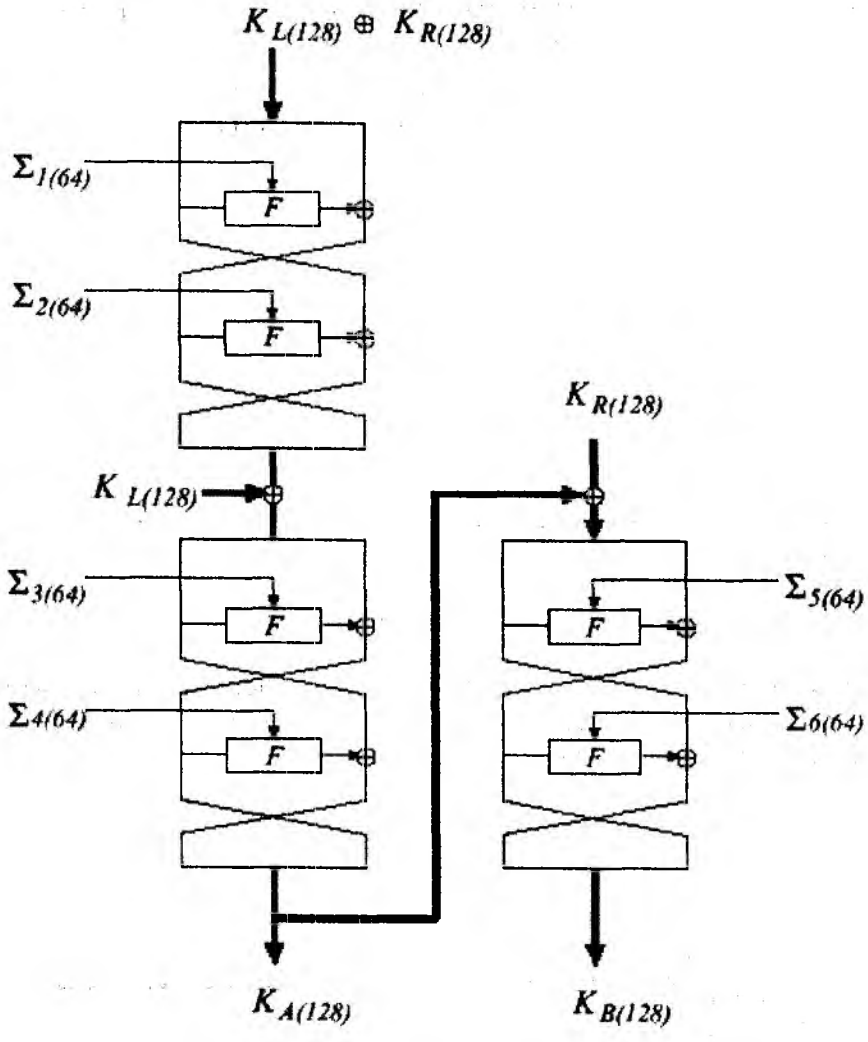

Рис. 5

Таблица 6

$\sum_{1(64)}$	0xA09E667F3BCC908B
$\sum_{2(64)}$	0xB67AE8584CAA73B2
$\sum_{3(64)}$	0xC6EF372FE94F82BE
$\sum_{4(64)}$	0x54FF53A5F1D36F1C
$\sum_{5(64)}$	0x10E527FADE682D1D
$\sum_{6(64)}$	0xB05688C2B3E6C1FD

5 Обратные преобразования (расшифрование)

Процедура расшифрования в алгоритме Camellia может быть выполнена таким же способом, как процедура зашифрования, путем изменения на обратный порядка подключей. На рис. 6 показана процедура расшифрования для 128-битового ключа. Алгоритм расшифрования также имеет 18-цикловую структуру Файстеля с двумя FL/FL^{-1}-функциональными слоями после 6-го и 12-го циклов и 128-битовыми операциями XOR перед первым циклом и после последнего цикла. Блок таблицы ключей генерирует подключи $kw_{t(64)}$ (t = 1, 2, 3, 4), $ku(64)$ (u = 1,2, … 18) и $kl_{v(64)}$ (v = 1,2,3,4) из секретного ключа K.

При расшифровании сначала выполняется операция XOR шифротекста $C_{(128)}$ с $kw_{3(64)}$ $kw_{4(64)}$ и выделение в $R_{18(64)}$ и $L_{18(64)}$ равной длины, то есть

$$C_{(128)} \oplus (kw_{3(64)} \| kw_{4(64)}) = R_{18(64)} \| L_{18(64)} .$$

Для r = от 18 до 1, выполняются следующие операции (кроме r = 13 и 7):

$$R_{r-1} = L_r \oplus F(R_r, k_t),$$
$$L_{r-1} = R_r .$$

Для r = 13 и 7 выполняется следующее:

$$R_{\bar{r}-1} = L_r \oplus F(R_r, k_r),$$
$$L_{\bar{r}-1} = R_r ,$$
$$R_{r-1} = FL(R_{\bar{r}-1}, kl_{2(r-1)/6-1}),$$
$$L_{r-1} = FL^{-1}(L_{\bar{r}-1}, kl_{2(r-1)/6-1}) .$$

И, наконец, $L_{0(64)}$ и $R_{0(64)}$ конкатенируются и XOR-ся с $kw_{1(64)} \| kw_{2(64)}$. Результирующее значение является открытым текстом, то есть

$$M_{(128)} = (L_{0(64)} \| R_{0(64)}) \oplus (kw_{1(64)} \| kw_{2(64)}) .$$

Схема расшифрования на длине l_k = 192, 256 бит выполняется аналогично зашифрованию, за исключением того, что в схему добавляется блок из шести циклов Фейстеля и перед 19-м циклом выполняются FL / FL^{-1} преобразования.

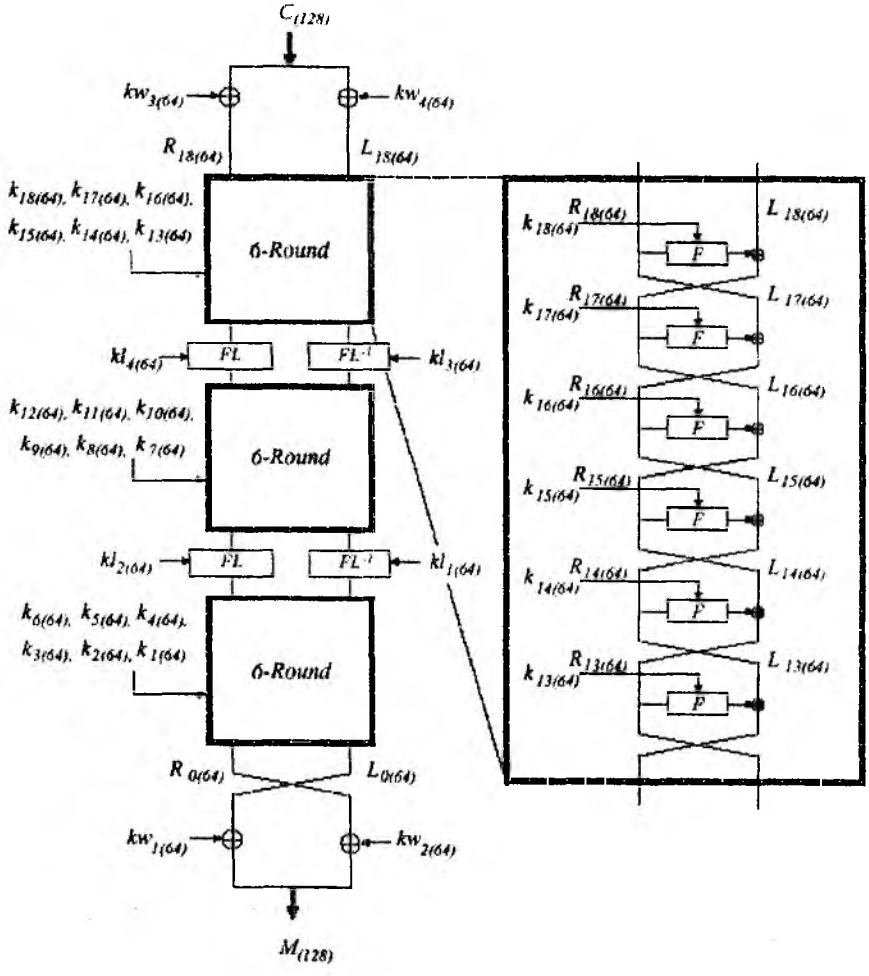

Рис. 6

6 Анализ стойкости криптоалгоритма

В процессе всех этапов «общественного», научного и практического обсуждения и анализа алгоритма Camellia, а также исследования и тестирования его со стороны Европейской комиссии NESSIE особое внимание уделялось анализу и оценке по критериям реальной криптозащищенности, статистической безопасности и надежности математической базы криптоалгоритма. На сегодняшний день относительно алгоритма Camellia не известно, а, возможно, и не существует ни одного метода криптоанализа, сложность реализации которого была бы меньше, чем методы, основанные на «грубой силе» (переборе). Алгоритм оказался защищенным от атак, реализованных на основе: дифференциального криптоанализа; поиска наилучшей дифференциальной характеристики; расширения для дифференциального криптоанализа; линейного криптоанализа; вторжения с использованием связанных ключей; вторжения с частичным угадыванием ключа; вторжения на основе обработки сбоев; интерполяционного (алгебраического) вторжения; поиска лазеек. Основу криптозащищенности в алгоритме составляют криптографические преобразования на множестве циклов.

Ниже будут рассмотрены результаты анализа статистической криптостойкости алгоритма, проведенные авторами. При анализе статистической безопасности алгоритма Camellia оценивалась связанность криптограмм между собой и со входными блоками открытых текстов, а также определялась избыточность криптограмм. Избыточность связана с зависимостью символов сообщений соответствующего алфавита и неодинаковой вероятностью их появления.

Пусть m_1, m_2, …, m_n – последовательность бит, принадлежащих открытому блоку исходного текста, а c_1, c_2, …, c_n – последовательность бит, принадлежащих соответствующей криптограмме. Символы m_i и c_i являются двоичными, то есть принимают значения 1 или 0. Связанность блоков M_i и C_i можно определить по формуле:

$$F = f(m_i, c_i) = \sum_{j=1}^{l_u} (m_i \oplus c_i),$$

а блоков C_i и C_k :

$$F = f(c_i, c_k) = \sum_{j=1}^{l_u} (c_i \oplus c_k),$$

где l_u длина блока шифруемой информации (128 бит).

Математическое ожидание зашифрованного блока можно определить по формуле:

$$M = f(b_j) = \frac{\sum_{j=1}^{l_u} b_j}{l_u}.$$

Расстояние Хемминга для блоков M_i и C_i можно найти как

$$H = f(M_j, C_j) = M_j \oplus C_j.$$

Будем считать блоки независимыми (некоррелированными), если $F = l_u / 2$, а математическое ожидание стремится к ½. Оценка статистической безопасности проведена при следующих параметрах работы криптоалгоритма: случайный ключ, случайные данные, ключ постоянный.

Таблица 7

Режим	Связанность блоков		Математическое ожидание	Расстояние Хемминга M_i, C_i
	M_i, C_i	C_k, C_i	C_i	
Работа алгоритма	64,0015	64,0494	0,499996	63,9985
Схема разворачивания ключей	64,0207	64,9193	0,48711	63,9793

При определении уровня корреляции блоков брался исходный файл и криптограмма, полученная в результате работы криптоалгоритма Camellia над этим исходным файлом. Затем считывались оба файла по 16 байт (128 бит), и находилась связанность каждого из прочитанных блоков. В конце результат делился на число прочитанных блоков. При поиске корреляции между криптограммами исходный текст изменялся, зашифровывался, и связанность искалась уже между этой криптограммой и криптограммой предыдущей. Такая же схема анализа связанности блоков использовалась и при нахождении связанности для схемы разворачивания ключей.

Математическое ожидание, также как и расстояние Хемминга, считалось для криптограмм, полученных в результате работы криптоалгоритма и для схемы разворачивания ключей.

Оценим качество зашифрования на основе анализа избыточности

Существование избыточности в зашифрованных текстах исследовалось с использованием программы – архиватора WinRAR. В роли входных текстов использовались:

DLL – динамическая библиотека WINDOWS;

DOC – документ Microsoft WORD 2000;

EXE – исполняемый файл WINDOWS;

PDF – документ переносимого формата;

TXT – обычный текст ANSI;

ZIP – файл-архив.

В качестве показателей сжатия использовались следующие соотношения:

$$k_1 = \frac{l'_m}{l_m} \text{ и } k_2 = \frac{l'_c}{l_m},$$

где

l'_m – длина сжатого открытого текста; l_m – длина входного открытого текста; l'_c – длина сжатой криптограммы.

В табл. 8 в качестве примеров приведены значения k_1 и k_2.

Таблица 8

Тип файла	l_m	l'_m	l'_c	k_1	k_2
DLL	1096736	524234	1071688	0,477994	0,977161
DOC	1085952	142893	607064	0,131583	0,559015
EXE	1060864	247488	810104	0,233289	0,763626
PDF	1062516	568422	887720	0,534977	0,835488
TXT	1318671	114096	707309	0,086523	0,536380
ZIP	1081634	1078088	1081727	0,996721	1,000085

Из анализа данных табл. 7 и 8 следует, что связанность блоков M_i и C_i стремится к $F = l_u / 2$ (то есть к 64), математическое ожидание стремится к ½, расстояние Хемминга – к 64, k_2 намного больше k_1, а в некоторых случаях приближается к единице. Следовательно, зашифрованный файл практически не имеет избыточности.

Дифференциальный и линейный криптоанализ

В 1990 г. Бихам и Шамир опубликовали статью, в которой описали аналитическую атаку на DES, сложность которой оказалась меньше, чем грубая сила (дифференциальный криптоанализ). Суть метода дифференциального криптоанализа сводится к нахождению разности между специально подбираемыми блоками M_i и M_j и соответствующими им блоками криптограмм C_i и C_j. То есть в начале подбираются пары сообщений M_i и M_j, которые имеют определенное расстояние между собой. Затем эти два блока пропускаются через схему шифрования, где просчитываются разности на каждом из циклов. Находится разность расстояний $\Delta M, \Delta C$. Далее для некоторых разностей, используя стандартные подстановки, можно найти вероятности появления ключей. По мере того, как все больше пар M_i и M_j проходят через алгоритм, производится уточнение вероятности появления ключей. Истинный ключ после большого числа испытаний становится высоко вероятным.

В 1993 году японский математик Мацуки опубликовал линейный метод криптоанализа. В нем ищутся произведения битов информации Mv на C_ξ биты криптограммы, которые равны произведению битов ключа $ПМ_v \cdot ПС_\xi = ПК_\varepsilon$. Решая это уравнение, осуществляем криптоанализ. Сложность линейного криптоанализа приблизительно равна сложности дифференциального криптоанализа.

Что касается криптоалгоритма Camellia, то для него вероятность нахождения s – блока $p_s = q_s = 2^{-6}$. Поскольку коэффициент линейной трансформации (P-функции) равен $B = 5$, то после 18 циклов Фейстеля без учета FL и FL^{-1} функций мы имеем следующие вероятности:

$$p_s^{2(2B+1)} = (2^{-6})^{22} = 2^{-132} \text{ и } q_s^{2(2B+1)} = (2^{-6})^{22} = 2^{-132}.$$

Обе вероятности p_s, q_s ниже порога безопасности для 128-битного блочного шифра: 2^{-128}. С учетом FL и FL^{-1} эти вероятности будут еще меньше.

7 Результаты анализа характеристик алгоритма Camellia

Экспериментально были исследованы следующие характеристики алгоритма Camellia – скорость зашифрования / расшифрования, сложность зашифрования / расшифрования и сложность разворачивания ключей.

В табл. 9 приведены результаты измерения скорости зашифрования.

Таблица 9

Процессор	Язык	Длина ключа	Зашифрование (Mbit/sec)
Pentium III (1)	Assembler	128	241,5
Pentium III (1)	Assembler	192	181
Pentium III (1)	Assembler	256	181
Pentium II (2)	ANSI C	128	66,6

(1) IBM совместимый компьютер с процессором Intel Pentium III (700 Mhz), 256Kb кэш L2, 128Mb оперативной памяти, операционная система FreeBSD 4.0R.
(2) IBM совместимый компьютер с процессором Intel Pentium II (300 Mhz), 512Kb кэш L2, 160Mb оперативной памяти, операционная система Windows 95.

В табл. 10 приведены результаты [6] измерения скоростей шифрования для ряда шифров.

Таблица 10

Алгоритм	Процессор	Скорость (cycles/byte)	
Camellia	Sparc	22,2	22,2
	Alpha	17,6	17,6
4-Way IDEA	PIII	13,75	
	PI/MMX	16,96	
Misty1	PII	26,6	26
	Alpha	25,4	25,8
RC6	P.ProII	16	
	Pentium	44	
	PIII	22,18	
Rijndael	PIII	14,13	14,93
	P.ProII	18	
	Pentium	20	
Mars	P.ProIII	20	
	Pentium	34	
Twofish	P.ProII	16	
	Pentium	19	

В табл. 11 приведены оценки сложности зашифрования / расшифрования / разворачивания ключей для различных шифров в числе циклов на байт. Эти результаты взяты из [6].

Таблица 11

название	$l_{k\ бит}$	Процессоры </операц. сист>		
		PIII/Linux	PIII/MS	Pentium4
Camellia	128	35/35/313	37/37/334	64/63/453
	192	45/45/420	49/48/434	86/86/546
	256	45/45/429	49/49/447	86/87/555
RC6 (20 rounds)	128	17/17/1054	24/25/1040	36/37/2056
	192	18/17/1175	30/25/1258	37/37/2141
	256	18/17/1168	25/25/1262	37/37/2153
Safer++	128	50/63/1333	46/60/1464	47/58/2918
	256	68/88/1805	63/84/1865	65/83/3998
Anubis	128	37/37/3550	37/37/3727	35/35/3750
	160	40/40/4519	39/39/4927	37/37/4845
	192	43/43/5857	42/42/6993	40/39/6054
	224	45/45/7356	44/44/8546	41/41/7338
	256	48/48/8876	47/47/10K	44/44/8902
	288	50/50/10K	49/49/12K	46/46/10K
	320	53/53/12K	51/51/13K	48/48/11K
Hiero-crypt-3	128	51/64/125K	50/56/13 IK	54/63/195K
	192	60/75/147K	59/67/1 54K	63/75/229K
	256	69/86/170K	67/76/1 78K	73/85/266K
Mars	128	31/30/2520	36/35/2354	82/72/3492
	192	31/30/2530	36/35/2337	82/72/3480
	256	31/30/2525	36/35/2336	82/73/3599
Rijndael	128	25/26/504	23/23/497	24/25/689
	192	30/31/601	27/27/552	28/29/820
	256	34/35/949	32/32/775	32/35/1327
Seed	128	45/45/409	51/50/421	63/63/401
Serpent	128	68/80/1577	59/57/1276	154/172/2235
	192	68/80/1578	59/57/1272	155/171/2232
	256	68/80/1566	59/57/1257	154/171/2231
Twofish	128	28/26/10K	28/29/15K	51/49/8871
	192	28/26/12K	30/30/17K	52/49/11K
	256	28/25/16K	28/29/20K	52/49/13K

Из табл. 11 следует, что по сложности алгоритм Camellia уступает RC6 и несколько уступает только Rijndael.

Заключение

Криптоалгоритм Camellia был разработан японскими Телеграфной и Телефонной Корпорацией Nippon и Электрической Корпорацией Mitsubishi [4]. В его разработке принимали участие такие известные специалисты в области практической криптографии как Kazumaro Aoki, Tetsuya Ichikawa, Masayuki Kanda, Mitsuru Matsui, Shiho Moriai, Junko Nakajima, Toshio Tokita.

Криптоалгоритм Camellia работает с блоками данных длиной 128 бит и длиной ключа 128, 192 и 256 бит, то есть он имеет такую же спецификацию интерфейса как и Advanced Encryption Standard (AES). Криптоалгоритм Camellia эффективно работает как в программной, так и в аппаратной реализации. По сравнению с финалистами AES, такими как MARS, RC6, Rijndael, Serpent и Twofish, Camellia показывает приемлемую скорость зашифрования / расшифрования. Оптимальная программная реализация криптоалгоритма Camellia, написанная на языке Assembler, позволяет достичь скорости зашифрования на Pentium III (1,13 GHz) 471 Мбит/сек. [4].

На сегодняшний день криптоалгоритм Camellia оказался устойчивым к следующим атакам:

- Дифференциальный и линейный криптоанализ.
- Усеченный дифференциальный криптоанализ.
- Усеченный линейный криптоанализ.
- Криптоанализ с невозможным дифференциалом.
- Атака типа бумеранг.
- Дифференциальная атака высших порядков.
- Квадратная атака.
- Интерполяционная и линейная суммарная атака.
- Атака на основе эквивалентных ключей.
- Скользящая атака.
- Атака на основе связанных ключей.
- Атаки, основанные на грубой силе.

Таким образом, международный проект NESSIE выполнен.

10 марта 2000 года NTT и Mitsubishi Electric Corporation представили совместно разработанный новый блочный симметричный шифр, названный Camellia. 17 апреля 2001 года NTT анонсировала, что NTT и Mitsubishi Electric Corporation намереваются разрешить бесплатное использование патентов Camellia.

24 сентября 2001 года алгоритм Camellia был представлен как один из криптоалгоритмов, участвующих во втором туре проекта NESSIE.

20 февраля 2003 года алгоритм Camellia был включен в список криптографических примитивов для использования в таких электронных правительственных системах Японии как: Министерство Иностранных Дел, Министерство Внутренних Дел, Министерство Экономики, Министерство Почты и Телекоммуникаций, Министерство Торговли и Индустрии. Криптостойкость алгоритма была оценена Комитетом Криптографических Оценок и Исследований (CRYPTREC) как высокая.

27 февраля 2003 года проект NESSIE [7] анонсировал финальный выбор криптографических алгоритмов, и Camellia была включена в список рекомендованных криптографических примитивов для использования в европейском сообществе.

Криптоалгоритм Camellia [8] эффективно реализуется как программно, так и аппаратно. В нем используются только 8-битовые таблицы подстановок (s-блоки) и логические операции, которые могут эффективно работать на различных аппаратных платформах. Поэтому криптоалгоритм способен работать на большой совокупности платформ, включая 8-, 32-, 64-битовые процессоры. Вместе с тем в криптоалгоритме Camellia не используется 32-битовая целочисленная операция сложения и умножения, которые широко используются некоторыми программно-ориентированными 128-битными блочными шифрами.

Схема разворачивания ключей очень проста и базируется на схеме зашифрования. Подключи, которые генерируются из основного ключа, могут быть вычислены в любом порядке. Для генерации подключей используется 32 байта для 128-битного ключа и 64 байта для ключей длиной 192/256 бит.

Список литературы: 1. *News* Releases № 2300 http://www.Global.MitsubishiElectric.com. 2 *Proposal* of addition of new cipher suites to TLS to support Camellia, EPOC, and PSEC 8/14/00 http://www.ietf.org 3. http: // www. cryptonessie.org 4. *K. Aoki,* T. Ichikawa, M. Kanda, M. Matsui, S. Moriai, J. Nakajima, and T. Tokita, \ Camellia: A 128-bit block cipher suitable for multiple platforms. See also http://info.isl.ntt.co.jp/camellia/. 5. *Specification* of Camellia – a 128-bit Block Cipher. 6. *Performance* of Optimized Implementations of the NESSIE Primitives B. Preneel, B. Van Rompay, S.B. Ors, A. Biryukov, L. Granboulan, E. Dottax. 7 *NESSIE* PROJECT ANNOUNCES FINAL SELECTION OF CRYPTO ALGORITHMS. 8 *Specification* of Camellia – 128-bit Block Cipher. Difference between specifications.

Харьковский национальный

университет радиоэлектроники

Поступила в редколлегию 25.06.2003