SEED-256 알고리즘 사양 및 세부 명세

2009.7



1. 개요

국내 블록 암호알고리즘 SEED-128은 민간 부분인 인터넷, 전자상거래, 무선 통신 등의 환경에서 민감한 정보의 보호와 개인 프라이버시 등을 보호하기 위하여 1999년 2월 한국정보보호센터(현 한국정보보호진흥원)에서 개발된 블록 암호알고리즘이다.

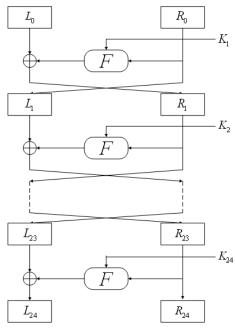
AES, Camellia 등 국외 블록 암호알고리즘은 128/192/256비트의 키를 지원함으로써 암호 알고리즘 적용 환경에 따라 보안강도를 다양하게 제공할 수 있다. 하지만 국내 블록 암호알고리즘 SEED-128은 128 비트 키만을 지원하므로 활용 분야가제한적이었다. 이에, 국내 암호알고리즘의 활용성 강화를 위하여 2009년 256비트의키를 지원하는 블록 암호알고리즘 SEED를 개발하였다.

2. 기호화 표기

- o a⊕b: 비트별 배타적 논리합
- o $a \boxplus b$: $(a+b) \mod 2^{32}$
- o a&b: a와 b의 비트별 논리곱
- o $X^{\ll s}$: 워드 X를 s비트만큼 왼쪽으로 순환 이동하는 연산
- o $X^{\gg s}$: 워드 X를 s비트만큼 오른쪽으로 순환 이동하는 연산
- o Li: i번째 라운드에서 출력된 왼쪽 메시지 블록(64비트)
- o R_i: i번째 라운드에서 출력된 오른쪽 메시지 블록(64비트)
- $o K_i = (K_{i,0}, K_{i,1})$: i번째 라운드에서 사용되는 64비트 라운드 키
- o $K_{i,0}$: i번째 라운드에서 F함수의 오른쪽 입력과 연산되는 라운드 키의 오른쪽 32비트
- o $K_{i,1}$: i번째 라운드에서 F함수의 왼쪽 입력과 연산되는 라운드 키의 왼쪽 32비트
- o $X = (X_3 || X_2 || X_1 || X_0)$: G 함수의 32비트 입력값
- o $Y = (Y_3 || Y_2 || Y_1 || Y_0)$: G 함수에서 S-box (S_1, S_2) 의 32비트 출력값
- o $Z = (Z_3 ||Z_1||Z_0)$: G 함수의 32비트 출력값
- o m_i : 상수
- o KC: 라운드 키 생성과정에서 사용되는 i+1번째 라운드 상수

3. 알고리즘 SEED-256의 구조

SEED-256의 전체 구조는 Feistel 구조로 이루어져 있으며, 128비트의 평문 블록과 256비트 키를 입력으로 사용하여, 24라운드를 거쳐 128비트 암호문 블록을 출력한다. (그림 1)은 SEED-256의 전체구조를 도식화한 것이다.



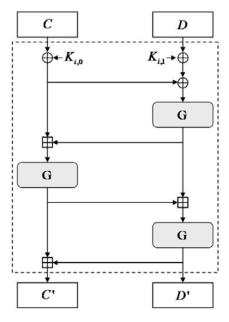
(그림 1) SEED-256 전체 구조도

3.1. F 함수

Feistel 구조의 블록암호 알고리즘은 F 함수라 불리는 내부 함수를 어떻게 구성하는가에 의해 구분된다. SEED-256의 F 함수는 그 자체로 변형된 Feistel 구조를 띠고 있다. F 함수는 2 개의 32비트 블록(C, D)으로 나뉘고, 라운드 키 역시 2 개의 32비트 블록 $(K_{i,0}, K_{i,1})$ 으로 나누어 연산되며 최종적으로 2 개의 32비트 출력 블록(C', D')을 출력한다. 이를 수식으로 표현하면 다음과 같으며(i)는 라운드 수), F 함수를 도식화 한 것은 (C', C') 같다.

$$\begin{split} C' &= G(G((C \oplus K_{i,0}) \oplus (D \oplus K_{i,1})) \boxplus (C \oplus K_{i,0})) \\ &\boxplus G(G(G((C \oplus K_{i,0}) \oplus (D \oplus K_{i,1})) \boxplus (C \oplus K_{i,0})) \boxplus G((C \oplus K_{i,0}) \oplus (D \oplus K_{i,1}))) \end{split}$$

 $D' = G(G(G((C \oplus K_{i,0}) \oplus (D \oplus K_{i,1})) \boxplus (C \oplus K_{i,0})) \boxplus G((C \oplus K_{i,0}) \oplus (D \oplus K_{i,1})))$



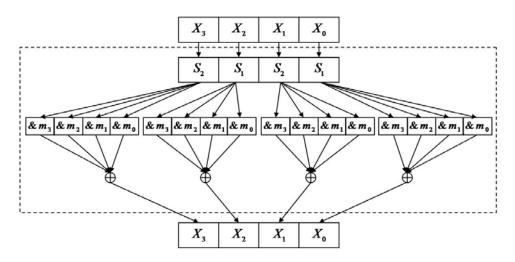
(그림 2) SEED-256 F 함수

3.2. G 함수

전체 G 함수는 다음과 같이 기술된다. (그림 3)과 같이 32비트 X를 입력 받아서 32비트 출력값 $Z(=Z_3||Z_1||Z_0)$ 를 생성한다. G 함수를 수식으로 표현하면 다음과 같다.

$$\begin{split} Y_3 &= S_2(X_3), \quad Y_2 = S_1(X_2), \quad Y_1 = S_2(X_1), \quad Y_0 = S_1(X_0) \\ Z_3 &= (Y_0 \wedge m_3) \oplus (Y_1 \wedge m_0) \oplus (Y_2 \wedge m_1) \oplus (Y_3 \wedge m_2), \\ Z_2 &= (Y_0 \wedge m_2) \oplus (Y_1 \wedge m_3) \oplus (Y_2 \wedge m_0) \oplus (Y_3 \wedge m_1), \\ Z_1 &= (Y_0 \wedge m_1) \oplus (Y_1 \wedge m_2) \oplus (Y_2 \wedge m_3) \oplus (Y_3 \wedge m_0), \\ Z_0 &= (Y_0 \wedge m_0) \oplus (Y_1 \wedge m_1) \oplus (Y_2 \wedge m_2) \oplus (Y_3 \wedge m_3). \end{split}$$

$$(m_0 = 0xfc, \quad m_1 = 0xf3, \quad m_2 = 0xcf, \quad m_3 = 0x3f)$$



(그림 3) SEED-256 G 함수

참고> 위의 G 함수는 구현의 효율성을 위해 4개의 확장된 4-바이트 SS-box의 XOR로 구현할 수 있다. 이를 위해 다음과 같은 4개의 SS-box 들을 저장해야 한다.

이 확장 SS-box들을 이용하면 G 함수는 다음과 같이 구현될 수 있다.

$$Z = SS_3(X_3) \oplus SS_2(X_2) \oplus SS_1(X_1) \oplus SS_0(X_0).$$

3.3. S-Box

G 함수의 내부에 사용되는 비선형 S-box S_1 , S_2 는 $S_i:GF(2^8) \to GF(2^8)$ 인 지수함수와 아핀변환(Affine transform)으로 정의되며, 다음과 같다.

$$S_i(x) = A^{(i)} \cdot x^{n_i} \oplus b_i, i = 1, 2.$$

여기서, $n_1=247,\ n_2=251,\ b_1=159,\ b_2=56$ 이다. 두 개의 지수 $n_1,\ n_2$ 는 각각 지

수함수 x^n 에서, 여기서 $0 \le n \le 255$ 인 정수, DC 특성 및 LC 특성, SAC 특성 등이 가장 우수한 두개의 지수를 선택한 것이고, 이때 사용된 8차의 기약다항식은 $x^8 + x^6 + x^5 + x + 1$ 이 사용되었다. 또한 고정점을 제거하기 위해 다음의 선형변환과 상수 덧셈으로 구성된 아핀변환이 추가 되었다.

$$A^{(1)} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}, \quad A^{(2)} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$b_1 = (1,0,1,0,1,0,0,1)^T = 169, \ b_2 = (0,0,1,1,1,0,0,0)^T = 56$$

i	$S_1(i)$	i	$S_1(i)$	i	$S_1(i)$	i	$S_{\!1}(i)$	i	$S_{\!1}(i)$	i	$S_{\!1}(i)$
0	169	1	133	2	214	3	211	4	84	5	29
6	172	7	37	8	93	9	67	10	24	11	30
12	81	13	252	14	202	15	99	16	40	17	68
18	32	19	157	20	224	21	226	22	200	23	23
24	165	25	143	26	3	27	123	28	187	29	19
30	210	31	238	32	112	33	140	34	63	35	168
36	50	37	221	38	246	39	116	40	236	41	149
42	11	43	87	44	92	45	91	46	189	47	1
48	36	49	28	50	115	51	152	52	16	53	204
54	242	55	217	56	44	57	231	58	114	59	131
60	155	61	209	62	134	63	201	64	96	65	80
66	163	67	235	68	13	69	182	70	158	71	79
72	183	73	90	74	198	75	120	76	166	77	18
78	175	79	213	80	97	81	195	82	180	83	65
84	82	85	125	86	141	87	8	88	31	89	153
90	0	91	25	92	4	93	83	94	247	95	225
96	253	97	118	98	47	99	39	100	176	101	139
102	14	103	171	104	162	105	110	106	147	107	77
108	105	109	124	110	9	111	10	112	191	113	239
114	243	115	197	116	135	117	20	118	254	119	100
120	222	121	46	122	75	123	26	124	6	125	33
126	107	127	102	128	2	129	245	130	146	131	138
132	12	133	179	134	126	135	208	136	122	137	71
138	150	139	229	140	38	141	128	142	173	143	223
144	161	145	48	146	55	147	174	148	54	149	21
150	34	151	56	152	244	153	167	154	69	155	76
156	129	157	233	158	132	159	151	160	53	161	203
162	206	163	60	164	113	165	17	166	199	167	137
168	117	169	251	170	218	171	248	172	148	173	89
174	130	175	196	176	255	177	73	178	57	179	103
180	192	181	207	182	215	183	184	184	15	185	142
186	66	187	35	188	145	189	108	190	219	191	164
192	52	193	241	194	72	195	194	196	111	197	61
198	45	199	64	200	190	201	62	202	188	203	193
204	170	205	186	206	78	207	85	208	59	209	220
210	104	211	127	212	156	213	216	214	74	215	86
216	119	217	160	218	237	219	70	220	181	221	43
222	101	223	250	224	227	225	185	226	177	227	159
228	94	229	249	230	230	231	178	232	49	233	234
234	109	235	95	236	228	237	240	238	205	239	136
240	22	241	58	242	88	243	212	244	98	245	41
246	7	247	51	248	232	249	27	250	5	251	121
252	144	253	106	254	42	255	154				

(班 1) S₁-box

i	$S_2(i)$	i	$S_{2}(i)$	i	$S_{2}(i)$	i	$S_{\!2}(i)$	i	$S_2(i)$	i	$S_{\!2}(i)$
0	56	1	232	2	45	3	166	4	207	5	222
6	179	7	184	8	175	9	96	10	85	11	199
12	68	13	111	14	107	15	91	16	195	17	98
18	51	19	181	20	41	21	160	22	226	23	167
24	211	25	145	26	17	27	6	28	28	29	188
30	54	31	75	32	239	33	136	34	108	35	168
36	23	37	196	38	22	39	244	40	194	41	69
42	225	43	214	44	63	45	61	46	142	47	152
48	40	49	78	50	246	51	62	52	165	53	249
54	13	55	223	56	216	57	43	58	102	59	122
60	39	61	47	62	241	63	114	64	66	65	212
66	65	67	192	68	115	69	103	70	172	71	139
72	247	73	173	74	128	75	31	76	202	77	44
78	170	79	52	80	210	81	11	82	238	83	233
84	93	85	148	86	24	87	248	88	87	89	174
90	8	91	197	92	19	93	205	94	134	95	185
96	225	97	125	98	193	99	49	100	245	101	138
102	106	103	177	104	209	105	32	106	215	107	2
108	34	109	4	110	104	111	113	112	7	113	219
114	157	115	153	116	97	117	190	118	230	119	89
120	221	121	81	122	144	123	220	124	154	125	163
126	171	127	208	128	129	129	15	130	71	131	26
132	227	133	236	134	141	135	191	136	150	137	123
138	92	139	162	140	161	141	99	142	35	143	77
144	200	145	158	146	156	147	58	148	12	149	46
150	186	151	110	152	159	153	90	154	242	155	146
156	243	157	73	158	120	159	204	160	21	161	251
162	112	163	117	164	127	165	53	166	16	167	3
168	100	169	109	170	198	171	116	172	213	173	180
174	234	175	9	176	118	177	25	178	254	179	64
180	18	181	224	182	189	183	5	184	250	185	1
186	240	187	42	188	94	189	169	190	86	191	67
192	133	193	20	194	137	195	155	196	176	197	229
198	72	199	121	200	151	201	252	202	30	203	130
204	33	205	140	206	27	207	95	208	119	209	84
210	178	211	29	212	37	213	79	214	0	215	70
216	237	217	88	218	82	219	235	220	126	221	218
222	201	223	253	224	48	225	149	226	101	227	60
228	182	229	228	230	187	231	124	232	14	233	80
234	57	235	38	236	50	237	132	238	105	239	147
240	55	241	231	242	36	243	164	244	203	245	83
246	10	247	135	248	217	249	76	250	131	251	143
252	206	253	59	254	74	255	183				

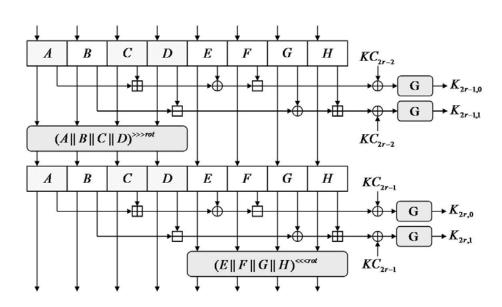
(班 2) S₂-box

3.4. 라운드 키 생성과정

SEED-256은 SEED-128의 라운드 키 생성과정과 유사한 논리로 설계되었으며, SEED-256은 8개의 레지스터를 사용한다.

알고리즘 SEED-256은 256비트 비밀키를 입력 받아 (그림 4)와 같은 과정을 수행하여 24 라운드에 대한 각각의 라운드 키를 생성한다. 8개의 레지스터를 사용하고, 128비트 단위 순환 이동 연산을 사용한다. (그림 4)에서 순환이동 수 rot는 on-the-fly를 만족하도록 9, 11, 12를 번갈아 사용한다. 그리고 라운드 상수 $KC_r(r=0,\cdots,23)$ 은 SEED-128과 유사한 방식으로 다음과 같이 생성한다.

$$KC_r = (KC_0)^{\ll r} \ (r = 0, \dots, 23)$$



(그림 4) SEED-256의 라운드 키 생성 과정