轻量级分组密码 PRESENT

1 PRESENT

PRESENT 是一个轻量级分组密码算法，由 Bogdanov、Knudsen 等人在 2007 年设计。

PRESENT 采用 SPN 结构设计，分组长度为 64 比特，密钥长度包括 80 比特和 128 比特两种，一共 31 轮。作者认为在 RFID 标签、传感器等物联网低功耗节点中，80 比特的密钥能够提供足够的安全性，因此推荐使用 80 比特的密钥，128 比特的密钥可以用在安全性要求更高的场景。

1.1 加密算法

PRESENT 加密算法的结构如图 1 所示。算法先将轮函数迭代 31 次，轮函数包括轮密钥加（addRoundKey）、非线性置换 S 层（sBoxLayer）、线性置换 P 层（pLayer），再经过一步轮密钥加后得到密文。

设 64 比特的明文

$$P = p_{63} \cdots p_2 p_1 p_0$$

其中 p_0 表示最低比特位， p_{63} 表示最高比特位。轮函数各步操作如下：

① 轮密钥加完成 64 比特中间状态与 64 比特轮子密钥异或；

② 设 S 层输入 $W = w_{63} \cdots w_2 w_1 w_0$，输出 $V = v_{63} \cdots v_2 v_1 v_0$，则：

$$\begin{aligned} v_3 v_2 v_1 v_0 &= S(w_3 w_2 w_1 w_0) \\ v_7 v_6 v_5 v_4 &= S(w_7 w_6 w_5 w_4) \\ &\cdots\cdots \\ v_{63} v_{62} v_{61} v_{60} &= S(v_{63} v_{62} v_{61} v_{60}) \end{aligned}$$

S 盒的取值如表 1；

③ P 层将输入的每一比特映射到输出的另一比特，取值如表 2 所示；

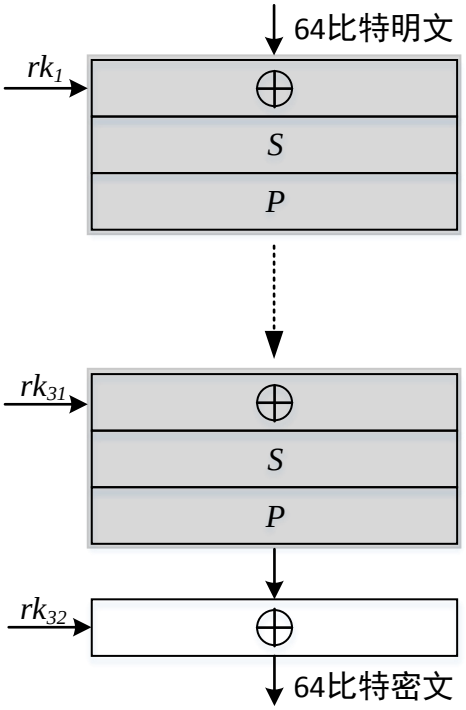

图1 PRESENT 加密结构

表1 PRESENT S 盒真值表

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S(x)$	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

表2 P 层置换表

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$P(i)$	0	16	32	48	1	17	33	49	2	18	34	50	3	19	35	51
i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$P(i)$	4	20	36	52	5	21	37	53	6	22	38	54	7	23	39	55
i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
$P(i)$	8	24	40	56	9	25	41	57	10	26	42	58	11	27	43	59
i	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$P(i)$	12	28	44	60	13	29	45	61	14	30	46	62	15	31	47	63

P 层表示为公式如 1)所示。

$$P(i)=\begin{cases}(16\times i)\%63 & ,i\neq 63\\ 63 & ,i=63\end{cases} \quad 1)$$

1.2 解密算法

解密先将解密轮函数迭代 31 次，最后经过一次轮密钥加得到明文。与加密不同的是：

① 解密轮函数依次是：轮密钥加、线性置换逆 P 层（P^{-1}）、非线性置换逆 S 层；

② 轮密钥加实现中间状态与轮子密钥异或，轮子密钥的使用顺序与加密相反，依次是 rk_{32}、rk_{31}、……、rk_2，最后一步轮密钥加使用 rk_1；

③ P^{-1} 层表示为公式如 2)；

$$P^{-1}(i)=\begin{cases}(4\times i)\%63 & ,i\neq 63\\ 63 & ,i=63\end{cases} \quad 2)$$

④ 逆 S 层依次将连续 4 比特经过逆 S 盒（S^{-1}），S^{-1} 的取值如表 3。

表3 PRESENT S^{-1} 盒真值表

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S^{-1}(x)$	5	E	2	8	C	1	2	D	B	4	6	3	0	7	9	A

1.3 密钥扩展算法

2 软件实现

2.1 基本实现

2.2 一个分组并行实现

2.3 64 分组并行

代码

[1] 基本实现：https://github.com/michaelkitson/Present-8bit
[2] 分组内部并行：https://github.com/bozhu/PRESENT-C
[3] 分组之间并行：https://github.com/pfasante/present
[4] https://github.com/pfasante/present 给出了几种实现方式