Attacking the IV Setup of Stream Cipher LEX

Hongjun Wu and Bart Preneel

Katholieke Universiteit Leuven, Dept. ESAT/COSIC
{wu.hongjun,bart.preneel}@esat.kuleuven.be

Abstract. In this paper, we point out that the initialization of stream cipher LEX is weak. If a key is used with about 2^{61} random IVs, and 20,000 keystream bytes are generated from each IV, then the key could be recovered easily.

1 Introduction

Stream cipher LEX [1] is based on block cipher AES [3]. The keystream bits are generated by extracting 32 bits from each round of AES in the 128-bit Output Feedback (OFB) mode [2]. The LEX is about 2.5 times faster than AES.

Extracting the middle value from a block cipher needs to be done very carefully. If the input/output and the middle value of a block cipher are known, then the complexity of the attack may be reduced dramatically. In LEX, we found that the input of AES could be revealed if a key is used with about 2^{61} IVs.

This paper is organized as follows. The stream cipher LEX is introduced in Section 2. The attack is provided in Section 3. Section 4 concludes this paper.

2 Description of Stream Cipher LEX

The Fig. 1 shows how the AES is initialized and chained. First a standard AES key-schedule for a secret 128-bit key K is performed. Then a given 128-bit IV is encrypted by a single AES invocation: $S = AES_K(IV)$. The S and the subkeys are the output of the initialization process.

S is encrypted by K in the 128-bit OFB mode (for more secure variant, K is changed every 500 AES encryptions). At each round, 32 bits of the middle value of AES are extracted to form the keystream. The bytes $b_{0,0}$, $b_{0,2}$, $b_{2,0}$, $b_{2,2}$ at every odd round and the bytes $b_{0,1}$, $b_{0,3}$, $b_{2,1}$, $b_{2,3}$ at every even round are selected, as shown in Fig. 2.

In [1], Biyukov mentioned using ten (maybe eleven, since there are eleven subkeys in AES) consecutive encryptions of the IV as subkeys. That initialization could resist the attack given in this report.

3 The Weakness in the Initialization Process

In Subsection 3.1, we will illustrate how to recover the AES key if the input and partial information from the first round output are known. In Subsection 3.2, we show how to recover the key of LEX.

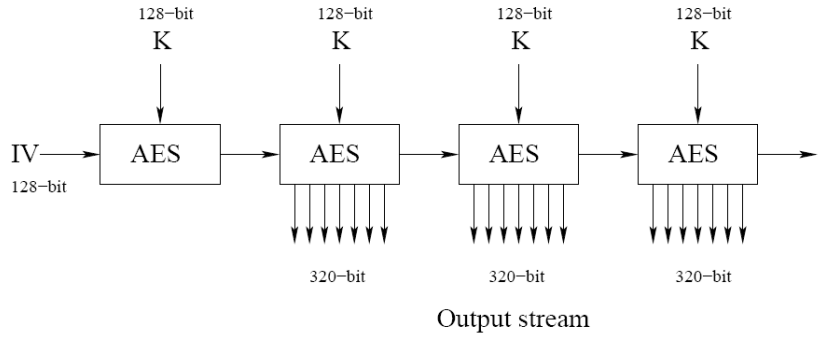

Fig. 1. Initialization and stream generation [1]

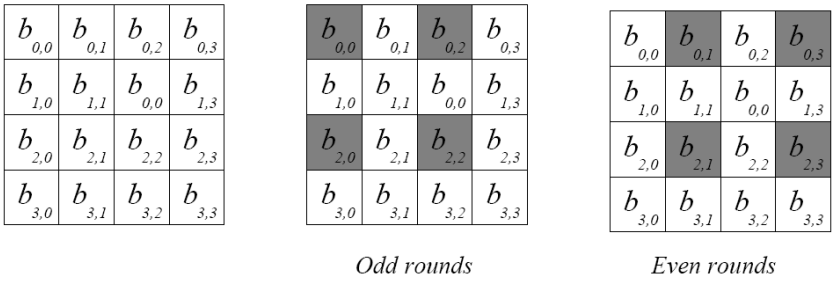

Fig. 2. The positions of leak in the even and odd rounds [1]

3.1 Recovering the AES key with partial information of the first round output

Denote the 16-byte output of the r-th round of AES as $m^r_{i,j}$ $(0 \le i,j \le 3)$. And denote the 16-byte round key at the end of the r-th round as $w^r_{i,j}$ $(0 \le i,j \le 3)$. Now if $m^1_{0,0}$, $m^1_{0,2}$, $m^1_{2,0}$, $m^1_{2,2}$ are known, i.e, four bytes of the first round output are known, then we obtain the following four equations:

$$\begin{aligned} m^1_{0,0} \oplus w^1_{0,0} = \text{MixColumn}((m^0_{0,0} \oplus w^0_{0,0})||(m^0_{1,3} \oplus w^0_{1,3}) \\ ||(m^0_{2,2} \oplus w^0_{2,2})||(m^0_{3,1} \oplus w^0_{3,1}))\&0xFF \quad (1) \\ m^1_{2,0} \oplus w^1_{2,0} = (\text{MixColumn}((m^0_{0,0} \oplus w^0_{0,0})||(m^0_{1,3} \oplus w^0_{1,3}) \\ ||(m^0_{2,2} \oplus w^0_{2,2})||(m^0_{3,1} \oplus w^0_{3,1})) >> 16)\&0xFF \quad (2) \\ m^1_{0,2} \oplus w^1_{0,2} = \text{MixColumn}((m^0_{0,2} \oplus w^0_{0,2})||(m^0_{1,1} \oplus w^0_{1,1}) \\ ||(m^0_{2,0} \oplus w^0_{2,0})||(m^0_{3,3} \oplus w^0_{3,3}))\&0xFF \quad (3) \\ m^1_{2,2} \oplus w^1_{2,2} = (\text{MixColumn}((m^0_{0,2} \oplus w^0_{0,2})||(m^0_{1,1} \oplus w^0_{1,1}) \\ ||(m^0_{2,0} \oplus w^0_{2,0})||(m^0_{3,3} \oplus w^0_{3,3})) >> 16)\&0xFF \quad (4) \end{aligned}$$

Each equation leaks one-byte information of the secret key. In the above four equations, 12 bytes of the subkey are involved. To recover all those 12 bytes, we need three inputs to AES and the related 32-bit first round outputs so that we could obtain 12 equations. Those 12 equations can be solved with about $\alpha \times 2^{32}$ operations, where α is a small constant. With 96 bits of the key being recovered, the rest of the 32 bits of AES could be recovered by exhaustive search.

3.2 Recovering the key of LEX

Denote $S_i = E_K^i(IV)$, where $E^i(m)$ means that m is encrypted by i times, $S_0 = IV$. And denote the 320 bits extracted from the i-th encryption as k_i for $i \geq 2$. For two IVs, IV' and IV'', if $k_2' = k_j''$ $(j > 2)$, then we know that $S_1' = S_{j-1}''$. Immediately, we know that $S_{j-2}'' = S_0' = IV'$. Note that k_{j-1}'' are extracted from $E_K(S_{j-2}'')$, so k_{j-1}'' are extracted from $E_K(IV')$, it means that we know the input to AES, and we know 32 bits from the output of the first round. From the attack given in Subsection 3.1, we can recover the key of LEX if three are three such collisions.

We now compute the number of IVs required to generate three collisions. Suppose that a secret key is used with about $2^{65.3}$ random IVs, and each IV^i is used to generate a 640-bit keystream k_2^i, k_3^i. Since the block size of AES is 128 bits, we know that with high chance there are three collisions $k_2^i = k_3^j$ for different i and j.

The number of IVs could be reduced if more keystream bits are generated from each IV. However in [1], it is suggested to change the key every 500 AES encryptions for strong variant of LEX. Suppose that each IV is applied to generate 500 320-bit outputs, then with $2^{60.8}$ IVs, we could find three collisions $k_2^i = k_x^j$ $(2 < x < 500)$ and recover the key of LEX.

4 Conclusion

In this paper, we show that LEX is insecure if a key is used with about 2^{61} random IVs. Due to the large amount of IVs required in the attack, it is not easy to exploit this weakness in practice. However, the attack shows that the initialization of LEX is imperfect and improvement is needed.

References

1. A. Biryukov, "A New 128-bit Key Stream Cipher LEX", *ECRYPT Stream Cipher Project Report 2005/013*. Available at http://www.ecrypt.eu.org/stream/
2. National Institute of Standards and Technology, "DES Modes of Operation", Federal Information Processing Standards Publication (FIPS) 81. Available at http://csrc.nist.gov/publications/fips/
3. National Institute of Standards and Technology, "ADVANCED ENCRYPTION STANDARD (AES) ", Federal Information Processing Standards Publication (FIPS) 197. Available at http://csrc.nist.gov/publications/fips/