Exploring the Stream Cipher: A Cryptographic Study of the E_0 Cipher

H3Art

International School

Jinan University

Guangzhou, China

Abstract

This report examines the E_0 stream cipher algorithm which provides encryption for Bluetooth wireless communications. E_0 generates a pseudorandom keystream through the combination of four linear feedback shift registers (LFSRs) and a non-linear blending element, enabling efficient real-time encryption of data. An overview of stream ciphers and their working mechanism is provided first. The specific design and operation of the E_0 cipher is then described in detail, including the configuration of its four LFSRs, the keystream generation process, and its role in the Bluetooth standard. A key focus is the analysis of potential cryptanalytic attacks against E_0. A basic approach to recover the initial LFSR states given a limited keystream by maintaining a system of linear equations on the LFSR output bits is presented. It suggest this attack achieves a complexity of $O(2^{83})$, indicating the true security level of E_0 may be significantly lower than specified parameters.

Keywords: Cryptology; Stream Cipher; Cryptanalysis; Bluetooth Security

1 Introduction

Among the myriad cryptographic techniques deployed to safeguard data, stream ciphers hold a pivotal role due to their efficiency and effectiveness in real-time encryption tasks. Unlike block ciphers, which encrypt data in fixed-size blocks, stream ciphers encrypt plaintext one byte or bit at a time, making them ideally suited for environments where speed and low latency are paramount. This report focuses on one such stream cipher: the E_0 encryption algorithm, which is integral to the Bluetooth standard—a ubiquitous technology enabling wireless communication between devices.

The E_0 cipher, while less well-known outside of cryptographic circles, is fundamental to the security framework of Bluetooth, providing confidentiality for billions of devices worldwide. Given its widespread application, the examination of E_0's design, functionality, and, crucially, its vulnerabilities, is of paramount importance. This report aims not only to dissect the E_0 cipher's inner workings but also to scrutinize its resilience against various cryptanalytic attacks. Through this analysis, I seek to contribute to the ongoing dialogue regarding the cipher's efficacy and explore potential avenues for bolstering its security.

2 Overview of Stream Ciphers

Stream ciphers are a fundamental class of encryption algorithms designed for the secure transmission of data. They operate by generating a seemingly random stream of bits, known

as the keystream, which is then combined with the plaintext to produce ciphertext. The primary characteristic that distinguishes stream ciphers from other cryptographic algorithms is their method of operation—encrypting one bit or byte of plaintext at a time. This feature makes stream ciphers particularly suited for applications requiring high speed and low latency, such as real-time communication and small, resource-constrained devices.

The core principle behind stream ciphers is the generation of a keystream that is as random as possible. The security of a stream cipher depends significantly on the unpredictability of this keystream; any pattern or weakness that allows an attacker to predict future bits in the stream can compromise the encryption. To encrypt data, the keystream is combined with the plaintext, typically using a simple operation like bitwise XOR. The same keystream is then used to decrypt the ciphertext, restoring the original plaintext.

Mathematically, the encryption process can be described as follows:

Algorithm 1 Stream Cipher Encryption

```
procedure ENCRYPT(plaintext, key)
    keystream ← GENERATEKEYSTREAM(key), LENGTH(plaintext)
    ciphertext ← empty string
    for i ← 1 to LENGTH(plaintext) do
        ciphertext[i] ← plaintext[i] ⊕ keystream[i]
    end for
    return ciphertext
end procedure
```

Where $ciphertext[i]$ is the i-th bit of the ciphertext, $plaintext[i]$ is the i-th bit of the plaintext, and $keystream[i]$ is the i-th bit of the keystream. The decryption process is identical due to the properties of XOR:

Algorithm 2 Stream Cipher Decryption

```
procedure DECRYPT(ciphertext, key)
    keystream ← GENERATEKEYSTREAM(key), LENGTH(ciphertext)
    plaintext ← empty string
    for i ← 1 to LENGTH(ciphertext) do
        plaintext[i] ← ciphertext[i] ⊕ keystream[i]
    end for
    return plaintext
end procedure
```

Stream ciphers play a crucial role in the realm of cryptographic protocols, offering a balance between security and performance that is essential for many modern applications. The following sections will delve deeper into the E_0 stream cipher, exploring its specific mechanisms and evaluating its security in the context of Bluetooth communications.

3 Overview of the E_0 Cipher

The E_0 encryption algorithm constitutes an integral element of the Bluetooth protocol, serving as the bulwark for secure wireless device communication. This stream cipher is adept at safeguarding data confidentiality through encryption of the data traversing Bluetooth connections. The cipher operates by generating a pseudorandom keystream that, when XORed with the plaintext, yields the ciphertext. Its architecture is a synthesis of LFSRs, arrayed to fabricate a sophisticated keystream intended to counteract cryptanalytic assaults.

A comprehensive elucidation of the E_0 encryption paradigm is delineated as follows:

- Two Bluetooth devices will secure their communication channel by employing a key exchange protocol. Participants in the dialogue establish a consensual secret known as the $Link\_Key$, which undergirds mutual authentication. This key is subsequently instrumental in generating the cipher key K_C.
- The authentication mechanism contemplates a 96-bit numeral, designated as the Ciphering Offset Number (COF).
- A 128-bit random number, recognized as the public variable (EN_RAND), is conveyed in plaintext.

Within the E_0 framework, the cipher key K_C undergoes a transformation to K_C^1. Thereafter, K_C^1 is employed linearly alongside universally known parameters: the 48-bit Bluetooth address and a 26-bit counter that remains unique for each packet.

The E_0 encryption algorithm engenders a binary keystream termed K_{cipher}. Subsequently, K_{cipher} undergoes a bitwise XOR operation with the plaintext to fabricate the ciphertext. The decryption process mirrors the encryption, utilizing the identical keystream. The E_0 stream cipher encompasses the following constituents:

- A suite of four LFSRs, aggregating to a span of 128 bits, detailed as:
  - LFSR1: A 25-bit register.
  - LFSR2: A 31-bit register.
  - LFSR3: A 33-bit register.
  - LFSR4: A 39-bit register.
- Four nonlinear storage bits, collectively referred to as the Blender.
- The binary operations of addition and exclusive disjunction (XOR).

In E_0, with each iteration, the LFSRs are clocked, eliciting a new state derived by the blender, which operates as a nonlinear function predicated on its current state. Utilizing the quartet of output bits from the LFSRs, a single keystream bit is computed via the XOR operation involving the four LFSR outputs and the output bit from the blender. Figure.1 depicts the entire E_0 encryption process.

The E_0 Encryption system is architectured with four LFSRs as input, and the outputs are conjoined with 16 states through a simple Finite State Machine (FSM), termed the summation combiner. The algorithm prominently utilizes:

- K_C
- Bluetooth address
- CLK_{26-1} master clock bits
- RAND

The specifications of the LFSRs are chronicled in Table 1. Here, the registers vary in length, but cumulatively they constitute 128 delay elements. The feedback polynomials, as tabulated, are primitive, ensuring the LFSRs' capability to generate maximal length sequences. Each primitive polynomial exhibits a hamming weight of five.

Denote the output of $LFSR_i$ at clock-time t as x_t^i. The value y_t is derived from the tuple $x_t^1, ..., x_t^4$ through the subsequent equation:

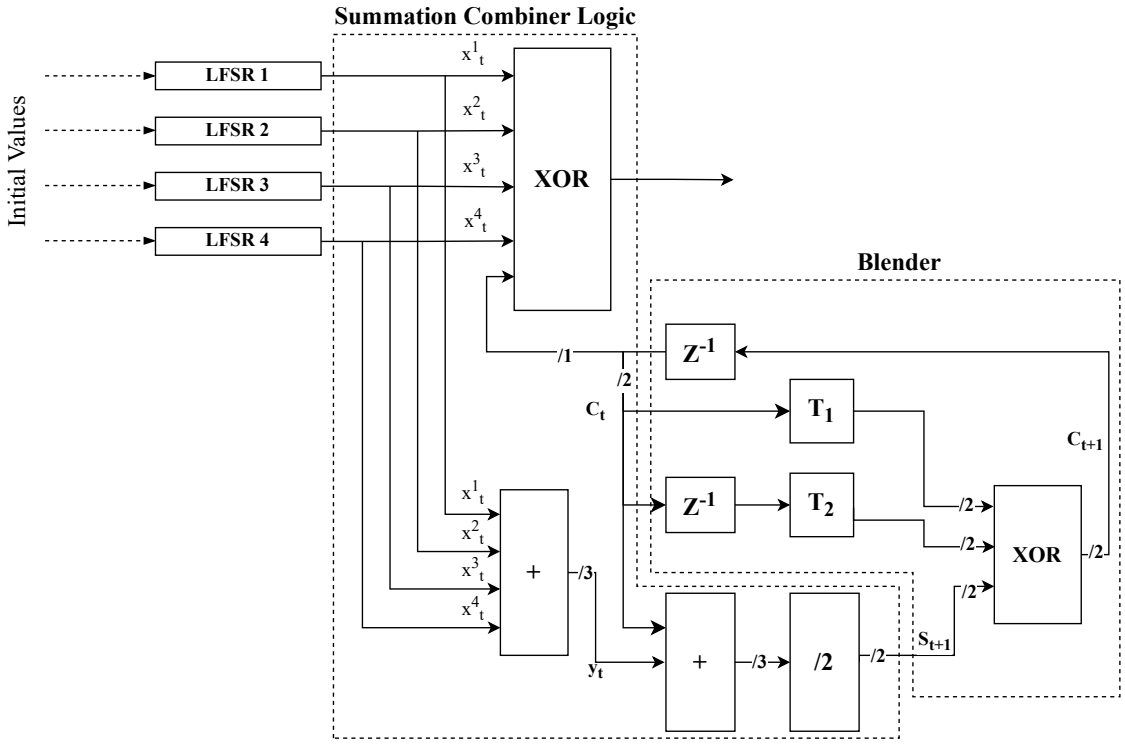

Figure 1: E_0 Encryption Procedure

Table 1: The four primitive feedback polynomials

LFSR	Length	Feedback polynomial
1	25	$1 + t^8 + t^{12} + t^{20} + t^{25}$
2	31	$1 + t^{12} + t^{16} + t^{24} + t^{31}$
3	33	$1 + t^4 + t^{24} + t^{28} + t^{33}$
4	39	$1 + t^4 + t^{28} + t^{36} + t^{39}$

$$y_t = \sum_{i=1}^{n} x_t^i$$

The sum here is calculated over integers, meaning that the value of y_t will be within the set $\{0, 1, 2, 3, 4\}$. The summation generator's output is determined by the ensuing equations:

Define two internal states with 2-bit values c_{t-1} and c_t. The new content c_{t+1} is calculated as:

$$c_{t+1} = (c_{t+1}^1, c_{t+1}^0) = s_{t+1} \oplus T_1[c_t] \oplus T_2[c_{t-1}]$$

wherein:

$$y_t = (y_t^2, y_t^1, y_t^0) = x_t^1 + x_t^2 + x_t^3 + x_t^4 \in \{0, 1, 2, 3\}$$

$$s_{t+1} = (s_{t+1}^1, s_{t+1}^0) = \left[\frac{y_t + c_t}{2} \right] \in \{0, 1, 2, 3\}$$

In the above equation, $T_1[\,\cdot\,]$ and $T_2[\,\cdot\,]$ are distinct linear bijections over the $GF(4)$. Assuming $GF(4)$ is constituted by the irreducible polynomial $x^2 + x + 1$, and α denotes a zero of this polynomial within $GF(4)$. The mappings T_1 and T_2 are defined thusly:

$$T_1 : GF(4) \to GF(4), x \mapsto x$$

$$T_2 : GF(4) \to GF(4), x \mapsto (\alpha + 1)x$$

Ultimately, the XOR operation of the four LFSR sequences x_t^i in conjunction with the FSM output bit c_t^0 yields the keystream bits z_t:

$$z_t = x_t^1 \oplus x_t^2 \oplus x_t^3 \oplus x_t^4 \oplus c_t^0 \in \{0, 1\}$$

The elements of $GF(4)$, represented by binary vectors, are tabulated in Table 2.

Table 2: The mappings T_1 and T_2

x	$T_1[x]$	$T_2[x]$
00	00	00
01	01	11
10	10	01
11	11	10

Ultimately, the encryption comprises an XOR operation of z_t with the data packet bit sequence. The maximum packet size approximates to 2745 bits. Post transmission, the cipher undergoes reinitialization.

4 Cryptanalysis of E_0

The resilience of the E_0 cipher hinges on the intricacies of its keystream generation mechanism as well as the confidentiality of the consensual key. Notwithstanding, an array of cryptanalytic inquiries has unveiled several potential design vulnerabilities in E_0, predisposing it to assorted attacks that could potentially jeopardize the secrecy of Bluetooth communications under specific scenarios.

Subsequent to this discourse, a foundational analytical method will be delineated to assail the E_0 cipher.

The principal attack strives to deduce the primordial configurations of the LFSRs contingent upon a circumscribed keystream output, nominally 132 bits. Predicated on this modus operandi, the presumptive initial conditions of the FSM and the contents of LFSR1 and LFSR2 are postulated. Concurrently, for each discrete state, the present configuration of the blending FSM is perpetuated, along with a repertoire, denoted as $\mathcal{L}$, of linear equations derived from the output bits of LFSR3 and LFSR4, which are hereby referred to as LFSR3_n and LFSR4_n respectively.

In the incipient phase, the repertoire $\mathcal{L}$ is initialized as a null set. The ensuing procedure is a depth-first traversal as follows:

1. Designate the state under scrutiny as n. Calculate the binary exclusive disjunction of the output n of LFSR1 and LFSR2, conjoined with the subsequent output of the blending FSM, predicated on its antecedent state, and the extant keystream bit Z_n. Should our postulations hold veracity to this juncture, it must correspond to the binary exclusive disjunction of the egress from LFSR3 and LFSR4.

2. In the event that the binary exclusive disjunction culminates in naught, we bifurcate our exploration to account for the possibilities that both LFSR3 and LFSR4 emanate

zero, or conversely, both yield one. An assumption of zero entails the incorporation of two linear equations, $\text{LFSR3}_n = 0$ and $\text{LFSR4}_n = 0$, into $\mathcal{L}$, while an assumption of one predicates the inclusion of $\text{LFSR3}_n = 1$ and $\text{LFSR4}_n = 1$ into $\mathcal{L}$.

3. Should the binary exclusive disjunction resolve to one, the solitary linear equation $\text{LFSR3}_n \neq \text{LFSR4}_n$ is assimilated into $\mathcal{L}$.

4. Contingent upon n being at least 33, the linear equation signified by the tap equations of LFSR3 is subsumed into $\mathcal{L}$. Similarly, when n is no less than 39, the equation intimated by the tap equations of LFSR4 is amalgamated into $\mathcal{L}$. Subsequent to each inclusion, a verification is conducted to ascertain the congruence of the new equations with the pre-existing ones in $\mathcal{L}$. Discordance prompts a reversion to scrutinize alternative suppositions.

5. Deduce the successor state of the blender FSM, an invariable possibility given that the successor state is predicated solely upon the known current state and the count of LFSRs emitting one.

6. If n exceeds the threshold of 132, the initial state of the encryption apparatus is surmised with considerable certitude. Failing which, the investigation extends to the state $n + 1$.

Two seminal concepts underscore this algorithm. The first is that the successor state function for the blender FSM is exclusively contingent upon the quantity of LSFRs dispensing one. Accordingly, when the outputs of LFSR3 and LFSR4 are conjectured to diverge, it is not obligatory to discern which specific LFSR outputs zero or one; rather, the differential state is duly noted, and the search persists.

The second premise posits that systems of linear equations within $GF(2)$ can be efficaciously scrutinized for contradictions.

The efficacy of this cryptanalytic stratagem is predicated on heuristic arguments. Foremost, envisage the scenario where all presupposed bits of LFSRs 1 and 2, along with the blender state, are accurate.

With each iterative measure, we ascertain whether the aggregate S of the two output bits either (a) subsists within the set $\{0, 2\}$ or (b) equals 1. Both eventualities, (a) and (b), possess equiprobable likelihoods.

It is noteworthy that $\mathbf{Pr}[S = 1] = 0.5$, whilst $\mathbf{Pr}[S = 0] = \mathbf{Pr}[S = 2] = 0.25$. In the event of $S = 1$, a singular linear equation pertaining to the state bits of LFSRs 3 and 4 is discerned (specifically, the XOR of the concurrent output bits). Conversely, if S is subsumed within $\{0, 2\}$, we bifurcate and ponder both $S = 0$ and $S = 2$. Each value, $S = 0$ and $S = 2$, bequeaths us with a duo of linear equations pertinent to the state bits of LFSRs 3 and 4.

On average, we anticipate gleaning 1.5 linear equations and necessitating a bifurcation 0.5 times per iterative step. Upon the acquisition of a cumulative total of $33 + 39 = 72$ equations, we arrive at a terminal leaf of the branching paradigm, wherein all bits within the system are known or posited. The totality of such terminal leaves delineates the quantum of computational effort required. (It is imperative to note that this analysis is anchored in the heuristic presumption that the equations are neither redundant nor contradictory, or that the impact of such equations on the computational effort is nullified.)

Henceforth, our branching paradigm boasts an "average" magnitude determined by $2^{72/3} = 2^{24}$ leaves. Initially postulating 60 bits, we can anticipate a correct surmise post 2^{59} attempts, which begets an average computational complexity of $O(2^{59+24}) = O(2^{83})$.

5 Conclusion

We provided an overview of stream ciphers in general and examined the specific stream cipher algorithm known as E_0 that is integral to Bluetooth encryption and described methods for rederiving the session key for E_0 given a limited amount of known keystream. This session key will allow the attacker to decrypt all messages in that session. We showed that the real security level of E_0 is no more than 84 bits (depending the amount of keystream available to the attacker). While E_0 plays a critical role in Bluetooth security, continued research into its design strengths and weaknesses is important to ensure the privacy and integrity of the billions of wireless connections it helps enable every day.

6 Contributions

All the reference search, writing work and graph production of this report were completed by H3Art.

References

[1] K. K. Deepthi, K. Singh, and S. K. Konduru, "Cube attack on stream cipher e0: revisited," *International Journal of Information Technology*, vol. 14, no. 5, pp. 2575–2584, 2022.

[2] 郭锋 and 庄奕琪, "蓝牙 e0 加密算法安全分析," 电子科技大学学报, pp. 160–163, 2006.

[3] S. Fluhrer and S. Lucks, "Analysis of the e 0 encryption system," in *Selected Areas in Cryptography: 8th Annual International Workshop, SAC 2001 Toronto, Ontario, Canada, August 16–17, 2001 Revised Papers 8*. Springer, 2001, pp. 38–48.

[4] R. La Scala, S. Polese, S. K. Tiwari, and A. Visconti, "An algebraic attack to the bluetooth stream cipher e0," *Finite Fields and Their Applications*, vol. 84, p. 102102, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1071579722001113

[5] G. Lamm, G. Falauto, J. Estrada, J. Gadiyaram, and D. Cockerham, "Bluetooth wireless networks security features," in *Proc. IEEE Workshop Information Assurance and Security*. Citeseer, 2001, pp. 265–272.

[6] Y. Shaked and A. Wool, "Cryptanalysis of the bluetooth e 0 cipher using obdd' s," in *Information Security: 9th International Conference, ISC 2006, Samos Island, Greece, August 30-September 2, 2006. Proceedings 9*. Springer, 2006, pp. 187–202.

[7] M. Hermelin and K. Nyberg, "Correlation properties of the bluetooth combiner," in *International Conference on Information Security and Cryptology*. Springer, 1999, pp. 17–29.