ASC-1: An Authenticated Encryption Stream Cipher

Goce Jakimoski[1] and Samant Khajuria[2,⋆]

[1] Stevens Institute of Technology, USA
[2] Aalborg University, Denmark

Abstract. The goal of the modes of operation for authenticated encryption is to achieve faster encryption and message authentication by performing both the encryption and the message authentication in a single pass as opposed to the traditional encrypt-then-mac approach, which requires two passes. Unfortunately, the use of a block cipher as a building block limits the performance of the authenticated encryption schemes to at most one message block per block cipher evaluation.

In this paper, we propose the authenticated encryption scheme ASC-1 (Authenticating Stream Cipher One). Similarly to LEX, ASC-1 uses leak extraction from different AES rounds to compute the key material that is XOR-ed with the message to compute the ciphertext. Unlike LEX, the ASC-1 operates in a CFB fashion to compute an authentication tag over the encrypted message. We argue that ASC-1 is secure by reducing its (IND-CCA , INT-CTXT) security to the problem of distinguishing the case when the round keys are uniformly random from the case when the round keys are generated by a key scheduling algorithm.

Keywords: authenticated encryption, stream ciphers, message authentication, universal hash functions, block ciphers, maximum differential probability.

1 Introduction

Confidentiality and message authentication are two fundamental information security goals. Confidentiality addresses the issue of keeping the information secret from unauthorized users. Often, this is achieved by encrypting the data using a symmetric-key encryption scheme. Message authentication addresses the issues of source corroboration and improper or unauthorized modification of data. To protect the message authenticity, the sender usually appends an authentication tag that is generated by the signing (tagging) algorithm of some message authentication scheme.

Although symmetric-key encryption and message authentication have been mainly studied in a separate context, there are many applications where both are needed. The cryptographic schemes that provide both confidentiality and

⋆ The research was supported in part by the Center for Wireless Systems and Applications - CTIF Copenhagen.

A. Miri and S. Vaudenay (Eds.): SAC 2011, LNCS 7118, pp. 356–372, 2012.

authenticity are called authenticated encryption schemes. The authenticated encryption schemes consist of three algorithms: a key generation algorithm, an encryption algorithm, and a decryption algorithm. The encryption algorithm takes a key, a plaintext and an initialization vector and it returns a ciphertext. Given the ciphertext and the secret key, the decryption algorithm returns plaintext when the ciphertext is authentic, and invalid when the ciphertext is not authentic. The scheme is secure if it is both unforgeable and secure encryption scheme [1]. Two block cipher modes of operation for authenticated encryption, IACBC and IAPM, supported by a claim of provable security were proposed in [14]. Provably secure authenticated encryption schemes that use a block cipher as a building block were also presented in [9,10,24]. The previous authenticated encryption schemes use a block cipher as a building block. The duplex construction [2] iteratively applies a bijective transformation, and its main application is authenticated encryption. One can also incorporate some message authentication mechanisms in a stream cipher. The drawback of this approach is that one cannot reduce the security of the scheme to a well-known problem such as the indistinguishability of block ciphers from random permutations. However, this approach promises better efficiency. One such authenticated encryption scheme is Helix [7]. Another example of a heuristically designed authenticated encryption scheme is SOBER-128 [11].

We propose the authenticated encryption scheme ASC-1. The design of the scheme has roots in message authentication and encryption schemes that use four rounds of AES [8] as a building block such as the LEX [3] stream cipher, the ALRED [4,5] MAC scheme, and the MAC schemes proposed in [20,13]. However, unlike the previous constructions, we use a single cryptographic primitive to achieve both message secrecy and authenticity. To argue the security of the scheme, we show that the scheme is secure if one cannot tell apart the case when the scheme uses random round keys from the case when the round keys are derived by a key scheduling algorithm. Our information-theoretic security analysis uses the approach taken in [12,15,16,17,18,19,21,22] to provide differential probability bounds.

2 ASC-1 Specification

ASC-1 is an authenticated encryption scheme. Its key size can vary depending on the block cipher that is used. Our block cipher suggestion is AES with 128-bit key. The encryption and decryption algorithms for a message $M = m_1||m_2||m_3$ consisting of three 128-bit blocks are depicted in Figure 1.

The schemes uses a 56-bit representation of a counter that provides a unique initialization vector for each encrypted message. The encryption algorithm derives an initial state X_0 and three keys $K_{1,0}$, $K_{2,0}$ and $K_{3,0}$ by applying a block cipher to $0^{70}||00||Cntr$, $0^{70}||01||Cntr$, $0^{70}||10||Cntr$ and $l(M)||00000011||Cntr$ respectively, where $l(M)$ is a 64-bit representation of the bit length of the message M. The message is then processed in a CFB-like mode using the 4R-AES transformation. The 4R-AES transformation takes as input a 128-bit input state

and outputs a 128-bit "random" leak r_i and a 128-bit output state. The first leak r_1 is used to encrypt the first message block m_1. The resulting ciphertext block c_1 is XOR-ed with the output state to give the input state for the second 4R-AES transformation. This process is repeated for all message blocks. The leak from the last 4R-AES application is ignored, and its output h is encrypted by $K_{3,0}$ to give the authentication tag. The ciphertext consists of the counter value, the ciphertext blocks and the authentication tag.

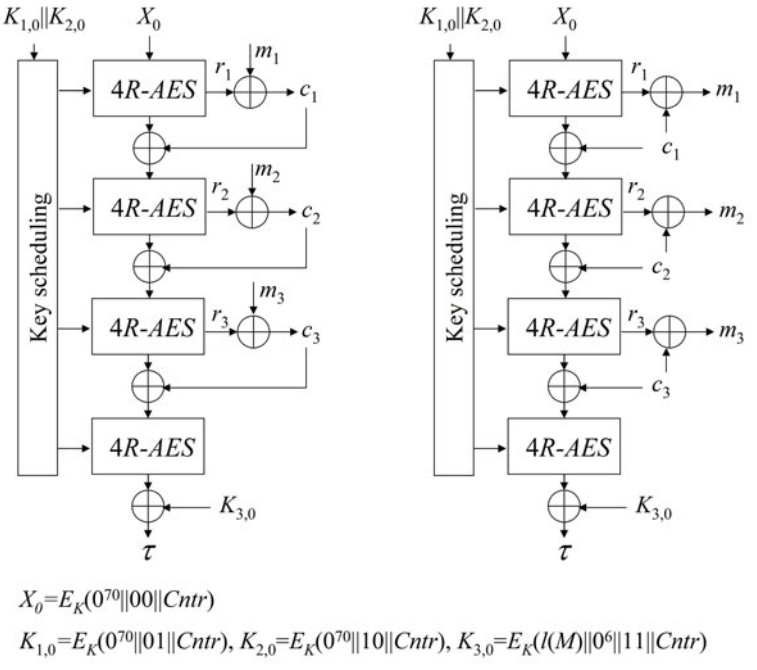

Fig. 1. The encryption and decryption algorithms of ASC-1. The message consists of three blocks. The ciphertext consists of the counter value, three ciphertext block and an authentication tag. The receiver recovers the original message and verifies its validity by checking whether the re-computed authentication tag is equal to the received one.

The decryption algorithm uses the same secret key and the received counter value to compute X_0, $K_{1,0}$, $K_{2,0}$ and $K_{3,0}$. The leak r_1 derived by applying 4R-AES to X_0 is used to decrypt c_1 into the original message block m_1. The output of the first 4R-AES is XOR-ed with the first ciphertext block to give the next input state, and the process is repeated until all message blocks are recovered and an authentication tag of the message is computed. If the computed tag is same as the one that was received, then the decrypted message is accepted as valid.

Although, we use 64-bit and 56-bit representations for the message length and the counter, we assume that both the maximum message length and the

maximum number of messages to be encrypted is 2^{48}. The message length might not be a multiple of the block length. In this case, the last message block m_n with length $l_n < 128$ is padded with zeros to get a 128-bit block m'_n. A 128-bit ciphertext block c'_n is derived as $c'_n = m'_n \oplus r_n$, and it is XOR-ed with the n-th output state to give the $(n+1)$-st input state. However, the sender will not transmit c'_n, but c_n, which consists of the first l_n bits of c'_n. This will enable the receiver to recover the message length.

The 4R-AES transformation is depicted in Figure 2. Four AES rounds are applied to the initial state $x = (x_1, \ldots, x_{16})$ to give a 128-bit leak $r = l_{1..4}||l_{5..8}||l_{9..12}||l_{13..16}$ and an output state $y = (y_1, \ldots, y_{16})$. Here, we assume that the key addition is the first operation of the AES rounds. Four bytes are leaked after the MixColumns transformation in each round. The leak positions are same as in LEX. However, unlike LEX, we add a whitening key byte before each extracted byte.

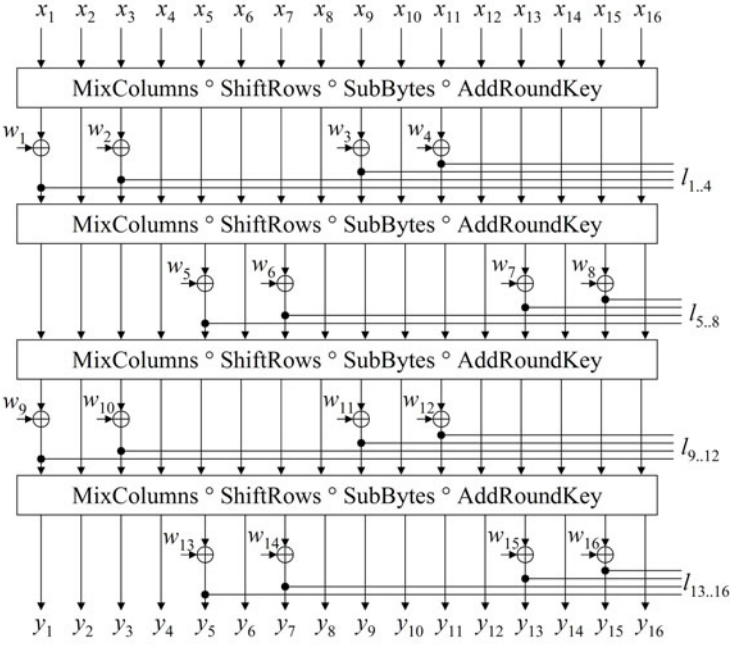

Fig. 2. The 4R-AES transfomration

The 4R-AES transformation uses five 128-bit keys: four round keys and one whitening key. These keys are derived from the 256-bit key $K_{1,0}||K_{2,0}$ as follows. The AES-256 key scheduling algorithm is applied to $K_{1,0}||K_{2,0}$ to derive 14 round keys $K_1, K_2, \ldots, K_{14}$. The keys K_2, K_3, K_4 and K_5 are used as round keys in the first 4R-AES transformation. The keys K_7, K_8, K_9 and K_{10} are used as round keys in the second 4R-AES transformation. The key K_1 is used as a whitening key in the second 4R-AES transformation, and the key K_{11} is used as

a whitening key in the first 4R-AES transformation. The AES-256 key scheduling algorithm is again applied to $K_{13}||K_{14}$ to derive 14 keys that are used by the third and the fourth 4R-AES transformation, and the process is repeated as long as we need new keys.

3 Authenticated Encryption Based on Leak-Safe AXU (LAXU) Hash Functions

In this section, we introduce the concept of a leak-safe almost XOR universal (LAXU) hash function, which is an extension of the notion of an AXU hash function [23]. We also show how LAXU hash functions can be used to construct an unconditionally secure authenticated encryption scheme. This construction is used in the information-theoretic part of the security proof of ASC-1 in Section 4.

Definition 1 (LAXU). *A family of hash functions* $\mathcal{H} = \{h(m) = (l, h)|m \in M, l \in \{0,1\}^k, h \in \{0,1\}^n\}$ *is* leak-safe ϵ-almost XOR universal$_2$*, written* ϵ-LAXU$_2$*, if for all distinct messages* $m, m' \in M$*, for all leaks* $l \in \{0,1\}^k$ *and any constant* $c \in \{0,1\}^n$*,*

$$\Pr_{h \in \mathcal{H}}[\pi_h(h(m)) \oplus \pi_h(h(m')) = c|\pi_l(h(m)) = l] \leq \epsilon,$$

where $\pi_h(l, h) = h$ *and* $\pi_l(l, h) = l$ *are projection functions.*

One can use a LAXU hash function family as a building block to construct an unconditionally secure authenticated encryption scheme as shown in Figure 3. We assume that the message M consists of d n-bit blocks. Some techniques that deal with arbitrary length messages are discussed later on. The ciphertext blocks are computed as follows. A hash function h_{K_1} is selected randomly from $\mathcal{H}$ and it is applied to an initial value IV to get a leak l_1 and hash value h_1. The leak l_1 is used to encrypt the message block m_1 into a ciphertext block $c_1 = m_1 \oplus l_1$. A new hash function h_{K_2} is randomly drawn from $\mathcal{H}$. It is applied to $i_2 = h_1 \oplus c_1 \oplus k_1$, where k_1 is a random key, to get a leak l_2 and hash value h_2. The leak l_2 is used to encrypt the message block m_2 into a ciphertext block c_2, and the process is repeated until the encryption of the last message block m_d. The authentication tag τ is computed as $\tau = K_T \oplus h_{d+1}$, where K_T is a random n-bit key, and h_{d+1} is the hash value that is obtained by applying a randomly drawn hash function $h_{K_{d+1}}$ to $c_d \oplus h_d$. The ciphertext $C = IV||c_1||c_2||\ldots||c_d||\tau$ is a concatenation of the initial value, the ciphertext blocks, and the authentication tag.

We assume that the recipient has knowledge of the secret keys that were used to encrypt the message. The decryption and verification of the ciphertext proceeds as follows. First, h_{K_1} is applied to IV to get a leak l_1 and hash value h_1. The leak l_1 is used to decrypt the ciphertext block c_1 into a message block $m_1 = c_1 \oplus l_1$. Then, the hash function h_{K_2} is applied to $i_2 = h_1 \oplus c_1 \oplus k_1$ to get a leak l_2 and hash value h_2. The second message block is obtained as $m_2 = c_2 \oplus l_2$, and the process is repeated until all message blocks $m_1, m_2, \ldots, m_d$ are decrypted. To verify the authenticity of the received ciphertext, the recipient

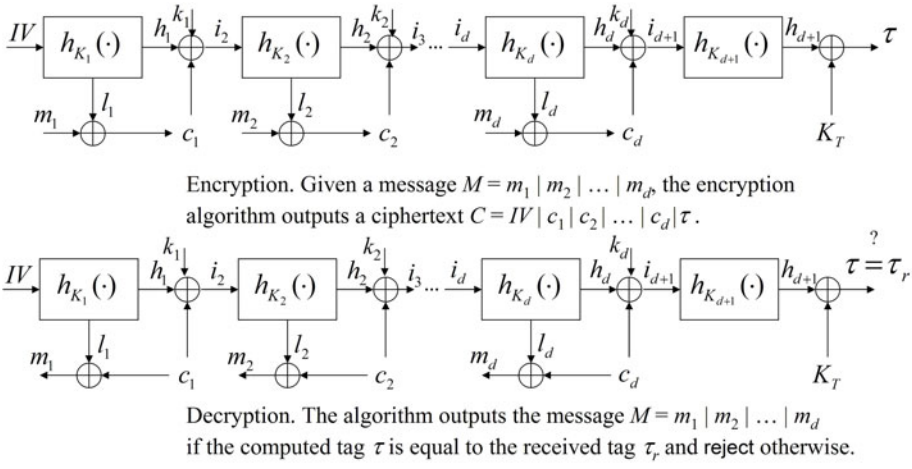

Fig. 3. An authenticated encryption scheme construction based on a LAXU hash function family in a CFB-like mode

recomputes the authentication tag τ as $\tau = h_{d+1} \oplus K_T$, where h_{d+1} is the hash value that is obtained when applying $h_{K_{d+1}}$ to $c_d \oplus h_d$. If the recomputed tag τ is equal to the received tag τ_r, then the decryption algorithm outputs the message $M = m_1||m_2||\ldots||m_d$. Otherwise, the decryption algorithm outputs reject.
The following theorem establishes the security of the previous construction.

Theorem 1. *Suppose that* $\mathcal{H} = \{h(m) = (l,h) | m \in \{0,1\}^n, l \in \{0,1\}^n, h \in \{0,1\}^n\}$ *is an* ϵ-LAXU$_2$ *family of hash functions such that (i)* $\pi_h(h(m))$ *is a bijection, and (ii)* $\Pr_{h \in_R \mathcal{H}}[\pi_l(h(m)) = l | m] = 2^{-n}$ *for any message* m *and any leak* l*. Then, the authenticated encryption scheme depicted in Figure 3 achieves:*

1. perfect secrecy. *The a posteriori probability that the message is* M *given a ciphertext* C *is equal to the a priori probability that the message is* M*.*
2. unconditionally secure ciphertext integrity. *The probability that a computationally unbounded adversary will successfully forge a ciphertext is at most* $q_v\epsilon$*, where* q_v *is the number of the verification queries that the adversary makes.*

Proof. The perfect secrecy of the scheme follows from the fact that the initial value IV is independent of the message, and all l_i and the key K_T have uniform probability distribution for any possible message. A more formal analysis is given below.

$$
\begin{aligned}
&\Pr[\mathbf{M} = m_1||\ldots||m_d | \mathbf{C} = IV||c_1||\ldots||c_d||\tau] = \\
&= \frac{\Pr[\mathbf{M} = m_1||\ldots||m_d] \times \Pr[\mathbf{C} = IV||c_1||\ldots||c_d||\tau | \mathbf{M} = m_1||\ldots||m_d]}{\sum_{\mathbf{M'}} \Pr[\mathbf{M'} = m'_1||\ldots||m'_d] \times \Pr[\mathbf{C} = IV||c_1||\ldots||c_d||\tau | \mathbf{M'} = m'_1||\ldots||m'_d]} \\
&= \frac{\Pr[\mathbf{M} = m_1||\ldots||m_d] \times 2^{-(d+1)n} \times \Pr[IV]}{\sum_{\mathbf{M'}} \Pr[\mathbf{M'} = m'_1||\ldots||m'_d] \times 2^{-(d+1)n} \times \Pr[IV]} \\
&= \Pr[\mathbf{M} = m_1||\ldots||m_d].
\end{aligned}
$$

In the previous analysis, we used the fact that for any message $\mathbf{M}'$:

$$\begin{aligned}
&\Pr[\mathbf{C} = IV||c_1||\dots||c_d||\tau|\mathbf{M}' = m_1'||\dots||m_d'] = \\
&= \Pr[c_1||\dots||c_d||\tau|IV, \mathbf{M}'] \times \Pr[IV|\mathbf{M}'] \\
&= \Pr[\mathbf{l} = m_1 \oplus c_1||\dots||m_d \oplus c_d, K_T = h_{d+1} \oplus \tau|IV, \mathbf{M}'] \times \Pr[IV] \\
&= 2^{-(d+1)n} \times \Pr[IV].
\end{aligned}$$

There are two possible types of attacks when considering the authenticity of the ciphertext: an impersonation attack and a substitution attack.

In the case of an impersonation attack, the attacker constructs and sends a ciphertext to the receiver before he sees the encryption of the message. Due to the fact that the key K_T is uniformly random, the probability of success of an impersonation attack is at most 2^{-n}. If the adversary makes q_I impersonation attempts, then the probability that at least one of this attempts will be successful ia $1 - (1 - 2^{-n})^{q_I} \leq q_I \times 2^{-n}$.

In the case of substitution attack, the adversary has intercepted the ciphertext of a given message and tries to replace it with a different ciphertext that will be accepted as valid by the receiver. We will show that the probability of success in this case is at most $q_S \times \epsilon$, where q_S is the number of substitution attempts made by the adversary.

Suppose that $\mathbf{C} = IV||c_1||\dots||c_d||\tau$ is the ciphertext of a chosen message $\mathbf{M}$ and $\mathbf{C}' = IV'||c_1'||\dots||c_d'||\tau'$ is the substitution ciphertext. If the two ciphertexts $\mathbf{C}$ and $\mathbf{C}'$ differ only in their authentication tags (i.e., $\tau' \neq \tau$, $IV' = IV$ and $c_j = c_j', 1 \leq j \leq d$), then the probability of successful substitution is zero. Therefore, the only interesting case is when the substitution ciphertext $\mathbf{C}'$ and the original ciphertext $\mathbf{C}$ differ in at least one block that is different from the tag block.

Let $0 \leq j \leq d$ be the index of the first block where $\mathbf{C}$ and $\mathbf{C}'$ differ, and let $\Delta i_{j+1} = c_j' \oplus c_j$ be the difference at the input of $h_{K_{j+1}}$, with $c_0 = IV$ and $c_0' = IV'$. Then, due to the ϵ-LAXU and invertibility properties of $\mathcal{H}$, we have that $\Pr[\Delta h_{j+1} = 0|\mathbf{M}, \mathbf{C}, \mathbf{C}'] = 0$ and $\forall_{\Delta \in \{0,1\}^n, \Delta \neq 0} \Pr[\Delta h_{j+1} = \Delta|\mathbf{M}, \mathbf{C}, \mathbf{C}'] \leq \epsilon$, where Δh_{j+1} is the difference at the output of $h_{K_{j+1}}$. Hence, for the difference $\Delta i_{j+2} = \Delta h_{j+1} \oplus \Delta c_{j+1}$, we get that $\forall_{\Delta \in \{0,1\}^n} \Pr[\Delta i_{j+2} = \Delta|\mathbf{M}, \mathbf{C}, \mathbf{C}'] \leq \epsilon$. The probability $\Pr[\Delta h_{j+2} = 0|\mathbf{M}, \mathbf{C}, \mathbf{C}']$ is equal to the probability that $\Pr[\Delta i_{j+2} = 0|\mathbf{M}, \mathbf{C}, \mathbf{C}']$, and is at most ϵ. When the input difference Δi_{j+2} is nonzero, we get that $\forall_{\Delta \in \{0,1\}^n, \Delta \neq 0} \Pr[\Delta h_{j+2} = \Delta|\mathbf{M}, \mathbf{C}, \mathbf{C}'] \leq \epsilon$. If we continue in this manner, we get that $\forall_{\Delta \in \{0,1\}^n} \Pr[\Delta h_{d+1} = \Delta|\mathbf{M}, \mathbf{C}, \mathbf{C}'] \leq \epsilon$. The substitution ciphertext will be accepted as valid only if $h_{d+1}' \oplus K_T = \tau'$, i.e., only if $\Delta h_{d+1} = \Delta\tau$, where $\Delta\tau = \tau \oplus \tau'$. Given the previous analysis, this will happen with probability no larger than ϵ.

The probability that at least one out of q_S substitution queries will be successful is at most $q_S\epsilon$. The probability of success when making at most $q_v = q_I + q_S$ verification queries is at most $q_v\epsilon$ due to the fact that $\epsilon \leq 2^{-n}$. □

To deal with messages of arbitrary length, one can generate uniformly at random a key K_T for each possible message length. Now, if one substitutes a ciphertext

with a different-length ciphertext, then the probability of success will be same as for the impersonation attack (i.e., 2^{-n}). In ASC-1, this is accomplished by having the message length as a part of the input when generating $K_{3,0}$.

4 Security of ASC-1

In this section, we show that if the block cipher used in ASC-1 is secure and one cannot tell apart the case when ASC-1 uses random round keys from the case when it uses round keys derived by a key scheduling algorithm, then ASC-1 is secure authenticated encryption scheme.

4.1 The Information-Theoretic Case

Here, we establish the unconditional security of ASC-1 with random keys. First, we consider the two round SPN structure of Figure 4. The input $\mathbf{x} = x_1||\ldots||x_n$ is an $n \times m$-bit string. The key addition operator is the bitwise XOR operator.

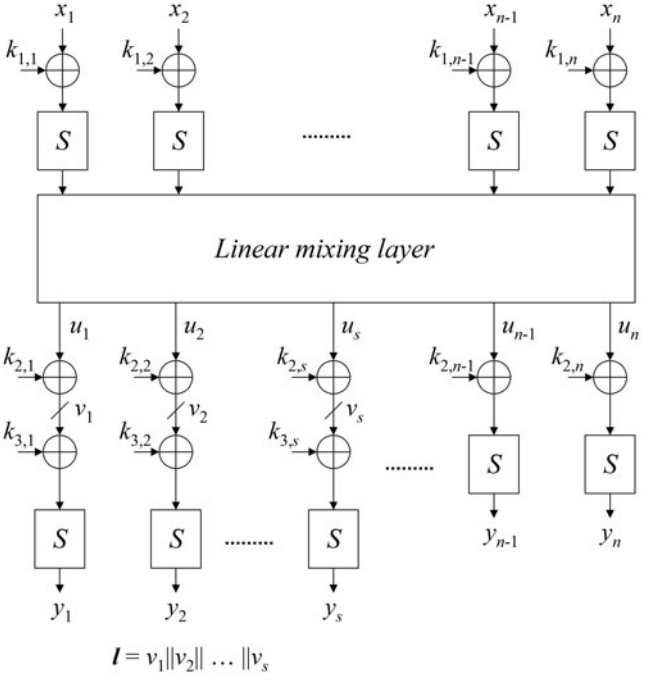

Fig. 4. A two round SPN structure with a leak. Each of the n S-boxes is a non-linear permutation on $\{0,1\}^m$, and the branch number of the linear mixing layer is $n+1$. Without loss of generality, we assume that the leak positions are the first s positions of $\mathbf{v}$ (i.e., $\mathbf{l} = v_1||v_2||\ldots||v_s$)

The non-linear substitution layer consists of n S-boxes. Each S-box is a non-linear permutation that transforms an m-bit string into an m-bit string. The mixing layer is defined by an $n \times n$ matrix. It is linear with respect to bitwise XOR and its branch number is $n+1$. We omit the mixing layer in the second round since it does not affect our analysis. The leak $\mathbf{l}$ consists of s values $v_1, \ldots, v_s$.

Each possible key $k_{1,1}, \ldots, k_{1,n}, k_{2,1}, \ldots k_{2,n}, k_{3,1}, \ldots, k_{3,s}$ defines a function that maps the input $\mathbf{x}$ into an output $\mathbf{y}$ and a leak $\mathbf{l}$. The collection of such functions $\mathcal{H}_{2R}$ forms a LAXU hash function family.

Lemma 1. *Suppose that the keys in the transformation depicted in Figure 4 are chosen uniformly at random. Then, we have that*

$$\Pr[\Delta\mathbf{y} = \Delta y | \mathbf{x} = x, \mathbf{x}' = x', \mathbf{l} = l] = \Pr[\Delta\mathbf{y} = \Delta y | \Delta\mathbf{x} = x \oplus x'].$$

Proof. Suppose that a function h (i.e., the key $k_{1,1}, \ldots, k_{1,n}$, $k_{2,1}, \ldots, k_{2,n}$, $k_{3,1}$, $\ldots, k_{3,s}$) is selected uniformly at random from $\mathcal{H}_{2R}$. Let $\mathbf{l}$ be the leak that is obtained when h is applied to an input $\mathbf{x}$, and let $\mathbf{x}'$ be an input bit string distinct from $\mathbf{x}$. The probability $\Pr[\Delta\mathbf{y} = \Delta y | \mathbf{x} = x, \mathbf{x}' = x', \mathbf{l} = l]$ is the probability that the output difference $\mathbf{y} \oplus \mathbf{y}'$ is Δy given $\mathbf{x} = x$, $\mathbf{x}' = x'$ and $\mathbf{l} = l$. Due to the initial key addition, this probability is equal to the probability $\Pr[\Delta\mathbf{y} = \Delta y | \Delta\mathbf{x} = x \oplus x', \mathbf{l} = l]$ that the output difference is Δy given the input difference is $\Delta x = x \oplus x'$ and the leak l. To prove the lemma, we use the following observations:

1. $\Pr[\Delta u | \Delta x, l] = \Pr[\Delta u | \Delta x]$, where $\Delta u = (\Delta u_1, \ldots, \Delta u_n)$, $\Delta u_i = u_i \oplus u_i'$. That is the difference Δu given input difference Δx is independent of the leak l. This is due to the second key addition, which makes the leak uniformly distributed for any possible value Δu.
2. $\Pr[\Delta y | \Delta u, l] = \prod_{i=1}^{s} \Pr[\Delta y_i | \Delta u_i, v_i] \times \prod_{i=s+1}^{n} \Pr[\Delta y_i | \Delta u_i]$. Given the difference Δu, the probability of having a difference $\Delta y_i = y_i \oplus y_i'$ at the output of the i-th S-box of the second round is independent of the probability of having a difference $\Delta y_j, j \neq i$ at the output of some other S-box in the second round.
3. $\Pr[\Delta y_i | \Delta u_i, v_i] = \Pr[\Delta y_i | \Delta u_i]$, $i = 1, \ldots, s$. After the third key addition, the input to the S-boxes is uniformly distributed and independent of the v_i values.

Using the previous observations, we can now prove the theorem.

$$\begin{aligned}
\Pr[\Delta y | \Delta x, l] &= \sum_{\Delta u} \Pr[\Delta y | \Delta u, \Delta x, l] \Pr[\Delta u | \Delta x, l] \\
&= \sum_{\Delta u} \Pr[\Delta y | \Delta u, l] \Pr[\Delta u | \Delta x] \\
&= \sum_{\Delta u} \Pr[\Delta u | \Delta x] \times \prod_{i=1}^{s} \Pr[\Delta y_i | \Delta u_i, v_i] \times \prod_{i=s+1}^{n} \Pr[\Delta y_i | \Delta u_i] \\
&= \sum_{\Delta u} \Pr[\Delta u | \Delta x] \times \prod_{i=1}^{s} \Pr[\Delta y_i | \Delta u_i] \times \prod_{i=s+1}^{n} \Pr[\Delta y_i | \Delta u_i]
\end{aligned}$$

$$= \sum_{\Delta u} \Pr[\Delta u|\Delta x] \times \prod_{i=1}^{n} \Pr[\Delta y_i|\Delta u_i]$$
$$= \sum_{\Delta u} \Pr[\Delta y|\Delta u] \times \Pr[\Delta u|\Delta x]$$
$$= \Pr[\Delta y|\Delta x]. \qquad \square$$

Corollary 1. *The family of functions $\mathcal{H}_{2R}$ defined by the 2-round transformation depicted in Figure 4 is ϵ-LAXU$_2$ with $\epsilon = \mathrm{DP}_{2R}$, where DP_{2R} is the maximum differential probability of the 2-round SPN structure when there is no leak.*

Proof. Due to the previous lemma, we get that $\Pr[\Delta\mathbf{y} = \Delta y|\mathbf{x} = x, \mathbf{x}' = x', \mathbf{l} = l] = \Pr[\Delta\mathbf{y} = \Delta y|\Delta\mathbf{x} = x \oplus x'] \leq \mathrm{DP}_{2R}$. $\square$

The previous results refer to two round SPN structures. In order to show that one can use four AES rounds to construct a LAXU hash function, we will first consider the composition of transformations depicted in Figure 5. The next lemma establishes independence of the differential probability of F_1 (resp., F_2) from the leak value l_2 (resp., l_1). This is due to the key addition operation that follows F_1 and precedes F_2.

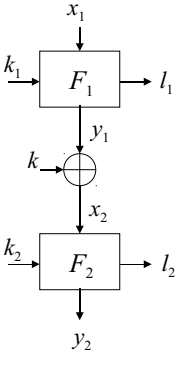

Fig. 5. A composition of a transformation F_1, key addition and transformation F_2. The length of the F_1's output y_1, the length of the F_2's input x_2 and the length of the key k are equal. Both F_1 and F_2 "leak" a value (l_1 and l_2 resp.).

Lemma 2. *The following holds for the differential probabilities of the transformations F_1 and F_2 depicted in Figure 5:*

$$\Pr[\Delta\mathbf{y_1} = \Delta y_1|\Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1, \mathbf{l_2} = l_2] = \Pr[\Delta\mathbf{y_1} = \Delta y_1|\Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1],$$

and

$$\Pr[\Delta\mathbf{y_2} = \Delta y_2|\Delta\mathbf{y_1} = \Delta y_1, \mathbf{l_1} = l_1, \mathbf{l_2} = l_2] = \Pr[\Delta\mathbf{y_2} = \Delta y_2|\Delta\mathbf{y_1} = \Delta y_1, \mathbf{l_2} = l_2].$$

Proof.

$$
\begin{aligned}
&\Pr[\Delta\mathbf{y_1} = \Delta y_1 | \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1, \mathbf{l_2} = l_2] \\
&= \sum_{y_1} \Pr[\Delta\mathbf{y_1} = \Delta y_1, \mathbf{y_1} = y_1 | \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1, \mathbf{l_2} = l_2] \\
&= \sum_{y_1} (\Pr[\Delta\mathbf{y_1} = \Delta y_1 | \mathbf{y_1} = y_1, \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1, \mathbf{l_2} = l_2] \times \\
&\quad \times \Pr[\mathbf{y_1} = y_1 | \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1, \mathbf{l_2} = l_2]) \\
&= \sum_{y_1} (\Pr[\Delta\mathbf{y_1} = \Delta y_1 | \mathbf{y_1} = y_1, \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1] \times \\
&\quad \times \Pr[\mathbf{y_1} = y_1 | \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1]) \\
&= \Pr[\Delta\mathbf{y_1} = \Delta y_1 | \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1].
\end{aligned}
$$

Here we used the fact that

$$
\begin{aligned}
&\Pr[\Delta\mathbf{y_1} = \Delta y_1 | \mathbf{y_1} = y_1, \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1, \mathbf{l_2} = l_2] \\
&= \frac{\Pr[\Delta\mathbf{y_1} = \Delta y_1, \mathbf{y_1} = y_1, \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1, \mathbf{l_2} = l_2]}{\Pr[\mathbf{y_1} = y_1, \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1, \mathbf{l_2} = l_2]} \\
&= \frac{\Pr[\mathbf{l_2} = l_2 | \Delta\mathbf{y_1} = \Delta y_1, \mathbf{y_1} = y_1, \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1]}{\Pr[\mathbf{l_2} = l_2 | \mathbf{y_1} = y_1, \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1]} \times \\
&\quad \times \frac{\Pr[\Delta\mathbf{y_1} = \Delta y_1, \mathbf{y_1} = y_1, \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1]}{\Pr[\mathbf{y_1} = y_1, \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1]} \\
&= \frac{\Pr[\mathbf{l_2} = l_2] \times \Pr[\Delta\mathbf{y_1} = \Delta y_1, \mathbf{y_1} = y_1, \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1]}{\Pr[\mathbf{l_2} = l_2] \times \Pr[\mathbf{y_1} = y_1, \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1]} \\
&= \Pr[\Delta\mathbf{y_1} = \Delta y_1 | \mathbf{y_1} = y_1, \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1].
\end{aligned}
$$

The equalities $\Pr[\mathbf{l_2} = l_2 | \mathbf{y_1} = y_1, \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1] = \Pr[\mathbf{l_2} = l_2]$ and $\Pr[\mathbf{l_2} = l_2 | \Delta\mathbf{y_1} = \Delta y_1, \mathbf{y_1} = y_1, \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1] = \Pr[\mathbf{l_2} = l_2]$ follow from the fact that the value of the second leak l_2 is independent of $\Delta x_1, y_1, \Delta y_1$ and l_1 since x_2 is uniformly distributed and independent of these values. Similarly, we can show that

$$\Pr[\mathbf{y_1} = y_1 | \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1, \mathbf{l_2} = l_2] = \Pr[\mathbf{y_1} = y_1 | \Delta\mathbf{x_1} = \Delta x_1, \mathbf{l_1} = l_1].$$

This concludes the first part of the proof. The second equation of the lemma can be proved in a similar fashion, and we omit its proof. □

Let us look now at the situation depicted in Figure 6. A keyed non-linear function F is applied to a vector $\mathbf{x}$ of n input values $(x_1, \ldots, x_n)$ to produce a vector $\mathbf{y} = (y_1, \ldots, y_n)$ of n output values. Without loss of generality, we assume that the first s output values are leaked after a uniformly random key is added to them. The knowledge of the leak $\mathbf{l'} = (l_1, \ldots, l_s)$ does not change the output differential probabilities of F.

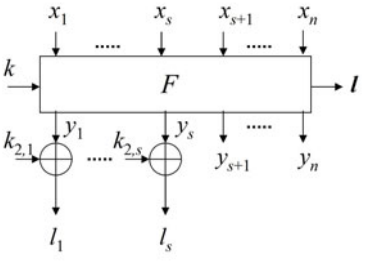

Fig. 6. The first s output values of a non-linear function F are "leaked" after a uniformly random key is added to them

Lemma 3. *Let* $\mathbf{o} = (l_1, \ldots, l_s, y_{s+1}, \ldots, y_n)$ *denote the output of the transformation depicted in Figure 6. The following holds for the output differential probability* $\Delta\mathbf{o}$*:*

$$\Pr[\Delta\mathbf{o}(\equiv \Delta\mathbf{y}) = \Delta o | \Delta\mathbf{x} = \Delta x, \mathbf{l} = l, \mathbf{l}' = l'] = \Pr[\Delta\mathbf{o} = \Delta o | \Delta\mathbf{x} = \Delta x, \mathbf{l} = l]$$

Proof. Since the output values are leaked after the random key is added, they tell nothing about the values $y_1, \ldots, y_s$ and do not affect the probability of having output difference Δy.

$$\begin{aligned}
&\Pr[\Delta\mathbf{o}(\equiv \Delta\mathbf{y}) = \Delta o | \Delta\mathbf{x} = \Delta x, \mathbf{l} = l, \mathbf{l}' = l'] \\
&= \sum_y \Pr[\Delta\mathbf{o} = \Delta o, \mathbf{y} = y | \Delta\mathbf{x} = \Delta x, \mathbf{l} = l, \mathbf{l}' = l'] \\
&= \sum_y (\Pr[\Delta\mathbf{o} = \Delta o | \mathbf{y} = y, \Delta\mathbf{x} = \Delta x, \mathbf{l} = l, \mathbf{l}' = l'] \times \\
&\quad \times Pr[\mathbf{y} = y | \Delta\mathbf{x} = \Delta x, \mathbf{l} = l, \mathbf{l}' = l']) \\
&= \sum_y \Pr[\Delta\mathbf{o} = \Delta o | \mathbf{y} = y, \Delta\mathbf{x} = \Delta x, \mathbf{l} = l] \times Pr[\mathbf{y} = y | \Delta\mathbf{x} = \Delta x, \mathbf{l} = l] \\
&= \Pr[\Delta\mathbf{o} = \Delta o | \Delta\mathbf{x} = \Delta x, \mathbf{l} = l].
\end{aligned}$$

□

The following theorem follows from the previous analysis.

Theorem 2. *Suppose that the initial state and all the keys in ASC-1 are uniformly random, then the scheme provides:*

- *perfect secrecy, and*
- *unconditional ciphertext integrity, where the probability of success of any adversary making* q_v *verifying queries is at most* $q_v \times 2^{-113}$*.*

Proof. We will show here that if the (round) keys are selected uniformly at random, then the family of functions defined by four rounds of AES with leak extraction is an ϵ-LAXU$_2$ hash function family with $\epsilon = 2^{-113}$. The first round key additions in the 4R-AES transformations play the role of the keys k_i of the

construction depicted in Figure 3. Clearly, the transformation defined by four rounds of AES is a bijection, and the leak values are uniformly random and independent of the input due to the uniform probability distribution of the keys. Therefore, the sufficient conditions of Theorem 1 are satisfied, and the scheme provides perfect secrecy and unconditional ciphertext integrity.

In our analysis, we assume that the key addition is the first round operation instead of a last one as in the AES specification. Furthermore, all the keys are independent with uniform probability distribution. We use the following notation:

- $\mathbf{x_i}$, $i = 0, \ldots, 3$ is the input to the i-th round and consists of 16 bytes $x_{i,0}, \ldots, x_{i,15}$;
- $\mathbf{y_i}$, $i = 0, \ldots, 3$ is the output of the MixColumns layer of the i-th round and consists of 16 bytes $y_{i,0}, \ldots, y_{i,15}$;
- $\mathbf{z_i}$, $i = 0, \ldots, 3$ is the state after the leak extraction layer of the i-th round and consists of 16 bytes $z_{i,0}, \ldots, z_{i,15}$;
- $\mathbf{l_i}$, $i = 0, \ldots, 3$ is the leak extracted in the i-th round and consists of 4 bytes $l_{i,0}, \ldots, l_{i,15}$;

Suppose that x'_0 and x''_0 are two distinct input values, and let us consider the output difference $\boldsymbol{\Delta z_3}$ given the input difference $\Delta \mathbf{x_0} = \mathbf{x'_0} \oplus \mathbf{x''_0}$. By applying the previously presented lemmas, we get:

$$
\begin{aligned}
&\Pr[\Delta \mathbf{z_3} = \Delta z_3 | \mathbf{x'_0} = x'_0, \mathbf{x''_0} = x''_0, \mathbf{l_0} = l_0, \mathbf{l_1} = l_1, \mathbf{l_2} = l_2, \mathbf{l_3} = l_3] \\
&= \Pr[\Delta \mathbf{z_3} = \Delta z_3 | \Delta \mathbf{x_0} = x'_0 \oplus x''_0, \mathbf{l_0} = l_0, \mathbf{l_1} = l_1, \mathbf{l_2} = l_2, \mathbf{l_3} = l_3] \\
&= \sum_{\Delta z_1} (\Pr[\Delta \mathbf{z_1} = \Delta z_1 | \Delta \mathbf{x_0} = x'_0 \oplus x''_0, \mathbf{l_0} = l_0, \mathbf{l_1} = l_1, \mathbf{l_2} = l_2, \mathbf{l_3} = l_3] \times \qquad (1) \\
&\quad \times \Pr[\Delta \mathbf{z_3} = \Delta z_3 | \Delta \mathbf{z_1} = \Delta z_1, \Delta \mathbf{x_0} = x'_0 \oplus x''_0, \mathbf{l_0} = l_0, \mathbf{l_1} = l_1, \mathbf{l_2} = l_2, \mathbf{l_3} = l_3]) \\
&= \sum_{\Delta z_1} (\Pr[\Delta \mathbf{z_1} = \Delta z_1 | \Delta \mathbf{x_0} = x'_0 \oplus x''_0, \mathbf{l_0} = l_0, \mathbf{l_1} = l_1] \times \\
&\quad \times \Pr[\Delta \mathbf{z_3} = \Delta z_3 | \Delta \mathbf{z_1} = \Delta z_1, \mathbf{l_2} = l_2, \mathbf{l_3} = l_3]) \qquad (2) \\
&= \sum_{\Delta z_1} (\Pr[\Delta \mathbf{z_1} = \Delta z_1 | \Delta \mathbf{x_0} = x'_0 \oplus x''_0, \mathbf{l_0} = l_0] \times \\
&\quad \times \Pr[\Delta \mathbf{z_3} = \Delta z_3 | \Delta \mathbf{z_1} = \Delta z_1, \mathbf{l_2} = l_2]) \qquad (3) \\
&= \sum_{\Delta z_1} \Pr[\Delta \mathbf{z_1} = \Delta z_1 | \Delta \mathbf{x_0} = x'_0 \oplus x''_0] \times \Pr[\Delta \mathbf{z_3} = \Delta z_3 | \Delta \mathbf{z_1} = \Delta z_1] \qquad (4) \\
&= \Pr[\Delta \mathbf{z_3} = \Delta z_3 | \Delta \mathbf{x_0} = x'_0 \oplus x''_0] \\
&\leq \mathrm{DP}_{4\mathrm{rAES}},
\end{aligned}
$$

where $\mathrm{DP}_{4\mathrm{rAES}}$ is the differential probability of the transformation defined by four rounds (with no leak extraction) of AES when the round keys are random. The equation (2) follows from Lemma 2, the equation (3) follows from Lemma 3, and the equation (4) follows from Lemma 1.

Having the previous inequality in mind, we get that the family of functions defined by four rounds of AES with leak extraction is an ϵ-LAXU$_2$ hash function family with $\epsilon = \mathrm{DP}_{4\mathrm{rAES}} \leq 2^{-113}$ [18]. □

4.2 Computational Security Analysis of ASC-1

In the previous subsection, we showed that if all the keys and the initial state are random, then ASC-1 is unconditionally secure authenticated encryption scheme. However, the keys and the initial state of ASC-1 are derived by combining a block cipher in a counter mode and a key scheduling algorithm. The security of the scheme in this case is based on two assumptions:

- the block cipher (e.g., AES) is indistinguishable from a random permutation, and
- one cannot tell apart the case when the initial state and the keys are random from the case when the initial state X_0 and the tag key $K_{3,0}$ are random, and the round keys are derived by applying a key scheduling algorithm to a random initial key $K_{1,0}||K_{2,0}$.

The first assumption is a standard assumption that is used in many security proofs such as the security proofs of the modes of operation for block ciphers. The second assumption is a novel one, and should be examined with more scrutiny. It asserts that an adversary cannot win in the following game. The adversary is given two oracles, an encryption oracle and a decryption oracle. A random coin b is flipped. If the outcome is zero, then a large table whose entries are random strings is generated. The number of entries in the table is equal to the maximum number of messages that can be encrypted. The length of each random string in the table is sufficient to encrypt a message of a maximum length. When the adversary submits an encryption query, the encryption oracle gets the next random string from the table, extracts the initial value and all the (round) keys from the random string, and encrypts the message. When the adversary submits a decryption query, the decryption oracle gets the random string corresponding to the counter value given in the ciphertext, and uses it to decrypt the ciphertext. If the outcome of the coin flipping is one, then the random strings in the table consist of four 128-bit random values: an initial state X_0 and three keys $K_{1,0}, K_{2,0}$ and $K_{3,0}$. When the adversary asks an encryption query, the encryption oracle uses the next available initial state and keys to encrypt the message following the ASC-1 algorithm. When the adversary asks a decryption query, the decryption oracle uses the initial state and keys corresponding to the counter value given in the ciphertext to decrypt the ciphertext. The goal of the adversary is to guess the outcome of the coin flipping. The adversary wins if it can guess the value of b with probability significantly greater than $\frac{1}{2}$.

It is not uncommon to make the assumption that the round keys are random when analyzing the security of cryptographic primitives. For instance, this assumption is always made when proving the resistance of a block cipher to linear and differential cryptanalysis (e.g., [22]). However, one can easily come up with

a stream cipher that is secure when the random round keys assumption is made, but is trivial to break otherwise. Since the design of ASC-1 was inspired by the LEX stream cipher, we are going to address the known attacks on LEX:

- LEX applies iteratively a block cipher transformation to some initial state. During this process some bytes are leaked from different rounds (i.e., states), and then used as randomness to encrypt the message. The attack presented in [6] analyzes the state differences to find highly probable differentials and deduce the secret key. However, in our case, we do not use the same round keys repeatedly. So, in order for a differential cryptanalysis to work, one has to be able to guess the round key differences as well. Since these round keys are far apart in the key scheduling process, this does not appear to be an easy task.
- The attack presented in [25] looks for a repetition of a state, which can be easily detected due to the fact that same states will generate the same pseudo-random key material. The state in LEX is a 128-bit string since the round keys are reused, and it is possible to find collisions. In our case, the state is a 384-bit string, and finding collisions should not be a straightforward problem.
- Some modified variants of the previous attacks might work if the key scheduling algorithm generates short cycles. However, the probability of having a cycle of length less than 2^{64} when considering a random permutation on $\{0,1\}^{256}$ is at most $\approx 2^{-128}$, and we are not aware of the existence of short cycles.

It is not hard to show that given an adversary A_{ROR} that can distinguish the ciphertext generated by ASC-1 from a random string, one can construct two adversaries A_{PRP}, which can tell apart the block cipher from a PRP, and A_{KSOR}, which can distinguish the case when the round keys are random from the case when the round keys are derived by a key scheduling algorithm, such that at least one of these adversary wins with significant probability. Namely, the A_{PRP} and A_{KSOR} will use their oracles to simulate ASC-1 and answer A_{ROR}'s queries. The output of A_{PRP} and A_{KSOR} will be same as A_{ROR}'s output. If the advantage of A_{ROR} is non-negligible, then at least one of A_{PRP} and A_{KSOR} will have non-negligible advantage. A similar result will hold in the case of a forging adversary A_F. So, we have the following informal theorems.

Theorem 3. *If the block cipher used by ASC-1 is a pseudo-random permutation and one cannot tell apart the case when ASC-1 uses random keys from the case when ASC-1 uses a key scheduling algorithm to derive the round keys, then ASC-1 is a secure encryption scheme in the Real-Or-Random sense.*

Theorem 4. *If the block cipher used by ASC-1 is a pseudo-random permutation and one cannot tell apart the case when ASC-1 uses random keys from the case when ASC-1 uses a key scheduling algorithm to derive the round keys, then ASC-1 is a secure message authentication scheme in the ciphertext-integrity sense.*

The definition of Real-Or-Random security of a symmetric encryption scheme and the definition of a ciphertext integrity for an authenticated encryption scheme can be found in [1].

5 Conclusions

We have proposed ASC-1, which is an authenticated encryption scheme that is designed using a stream cipher approach instead of a block cipher mode approach. We argued the security of ASC-1 by showing that it is secure if one cannot distinguish the case when the round keys are uniformly random from the case when the round keys are derived by the key scheduling algorithm of ASC-1.

References

1. Bellare, M., Namprempre, C.: Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 531–545. Springer, Heidelberg (2000)
2. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Duplexing the Sponge: Authenticated Encryption and Other Applications. In: The Second SHA-3 Candidate Conference (2010)
3. Biryukov, A.: The Design of a Stream Cipher LEX. In: Biham, E., Youssef, A.M. (eds.) SAC 2006. LNCS, vol. 4356, pp. 67–75. Springer, Heidelberg (2007)
4. Daemen, J., Rijmen, V.: A New MAC Construction ALRED and a Specific Instance ALPHA-MAC. In: Gilbert, H., Handschuh, H. (eds.) FSE 2005. LNCS, vol. 3557, pp. 1–17. Springer, Heidelberg (2005)
5. Daemen, J., Rijmen, V.: The Pelican MAC Function, IACR ePrint Archive, 2005/088
6. Dunkelman, O., Keller, N.: A New Attack on the LEX Stream Cipher. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 539–556. Springer, Heidelberg (2008)
7. Ferguson, N., Whiting, D., Schneier, B., Kelsey, J., Lucks, S., Kohno, T.: Helix: Fast Encryption and Authentication in a Single Cryptographic Primitive. In: Johansson, T. (ed.) FSE 2003. LNCS, vol. 2887, pp. 330–346. Springer, Heidelberg (2003)
8. Advanced Encryption Standard (AES), FIPS Publication 197 (November 26, 2001), `http://csrc.nist.gov/encryption/aes`
9. Gligor, V., Donescu, P.: Fast Encryption and Authentication: XCBC Encryption and XECB Authentication Modes. Presented at the 2nd NIST Workshop on AES Modes of Operation, Santa Barbara, CA (August 24, 2001)
10. Gligor, V.D., Donescu, P.: Fast Encryption and Authentication: XCBC Encryption and XECB Authentication Modes. In: Matsui, M. (ed.) FSE 2001. LNCS, vol. 2355, pp. 1–20. Springer, Heidelberg (2002)
11. Hawkes, P., Rose, G.: Primitive Specification for SOBER-128, `http://www.qualcomm.com.au/Sober128.html`
12. Hong, S., Lee, S., Lim, J., Sung, J., Cheon, D., Cho, I.: Provable Security against Differential and Linear Cryptanalysis for the SPN Structure. In: Schneier, B. (ed.) FSE 2000. LNCS, vol. 1978, pp. 273–283. Springer, Heidelberg (2001)

13. Jakimoski, G., Subbalakshmi, K.P.: On Efficient Message Authentication Via Block Cipher Design Techniques. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 232–248. Springer, Heidelberg (2007)
14. Jutla, C.S.: Encryption Modes with Almost Free Message Integrity. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 529–544. Springer, Heidelberg (2001)
15. Kang, J.-S., Hong, S., Lee, S., Yi, O., Park, C., Lim, J.: Practical and Provable Security Against Differential and Linear Cryptanalysis for Ssubstitution-Permutation Networks. ETRI Journal 23(4), 158–167 (2001)
16. Keliher, L., Meijer, H., Tavares, S.: New Method for Upper Bounding the Maximum Average Linear Hull Probability for sPNs. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 420–436. Springer, Heidelberg (2001)
17. Keliher, L., Meijer, H., Tavares, S.: Improving the Upper Bound on the Maximum Average Linear Hull Probability for Rijndael. In: Vaudenay, S., Youssef, A.M. (eds.) SAC 2001. LNCS, vol. 2259, pp. 112–128. Springer, Heidelberg (2001)
18. Keliher, L., Sui, J.: Exact Maximum Expected Differential and Linear Probability for 2-Round Advanced Encryption Standard (AES). IACR ePrint Archive, 2005/321
19. Matsui, M.: New Structure of Block Ciphers with Provable Security against Differential and Linear Cryptanalysis. In: Gollmann, D. (ed.) FSE 1996. LNCS, vol. 1039, pp. 205–218. Springer, Heidelberg (1996)
20. Minematsu, K., Tsunoo, Y.: Provably Secure MACs from Differentially-Uniform Permutations and AES-Based Implementations. In: Robshaw, M.J.B. (ed.) FSE 2006. LNCS, vol. 4047, pp. 226–241. Springer, Heidelberg (2006)
21. Park, S., Sung, S.H., Chee, S., Yoon, E.-J., Lim, J.: On the Security of Rijndael-Like Structures against Differential and Linear Cryptanalysis. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 176–191. Springer, Heidelberg (2002)
22. Park, S., Sung, S.H., Lee, S., Lim, J.: Improving the Upper Bound on the Maximum Differential and the Maximum Linear Hull Probability for SPN Structures and AES. In: Johansson, T. (ed.) FSE 2003. LNCS, vol. 2887, pp. 247–260. Springer, Heidelberg (2003)
23. Rogaway, P.: Bucket Hashing and Its Application to Fast Message Authentication. In: Coppersmith, D. (ed.) CRYPTO 1995. LNCS, vol. 963, pp. 29–42. Springer, Heidelberg (1995)
24. Rogaway, P., Bellare, M., Black, J., Krovetz, T.: OCB: A block-cipher mode of operation for efficient authenticated encryption. In: Proc. 8th ACM Conf. Comp. and Comm. Security, CCS (2001)
25. Wu, H., Preneel, B.: Resynchronization Attacks on WG and LEX. In: Robshaw, M.J.B. (ed.) FSE 2006. LNCS, vol. 4047, pp. 422–432. Springer, Heidelberg (2006)