Hardness of Learning AES Key (Student Abstract)

Artur Pak, Sultan Nurmukhamedov, Rustem Takhanov, Zhenisbek Assylbekov

Department of Mathematics, Nazarbayev University
53, Kabanbay Batyr ave., Astana, Kazakhstan
zhassylbekov@nu.edu.kz

Abstract

We show hardness of learning AES key from pairs of ciphertexts under the assumption of *computational* closeness of AES to pairwise independence. The latter is motivated by a recent result on statistical closeness of AES to pairwise independence.

Introduction and Main Result

Advanced Encryption Standard (AES) is one of the most popular encryption algorithms today. It underlies the TLS 1.3 protocol, which is used by most modern websites, email services, instant messengers, etc. However, AES is not based on any hard mathematical problem (or at least we do not know there is one), and we currently have little understanding of its *provable* security. A recent work by Liu et al. (2021) shows that for a pair of distinct inputs the corresponding pair of AES outputs is *statistically* close to pairwise independence (i.e. statistically indistinguishable from a pair of uniformly sampled distinct random n-bit strings) under the assumption of independence and randomness of keys at *each* round. This rules out attacks based on differential and linear cryptanalysis.

Let $F : \{0,1\}^m \times \{0,1\}^n \to \{0,1\}^n$ be a permutation family, denoted as $F_{\mathbf{k}}(\mathbf{x})$, where $\mathbf{k} \in \{0,1\}^m$ is a key, and $\mathbf{x} \in \{0,1\}^n$ is an input. AES is a special case of F with $m \in \{128, 192, 256\}$ and $n = 128$. In this work, we prove the resistance of a permutation family F to attacks based on machine learning under the following

Assumption 1. *For a pair of distinct inputs* $\mathbf{x}$ *and* $\mathbf{x}'$*, and a uniformly sampled key* $\mathbf{k}$*, the distribution of the corresponding pair* $[F_{\mathbf{k}}(\mathbf{x}), F_{\mathbf{k}}(\mathbf{x}')]$ *is computationally indistinguishable from the uniform distribution of two random distinct* n*-bit strings* $[\mathbf{u}, \mathbf{u}']$*, i.e. for any* $\mathrm{poly}(n)$*-time algorithm* D

$$\left| \Pr_{\mathbf{k}}[D(F_{\mathbf{k}}(\mathbf{x}), F_{\mathbf{k}}(\mathbf{x}')) = 1] - \Pr_{\mathbf{u},\mathbf{u}'}[D(\mathbf{u}, \mathbf{u}') = 1] \right| \leq 1/\mathrm{poly}(n) \quad (1)$$

Note that the result of Liu et al. (2021) differs from Assumption 1 in that we require only the initial key to be random, as is the case in the real AES.

We show that existence of a function computable in $\mathrm{poly}(n)$ time that, given a pair of arbitrary ciphertexts, can recover one of the keys consistent with those ciphertexts, would result in a polynomial distinguisher that contradicts Assumption 1. Our main result is the following

Theorem 1. *Let* $\mathbf{x}$ *and* $\mathbf{x}'$ *be arbitrary distinct* n*-bit strings and assume there exists a function* $h_{\mathbf{x},\mathbf{x}'} : \{0,1\}^{2n} \to \{0,1\}^m$ *such that*

$$h_{\mathbf{x},\mathbf{x}'}(\mathbf{y}, \mathbf{y}') = \begin{cases} \mathbf{k}, \text{ if } \exists \mathbf{k} : [F_{\mathbf{k}}(\mathbf{x}), F_{\mathbf{k}}(\mathbf{x}')] = [\mathbf{y}, \mathbf{y}'] \\ \mathbf{0}, \text{ otherwise} \end{cases}, \quad (2)$$

and $h_{\mathbf{x},\mathbf{x}'}$ *is computable in* $\mathrm{poly}(n)$ *time. Then for a random uniform* m*-bit string* $\mathbf{k}$ *the distribution of* $[F_{\mathbf{k}}(\mathbf{x}), F_{\mathbf{k}}(\mathbf{x}')]$ *is* computationally distinguishable *from that of two uniformly sampled distinct* n*-bit vectors.*

Remark. Under Assumption 1 there is no efficient learner for the class $\mathcal{H} := \{h_{\mathbf{x},\mathbf{x}'} \mid \mathbf{x}, \mathbf{x}' \in \{0,1\}^n, \mathbf{x} \neq \mathbf{x}'\}$, where each $h_{\mathbf{x},\mathbf{x}'}$ is given by (2). If there were such a learner, then by sampling uniformly at random $\ell = \mathrm{poly}(n)$ keys $\{\mathbf{k}_i\}_{i=1}^{\ell}$, and computing $[F_{\mathbf{k}_i}(\mathbf{x}), F_{\mathbf{k}_i}(\mathbf{x}')]$, we could generate a labeled training sample of pairs $([F_{\mathbf{k}_i}(\mathbf{x}), F_{\mathbf{k}_i}(\mathbf{x}')], \mathbf{k}_i)$, which should suffice for our learner to figure out an (ϵ, δ) approximation (in PAC sense) of $h_{\mathbf{x},\mathbf{x}'}$, which by Theorem 1 would result in a polynomial time distinguisher that contradicts Assumption 1.

Proof of Theorem 1

Fix arbitrary distinct $\mathbf{x}, \mathbf{x}' \in \{0,1\}^n$, and let $h_{\mathbf{x},\mathbf{x}'}$ be defined by (2). Consider Algorithm 1, which we denote $D_{\mathbf{x},\mathbf{x}'}(\mathbf{y}, \mathbf{y}')$ for brevity. Randomly pick $\mathbf{k}$ from a uniform distribution over $\{0,1\}^m$. Feeding $F_{\mathbf{k}}(\mathbf{x})$, $F_{\mathbf{k}}(\mathbf{x}')$ as input to $D_{\mathbf{x},\mathbf{x}'}$, Line 1 produces $\boldsymbol{\kappa}$ such that $F_{\boldsymbol{\kappa}}(\mathbf{x}) = F_{\mathbf{k}}(\mathbf{x})$ and $F_{\boldsymbol{\kappa}}(\mathbf{x}') = F_{\mathbf{k}}(\mathbf{x}')$. Thus Line 2 gives us

$$\boldsymbol{\xi} \leftarrow F_{\boldsymbol{\kappa}}^{-1}(F_{\mathbf{k}}(\mathbf{x})) = F_{\boldsymbol{\kappa}}^{-1}(F_{\boldsymbol{\kappa}}(\mathbf{x})) = \mathbf{x},$$
$$\boldsymbol{\xi}' \leftarrow F_{\boldsymbol{\kappa}}^{-1}(F_{\mathbf{k}}(\mathbf{x}')) = F_{\boldsymbol{\kappa}}^{-1}(F_{\boldsymbol{\kappa}}(\mathbf{x}')) = \mathbf{x}',$$

and the algorithm outputs 1. Hence

$$\Pr_{\mathbf{k}}[D_{\mathbf{x},\mathbf{x}'}(F_{\mathbf{k}}(\mathbf{x}), F_{\mathbf{k}}(\mathbf{x}')) = 1] = 1 \quad (3)$$

Now randomly pick n-bit strings $\mathbf{u}$, $\mathbf{u}'$ without replacement from the uniform distribution over $\{0,1\}^n$ and feed

Algorithm 1: Distinguisher

Input: $\mathbf{y}, \mathbf{y}' \in \{0,1\}^n$ s.t. $\mathbf{y} \neq \mathbf{y}'$
Parameter: $\mathbf{x}, \mathbf{x}' \in \{0,1\}^n$ s.t. $\mathbf{x} \neq \mathbf{x}'$

$\boldsymbol{\kappa} \leftarrow h_{\mathbf{x},\mathbf{x}'}(\mathbf{y}, \mathbf{y}')$
$\boldsymbol{\xi} \leftarrow F_{\boldsymbol{\kappa}}^{-1}(\mathbf{y}), \quad \boldsymbol{\xi}' \leftarrow F_{\boldsymbol{\kappa}}^{-1}(\mathbf{y}')$
if $\boldsymbol{\xi} = \mathbf{x}$ **and** $\boldsymbol{\xi}' = \mathbf{x}'$ **then**
  return 1.
else
  return 0.
end if

them as input to $D_{\mathbf{x},\mathbf{x}'}$. Intuitively, in this case the event $A := \{h_{\mathbf{x},\mathbf{x}'}(\mathbf{u},\mathbf{u}') \neq \mathbf{0}\}$ has low probability. Let us upperbound the latter using the union bound:

$$\begin{aligned}\Pr[A] &= \Pr_{\mathbf{u},\mathbf{u}'}[h_{\mathbf{x},\mathbf{x}'}(\mathbf{u},\mathbf{u}') \neq \mathbf{0}] \\ &= \Pr_{\mathbf{u},\mathbf{u}'}[\exists \boldsymbol{\kappa} \neq \mathbf{0} : [F_{\boldsymbol{\kappa}}(\mathbf{x}), F_{\boldsymbol{\kappa}}(\mathbf{x}')] = [\mathbf{u},\mathbf{u}']] \\ &= \Pr_{\mathbf{u},\mathbf{u}'}\left[\bigcup_{\boldsymbol{\kappa}\neq\mathbf{0}} [F_{\boldsymbol{\kappa}}(\mathbf{x}), F_{\boldsymbol{\kappa}}(\mathbf{x}')] = [\mathbf{u},\mathbf{u}'])\right] \\ &\leq \sum_{\boldsymbol{\kappa}\neq\mathbf{0}} \Pr_{\mathbf{u},\mathbf{u}'}[[F_{\boldsymbol{\kappa}}(\mathbf{x}), F_{\boldsymbol{\kappa}}(\mathbf{x}')] = [\mathbf{u},\mathbf{u}']] \qquad (4)\end{aligned}$$

Notice that $[F_{\boldsymbol{\kappa}}(\mathbf{x}), F_{\boldsymbol{\kappa}}(\mathbf{x}')]$ is a fixed $2n$-bit string, and the joint p.d.f. of $\mathbf{u}, \mathbf{u}'$ has the form

$$\Pr_{\mathbf{u},\mathbf{u}'}(\mathbf{u} = \boldsymbol{\upsilon}, \mathbf{u}' = \boldsymbol{\upsilon}') = \frac{1}{2^n(2^n-1)}, \qquad \boldsymbol{\upsilon} \neq \boldsymbol{\upsilon}'. \quad (5)$$

Combining (4) and (5), we have

$$\Pr_{\mathbf{u},\mathbf{u}'}[A] \leq \sum_{\boldsymbol{\kappa}\neq\mathbf{0}} \frac{1}{2^n(2^n-1)} = \frac{1}{2^n}. \quad (6)$$

When $h_{\mathbf{x},\mathbf{x}'}(\mathbf{u},\mathbf{u}') = \boldsymbol{\kappa} \neq \mathbf{0}$, we have $F_{\boldsymbol{\kappa}}^{-1}(\mathbf{u}) = \mathbf{x}$, $F_{\boldsymbol{\kappa}}^{-1}(\mathbf{u}') = \mathbf{x}'$, and thus we can write

$$\Pr_{\mathbf{u},\mathbf{u}'}[D_{\mathbf{x},\mathbf{x}'}(\mathbf{u},\mathbf{u}') = 1 \mid A] = 1 \quad (7)$$

Now we turn to the event when $h_{\mathbf{x},\mathbf{x}'}(\mathbf{u},\mathbf{u}')$ outputs the zero key. This happens if one of the following events occurs: $B := \{h_{\mathbf{x},\mathbf{x}'}(\mathbf{u},\mathbf{u}') = \mathbf{0}\} \cap \{[F_{\mathbf{0}}(\mathbf{x}), F_{\mathbf{0}}(\mathbf{x}')] = [\mathbf{u},\mathbf{u}']\}$, or $C := \{h_{\mathbf{x},\mathbf{x}'}(\mathbf{u},\mathbf{u}') = \mathbf{0}\} \cap \{\nexists \boldsymbol{\kappa} \in \{0,1\}^n : [F_{\boldsymbol{\kappa}}(\mathbf{x}), F_{\boldsymbol{\kappa}}(\mathbf{x}')] = [\mathbf{u},\mathbf{u}']\}$.

By Eq. (5), we have

$$\Pr_{\mathbf{u},\mathbf{u}'}[B] \leq \Pr[[F_{\mathbf{0}}(\mathbf{x}), F_{\mathbf{0}}(\mathbf{x}')] = [\mathbf{u},\mathbf{u}']] = \frac{1}{2^n(2^n-1)}. \quad (8)$$

In the event B, we have $[F_{\mathbf{0}}^{-1}(\mathbf{u}), F_{\mathbf{0}}^{-1}(\mathbf{u}')] = [\mathbf{x},\mathbf{x}']$, and thus Alg. 1 produces 1 in this case, i.e.

$$\Pr_{\mathbf{u},\mathbf{u}'}[D_{\mathbf{x},\mathbf{x}'}(\mathbf{u},\mathbf{u}') = 1 \mid B] = 1. \quad (9)$$

In the event C, $h_S(\mathbf{u},\mathbf{u}')$ outputs $\mathbf{0}$ which is *not* a key that maps $[\mathbf{x},\mathbf{x}']$ to $[\mathbf{u},\mathbf{u}']$ under AES, and we have

$$\begin{aligned}&\Pr_{\mathbf{u},\mathbf{u}'}[D_{\mathbf{x},\mathbf{x}'}(\mathbf{u},\mathbf{u}') = 1 \mid C] \\ &= \Pr_{\mathbf{u},\mathbf{u}'}[F_{\mathbf{0}}^{-1}(\mathbf{u}) = \mathbf{x}, F_{\mathbf{0}}^{-1}(\mathbf{u}') = \mathbf{x}' \mid C] \\ &= \Pr_{\mathbf{u},\mathbf{u}'}[\mathbf{u} = F_{\mathbf{0}}(\mathbf{x}), \mathbf{u}' = F_{\mathbf{0}}(\mathbf{x}') \mid C] = 0 \qquad (10)\end{aligned}$$

Now we can decompose the probability that $D_{\mathbf{x},\mathbf{x}'}(\mathbf{u},\mathbf{u}')$ outputs 1 as follows:

$$\begin{aligned}&\Pr_{\mathbf{u},\mathbf{u}'}[D_{\mathbf{x},\mathbf{x}'}(\mathbf{u},\mathbf{u}') = 1] \\ &= \Pr_{\mathbf{u},\mathbf{u}'}[D_{\mathbf{x},\mathbf{x}'}(\mathbf{u},\mathbf{u}') = 1 \mid A] \cdot \Pr_{\mathbf{u},\mathbf{u}'}[A] \\ &+ \Pr_{\mathbf{u},\mathbf{u}'}[D_{\mathbf{x},\mathbf{x}'}(\mathbf{u},\mathbf{u}') = 1 \mid B] \cdot \Pr_{\mathbf{u},\mathbf{u}'}[B] \\ &+ \Pr_{\mathbf{u},\mathbf{u}'}[D_{\mathbf{x},\mathbf{x}'}(\mathbf{u},\mathbf{u}') = 1 \mid C] \cdot \Pr_{\mathbf{u},\mathbf{u}'}[C]. \qquad (11)\end{aligned}$$

Plugging (6), (7), (9), (8), (10) into (11), we have

$$\begin{aligned}&\Pr_{\mathbf{u},\mathbf{u}'}[D_{\mathbf{x},\mathbf{x}'}(\mathbf{u},\mathbf{u}') = 1] \leq 1 \cdot \frac{1}{2^n} + 1 \cdot \frac{1}{2^n(2^n-1)} + 0 \\ &= \frac{2^n - 1 + 1}{2^n(2^n-1)} = \frac{1}{2^n - 1}. \qquad (12)\end{aligned}$$

Finally, combining (3) and (12), we get

$$\begin{aligned}&\left|\Pr_{\mathbf{k}}[D_{\mathbf{x},\mathbf{x}'}(F_{\mathbf{k}}(\mathbf{x}), F_{\mathbf{k}}(\mathbf{x}')) = 1] - \Pr_{\mathbf{u},\mathbf{u}'}[D_{\mathbf{x},\mathbf{x}'}(\mathbf{u},\mathbf{u}') = 1]\right| \\ &\geq 1 - \frac{1}{2^n - 1},\end{aligned}$$

which means that Alg. 1 is a $\mathrm{poly}(n)$-time distinguisher between the distribution of $[F_{\mathbf{k}}(\mathbf{x}), F_{\mathbf{k}}(\mathbf{x}')]$ and the distribution of two distinct random n-bit strings, and this concludes the proof.

Conclusion

Inspired by the recent result of Liu et al. (2021) on statistical closeness of AES to pairwise independence under randomness of all round keys, we make a relevant assumption on *computational* closeness of AES to pairwise independence under randomness of just the initial key. Under this assumption we prove the resistance of AES against attacks based on machine learning algorithms that aim to recover AES key from pairs of ciphertexts. Our proof is elementary and uses only college-level probability. We argue that Assumption 1 is realistic and is a reasonable alternative to common cryptographic assumptions such as existence of a one-way function.

Acknowledgements

This work was supported by the Program of Targeted Funding "Economy of the Future" #0054/ПЦФ-НС-19.

References

Liu, T.; Tessaro, S.; and Vaikuntanathan, V. 2021. The t-wise Independence of Substitution-Permutation Networks. In Malkin, T.; and Peikert, C., eds., *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part IV*, volume 12828 of *Lecture Notes in Computer Science*, 454–483. Springer.