# Machine-Checked Proofs for AES: High-Assurance Security

**v 1.0**

**Ji, Yong-hyeon**

(`hacker3740@kookmin.ac.kr`)

**Department of Information Security, Cryptology, and Mathematics**

College of Science and Technology

Kookmin University

December 25, 2024

## File Structure

- Machine-checked Proofs for AES
  - build/
    - aeslib/
    - example/
  - extraction/
    - aeslib/
    - example/
  - proof/
    - aeslib/
      - `AES_ct_proof.ec`
      - `AES_proof.ec`
      - `AES_spec.ec`
    - example/
      - `NbAESEnc_ct_proof.ec`
      - `NbAESEnc_mem_ct_proof.ec`
      - `NbAESEnc_mem_proof.ec`
      - `NbAESEnc_proof.ec`
      - `NbAESEnc_spec.ec`
      - ...
  - src/
    - aeslib/
      - `aes.jazz`
    - example/
      - `NbAESEnc.jazz`
      - `NbAESEnc_mem.jazz`

## Copyright



## Changelog

| | | |
|---|---|---|
| v1.0 | 2024-12-24 | Initial release: |

# Contents

# 1 Preliminaries

## 1.1 Cryptosystem and Encryption Scheme

**Cryptosystem**

**Definition 1.** A **cryptosystem** is a five-tuple

$$(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D}),$$

where

(i) $\boxed{\mathcal{P}}$ is a finite set of plaintexts[a].

(ii) $\boxed{\mathcal{C}}$ is a finite set of ciphertexts[b].

(iii) $\boxed{\mathcal{K}}$ is a finite set of possible keys[c].

(iv) $\boxed{\mathcal{E} : \mathcal{K} \times \mathcal{P} \to \mathcal{C}}$ is a deterministic function that maps a key $k \in \mathcal{K}$ and a plaintext $p \in \mathcal{P}$ to a ciphertext $c \in \mathcal{C}$. Formally:

$$\begin{array}{rcccl} \mathcal{E} & : & \mathcal{K} \times \mathcal{P} & \longrightarrow & \mathcal{C} \\ & & (k, p) & \longmapsto & c \end{array}.$$

(v) $\boxed{\mathcal{D} : \mathcal{K} \times \mathcal{C} \to \mathcal{P}}$ is a deterministic function that maps a key $k \in \mathcal{K}$ and a ciphertext $c \in \mathcal{C}$ to a ciphertext $p \in \mathcal{P}$. Formally:

$$\begin{array}{rcccl} \mathcal{D} & : & \mathcal{K} \times \mathcal{C} & \longrightarrow & \mathcal{P} \\ & & (k, c) & \longmapsto & p \end{array}.$$

[a] These are the possible inputs to the encryption algorithm and typically represent meaningful data to be protected.

[b] These are the encrypted outputs of the encryption algorithm corresponding to plaintexts in $\mathcal{P}$.

[c] Each key $k \in \mathcal{K}$ determines a specific encryption and decryption function.

**Remark 1** (Correctness Property)**.** For every key $k \in \mathcal{K}$ and every plaintext $p \in \mathcal{P}$, the decryption function is the inverse of the encryption function. That is:

$$\mathcal{D}(k, \mathcal{E}(k, p)) = p.$$

**Remark 2** (Security)**.** The security of the cryptosystem is defined with respect to a particular adversarial model. Informally, a cryptosystem is secure if an adversary with limited computational resources cannot distinguish between the ciphertexts of any two plaintexts, even if they know the encryption algorithm but do not know the key.

### Encryption Scheme

**Definition 2.** An **encryption scheme** is a three-tuple

$$\Pi := (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}).$$

where

(i) KeyGen is a probabilistic algorithm that ouputs a key $k \in \mathcal{K}$, where $\mathcal{K}$ is the key space. Formally:

$$\boxed{\mathsf{KeyGen} : \{0,1\}^* \to \mathcal{K}},$$

where $\{0,1\}^*$ is the set of binary strings of arbitrary length (representing randomness or input seed). The ouput $k$ is uniformly distributed over $\mathcal{K}$.

(ii) Enc is a (possibly probabilistic) algorithm that takes a key $k \in \mathcal{K}$ and a message $p \in \mathcal{M}$ (message space) and outpus a ciphertext $c \in \mathcal{C}$ (ciphertext sapce). Formally:

$$\boxed{\mathsf{Enc} : \mathcal{K} \times \mathcal{M} \times \{0,1\}^* \to \mathcal{C}}.$$

The algorithm may use randomness (from $\{0,1\}^*$) to ensure that repeated encryptions of the same message $m \in \mathcal{M}$ under the same key $k \in \mathcal{K}$ yield different ciphertexts $c$.

(iii) Dec is a deterministic algorithms that takes a key $k \in \mathcal{K}$ and a ciphertext $c \in \mathcal{C}$ and ouputs the corresponding message $m \in \mathcal{M}$. Formally:

$$\boxed{\mathsf{Dec} : \mathcal{K} \times \mathcal{C} \to \mathcal{M}}.$$

**Remark 3** (Correctness Property)**.** For every $k \in \mathcal{K}$, $m \in \mathcal{M}$, and $c \in \mathcal{C}$, the scheme must satisfy

$$\mathsf{Dec}(k, \mathsf{Enc}(k, m; r)) = m$$

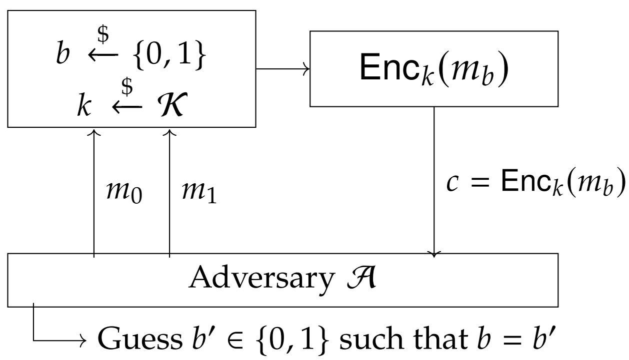
where $r$ represents the random bits used by Enc.

**Remark 4** (Security)**.** The security of an encryption scheme depends on the adversarial model. For **semantic security**, an encryption scheme must satisfy the following:

> "Given a ciphertext $c$, no computationally bounded adversary can distinguish between encryptions of any two messages $m_0, m_1$, even if they are chosen adaptively by the adversary."

**Example 1** (IND-CPA)**.**
The **indistinguishability under chosen plaintext attack (IND-CPA)** model:

1. The adversary chooses two messages $m_0, m_1$.
2. A random bit $b \in \{0, 1\}$ is chosen, and the ciphertext $c = \mathsf{Enc}(k, m_b)$ is provided to the adversary.
3. The adversary outputs a guess $b' \in \{0, 1\}$ .




The scheme is secure if the adversary's advantage is negligible:

$$\mathrm{Adv}_{\Pi}^{\text{IND-CPA}}(\mathcal{A}) := \left| \Pr[b' = b] - \frac{1}{2} \right| \leq \mathrm{negl}(\lambda),$$

where $\lambda$ is the security parameter.

## 1.2 Perfect Security

**Perfect Security of an Encryption Scheme**

**Definition 3.** An encryption scheme $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ is **perfect security** if, for every $m \in \mathcal{M}$, $c \in \mathcal{C}$, and $k \in \mathcal{K}$ such that $\mathsf{Enc}(k, m) = c$, the following holds:

(i) **Ciphertext Independence**:

$$\Pr[M = m \mid C = c] = \Pr[M = m],$$

where

- $M$ is the random variable representing the plaintext.
- $C$ is the random variable representing the ciphertext.

(ii) **Key Uniformity**: The key $K$ must satisfy:

$$\Pr[\mathsf{Enc}(K, m) = c] = \Pr[C = c],$$

for all $m \in \mathcal{M}$, $c \in \mathcal{C}$, and uniformly random $K$.

**Remark 5.**

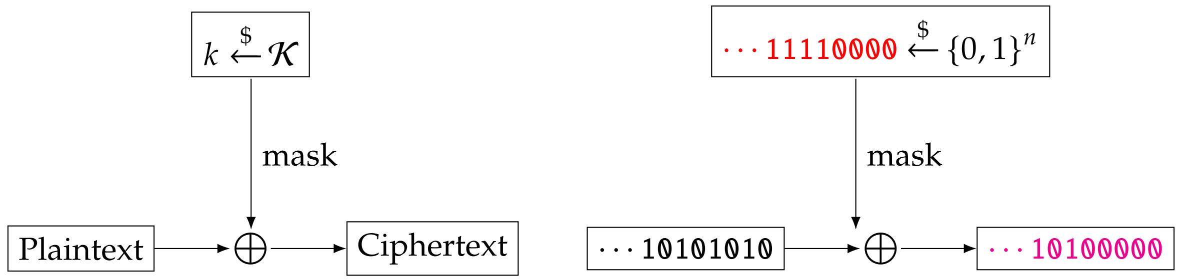
$$\Pr[M = m \mid C = c] = \Pr[M = m] \iff \Pr[C = c \mid M = m] = \Pr[C = c]$$

**Example 2** (One-Time Pad)**.** The one-time pad encryption scheme is perfect security.

- $\mathcal{M} = \mathcal{C} = \mathcal{K} = \{0, 1\}^n$;
- $\mathsf{Enc}(k, m) = m \oplus k$, where $\oplus$ is bitwise XOR;
- $\mathsf{Dec}(k, c) = c \oplus k$.

We must show that $\Pr[C = c \mid M = m] = \Pr[C = c]$. For $m \in \mathcal{M}$ and $c \in \mathcal{C}$,

$$\Pr[C = c \mid M = m] = \sum_{k \in \mathcal{K}} \Pr[K = k]$$




**Theorem 1.** *The one-time pad encryption scheme is perfectly secret.*

*Proof.*

$$\begin{aligned}
\Pr[C = c \mid M = m] = \Pr[c = \mathsf{Enc}(K, m)] &= \Pr[c = m \oplus K] \\
&= \Pr[K = m \oplus c] \\
&= 2^{-n} \quad \text{if } K \xleftarrow{\$} \mathcal{K} = \{0,1\}^n
\end{aligned}$$

Fix any distribution over $\mathcal{M}$. For any $c \in \mathcal{C}$, we have

$$\begin{aligned}
\Pr[C = c] &= \sum_{m \in \mathcal{M}} \Pr[C = c \mid M = m] \cdot \Pr[M = m] \\
&= 2^{-n} \cdot \Pr[M = m]
\end{aligned}$$

By Bayes' Theorem, we obtain

$$\begin{aligned}
\Pr[M = m \mid C = c] &= \frac{\Pr[C = c \mid M = m] \cdot \Pr[M = m]}{\Pr[C = c]} \\
&= \frac{2^{-n} \cdot \Pr[M = m]}{2^{-n}} \\
&= \Pr[M = m].
\end{aligned}$$

□

# 2 Block Cipher

## 2.1 Formal Definition

**Pseudo-Random Permutation (PRP)**

**Definition 4.** Consider a mapping

$$f : \{0,1\}^m \times \{0,1\}^n \to \{0,1\}^n, \quad \text{i.e.,} \quad f : \{0,1\}^m \to \mathrm{Perm}(\{0,1\}^n).$$

Let

$$\mathcal{F} := \left\{f_k\right\}_{k\in\{0,1\}^m} \text{ where } f_k \in \mathrm{Perm}(\{0,1\}^n)$$

be a family of permutations, where $n$ is the block length and $m$ is key length. The family $\mathcal{F}$ is said to be a **pseudo-random permutation** (PRP) if it satisfies the following properties:

(i) **Permutation Property**: For every $k \in \{0,1\}^m$, the function $f_k : \{0,1\}^n \to \{0,1\}^n$ is a bijection. That is,

$$f_k^{-1}(f_k(x)) = x \quad \text{and} \quad f_k(f_k^{-1}(y)) = y, \quad \forall x, y \in \{0,1\}^n.$$

(ii) **Indistinguishability from Random Permutation**: Define the advantage of an adversary $\mathcal{A}$ as

$$\mathrm{Adv}_{\mathcal{F}}^{\mathrm{PRP}}(\mathcal{A}) := \left|\Pr[\mathcal{A}^{f_k, f_k^{-1}} = 1] - \Pr[\mathcal{A}^{P, P^{-1}} = 1]\right|,$$

where

- $k \xleftarrow{\$} \{0,1\}^m$ (uniformly sampled)
- $P \xleftarrow{\$} \mathrm{Perm}\left(\{0,1\}^n\right)$ (a uniformly random permutation)
- $\mathcal{A}^{f_k, f_k^{-1}}$ is the adversary $\mathcal{A}$ interacting with the oracle for $f_k$ and $f_k^{-1}$, while
- $\mathcal{A}^{P, P^{-1}}$ is the adversary $\mathcal{A}$ interacting with the oracle for $P$ and $P^{-1}$.

(iii) **Efficiency**: The functions $f_k$ and $f_k^{-1}$ must be efficiently computable, meaning there exits deterministic algorithms that compute $f_k(x)$ and $f_k^{-1}(y)$ in time polynomial in $n$ and $m$.

**Remark 6** (Secure PRP)**.** The family $\mathcal{F}$ is a **secure PRP** if, for all probabilistic polynomial-time (PPT) adversary $\mathcal{A}$, the advantage $\mathrm{Adv}_{\mathcal{F}}^{\mathrm{PRP}}(\mathcal{A})$ is negligible in $m$, i.e.,

$$\mathrm{Adv}_{\mathcal{F}}^{\mathrm{PRP}}(\mathcal{A}) \leq \mathrm{negl}(m).$$

**Block Cipher**

**Definition 5.** A **block cipher** is defined as a family of functions

$$\{E_k : \{0,1\}^n \to \{0,1\}^n\}_{k \in \{0,1\}^k},$$

where:

- Each function $E_k$ is a bijection over $\{0,1\}^n$, meaning there exists a corresponding decryption function $D_k$ such that

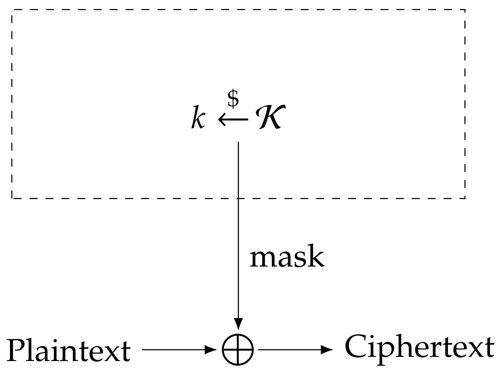
$$D_k(E_k(x)) = x, \quad \forall x \in \{0,1\}^n.$$

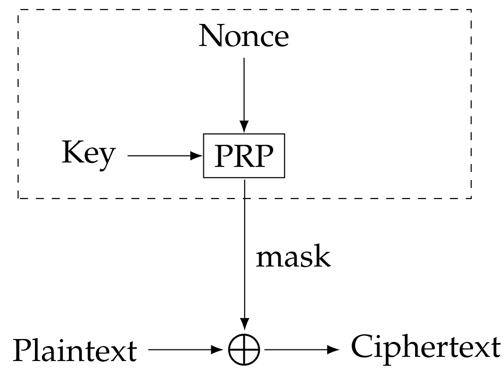
- The family of functions satisfies the *secure pseudo-random permutation (PRP)* property: for a uniformly chosen $k \in \{0,1\}^k$, no computationally bounded adversary can distinguish $E_k$ from a truly random permutation $P : \{0,1\}^n \to \{0,1\}^n$ with non-negligible advantage.

- The block cipher operates on fixed-length input blocks of size $n$, and the key $k$ is sampled uniformly from the key space $\{0,1\}^k$.

In summary, a block cipher is a deterministic, key-dependent, reversible function family over fixed-length input blocks, which achieves the properties of a secure pseudo-random permutation when the key is secret.

# 3 Machine-checked Proofs for AES




- The **one-time pad (OTP) encryption scheme** is a cryptographic construct that achieves perfect security.

$$\Pi_{\mathrm{OTP}} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}),$$

where

(i) $\mathsf{KeyGen} : \{0,1\}^n \to \{0,1\}^n\,, \quad k \sim \mathrm{Uniform}(\{0,1\}^n);$

(ii)

$$\begin{array}{rcrcl} \mathsf{Enc} & : & \mathcal{K} \times \mathcal{M} & \longrightarrow & \mathcal{C} \\ & & (k, m) & \longmapsto & c = k \oplus m \end{array}$$

$$\begin{array}{rcrcl} \mathsf{Enc} & : & \mathcal{K} & \longrightarrow & \mathcal{C}^{\mathcal{M}} \\ & & k & \longmapsto & c = \mathsf{Enc}_k(m) = k \oplus m \end{array} \quad \text{where} \quad \begin{array}{rcrcl} \mathsf{Enc}_k & : & \mathcal{M} & \longrightarrow & \mathcal{C} \\ & & m & \longmapsto & c = k \oplus m \end{array}$$

- A **nonce-based PRP encryption scheme** is a cryptographic construct where a nonce (number used once) is incorporated to ensure unique ciphertexts for the same plaintext under the same key.

$$\Pi_{\mathcal{N}-\mathrm{PRP}} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}),$$

where

(i) $\mathsf{KeyGen} : \{0,1\}^* \to \mathcal{K};$

(ii)

$$\begin{array}{rcrcl} \mathsf{Enc} & : & \mathcal{K} \times \mathcal{N} \times \mathcal{M} & \longrightarrow & \mathcal{C} \\ & & (k, m, m) & \longmapsto & c = \mathsf{Enc}_k(n) \oplus m \end{array}$$

$$\begin{array}{rcrcl} \mathsf{Enc} & : & \mathcal{K}\times\mathcal{N}\times\mathcal{M} & \longrightarrow & \mathcal{C} \\ & & (k,n,m) & \longmapsto & c = \mathsf{Enc}_k(n)\oplus m \end{array}$$

$$\begin{array}{rcrcl} \mathsf{Enc} & : & \mathcal{K}\times\mathcal{N} & \longrightarrow & \mathcal{C}^{\mathcal{M}} \\ & & (k,n) & \longmapsto & c = \mathrm{Xor}(k,n) = k\oplus m \end{array} \text{ where } \begin{array}{rcrcl} \mathsf{Enc}_k & : & \mathcal{M} & \longrightarrow & \mathcal{C} \\ & & m & \longmapsto & c = k\oplus m \end{array}$$

$$\begin{array}{rcrcl} \mathsf{Enc} & : & \mathcal{K} & \longrightarrow & [\mathcal{N}\to[\mathcal{M}\to\mathcal{C}]] \\ & & k & \longmapsto & c = \mathsf{Enc}_k(n)\oplus m \end{array} \text{ where } \begin{array}{rcrcl} \mathsf{Enc}_k & : & \mathcal{N} & \longrightarrow & [\mathcal{M}\to\mathcal{C}] \\ & & m & \longmapsto & c = k\oplus m \end{array}$$

# 4 Tutorials

## 4.1 Introduction and Goals of the EasyCrypt and Jasmine

## 4.2 Exploring Jasmine Language and Compiler

## 4.3 Connecting Jasmine with EasyCrypt for Verification

## 4.4 Exploring EasyCrypt and Jasmine Integration

## 4.5 Examining Jasmine and EasyCrypt Verification Process

## 4.6 Enhancing Cryptographic Verification with Jasmine and EasyCrypt

# References

[1] Jonathan, Katz. *Introduction to Modern Cryptography, Second Edition.*, n.d.

[2] Smart, Nigel P. *Cryptography Made Simple. Information Security and Cryptography*. Cham: Springer International Publishing, 2016. https://doi.org/10.1007/978-3-319-21936-3.