EasyAES API Reference Book

Wayne Chiu

Information Security Laboratory, National Donghwa University

Fundamental Computer Science Laboratory, University of Aizu

2018/05/14 Version 1

System Requirements

Hardware Requirement:

Java is architecture independent.

Software Requirements:

Any platform equipped with Java Virtual Machine, please note that JVM version must be higher than JRE 8. The code also has been tested on JRE9 and JRE10 environment.

Development Environment of EasyAES:

OS: Windows Server 2016
IDE: Eclipse 4.8.3a (Oxygen 3A)
Java Runtime Environment: JDK 10

If you like to import it directly into Eclipse:

Please be sure that:

Eclipse Version >= 4.8.3A (Version 20180405-1200)
Compiler compliance level: 10

Disclaimer

This project is based on Mr. Codahale's AES-GCM-SIV project; I only perform some slight fixes for JRE 10 and provide an interface with easy access. The benchmark program, which originally included in the project, has been removed.

For more information, please check on Codahale's github project page and IETF page.

Getting Started with EasyAES

EasyAES is an easy-to-use AES-GCM-SIV Java implements. Before using EasyAES, you should be able to understand the basic elements for AES-GCM-SIV:

Key: AES Key

Nonce: Nonce

Credential Data: AES-GCM-SIV equipped AEAD (Authenticated Encryption with Additional Data), this mostly considered as kind of shared secret. [RFC5116]

Any encryption and decryption process relies on these three elements, if any elements differ between encryption process and decryption process, the decryption process would failed.

AES-GCM-SIV only supports two key Size: 128 bits and 256 bits, the nonce size is 96 bits. The length of credential data is not limited. That is:

Key: [16 bytes or 32 bytes long]

Nonce: [12 bytes long]

Credential data: [No limitation]

Please at least read the introduction part of the IETF document of AES-GCM-SIV

Function

- Constructors
  - EasyAES (byte[] key, byte[] nonce, byte[] cdata)
  - EasyAES (byte[] key, byte[] nonce)
  - EasyAES (int keySize)
  - EasyAES ()
- Functions (Sets and Gets)
  - byte[] getKey ()
  - byte[] getNonce ()
  - byte[] getCdata ()
  - void setKey (byte[] key)
  - void setNonce (byte[] nonce)
  - void setCdata (byte[] cdata)
- Functions (Encryption and Decryption)
  - SIV_encrypt (byte[] data)
  - SIV_decrypt (byte[] sivCipher)
  - GCM_encrypt (byte[] data)
  - GCM_decrypt (byte[] gcmCipher)
- Example
  - Encryption and decryption demo
  - Encryption and decryption demo with specified param.

Constructor

Name:	EasyAES (*byte[] key, byte[] nonce, byte[] cdata*)
Caption:	Construct the AES engine with “key”, “nonce”, “credential data” param. Please be advised that: ● Key should be 16 or 32 bytes long ● Nonce should be 12 bytes long
Example:	EasyAES eaes = new EasyAES(key, nonce, cdata)

Name:	EasyAES (*byte[] key, byte[] nonce*)
Caption:	Construct the AES engine with “key”, “nonce” param. Since no “credential data” specified, SIV functions are disabled. Please be advised that: ● Key should be 16 or 32 bytes long ● Nonce should be 12 bytes long
Example:	EasyAES eaes = new EasyAES(key, nonce)

Name:	EasyAES ()
Caption:	Construct the AES engine with no param.
Example:	EasyAES eaes = new EasyAES()

Name:	EasyAES (*int keySize*)
Caption:	Construct the AES engine with desired keySize. The key, nonce, credential data is generated automatically by SecureRandom. Please be advised that, there are only two keySize supported: both 128 and 256.
Example:	EasyAES eaes = new EasyAES(128)

Functions (Gets)

Name:	Return:
getKey ()	byte[] key, null if empty
Input Variables:	
N/A	
Caption:	
Return the key using by the AES engine.	
Example:	
byte[] key = EasyAES.getKey()	

Name:	Return:
getNonce ()	byte[] nonce, null if empty
Input Variables:	
N/A	
Caption:	
Return the nonce using by the AES engine.	
Example:	
byte[] nonce = EasyAES.getNonce()	

Name:	Return:
getCdata ()	byte[] cdata, null if empty
Input Variables:	
N/A	
Caption:	
Return the credential data using by the AES engine.	
Example:	
byte[] cdata = EasyAES.getCdata()	

Function (Sets)

Name:	Return:
setKey(byte[] key)	N/A
Input Variables:	
byte[] key	
Caption:	
Change the key using by the AES engine	
Example:	
EasyAES.setKey(key)	

Name:	Return:
setNonce(byte[] nonce)	N/A
Input Variables:	
byte[] nonce	
Caption:	
Change the nonce using by the AES engine	
Example:	
EasyAES.setNonce(nonce)	

Name:	Return:
setCdata(byte[] cdata)	N/A
Input Variables:	
byte[] cdata	
Caption:	
Change the Credential Data using by the AES engine	
Example:	
EasyAES.setCdata(cdata)	

Functions (Encryption and Decryption)

Name:	Return:
SIV_encrypt (byte[] data)	byte[] sivCipher
Input Variables:	
byte[] data	
Caption:	
Using AES engine perform encryption on plaintext “byte[] data”, which returns byte[] sivCipher as AES-GCM-SIV encrypted cipher. If you are considering using SIV_encrypt function, please ensure that key, nonce, credential data in the AES engine are set up correctly.	
Example:	
byte[] cipher = EasyAES.SIV_encrypt(plaintext)	

Name:	Return:
SIV_decrypt (byte[] sivCipher)	byte[] plainText
Input Variables:	
byte[] sivCipher	
Caption:	
Using AES engine perform decryption on cipher “byte[] sivCipher”, which returns byte[] plainText as decrypted result. If you are considering using SIV_encrypt function, please ensure that key, nonce, credential data in the AES engine are set up correctly.	
Example:	
byte[] plainText = EasyAES.SIV_decrypt(sivCipher)	

Name:	Return:
GCM_encrypt (byte[] data)	byte[] gcmCipher thorws Exception
Input Variables:	
byte[] data	
Caption:	
Using AES engine perform encryption on plaintext “byte[] data”, which returns byte[] gcmCipher as AES-GCM encrypted cipher. Credential data is not used in AES-GCM, please ensure that both key and nonce in the AES engine are set up correctly.	
Example:	
byte[] cipher = EasyAES.GCM_encrypt(plaintext)	

Name:	Return:
GCM_decrypt (byte[] gcmCipher)	byte[] plainText thorws Exception
Input Variables:	
byte[] gcmCipher	
Caption:	
Using AES engine perform decryption on cipher “byte[] gcmCipher”, which returns byte[] plainText as decrypted result. Credential data is not used in AES-GCM, please ensure that both key and nonce in the AES engine are set up correctly.	
Example:	
byte[] plainText = EasyAES.GCM_decrypt(gcmCipher)	

Example

Encryption and decryption demo

```
import easy.aesgcmsiv;

public class AutoTest {
	public static void main(String args[]){
		byte[] plainText = "plainText".getBytes();
		EasyAES eaes = new EasyAES(256);
		byte[] cipher = eaes.SIV_encrypt(plainText);
		byte[] decrypt = eaes.SIV_decrypt(cipher);
		System.out.println(new String(plainText));
		System.out.println(new String(decrypt));
	}
}
```

Encryption and decryption demo with specified param.

```
import java.security.SecureRandom;
import easy.aesgcmsiv;

public class ManTest {
	public static void main(String args[]) {
		SecureRandom sr = new SecureRandom();
		byte[] key = new byte[16];
		byte[] nonce = new byte[12];
		byte[] cdata = "CREDENTIAL".getBytes();
		byte[] plainText = "plainText".getBytes();
		sr.nextBytes(key);
		sr.nextBytes(nonce);

		EasyAES eaes = new EasyAES(key, nonce, cdata);
		byte[] cipher = eaes.SIV_encrypt(plainText);
		byte[] decrypt = eaes.SIV_decrypt(cipher);

		System.out.println(new String(plainText));
		System.out.println(new String(decrypt));
	}
}
```