CRIPTOGRAFIA - AES E MODOS DE OPERAÇÃO

Virgilius Santos <virgilius.santos@acad.pucrs.br>

Pontifícia Universidade Católica do Rio Grande do Sul – Escola Politécnica – Curso de Engenharia de Software
Av. Ipiranga, 6681 Prédio 32 Sala 505 – Bairro Partenon – CEP 90619-900 – Porto Alegre – RS

01 de maio de 2020

RESUMO

Este artigo irá descrever como conseguimos cifrar e decifrar usando a cifra de blocos AES, com os modos de operação CBC e CTR.

1 INTRODUÇÃO

O objetivo deste trabalho é aplicar os conhecimentos sobre a cifra de blocos AES e os modos de operação CBC e CTR.

Usando a linguagem Java foi criado um algoritmo que recebe um texto, a chave e o IV e:

- Se o texto estiver codificado, decodificamos para exibir o texto claro.

  Além disso codificamos o texto novamente para validar se texto cifrado gerado é igual ao texto cifrado recebido.

- Se o texto estiver decodificado, codificamos ele para mostrar o texto cifrado.

  Além disso decodificamos o texto novamente para validar se texto decifrado gerado é igual ao texto decifrado recebido.

2 ALGORITMO

O Algoritmo foi desenvolvido em torno uma função *processCrypt*.

2.1 ProcessCrypt

Essa função foi feita para codificar e decodificar o texto de acordo com seguintes parâmetros:

- *Title*: é o nome dado ao processo em execução.
- *Type*: define o tipo de decodificação será usada, o modo de operação e como o *padding* será fornecido. Neste projeto usamos: AES/CBC/PKCS5PADDING e AES/CTR/NoPadding.
- *Secret*: é chave de 16 bytes passada no formato de uma *string* de hexadecimais.
- *IV*: é um valor de 16 bytes que foi gerado de maneira aleatória no formato de uma string de hexadecimais.
- *Text*: é o valor que será codificado ou decodificado no formato de uma string de hexadecimais.
- *Mode*: pode ser de codificação *ENCRYPT_MODE*, ou decodificação *DECRYPT_MODE*.
- *MessageError*: mensagem exibida em caso de erro.

Também foi usada o *framework Cypher* que fornece a funcionalidade de uma cifra criptográfica para criptografia e descriptografia. Ele forma o núcleo da estrutura da *Java Cryptographic Extension (JCE)*.

Para realizar o processo é preciso transformar de hexadecimal em um *array* bytes o *Secret* e o *IV* e usar esses valores para, juntamente com o *Mode* e o *Type* configurar o *Cypher*.

Por fim o *Text* é convertido de hexadecimal em um *array bytes* para processa-lo usando o *Cypher* configurado. O *Cypher* retornar um *array* de *bytes*. Este *array* é convertido para uma *string* de hexadecimais. Vale lembrar que esse resultado pode ser o texto cifrado ou decodificado de acordo com o modo informado.

2.2 Conversão de Texto

Pare exibir o texto claro precisamos pegar a *string* em hexadecimal e converter para texto normal, isso foi feito usando a classe *HexToString* que possui as funções de converter:

- *string* hexadecimal para *string* normal ou *array* de *bytes*.
- *array* de *bytes* para *string* hexadecimal.
- *string* normal para *string* hexadecimal.

2.3 Entrada de dados

Os dados foram passados no formato *string* hexadecimal. A chave foi passada separada da mensagem, já a mensagem foi passada contendo o *IV* e o *Text*, como a cada 2 caracteres em hexadecimal temos um *byte*, pegamos os 32 primeiros caracteres para formar o *IV*, a seguir um exemplo

- *Key*: 140b41b22a29beb4061bda66b6747e14.
- *Ciphertext*:
  4ca00ff4c898d61e1edbf1800618fb2828a226d160dad07883d04e008a7897ee2e4b7465d5290d0c0e6c6822236e1daafb94ffe0c5da05d9476be028ad7c1d81.

  Sendo a parte em destaque (negrito e azul) o *IV* e o restante o *Text*.

3 RESULTADO

A seguir estão os resultados das 6 tarefas.

Tarefa 1, texto decodificado: *Basic CBC mode encryption needs padding*

Tarefa 2, texto decodificado: *Our implementation uses rand. IV*

Tarefa 3, texto decodificado: *CTR mode lets you build a stream cipher from a block cipher*.

Tarefa 4, texto decodificado: Always avoid the two time pad!

Tarefa 5, texto codificado:
266D7A7E13E6A4C5059DA6878E44D6C1D9FFE6260E6F8B8BBF3B32E35A9BA48B6609F1490962E4C41A87FD163945377A6B264DA16735A99E4DCE

Tarefa 6, texto codificado:
1C8C4995B0FF0A2EC2B111E08824947BAB0AC7B209D8026CBE7FC572A2FA0390DB08EC936DF8144AEB8C2018F5A15B1A36784356228B875924B689AA8E42962889519C1C99C5BC6EB0E155DECE937ABC4918C694CE2DB4A2CA8B021DEFC4D329D59A4CA3FADAA6AEE8C6DEDDFE45681F725DE4F460CAEFFBDDA8A4E76A4D71D7

As tarefas 5 e 6 foram passadas com texto claro, mas no formato hexadecimal. Como além de codificar, também é realizado o processo de decodificação, segue abaixo o texto claro:

Tarefa 5, texto decodificado: *This is a sentence to be encrypted using AES and CTR mode*.

Tarefa 6, texto decodificado: Next Thursday one of the best teams in the world will face a big challenge in the Libertadores da America Championship.

4 CONCLUSÃO

AES é a abreviação de *Advanced Encryption Standard* (Padrão Avançado de Criptografia). É uma codificação em bloco simétrica usada para criptografar dados confidenciais.

Em criptografia, um modo de operação é um algoritmo que usa uma cifra de bloco para fornecer um serviço de informação, tais como confidencialidade ou autenticidade. Um modo de operação descreve como aplicar repetidamente a operação do bloco único de uma cifra para transformar de forma segura quantidades de dados maiores do que um bloco.

Neste artigo usamos uma classe Java para codificar e decodificar textos usando AES e os modos de operação CBC e CTR.

Recebemos a chave, o IV e o texto codificado (tarefas 1, 2, 3 e 4) e decodificado (tarefas 5 e 6) e fizemos a conversão com sucesso conforme resultados apresentados.

5 REFERÊNCIA:

https://docs.oracle.com/javase/7/docs/api/javax/crypto/Cipher.html
https://www.cyclonis.com/pt/que-e-criptografia-aes-256/