Segurança de Sistemas – Trabalho 2

Igor S. Brehm[1]

[1] Escola Politécnica – Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
Caixa Postal 1429 – 90619-900 – Porto Alegre – RS – Brazil

igor.brehm@edu.pucrs.br

Abstract. *This paper aims to report to the reader the work done by this student in order to solve the problem scenario which was given by teacher Avelino Zorzo as the second evaluative task for the Security Systems discipline of the Software Engineering course at PUCRS university.*

Resumo. *Este relatório tem como objetivo relatar ao leitor o trabalho realizado por este aluno para solucionar o cenário problemático apresentado pelo professor Avelino Zorzo como segunda tarefa avaliadora da disciplina de Sistemas de Segurança do curso de Engenharia de Software da PUCRS.*

1. Cifrar e Decifrar com AES, modo CBC com PKCS5

Os 16 primeiros bytes, tanto na cifragem quanto na decifragem, são considerados como sendo o IV. Na cifragem o IV é gerado randomicamente e adicionado na frente da mensagem cifrada a ser enviada. Uma biblioteca/dependência externa foi utilizada no código e mais informações estão disponíveis sobre ela no arquivo Readme.

1.1. Resultados Obtidos

a) CBC key: 140b41b22a29beb4061bda66b6747e14
CBC Ciphertext:
4ca00ff4c898d61e1edbf1800618fb2828a226d160dad07883d04e008a7897ee2e4
b7465d5290d0c0e6c6822236e1daafb94ffe0c5da05d9476be028ad7c1d81
Resultado: Basic CBC mode encryption needs padding.

b) CBC key: 140b41b22a29beb4061bda66b6747e14
CBC Ciphertext:
5b68629feb8606f9a6667670b75b38a5b4832d0f26e1ab7da33249de7d4afc48e71
3ac646ace36e872ad5fb8a512428a6e21364b0c374df45503473c5242a253
Resultado: Our implementation uses rand. IV

c) CBC key: 140b41b22a29beb4061bda66b6747e14
CBC Plaintext:
4e6578742054687572736461792 06f6e65206f6620746865206265737420746561
6d7320696e2074686520776f726c642077696c6c206661636520612062696720
6368616c6c656e676520696e20746865204c6962657274617 46f7265732064612
0416d6572696361204368616d70696f6e736869702e
Resultado:
e4b9b5ffc0412348df37f4becd0f4127419326ce9401ee8da09e20e1197356a6587c
8978b0875abd1e0d69480c6134d17f5aedfcf2478dad5c5d6d641a7385e95dcacad
04576105a0371a2a0a83a2c7a9a5b14ec23e6c6ebcbe01d82dabf10d1a617ddf781
88964f59fb476b13c260b4f196b34abe24009a52045221f9f21634a87b9401002d7
c0b6832340425c1adfdc959ebaca765b08654c6da209752e649c2d1bbea52080f0b

46fe5cf6b365791eb4923dccfa64202eb096f8cf1321bbe96aec29fef05afd8960582
6b93ec117fd9a609951bc7d2e462880e50ecc2bc1d05ca4edd47b205fba94a6b658
de5999ac1c7a8495fde51e577f1eab3181cf197c

2. Cifrar e Decifrar com AES, modo CTR

Os 16 primeiros bytes, tanto na cifragem quanto na decifragem, foram novamente considerados como sendo o IV. Na cifragem o IV é gerado randomicamente e adicionado na frente da mensagem cifrada a ser enviada. Uma biblioteca/dependência externa foi utilizada no código e mais informações estão disponíveis sobre ela no arquivo Readme.

2.1. Resultados Obtidos

a) CTR key: 36f18357be4dbd77f050515c73fcf9f2
CTR Ciphertext:
69dda8455c7dd4254bf353b773304eec0ec7702330098ce7f7520d1cbbb20fc388d
1b0adb5054dbd7370849dbf0b88d393f252e764f1f5f7ad97ef79d59ce29f5f51eec
a32eabedd9afa9329
Resultado: CTR mode lets you build a stream cipher from a block cipher.

b) CTR key: 36f18357be4dbd77f050515c73fcf9f2
CTR Ciphertext:
770b80259ec33beb2561358a9f2dc617e46218c0a53cbeca695ae45faa8952aa0e3
11bde9d4e01726d3184c34451
Resultado: Always avoid the two time pad!

c) CTR key: 36f18357be4dbd77f050515c73fcf9f2
CTR Plaintext:
54686973206973206120 73656e74656e636520746f20626520656e63727970746
564207573696e67204145532061 6e6420435452206d6f64652e
Resultado:
0f416b123990ef705458a8d63d1a360fa94a7b03fafd6f6bd4b2f011a4394396fa90
8039e4e850b2b6d29587fcb0b129331c4cfca91e1b4cc711e9f722f27b20ff0aeec0
cf47244256aae5337123703116471f037863db5212b13a6066c9de7007cb5f34aed
61ac539a4bdd69b91447dc310950366c8271986a47ffc555d88d6d23886db

References

1. Java Code Geeks “AES Encryption and Decryption in Java(CBC Mode)”, https://www.javacodegeeks.com/2018/03/aes-encryption-and-decryption-in-javacbc-mode.html, Abril.