Projeto de Sistema de Arquivos Criptografados

Camila de Lima Martins, Vanderley Sousa da Silva Junior
Departamento Acadêmico de Ciência da Computação.
Tecnologia em Análise e Desenvolvimento de Sistemas.
Universidade Tecnológica Federal do Paraná
camilalm31@gmail.com, vanderley_1998@hotmail.com.br

Luís Gustavo Stelmastchuk Mandello, Vinicius Lopes Lievore
Departamento Acadêmico de Ciência da Computação
Engenharia de Computação
Universidade Tecnológica Federal do Paraná
luis.stelmastchuk, vinicius_lievore @hotmail.com

***Resumo*—Projeto Final da disciplina de Segurança e Auditoria de Sistemas, ministrada pelo Prof. Lucas Dias H. Sampaio na Universidade Tecnológica Federal do Paraná em Cornélio Procópio - PR**

Chaves*—*Criptografia, Web, Sistema, Segurança, HTTP, JavaScript, Sistema, Arquivos.

I. Introdução

Em sistemas computacionais, principalmente aqueles que envolvem troca de dados através da rede mundial de computadores, necessitam fortemente que os dados que estão sendo trafegados pela rede, sejam apenas visíveis pelo remetente e destinatário e não disponíveis para alguém mal intencionado. A partir deste viés, que foram criados protocolos de comunicação segura entre as pontas da rede (cliente-servidor), sendo um deles o protocolo HTTPS (Hyper Text Transfer Protocol Secure) desenvolvido pela empresa Netscape Communication em 1994, que é a implementação do protocolo HTTP com uma camada adicional de segurança que utiliza o protocolo SSL/TLS, permitindo assim que os dados sejam transmitidos por meio de uma conexão criptografada e validada a partir de certificados digitais. Tendo em vista as informações descritas anteriormente, iremos mostrar neste projeto, uma maneira de compartilhar arquivos entre cliente e servidor, utilizando um servidor HTTP (protocolo que não implementa camada adicional de criptografia sobre os dados) e garantir que os dados sejam criptografados, de tal forma, que apenas o cliente tenha acesso aos dados e envia-los a um servidor.

II. Proposta

Este trabalho terá como objetivo o desenvolvimento de um Sistema de Arquivos criptografados. O servidor utilizado para o trabalho terá o módulo HTTPS desativado e trabalhará somente com protocolo HTTP. Os arquivos devem ser previamente criptografados do lado do cliente, sendo que cada usuário possuirá uma chave de criptografia. A chave para cada usuário será gerada no servidor e salva no banco de dados em base64. O usuário e senha serão salvos como hash no banco de dados.

A. Tecnologia e Ferramentas

1) Criptografia AES (*Advanced Encryption Standard*): é uma cifra de bloco, que utiliza chave simétrica para criptografia. Será utilizado no projeto o modelo AES-256. O AES utiliza o Rijndael, este é um algoritmo de criptografia de blocos que trabalha com blocos de 128 bits e chaves de 128, 192 ou 256 bits. O Rijndael original foi desenvolvido para suportar tamanhos diferentes de blocos de dados e de chaves, porém, estes não são adotados na versão AES. O AES foi anunciado pelo National Institute of Standards and Technology (NIST) em 2001 e tornou-se um padrão efetivo em 2002.

2) phpseclib (PHP Secure Communications Library): É uma biblioteca PHP, que contém implementações de PHP puro com licença LGPL (Library GPL. Esta é uma variação da licença GPL que permite o desenvolvimento de programas de código aberto que contenham módulos proprietários) de inteiros de precisão arbitrária, é compatível com RSA, DES, 3DES, RC4, Rijndael, AES, SSH-1, SSH-2, e SFTP.

3) Criptografia RSA (Rivest-Shamir-Adleman): É um sistema de criptografia de chave pública amplamente utilizado para transmissão segura de dados. Neste sistema, a chave de encriptação é pública e é diferente da chave de decriptação que é secreta (privada). No RSA, essa assimetria é baseada na dificuldade prática da fatoração do produto de dois grandes números primos.

4) CryptoJS: Esta ferramenta é uma coleção de algoritmos criptográficos seguros, implementados em JavaScript usando as melhores práticas e padrões. Será utilizada para criptografar os dados no browser do usuário e utilizará a criptografia AES.

5) Apache: Servidor HTTP que será utilizado, como o servidor possui um módulo que garante a segurança dos dados transmitidos, utilizando HTTPS, o módulo em questão, *mod_ssl* será desativado para que utilize apenas o protocolo HTTP.

6) Gerenciamento de sessão: Para que seja garantido a troca segura da chave AES entre cliente-servidor, será criada uma sessão para cada autenticação de usuário e a chave será criptografada utilizando RSA, gerando assim, um conjunto de pares de chaves a cada sessão.

7) Ambiente: O ambiente no lado do cliente será implementado com uso de HTML, CSS e JavaScript. No lado do servidor o sistema será implementado utilizando PHP. A comunicação entre o Servidor e o Cliente será feita utilizando protocolo HTTP.

8) Metodologia de desenvolvimento Kanban: É um método de desenvolvimento enxuto para gerenciar melhor o trabalho em sistemas humanos. Visa gerenciar o trabalho equilibrando as demandas com a capacidade disponível e melhorar o tratamento dos gargalos no nível do sistema. O Kanban se concentra no cliente e no trabalho que atende às suas necessidades, em vez de atividades individuais. Possui seis práticas gerais: visualização, limitação do trabalho em andamento, gerenciamento de fluxo, explicitação de

políticas, uso de ciclos de feedback e evolução colaborativa ou experimental. Elas envolvem ver o trabalho e seu processo e melhorar o processo, mantendo e ampliando mudanças úteis e aprendendo, revertendo e amortecendo os ineficazes.

III. METODOLOGIA

Para que seja possível a implementação do sistema de arquivos, serão seguidos os seguintes passos de acordo com cada requisito a seguir:

1) Login do sistema: quando for solicitado pelo cliente acesso ao sistema de arquivos, o mesmo enviará ao servidor, utilizando criptografia AES, informações de login e senha e conferir com os dados do usuário no banco de dados. Caso as informações sejam válidas, o servidor retornará a chave para que o usuário faça uso.

2) Envio de arquivos: para que um arquivo seja enviado ao servidor, o cliente utilizará a chave que fora recebida anteriormente ao fazer login, irá criptografá-lo utilizando a ferramenta CryptoJS e enviará para o servidor.

3) Recebimento de arquivos: para que seja recebido um arquivo, o usuário escolherá por meio de uma lista, qual arquivo que gostaria de ter acesso. A partir daí o cliente irá fazer a requisição ao servidor do arquivo escolhido e assim que recebê-lo irá descriptografá-lo utilizando a chave e a ferramenta CryptoJS para dar acesso ao usuário ao arquivo.

4) Compartilhamento de arquivos: quando for solicitado pelo usuário que um arquivo seja compartilhado com outro usuário, o cliente deverá fazer uma requisição ao servidor, que irá criptografar uma cópia do arquivo com a chave do usuário que o mesmo será compartilhado e em seguida entregue ao repositório do usuário correspondente.

IV. RESUMO

As primeiras etapas do projeto foram a elaboração do modelo de banco de dados, que irá receber dados de acesso do usuário (login e senha), assim como sua chave de criptografia para manipular seus arquivos. Como mostrado na figura 1.

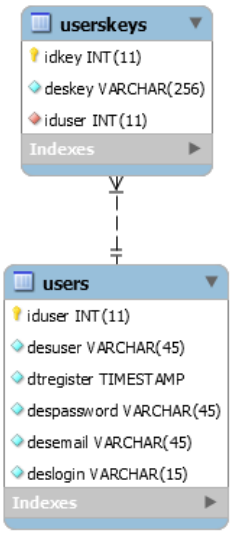

Figura 1 – Modelo do banco de dados.

Foram criados as pastas e o ambiente de desenvolvimento, seguindo o padrão habitual de disposição dos arquivos e nomenclatura das pastas para projetos web.

Após um período de desenvolvimento das telas utilizando HTML e CSS, foram realizados testes em PHP e JavaScript com o intuito de conhecer os métodos de criptografia AES e RSA utilizados por cada linguagem. Foram criados então as rotas de transporte das chaves entre o cliente e o servidor e implementado a troca de chaves entre cliente-servidor, como mostrado na figura 2.

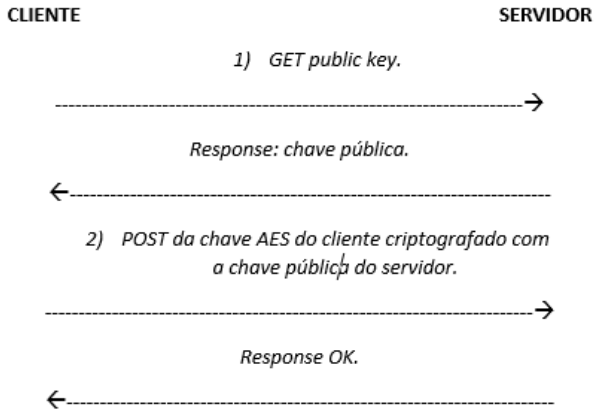

Figura 2 – Handshake de troca de chaves entre cliente-servidor.

Foram implementadas as requisições Ajax responsáveis por receber a chave pública do servidor e enviar a chave privada do cliente para o servidor já criptografada pela chave pública do servidor.

O handshake possui as funções descritas a seguir:

1) makeClientKey*:* responsável por gerar uma chave AES aleatória de 32 bits.
2) CryptoJSAESEncrypt*:* responsável pela encriptação dos dados de login e senha do cliente, gerando de forma randômica um vetor de inicialização.
3) encryptPublicKey*:* responsável pela criptografia da chave AES usada pelo cliente, com a chave pública do servidor.
4) getPublicKey*:* responsável pela obtenção da chave pública do servidor e enviará a chave AES usada pelo cliente para o servidor por meio de uma callback, descrita no item 5 desta sessão.
5) sendAESKey: responsável pelo envio da chave AES usada pelo cliente para a criptografia de dados ao servidor, essa chave é criptografada com a chave pública do servidor e assim enviada.

Para a realização do login no sistema, o cliente deve fazer a requisição da chave pública do servidor através da função *getPublicKey(sendAESKey)*, onde *getPublicKey* recebe como parâmetro um callback e *sendAESKey* envia ao servidor a chave AES que o cliente está usando para a criptografia dos dados. A chave AES é criptografada com a chave pública do servidor e assim enviada. Na tela de login também possui as funções *clearFields*, responsável por

limpar os dados do campo de formulário de login, como medida de precaução para os casos em que os dados de acesso sejam exibidos pelo browser e a função *validar*, responsável pela validação dos campos do formulário de login antes de seu envio.

REFERENCES

"Hyper Text Transfer Protocol Secure", Wikipedia, https://pt.wikipedia.org/wiki/Hyper_Text_Transfer_Protocol_Secure, (25 de ago. de 2018).

"Hyper Text Transfer Protocol", Wikipedia, https://pt.wikipedia.org/wiki/Hypertext_Transfer_Protocol (25 de ago. de 2018)

Estudo do Padrão Avançado de Criptografia AES – Advanced Encryption Standard. Trevisan, Diogo Fernando; Sacchi, Rodrigo p. da Silva; Sanabria, Lino. Disponível em: <http://seer.ufrgs.br/rita/article/download/rita_v20_n1_p13/23763>. Acesso em: 25 de agosto de 2018.

"Advanced Encryption Standard", Wikipedia, https://en.wikipedia.org/wiki/Advanced_Encryption_Standard, (25 de agosto de 2018).

"Crypto-JS", Google Code, https://code.google.com/archive/p/crypto-js, (25 de agosto de 2018).

"MySQL", Wikipedia, https://pt.wikipedia.org/wiki/MySQL, (26 de agosto de 2018).

"Download MySQL Community Server" MySQL TM, https://dev.mysql.com/downloads/mysql/>, (26 de agosto de 2018).

"RSA (cryptosystem)", Wikipedia, https://en.wikipedia.org/wiki/RSA_(cryptosystem)#History>, (27 de agosto de 2018).

"PHP Secure Communications Library", Jim Wigginton, http://phpseclib.sourceforge.net/documentation/index.html, (25 de agosto de 2018).

"Kanban (development)", Wikipedia, https://en.wikipedia.org/wiki/Kanban_(development), (26 de outubro de 2018).