



UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
INSTITUTO METRÓPOLE DIGITAL
BACHARELADO EM TECNOLOGIA DA INFORMAÇÃO



RELATÓRIO: implementação de Chat Seguro

ACSA LAIANE ARCANJO AUGUSTO
GABRIEL MARTINS SPÍNOLA

NATAL, RN
2019

ACSA LAIANE ARCANJO AUGUSTO
GABRIEL MARTINS SPÍNOLA

RELATÓRIO: implementação de Chat Seguro

Relatório apresentado como requisito para
obtenção de aprovação na disciplina
Segurança de Redes, no Curso de
Bacharelado em Tecnologia da Informação ,
na Universidade do Rio Grande do Norte.

Prof. Dr. Silvio Costa Sampaio

NATAL, RN

2019

LISTA DE ILUSTRAÇÕES

Figura 1 - mudando criptografia para SDES	5
Figura 2 - mudando criptografia para RC4	5
Figura 3 - Desabilitando criptografia	5
Figura 4 - mostrando todos os comandos do programa	5
Figura 5 - imprimindo chave pública	5
Figura 6 - imprimindo chave de sessão	6
Figura 7 - alterando parâmetros	6
Figura 8 - imprimindo usuários conectados	6
Figura 9 - imprimindo informações	6
Figura 10 - finalizando cliente chat seguro	6
Figura 11 - Saída do terminal, algoritmo para criptografia RC4 feito no c++	7
Figura 12 - Saída do terminal, servidor chat seguro	7
Figura 13 - Saída do terminal, cliente chat seguro	7

SUMÁRIO

1	INTRODUÇÃO.....	4
2	COMO EXECUTAR.....	5 e 6
3	SAÍDA DO PROGRAMA.....	7
4	CONSIDERAÇÕES FINAIS.....	8
5	REFERÊNCIAS.....	9

1. INTRODUÇÃO

O projeto trata-se de uma aplicação para troca de mensagens de texto (estilo chat) de maneira que seja possível a troca entre pares utilizando criptografias específicas.

O objetivo em si é executar os algoritmos de criptografias Simple DES (S-DES), Rivest Cipher 4 (RC4) e o de troca de chaves Diffie-Hellman, e usá-los na implementação do Chat com sistema interno de criptografia de mensagem, usando uma porta 5354, com socket TCP. No qual, ambos os pares geram chave de segurança (com o Diffie-Hellman) e essa poderá ser modificada pelo usuário em tempo de execução da aplicação, bem como o algoritmo de criptografia utilizado (S-DES ou RC4).

Esse trabalho foi implementado nas linguagens de Python e em C++, sendo dividido em quatro partes. Dentre elas, a implementação de cada algoritmo de criptografia para cifrar e decifrar, tal como o S-DES em python e o RC4 em C++, com leitura do arquivo que contém a mensagem e decodificando também em um arquivo. E os demais em python, como o algoritmo de troca de chaves Diffie-Hellman e o do próprio Chat que faz uso dos códigos (adaptados a classes em python) anteriormente citados.

2. COMO EXECUTAR

Para mudar a criptografia para o modelo SDES, utiliza-se o comando ‘\crypt sdes’.

Certifique-se de que já existe outro usuário conectado para usar os comandos de alteração de criptografia.

```
Acsa entrou no chat
\crypt sdes
A criptografia foi mudada para o modelo SDES
```

Figura 1 - mudando criptografia para SDES

Para mudar a criptografia para o modelo RC4, utiliza-se o comando ‘\crypt rc4’.

```
\crypt rc4
A criptografia foi mudada para o modelo RC4
```

Figura 2 - mudando criptografia para RC4

Para desabilitar a criptografia e enviar as mensagens do jeito que foi enviado de um cliente para o outro, utiliza-se o comando ‘\crypt none’.

```
\crypt none
Criptografia desabilitada
```

Figura 3 - Desabilitando criptografia

O comando ‘\commands’ imprime todos os comandos disponíveis no cliente.

```
\commands
-----
Lista de comandos:
\crypt sdes --> muda a criptografia atual para sdes com uma chave gerada por Diffie Hellman com os parametros escolhidos
\crypt rc4 --> muda a criptografia atual para rc4 com uma chave gerada por Diffie Hellman com os parametros escolhidos
\crypt none --> desabilita a criptografia
\getpublickey --> imprime a chave publica gerada pelo Diffie Hellman
\getsessionkey --> imprime a chave de sessao(IMPORTANTE: essa chave deve ser mantida em sigilo!!!)
\changeparameters --> muda os valores dos parametros do Diffie Hellman
\userlist --> mostra todos os usuarios conectados
\info --> mostra as informacoes do programa(Criptografia atual e informacoes do servidor)
\end --> finaliza a conexao e fecha o programa
-----
```

Figura 4 - mostrando todos os comandos do programa

O comando ‘\getpublickey’ imprime a chave pública do cliente que foi gerado a partir dos parâmetros que foram inicializados na abertura do programa.

```
\getpublickey
Chave publica: 273
```

Figura 5 - imprimindo chave pública

O comando ‘getsessionkey’ imprime a chave de sessão. IMPORTANTE: essa chave deve ser mantida em sigilo.

```
\getsessionkey  
Chave de sessao: 0000001100
```

Figura 6 - imprimindo chave de sessão

O comando ‘\changeparameters’ muda os parâmetros utilizados para gerar a chave pública pelo algoritmo Diffie Hellman.

Figura 7 - alterando parâmetros

```
\changeparameters  
Digite o novo valor de 'q'(OBS: o valor de 'q' so mudara no seu cliente, certifique-se de usar os mesmos parametros com outros clientes): 97  
Digite o novo valor de 'a'(OBS: o valor de 'a' so mudara no seu cliente, certifique-se de usar os mesmos parametros com outros clientes): 5
```

O comando ‘\userlist’ imprime os usuários conectados ao chat.

```
\userlist  
Lista de usuarios:  
Acса
```

Figura 8 - imprimindo usuários conectados

O comando ‘\info’ imprime as informações sobre a criptografia que está sendo utilizada no momento, a qual servidor está conectado e os parâmetros sendo utilizados pelo algoritmo Diffie Hellman.

```
\info  
Criptografia atual: desabilitada  
conectador ao servidor: IP -> 127.0.0.1 porta -> 5354  
Diffie Hellman: Valor de 'q' -> 97 Valor de 'alpha' -> 5
```

Figura 9 - imprimindo informações

O comando ‘\end’ finaliza o cliente.

```
\end  
Saindo do chat...
```

Figura 10 - finalizando cliente chat seguro

3. SAÍDA DO PROGRAMA

```
martinsspn@Martinsspn:~/Documentos/seguranca/algoritmorc4$ cat mensagem.txt
testando o algoritmo
martinsspn@Martinsspn:~/Documentos/seguranca/algoritmorc4$ ./algoritmorc4 mensagem.txt chave
martinsspn@Martinsspn:~/Documentos/seguranca/algoritmorc4$ cat cifrado.txt
♦♦GMnñE♦J>"♦♦♦♦♦2♦
martinsspn@Martinsspn:~/Documentos/seguranca/algoritmorc4$ mv cifrado.txt descifrar.txt
martinsspn@Martinsspn:~/Documentos/seguranca/algoritmorc4$ ./algoritmorc4 descifrar.txt chave
martinsspn@Martinsspn:~/Documentos/seguranca/algoritmorc4$ cat cifrado.txt
testando o algoritmo
martinsspn@Martinsspn:~/Documentos/seguranca/algoritmorc4$ █
```

Figura 11 - Saída do terminal, algoritmo para criptografia RC4 feito no c++

Para inicializar o servidor utilize a linha de comando 'python servidorChat.py IPdoServidor'.

```
martinsspn@Martinsspn:~/Documentos/chat$ python servidorChat.py '192.168.0.87'
Servidor iniciado no IP: 192.168.0.87 Porta: 5354
█
```

Figura 12 - Saída do terminal, servidor chat seguro

Para inicializar o cliente utilize a linha de comando 'python clienteChat.py IPdoServidor 'SeuNick' 'parametro Diffie Hellman q' 'parametro Diffie Hellman alpha'.

```
martinsspn@Martinsspn:~/Documentos/chat$ python clienteChat.py 127.0.0.1 Gabriel 353 3
Welcome to this chatroom!
```

Figura 13 - Saída do terminal, cliente chat seguro

4. CONSIDERAÇÕES FINAIS

Este trabalho apresentou a implementação dos algoritmos criptográficos S-DES, RC4, e o de troca de chaves Diffie-Hellman. Esses utilizados na implementação do Chat Seguro, com sistema interno de escolha de cifra das mensagens, fazendo uso do código do Diffie-Hellman para gerar a chave inicial.

Além disso, com o objetivo de visualizar o funcionamento das criptografias especificadas e observá-las em uma aplicação simples, tal como o Chat, usando uma porta particularizada, com socket TCP, constatando um dos pilares para ajudar a delimitar e incrementar o nível de segurança da aplicação. Também podendo avaliar o nível de eficiência e praticidade oferecido por cada uma.

5. REFERÊNCIAS

GeeksforGeeks A computer science portal for geeks. Simple Chat Room using Python. Disponível em <<https://www.geeksforgeeks.org/simple-chat-room-using-python>>. Acesso em 05 de setembro de 2019.