Sprint 1

Segurança de Aplicações e Dados

Rúben Medeiros (1201679)

Docentes:

Marelo Santos

Vladir Vicente

Data: 25/11/2022

Índice

Introdução

Foi proposta pelos docentes Marcelo e Vladir a realização de uma cifra chamada “Cifra de Ceaser”.

Cifra de Ceaser é um dos métodos mais antigos e mais conhecidos da criptografia. Esta cifra baseia-se numa substituição do alfabeto. Com isto quero dizer que a mensagem de input chega a um *plugboard*, onde este é diferente de aluno para aluno, passando de seguida por rotores e volta novamente ao *plugboard* para mostrar a mensagem alterada.

Este trabalho teve a ajuda de dois colegas meus: “Diogo Sousa e André Padilha” que se disponibilizaram para me ajudar durante toda a realização do projeto.

Carregar Plugboard

Inicialemente comecei por carregar a plugboard e com isso utilizei uma variável estática e final, no qual tem a minha plugboard que os professores forneceram no enunciado.

```
1 usage
private static final String plugboard = "{'N': 'Y', 'X': 'P', 'M': 'R', 'Z': 'G', 'E': 'O', 'B': 'T', 'A': 'V', 'S': 'H', 'Q': 'U', 'F': 'W'}";
```

De seguida, criei outra variável onde vai conter a minha plugboard com apenas as letras.

```
3 usages
private static final HashMap<String, String> plugboardMap = new HashMap<>();
```

Finalmente criei o método para carregar a plugboard.

```
/**
 * CARREGA A PLUGBOARD SEM {}, SEM ' E SEM :
 *
 * @param s
 */
private static void carregarPlugboard(String s) {
    // Apaga tudo o que estiver dentro da variável criada para evitar ter alguma coisa sempre que carrego a plugboard
    plugboardMap.clear();
    // Verifica se a plugboad tem {}
    if (s.equals("{}")) {
        return;
    }
    // Substitui a { por vazio
    s = s.replace( target: "{", replacement: "");
    // Substitui a } por vazio
    s = s.replace( target: "}", replacement: "");
    // Ciclo que separa todas as palavras que tiverem uma , e um espaço entre elas
    for (String s1 : s.split( regex: ", ")) {
        // Substitui as ' por vazio e de seguida separa as letras que tiverem : e um espaço entre elas
        String[] s2 = s1.replace( target: "'", replacement: "").split( regex: ": ");
        // Demonstração de como está neste momento -> [N,Y]
        // Crio um char onde na posição 0 guardo a letra da esquerda
        char c = s2[0].charAt(0);
        // Crio um char onde na posição 1 guardo a letra da direita
        char c1 = s2[1].charAt(0);
        // Validação se as letras da equerda e da direita são realmente letras
        if (Character.isLetter(c) && Character.isLetter(c1)) {
            // Insere as letras dentro da nova plugboard
            plugboardMap.put("" + c, "" + c1);
        }
    }
}
```

Substituir as Letras da Plugboard

Para ver se a plugboard estava a funcionar, devidi criar um método onde inseria uma palavra a meu gosto e verificava através do plugboard se as letras da palavra tinham sido alteradas.

```java
/**
 * SUBSTITUI AS LETRAS DO PLUGBOARD ONDE ENVIO UMA PALAVRA À SORTE PARA SER ALTERADA PELO PLUGBOARD
 *
 * @param palavra
 * @return
 */
2 usages  "Rúben *
private static String substituirPlugboard(String palavra) {
    String novaPalavra = "";
    // Ciclo onde separa as letras da palavra por espaços vazios
    for (String letra : palavra.split( regex: "")) {
        // Insere a nova palavra dentro da plugboard
        novaPalavra = novaPalavra + plugboardMap.getOrDefault(letra, letra);
    }
    // Retorna a nova palavra alterada
    return novaPalavra;
}
```

Salt para Testes de KAT

Para realizar todos os testes KAT tive que inicialmente criar uma variável com todas as letras do alfabeto.

```
private static final String alfabeto = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
```

Depois disso, criei o método para fazer o salt, tanto à esquerda do alfabeto, tanto à direita.

```
/**
 * SALT PARA FAZER TESTES DE KAT ONDE RECEBE COMO PARÂMETRO UMA PALAVRA DE INPUT
 *
 * @return
 */
 "Rúben *
private static List<String> saltKat(String palavra) {
    // Lista onde vai guardar todo o processo SALT num array
    List<String> lista = new ArrayList<>();
    // Ciclo onde vai subistuir todas as letras do alfabeto por espaços vazios
    for (String a : alfabeto.split( regex: "")) {
        for (String b : alfabeto.split( regex: "")) {
            //SALT À ESQUERDA
            lista.add(a + b + ";" + palavra);
            //SALT À DIREITA
            lista.add(palavra + ";" + a + b);
        }
    }
    return lista;
}
```

Salt para realizar o Brute Force

Depois de ter criado o primeiro salt fiz exatamente a mesma coisa que fiz no método anterior, sendo que a única diferença é que não utilizei o alfabeto normal, mas sim o dado pelo enunciado. Para isso tive que criar uma variável nova com o novo alfabeto.

```
2 usages
private static final String alfabetoSalt = "ABCDEFGHIJKLM";
```

Logo a seguir criei o método para fazer o Salt.

```
/**
 * SALT PARA FAZER O BRUTE FORCE ONDE RECEBE UMA PALAVRA COMO PARÂMETRO
 * @param palavra
 * @return
 */
1 usage   "Rúben"
private static List<String> salt(String palavra) {
    // Lista onde vai ser recebida o salt final
    List<String> lista = new ArrayList<>();
    // Ciclo onde corre todas as primeiras letras no alfabeto salt
    for (String a : alfabetoSalt.split( regex: "")) {
        // Ciclo onde corre todas as segundas letras no alfabeto salt
        for (String b : alfabetoSalt.split( regex: "")) {
            // Aplica o SALT à esquerda
            lista.add(a + b + ";" + palavra);
            // Aplica o SALTO à direita
            lista.add(palavra + ";" + a + b);
        }
    }
    return lista;
}
```

Cifra de Ceaser

Logo a seguir ao método anterior, resolvi fazer o método para calcular a cifra de ceaser.

```java
/**
 * CIFRA DE CEASER ONDE RECEBE A PALAVRA, AS ROTAÇÕES E O INCREMENTO COMO PARÂMETRO
 * @param palavra
 * @param rot
 * @param F
 * @return
 */
private static String cifraCeaser(String palavra, int rot, int F) {
    int index = 0;
    String novaPalavra = "";
    // Lista de Strings onde vai dividir todo os letras do alfabeto em espaços vazios
    List<String> alfabeto = Arrays.stream(Main.alfabeto.split( regex: "")).toList();
    // Pega num caracter e divide por espaços vazios
    for (String caracter : palavra.split( regex: "")) {
        // Formula do enunciado
        int inc = index * F;
        // Pega no alfabeto e indexa os caracteres todos
        int alfabetoIndex = alfabeto.indexOf(caracter);
        // A nova letra vai ter o alfabeto indexado e vai somar as rotações e o incremento
        int novaLetra = alfabetoIndex + rot + inc;
        // Ciclo que valida que enquanto a nova letra é maior do que o tamanho do alfabeto (Não há letras depois do Z)
        while (novaLetra >= alfabeto.size()) {
            // A nova letra tem que ser a nova letra menos o tramanho do alfabeto para tudo correr bem
            novaLetra = novaLetra - alfabeto.size();
        }
        // A letra vai receber a nova letra do alfabeto quando sai do ciclo
        String letra = alfabeto.get(novaLetra);
        // Na nova palavra vai ser adicionada a nova letra
        novaPalavra = novaPalavra + letra;
        index++;
    }
    return novaPalavra;
}
```

Execução dos Testes KAT

Depois de ter feito o método da cifra de ceaser decidi começar com a realização de todos os testes KAT o que felizmente foram todos bem sucedidos.

```
// TESTES KAT
carregarPlugboard( s: "{'N': 'K', 'V': 'E', 'J': 'S', 'W': 'T', 'R': 'M', 'P': 'H', 'Y': 'F', 'X': 'O', 'A': 'I', 'Q': 'C', 'G': 'D'}");

// CICLO QUE RECEBE O INPUT DA PALAVRA DADA PELOS DOCENTES
for (String salt : saltKat( palavra: "X")) {
    // Substitui a palavra de ; para espaços vazios
    String word = salt.replace( target: ";", replacement: "");
    // Substitui a palavra na nova plugboard
    String plugboard1 = substituirPlugboard(word);
    // Ciclo que os docentes também dizem no enunciado para fazer as rotações
    for (int i = 0; i <= 30; i++) {
        // Ciclo que os docentes também dizem no enunciado para fazer os incrementos
        for (int j = 0; j <= 30; j++) {
            // Faz a cifra de ceaser na plugboard, com as rotações e incrementos
            String cesar = cifraCeaser(plugboard1, i, j);
            // Faz a substituição de uma nova plugboard
            String plugboard2 = substituirPlugboard(cesar);
            // Verifica se o plugboard tem a palavra final dada também pelos docentes
            if (plugboard2.equalsIgnoreCase( anotherString: "SSS")) {
                System.out.println("PALAVRA: " + word);
                System.out.println("ROTAÇÃO: " + i);
                System.out.println("INCREMENTO: " + j);
                System.out.println("SALT: " + Arrays.toString(salt.split( regex: ";")));
            }
        }
    }
}
```

Ler o Ficheiro da Wordlist

Para começar resolvi por criar uma String onde tinha o caminho da wordlist.

```
1 usage
private static final String wordlist = "C:\\sad\\src\\main\\java\\wordlist.txt";
```

De seguida, criei um método simples para ler e guardar o conteúdo do ficheiro numa lista de Strings.

```
/**
 * MÉTODO QUE LÊ O FICHEIRO WORDLIST.TXT QUE CONTÉM TODAS AS PALAVRAS DADAS PELOS DOCENTES
 * @param path
 * @return
 * @throws IOException
 */
1 usage   "Rúben"
private static List<String> lerWordlist(String path) throws IOException {
    // Lista de String onde vai guardar todas as palavras do ficheiro
    List<String> listaPalavras = new ArrayList<String>();
    // Um simples BufferReader onde recebe como parâmetro o caminho do ficheiro
    BufferedReader br = new BufferedReader(new FileReader(path));
    // Lê o ficheiro linha a linha
    String linha = br.readLine();
    // Ciclo que valida que enquanto a linha não for nula adiciona o conteúdo à lista de palavras e lê a linha seguinte
    while (linha!= null) {
        listaPalavras.add(linha);
        linha = br.readLine();
    }
    // Fecha o ficheiro
    br.close();
    return listaPalavras;
}
```

SHA-512

Nesta parte decidi ir à internet ver métodos para o SHA-512. Confesso que tive bastante dificuldades e a ajuda dos meus colegas neste método foi crucial.

```java
/**
 * MÉTODO PARA CALCULAR O SHA-512
 * <a href="https://stackoverflow.com/questions/33085493/how-to-hash-a-password-with-sha-512-in-java">LINK ONDE ADAPETEI O CODIGO</a>
 * @param palavra
 * @return
 */
1 usage  "Rúben"
private static String calcularSHA(String palavra) {
    // Constroi uma string
    StringBuilder stringb = new StringBuilder();
    try {
        MessageDigest md = MessageDigest.getInstance( algorithm: "SHA-512");
        byte[] data = md.digest(palavra.getBytes());
        for (byte datum : data) {
            stringb.append(Integer.toString( i: (datum & 0xff) + 0x100,  radix: 16).substring( beginIndex: 1));
        }
    } catch (Exception erro) {
        System.out.println(erro);
    }
    return stringb.toString();
}
```

Brute Force

O meu brute force foi feito dentro do método Main. Com isto, comecei por criar uma variável chamada foiAchado que verifica se encontrou ou não a password.

```
// Variável que apenas verifica quando encontra a password
4 usages
private static boolean foiAchado = false;
```

De seguida foi feito vários ciclos para fazer o Brute Force.

```
public static void main(String[] args) throws IOException {
    // Carrega a minha plugboard
    carregarPlugboard(plugboard);
    // Ciclo que lê todas as palavras da wordlist e itera uma por uma
    for (String palavra: lerWordlist(Main.wordlist)) {
        // Itera pelas palavras da wordlist e aplica o salt às mesmas
        System.out.println("PALAVRA ATUAL:  " + palavra);
        for (String palavraComSalt: salt(palavra)) {
            String word = palavraComSalt.replace( target: ";", replacement: "");
            String plugboard1 = substituirPlugboard(word);
            // Ciclo que faz as Rotações no Rotor
            for (int i = 0; i <= 26 ; i++) {
                // Ciclo que aumenta o fator de incremento
                for (int j = 0; j <= 26 ; j++) {
                    // Aplica a cifra de ceaser
                    String cesar = cifraCeaser(plugboard1, i, j);
                    // Substitui na nova plugboard a cifra feita
                    String plugboard2 = substituirPlugboard(cesar);
                    // Calcula o sha-512 da nossa plugboard
                    String hash = calcularSHA(plugboard2);
                    // Validação que compara a Hash calculada pela Hash recebida
                    if (hash.equalsIgnoreCase(sha512)) {
                        System.out.println("-----------------------------------------");
                        System.out.println("PASSWORD FINAL: " + word);
                        System.out.println("ROTAÇÃO: " + i);
                        System.out.println("INCREMENTO: " + j);
                        System.out.println("SALT: " + Arrays.toString(palavraComSalt.split( regex: ";")));
                        System.out.println("-----------------------------------------");
                        Main.foiAchado = true;
                        break;
                    }
                }
                // Se a password não for encontrada sai do ciclo
                if (foiAchado)break;
            }
            // Se a password não for encontrada sai do ciclo
            if (foiAchado)break;
        }
        // Se a password não for encontrada sai do ciclo
        if (foiAchado)break;
        System.out.println("PASSWORD NÃO FOI ENCONTRADA!");
    }
```

Password Final

Depois de 5 horas à espera de resultados, o output do brute force foi o seguinte:

Password: JRCFYCI

ROTAÇÃO: 11

INCREMENTO: 8

PALAVRA + SALT: JRCRYF + CI

Conclusão

Este projeto no sprint 1 não foi de acordo com as minhas espectativas, porque nunca estive com tantas dificuldades como estive com este projeto. Quero desde já agradecer aos meus colegas pelo tempo perdido comigo e pela sua disponibilidade. Neste sprint consegui aprender algumas coisas sobre criptografia e o quanto dificil é para alguém que domina e não domina a área.