Sessão 9: Mecanismo criptográficos

1. Atividade – Vulnerabilidades no transporte de informações

Esta atividade compreende os testes para detecção de vulnerabilidades no transporte de informações. Para iniciá-la, carregue as máquinas virtuais do aluno e do servidor (Fedora) e execute os roteiros na primeira delas.

Teste das suítes criptográficas suportadas pelo servidor SSL/TLS

Nesta parte serão exercitados três métodos diferentes para detecção das suítes criptográficas suportadas por um servidor.

OpenSSL

A opção do OpenSSL que testa se uma determinada suíte é suportada é `-cipher` :

1. Abra uma janela de terminal.
2. Digite o seguinte comando para verificar se o servidor exemplo.esr.rnp.br suporta a suíte `NULL-SHA` :

```
~$ openssl s_client -cipher NULL-SHA -connect exemplo.esr.rnp.br:443
```

3. Se a sessão for estabelecida sem erro, o servidor suporta a suíte vulnerável.
4. Pressione Ctrl+C para encerrar a conexão com o servidor.
5. Encerre a janela de terminal.

THCSSLCheck

Como o THCSSLCheck é compilado para plataformas Windows, é necessário utilizar o aplicativo Wine para executá-lo em sistemas Linux:

1. Inicie o THCSSLCheck a partir do menu Aplicativos\Curso – Ferramentas.
2. Execute o comando:

```
~$ wine /usr/share/thcsslcheck/THCSSLCheck.exe exemplo.esr.rnp.br 443
```

3. Role a janela e analise o relatório gerado. Quais suítes fracas são suportadas?

Resposta: NULL-SHA

4. Encerre a janela de terminal.

SSL Scan

Um ponto interessante do SSL Scan é que ele exibe, para cada versão de protocolo, as suítes criptográficas utilizadas por padrão pelo servidor sendo testado:

1. Abra uma janela de terminal.
2. Digite o seguinte comando, para avaliar o servidor exemplo.esr.rnp.br:

```
~$ sslscan exemplo.esr.rnp.br
```

3. Role a janela e analise o relatório gerado. Quais as diferenças para a saída do THCSSLCheck?

Resposta: no thcsslcheck não aparece os algoritmos de cifra nula como NULL-SHA e NUL-MD5. O resto é igual.

4. Encerre a janela de terminal.

Prova de conceito de tráfego TLS em claro

Este exercício ilustra que é possível fazer com que os protocolos SSL e TLS não cifrem os dados que protegem. O roteiro abaixo consiste em um autoataque, mas isso não é importante por se tratar de uma prova de conceito.

1. Inicie o Wireshark:

```
~$ sudo wireshark
```

2. Clique no primeiro ícone da barra de ferramentas, para listar as interfaces de rede disponíveis para captura. Na caixa de diálogo que aparece, clique em `Options` da linha `eth1`.

   No campo `Capture filter`, digite `tcp port https` e clique em `Start` para iniciar a captura de pacotes.

3. Abra uma janela de terminal.
4. Conecte-se ao servidor web exemplo.esr.rnp.br, com o OpenSSL:

```
~$ openssl s_client -connect exemplo.esr.rnp.br:443
```

5. Digite a requisição abaixo, finalizando-a com `Enter` duas vezes:

```
GET / HTTP/1.1
Host: exemplo.esr.rnp.br
```

6. Pare a captura de pacotes no Wireshark, clicando no quarto botão da barra de ferramentas (`Stop the running live capture`).
7. Procure pela primeira linha contendo `TLSv1 e Application Data` e a selecione.
8. Na região central da tela do Wireshark, selecione a linha `Secure Socket Layer`
9. Examine o conteúdo na parte inferior e veja que ele está cifrado.
10. Inicie nova captura no Wireshark, clicando no terceiro botão da barra de ferramentas (`Start new live capture`).
11. Clique em `Continue without saving`, na caixa de diálogo que aparece.
12. Na janela de terminal, digite:

```
~$ openssl s_client -cipher NULL-SHA -connect exemplo.esr.rnp.br:443
```

13. Repita os Passos 5 a 8 acima, mas, em vez da primeira, procure pela última linha cujo `Protocol` seja igual a `HTTP`. Veja que o conteúdo agora está em claro, apesar de encapsulado por TLS.
14. Encerre o Wireshark e o terminal.

Certificado inválido

O propósito desta atividade é verificar uma das situações em que o navegador é impossibilitado de realizar a negociação SSL/TLS com o servidor, devido a um problema no certificado:

1. Inicie o Firefox, presente no menu `Usual application\Internet`.
2. Acesse https://exemplo.esr.rnp.br.

3. Uma mensagem de erro é exibida, informando que não é possível estabelecer uma conexão segura. Clique em `Learn more` e verifique o motivo do problema.
4. Encerre o Firefox.

> Recursos sensíveis protegidos por HTTPS e HTTP Informações sensíveis fornecidas por meio de HTTPS nunca devem ser disponibilizadas também por HTTP simples, pois o usuário pode ser induzido a acessar a versão desprotegida da página, que pode ser facilmente capturada em trânsito.

5. Inicie o Firefox, presente no menu `Usual application\Internet`.
6. Acesse https://w3s.esr.rnp.br e observe que a página é protegida por HTTPS.
7. Tente, agora, acessar o mesmo endereço pelo protocolo HTTP e observe o que acontece. Neste caso, o servidor está configurado para redirecionar o usuário para a página protegida, evitando que informações sejam expostas a pessoas não autorizadas.
8. Encerre o Firefox

2. Atividade – Vulnerabilidades no armazenamento de informações

O objetivo desta atividade é capacitar o aluno em técnicas básicas de criptoanálise, para que seja capaz de recuperar informações sensíveis, protegidas por cifras fracas ou por criptossistemas utilizados incorretamente. Todos os exercícios desta atividade necessitam da máquina virtual do aluno e de arquivos contidos na pasta `/home/esruser/Arquivos do Curso/sessao-09`.

Dados `protegidos` com BASE64

Conforme explicado, BASE64 é simplesmente um esquema de codificação e, por isso, não pode ser utilizado na proteção do sigilo de informações. Siga o roteiro abaixo para entender como é fácil recuperar a mensagem original a partir de um texto codificado em BASE64.

1. Abra uma janela de terminal e acesse o diretório `/home/esruser/Arquivos do Curso/sessao-09`.
2. Visualize o conteúdo do arquivo `arquivo.b64`:

```
~$ cat arquivo.b64
```

3. Decodifique o arquivo com o comando:

```
~$ base64 -d arquivo.b64
```

   Observe que não é necessário fornecer qualquer segredo para recuperar a mensagem original.

4. Visualize o conteúdo do arquivo `arquivo2.b64`:

```
~$ cat arquivo2.b64
```

5. Decodifique o arquivo com o comando:

```
~$ base64 -d arquivo2.b64 > arquivo2.txt
```

6. Visualize o arquivo gerado:

```
~$ cat arquivo2.txt
```

O arquivo resultante não está legível e, aparentemente, está protegido por uma cifra de substituição simples, que será criptoanalisada em atividade posterior.

7. Encerre a janela de terminal.

Índice de coincidência

O índice de coincidência é uma ferramenta que auxilia o processo de criptoanálise, ao indicar o tipo de cifra utilizada e o provável idioma da mensagem original. A última parte, porém, requer uma mensagem suficientemente longa, que seja válida estatisticamente.

1. Abra uma janela de terminal e acesse o diretório `/home/esruser/Arquivos do Curso/sessao-09`.
2. Verifique o índice de coincidência do arquivo em claro `ingles.txt`:

```
~$ /usr/local/sbin/ioc ingles.txt
```

3. Calcule o índice de coincidência do arquivo `ingles.enc`, resultante do ciframento de `ingles.txt`, por um algoritmo de substituição simples:

```
~$ /usr/local/sbin/ioc ingles.enc
```

4. Observe que os índices do arquivo em claro e do cifrado são praticamente idênticos, o que é coerente com o fato de que cifras monoalfabéticas apenas transferem as frequências individuais dos símbolos para outros.
5. Verifique o índice de coincidência do arquivo em claro `portugues2.txt`:

```
~$ /usr/local/sbin/ioc portugues2.txt
```

6. Calcule o índice de coincidência do arquivo `portugues2.poli.enc`, resultante do ciframento de `portugues2.txt`, por um algoritmo de substituição polialfabética:

```
~$ /usr/local/sbin/ioc portugues2.poli.enc
```

Note que o índice de coincidência é bem menor que o apresentado por textos em linguagem natural, conforme esperado.

7. Encerre a janela de terminal

Quebra da cifra de Cesar

A cifra de Cesar, assim como a codificação em BASE64, é uma transformação fixa, que não utiliza uma chave. Por esse motivo, é muito fácil quebrá-la e recuperar o texto legível.

1. Abra uma janela de terminal e acesse o diretório `/home/esruser/Arquivos do Curso/sessao-09`.
2. Visualize o conteúdo do arquivo `cesar.enc`:

```
~$ cat cesar.enc
```

3. Calcule o índice de coincidência do arquivo:

```
~$ /usr/local/sbin/ioc cesar.enc
```

Observe que o valor encontrado indica corretamente que o texto está protegido por uma cifra monoalfabética ou de transposição.

4. Aplique a transformação de deciframento da cifra de Cesar:

```
~$ rotix -f cesar.enc -o cesar.plain -r 3 -L
```

5. Visualize o conteúdo do arquivo `cesar.plain` e veja que texto legível foi recuperado:

```
~$ cat cesar.plain
```

6. Suponha que o arquivo `arquivo2.txt` , gerado no exercício sobre BASE64, esteja protegido com Cesar. Tente decifrá-lo por meio do comando:

```
~$ rotix -f arquivo2.txt -o arquivo2.plain -r 3 -L
```

7. Visualize o conteúdo do arquivo `arquivo2.plain` e veja que texto legível foi recuperado:

```
~$ cat arquivo2.plain
```

8. Encerre a janela de terminal.

Quebra da cifra de deslocamento

O grande problema da cifra de deslocamento é que o espaço de chaves utilizado contém apenas 26 elementos, que podem ser testados exaustivamente até que o texto original seja encontrado. Tendo isso em mente, aplique a técnica ao arquivo `deslocamento.enc` :

1. Abra uma janela de terminal e acesse o diretório `/home/esruser/Arquivos do Curso/sessao-09` .
2. Visualize o conteúdo de `deslocamento.enc` :

```
~$ cat deslocamento.enc
```

3. Calcule o índice de coincidência e observe a classe provável de cifra utilizada:

```
~$ /usr/local/sbin/ioc deslocamento.enc
```

   Este é um exemplo de que uma classe incorreta de cifra pode ser identificada, quando o texto cifrado for muito pequeno.

4. Tente realizar o deciframento com a chave k = 1 e veja se texto em claro é recuperado:

```
~$ rotix -f deslocamento.enc -r 1 -L
```

5. Repita o passo anterior, variando k de 2 a 25 (opção -r), até que texto legível seja obtido.

Resposta: o valor de K é 7.

6. Encerre a janela de terminal.

Quebra de ROT13

ROT13 é outro exemplo de transformação fixa, baseada em uma especialização da cifra de deslocamento, com chave k = . A recuperação de texto protegido por este mecanismo é trivial, conforme ilustrado pela presente atividade:

1. Abra uma janela de terminal e acesse o diretório `/home/esruser/Arquivos do Curso/sessao-09` .

2. Visualize o conteúdo do arquivo `rot13.enc` :

```
~$ cat rot13.enc
```

3. Calcule o índice de coincidência do arquivo:

```
~$ /usr/local/sbin/ioc rot13.enc
```

4. Aplique a transformação descrita pelo algoritmo ROT13:

```
~$ rotix -f rot13.enc
```

5. Encerre a janela de terminal.

Quebra da cifra de substituição simples

Uma cifra de substituição simples pode ser quebrada, como visto, por meio da técnica de análise de frequências, introduzida por al-Kindī. Aplique este método para criptoanalisar o arquivo `portugues3.mono.enc` , utilizando as ferramentas fornecidas pelo sítio web `The Black Chamber` , de Simon Singh:

1. Inicie o Firefox, presente no menu `Usual application\Internet` .
2. Acesse https://www.simonsingh.net/The_Black_Chamber/substitutioncrackingtool.html. (outra página que pode ser utilizada neste exercício é https://cryptoclub.org/tools/cracksub_topframe.php mas neste caso os passos abaixo não são validos)
3. Abra uma janela de terminal e acesse o diretório `/home/esruser/Arquivos do Curso/sessao-09` .
4. Calcule o índice de coincidência para o arquivo `portugues3.mono.enc` :

```
~$ /usr/local/sbin/ioc portugues3.mono.enc
```

   Observe que tanto a cifra quanto o idioma foram corretamente identificados.

5. Abra o arquivo `portugues3.mono.enc` :

```
~$ gedit portugues3.mono.enc &
```

6. Selecione todo o texto (Ctrl-A) e o copie para a área de transferência (Ctrl-C).
7. Retorne ao Firefox e cole o texto cifrado no campo `Ciphertext` .
8. Clique no link `Frequency of Individual Letters` .
9. Veja no histograma quais são, ordenadamente, as três letras mais frequentes.
10. Substitua, em ordem, as três letras mais comuns na língua portuguesa por aquelas encontradas no passo anterior. Para isso, preencha os campos correspondentes em `Plaintext Alphabet` .
11. Procure no campo `Plaintext` agrupamentos de letras que possam ser identificadas como palavras. Para cada uma que encontrar, veja que mapeamentos podem ser inferidos entre letras do texto em claro e do texto cifrado e efetue a substituição em `Plaintext Alphabet` .
12. Repita o Passo 11 até que o texto inteiro seja recuperado.
13. Encerre o Firefox e janela de terminal.

Comentário: Analisando a frequência de cada letra temos:

Letra Incidência no texto cifrado Equivalência observando a figura 9.62

B	**14**	**a**
F	13	e
P	11	o
I	8	i
T	7	s
Q	6	r
N	6	n
D	5	d
J	4	m
K	4	l
V	4	t
W	3	u
C	3	c
L	3	p
S	2	v
H	2	g
A	1	b
R	1	q

Ordem das probabilidade das letras na língua portuguesa.

A E O S R I N D M U T C L P Q V F G H B J Z X K W Y

Mensagem decifrada:

Possa ser bem aceita por eles esta lembrança de um correligionário ausente, mandada do exterior, donde se ama mais a pátria do que no próprio país - pela contingência de não tornar a vê-la, pelo trabalho constante da imaginação, e pela saudade que Garret nunca teria pintado ao vivo se não tivesse sentido a nostalgia - e onde o patriotismo, por isso mesmo que o Brasil é visto como um todo no qual homens e partidos, amigos e adversários se confundem na superfície alumiada pelo sol dos trópicos, parece mais largo, generoso e tolerante.

Quanto a mim, julgar-me-ei mais do que recompensado, se as sementes de liberdade, direito e justiça, que estas páginas contêm, derem uma boa colheita no solo ainda virgem da nova geração; e se este livro concorrer, unindo em uma só legião os abolicionistas brasileiros, para apressar, ainda que seja de uma hora, o dia em vejamos a independência completada pela abolição, e o Brasil elevado à dignidade de país livre, como o foi em 1822 à de nação soberana, perante a América e o mundo.

Assim sintetiza o grande liberal brasileiro Joaquim Nabuco (1849-1910)

Quebra da cifra de Vigenère

Nesta atividade, um texto cifrado com Vigenère será criptoanalisado pelo método de Babbage, com o auxílio de ferramentas do sítio web `The Black Chamber`, de Simon Singh:

1. Inicie o Firefox, presente no menu `Usual application\Internet`.
2. Acesse http://www.simonsingh.net/The_Black_Chamber/vigenere_cracking_tool.html.
3. Limpe a caixa de texto da tela inicial.
4. Abra uma janela de terminal e acesse o diretório `/home/esruser/Arquivos do Curso/sessao-09`.
5. Abra o arquivo `portugues2.poli.enc`:

```
~$ gedit portugues2.poli.enc &
```

6. Selecione todo o texto (Ctrl-A) e o copie para a área de transferência (Ctrl-C).
7. Retorne ao Firefox e cole o texto cifrado na caixa de texto.
8. Clique em `Find Repeated Sequences`.
9. Role a tela, observe a coluna que possui o maior número de linhas marcadas com `X` e clique no cabeçalho correspondente.
10. Vá ao final da página e clique no botão `L1`, para analisar a cifra monoalfabética definida para a primeira posição de cada bloco.
11. Clicando nas setas para a direita ou para a esquerda, alinhe o histograma em vermelho com histograma roxo. Quando os dois histogramas estiverem parecido clique no botão `L2` para tentar identificar a próxima letra. Faça isso para `L3`, `L4` e `L5`
    a. Oriente-se pelos picos das letras `A`, `E` e `O` e pelo vale formado pelas quatro últimas letras. Para ajudar, saiba que `L1` é a letra `L`

Resposta: A senha é a palavra LAPIS.

Uso de modo de operação inadequado

Vimos que o uso do modo ECB, para mensagens maiores que o tamanho do bloco da cifra utilizada, revela informações sobre o texto em claro. Como exemplo, a presente atividade visa ilustrar esta vulnerabilidade no contexto de proteção de imagens:

1. Abra uma janela de terminal e acesse o diretório `/home/esruser/Arquivos do Curso/sessao-09`.
2. Visualize com o gimp a imagem `figura.bmp`:

```
~$ gimp figura.bmp &
```

3. Retorne à janela de terminal.
4. Cifre a imagem `figura.bmp` com o algoritmo AES-128 em modo ECB:

```
~$ openssl enc -aes-128-ecb -in figura.bmp -out figura.bmp.ecb -K 01234567890123456789012345678901 -iv 0
```

5. Cifre a imagem `figura.bmp` com o algoritmo AES-128 em modo CBC:

```
~$ openssl enc -aes-128-cbc -in figura.bmp -out figura.bmp.cbc -K 01234567890123456789012345678901 -iv 0
```

6. Liste os arquivos do diretório e observe que o tamanho dos arquivos cifrados é maior que o original em alguns bytes:

```
~$ ls -l
```

   Isso se deve ao processo de padding, que consiste em se preencher a mensagem para que o tamanho dela fique múltiplo daquele definido pelo bloco da cifra utilizada.

7. Compacte com o gzip o arquivo `figura.bmp.ecb` :

```
~$ gzip -c figura.bmp.ecb > figura.bmp.ecb.gz
```

8. Compacte com o gzip o arquivo `figura.bmp.cbc` :

```
~$ gzip -c figura.bmp.cbc > figura.bmp.cbc.gz
```

9. Liste os arquivos do diretório novamente:

```
~$ ls -l
```

   Note que, enquanto não houve compressão do arquivo cifrado em modo CBC, a contraparte em modo ECB teve o tamanho reduzido para apenas 3,7% do original. Isto indica a presença de muita redundância no arquivo, o que não se espera da saída de uma cifra utilizada corretamente.

10. Tente abrir o arquivo cifrado em modo ECB com o gimp:

```
~$ gimp figura.bmp.ecb &
```

    Uma mensagem de erro indicando formato de arquivo desconhecido é exibida, pois o ciframento transforma toda a informação, inclusive o cabeçalho da imagem. Clique em `OK` para encerrá-la.

11. Como o formato BMP descreve um mapa de bits, copiando o cabeçalho da imagem original para os arquivos cifrados, é possível visualizá-los e observe as mudanças que ocorreram em cada bit, no processo de ciframento:

```
~$ ./bmphc figura.bmp figura.bmp.ecb
~$ ./bmphc figura.bmp figura.bmp.cbc
```

12. Visualize o arquivo `figura.bmp.ecb` :

```
~$ gimp figura.bmp.ecb &
```

    Observe que a imagem do arquivo original é preservada, mas com cores diferentes.

13. Retorne à janela de terminal.
14. Visualize o arquivo `figura.bmp.cbc` e note que a imagem não possui relação nenhuma com a figura original:

```
~$ gimp figura.bmp.cbc &
```

15. Encerre o gimp e a janela de terminal.

Uso incorreto de algoritmo criptográfico

O RC4 pertence à classe de cifras de fluxo aditivas e binárias e, portanto, não deve nunca ser usado para cifrar textos diferentes com a mesma chave. Este exercício ilustrará um ataque de texto em claro conhecido, que é possível quando essa restrição não é satisfeita:

1. Abra uma janela de terminal e acesse o diretório `/home/esruser/Arquivos do Curso/sessao-09`.
2. Visualize os conteúdos dos arquivos `1.txt`, `2.txt` e `3.txt`:

```
~$ cat 1.txt; echo
~$ cat 2.txt; echo
~$ cat 3.txt; echo
```

3. Cifre os arquivos `1.txt`, `2.txt` e `3.txt` com o algoritmo RC4:

```
~$ openssl enc -rc4-40 -in 1.txt -out 1.enc -K 0123456789
~$ openssl enc -rc4-40 -in 2.txt -out 2.enc -K 0123456789
~$ openssl enc -rc4-40 -in 3.txt -out 3.enc -K 0123456789
```

4. Analise os textos cifrados resultantes, comparando-os com os respectivos textos em claro:

```
~$ hexdump 1.txt 1.enc
~$ hexdump 2.txt 2.enc
~$ hexdump 3.txt 3.enc
```

5. Considere que o atacante tenha acesso somente aos arquivos cifrados, mas que conhece o texto em claro correspondente ao primeiro deles, isto é, o conteúdo de `1.txt`. Calculando o XOR byte a byte entre este arquivo e o `1.enc`, obtém-se o fluxo de chaves gerado pelo RC4, que não varia quando uma chave fixa é empregada. Isso pode ser executado por meio do comando abaixo, que grava o resultado no arquivo `ks.bin`:

```
~$ /usr/local/sbin/fxor 1.enc 1.txt ks.bin
```

6. Visualize o fluxo de chaves:

```
~$ hexdump ks.bin
```

7. O conhecimento do fluxo de chaves gerado implica que é irrelevante saber a chave específica que foi utilizada no ciframento, pois, calculando o XOR byte a byte entre o primeiro e qualquer texto protegido por ele, obtém-se o texto legível correspondente:

```
~$ /usr/local/sbin/fxor 2.enc ks.bin 2.plain
~$ /usr/local/sbin/fxor 3.enc ks.bin 3.plain
```

8. Visualize os conteúdos dos arquivos `2.plain` e `3.plain` e veja que correspondem aos textos em claro originais, presentes nos arquivos `2.txt` e `3.txt`, respectivamente:

```
~$ cat 2.plain; echo
~$ cat 3.plain; echo
```

9. Encerre a janela de terminal.

Nível de segurança dos algoritmos criptográficos

Indique o nível de segurança em bits das combinações de algoritmos abaixo enumeradas:

- 2-TDES, RSA-3072, RIPEMD-160.

Resposta: 2-TDES (80), RSA-3072 (128), RIPEMD-160 (80). 80

- AES-128, RSA-1024, SHA-512.

Resposta: AES-128 (128), RSA-1024 (80), SHA-512 (256).80

- AES-192, RSA-2048, SHA-256.

Resposta: AES-192 (192), RSA-2048 (112), SHA-256 (128).112

- AES-256, ECDSA-512, SHA-512.

Resposta: AES-256 (256), ECDSA-512 (256), SHA-512 (256).256

Comentário Com a segurança é definida pelo elo mais fraco, neste caso será considerado o mais fraco dos algoritmos.

A partir desse requisito, é natural questionar como se deve medir o nível de segurança de um criptossistema qualquer. A maneira mais direta leva em consideração o custo do melhor ataque conhecido; se para executá-lo, são necessárias 2n operações, é dito que o algoritmo fornece n bits de segurança.

Tabela 1. Tabela Apostila pg. 451

80	2-TDES RSA-1024 DH-1024 ECDSA-160 RIPEMD-160
112	3-TDES RSA-2048 DH-2048 ECDSA-224 SHA-224
128	AES-128 RSA-3072 DH-3072 ECDSA-256 SHA-256
192	AES-192 RSA-7680 DH-7680 ECDSA-384 SHA-384
256	AES-256 RSA-15360 DH-15360 ECDSA-512 SHA-512

Proteção inadequada de dados de domínio de pequena cardinalidade

O propósito desta atividade é constatar a velocidade com que é possível gerar um dicionário de hashes para senhas numéricas pequenas:

1. Abra uma janela de terminal e acesse o diretório `/home/esruser/Arquivos do Curso/sessao-09`.
2. Digite o comando abaixo para gerar um dicionário para números de seis dígitos:

```
~$ /usr/local/sbin/dictbuilder 6 list.txt
```

3. Abra o dicionário gerado com o gedit:

```
~$ gedit list.txt &
```

4. Veja o tamanho em MBytes do arquivo gerado:

```
~$ du -m list.txt
```

5. Encerre a janela de terminal e o gedit.

ENTREGA DA TAREFA

Para que seja considerada entregue você deve anexar a esta atividade no AVA o conteúdo do arquivo `arquivo2.txt`

Obs.: O arquivo resultado pode estar em formato de imagem ou texto

Última atualização 2020-09-02 18:44:19 -0300