HomeWork 2
Encryption Modes and Meet in the Middle

Universidade da Beira Interior
Mestrado Engenharia Informática
Ricardo Costa Oliveira, m8885

1) Compile the double s-des program(s2des.c) and experiment encrypting and decrypting some values.
The S-DES program is a simplified version of des and operates on blocks of size 8bits (i.e 1 byte) and keys of integer value 0-1023 (2^{10} i.e 10 bits). The Double S-DES program is simply an adaption of the original program to execute double s-des:

```
> s2des -d 101 123 < output > input2
> diff input input2
```

NOTES

- For the S-DES paper consult [5]
- The original code can be found at [3]
- The S-Des function was altered to always add a '\0' at the end of a string
- If you have problems with nonprintable asci characters change to reading bytes directly instead of gets()
- Read the sdes code!
- It might be useful to create a main function that only does single des.

Solution: Para a concretização dos exercícios apresentados, foram realizados primeiramente os passos seguintes:

1. Criada uma *shared library* a partir do código disponível em [3];

```
> gcc -fPIC -shared -o _sdes.so -lm sdes.c
```

2. A biblioteca partilhada foi carregada em Python e todas as funções que compõem o SDES assim como o S2DES foram desenvolvidas nessa linguagem.

2) Encrypt two strings S1="xxbeira" and S2="something" of your choice using keys of your choice and create the cypher texts.
Using a different language (python, ocaml, java, etc).

(a) Execute a brute force attack on the entire cypher text.
Output all key pairs and plain texts that where for instance the plain text contains a known word. For instance 'beira' in the example above.

Solution: O tamanho da chave define o limite superior de segurança de uma cifra por blocos.[4] Uma vez que o tamanho da chave empregue na cifra por blocos apresentada é inadequado, a utilização de uma cifra mais do que uma vez sobre a mesma mensagem pode aumentar a segurança do esquema utilizado.[4]
Com esse propósito em mente é possível utilizar uma cifra em cascata, que consiste na concatenação de duas ou mais cifras em blocos com chaves independentes.

As cifras utilizadas podem ser distintas, *general cascade of ciphers*, ou idênticas, *cascade of identical ciphers*.
Outro defecho possível passa pela utilização de *multiple encryption*, embora semelhante a uma cifra em cascata, não é imposta a utilização de chaves independentes. No entanto são usadas tipicamente chaves independentes, k_1, k_2, quando o mesmo texto limpo é cifrado duas vezes.[4] Para além disso, podem ser aplicados ambos os algoritmos da cifra (Enc, Dec). Nesse sentido para o exercício em causa (S2DES) tem-se que,

$$\mathsf{Enc}(\mathrm{x}) = \mathrm{EK2}(\mathrm{EK1}(\mathrm{x}))$$

onde EK denota a cifra por blocos E com a chave K.
Algoritmo:

1. Progressão por todo o espaço das chaves;
2. Decifra do criptograma C com cada uma das chaves do espaço da chave;
3. Descartar chaves cujo resultado da decifra não contem uma fração conhecida do texto limpo P.
4. Devolver o texto limpo e o par de chaves encontrados.

É esperado que a chave correta seja encontrada após pesquisa de metade do espaço das chaves. Para o S-DES, onde K = 10 e n = 8, é esperado que com apenas um par (P,C) a chave correta seja encontrada após 2^9 operações de decifra. Tem-se ainda que se forem conhecidas redundâncias no texto limpo é possível efetuar uma pesquisa exaustiva da chave apenas com o conhecimento de um número reduzido de criptogramas.
Demonstração:

```
> python skeleton.py --plaintext 'xxxbeira' -s2des --key 1021 --key 1022 --
    ↪ bruteforce
[(989, 958, 'xxxbeira'), (1021, 1022, 'xxxbeira')]

> python skeleton.py --plaintext 'something' -s2des --key 689 --key 937 --
    ↪ bruteforce
[(689, 937, 'something')]
```

(b) Execute a meet in the middle attack on the first character (block) of the cypher text.
Use subsequent characters to confirm or not the candidate key pair and then decrypt the cypher text.

Solution: Como referido anteriormente a pesquisa exaustiva pela chave usada investiga $\{0,1\}^{\mathrm{n}} \times \{0,1\}^{\mathrm{m}}$ chaves pelo que requer $\mathcal{O}(2^{n+m})$ operações. O ataque produzido neste exercício reduz a quantidade de operações necessárias a custo de um aumento substâncial de espaço utilizado. O *meet-in-the-middle attack* é um ataque do tipo *know plaintext attack*, inicialmente apresentado em [1] por Diffie e Hellman para uma criptoanálise ao algoritmo DES.
Sejam Enc_k e Dec_k os algoritmos de cifra e decifra utilizados pela cifra e $\mathsf{k} \in \{0,1\}^{\mathrm{n}}$ a chave empregue por ambos. Considere-se ainda o esquema de cifra múltipla que calcula o criptograma C a partir de um texto limpo P e das chaves $\mathsf{k}_1 \in \{0,1\}^{\mathrm{n}}$ e $\mathsf{k}_2 \in \{0,1\}^{\mathrm{m}}$:

$$\mathrm{C} = \mathsf{Enc}(\mathsf{k}_2, \mathsf{Enc}(\mathsf{k}_1, \mathrm{P}))$$
$$\mathrm{P} = \mathsf{Dec}(\mathsf{k}_1, \mathsf{Enc}(\mathsf{k}_2, \mathrm{P}))$$

Contudo é possível fazer uma derivação importante do esquema de cifra múltipla apresentado.

$$\mathrm{C} = \mathsf{Enc}(\mathsf{k}_2, \mathsf{Enc}(\mathsf{k}_1, \mathrm{P}))$$
$$\mathsf{Dec}'(\mathsf{k}_2, \mathrm{C}) = \mathsf{Dec}'(\mathsf{k}_2, \mathsf{Enc}(\mathsf{k}_2, \mathsf{Enc}(\mathsf{k}_1, \mathrm{P})))$$
$$\mathsf{Dec}'(\mathsf{k}_2, \mathrm{C}) = \mathsf{Enc}(\mathsf{k}_1, \mathrm{P})$$

Esta derivação permite quebrar a cifra em análise, de uma forma mais eficiente, assim como qualquer cifra múltipla que utilize duas ou mais chaves.

Algoritmo:

1. Contrução do conjunto de todos os criptogramas gerados por $\mathsf{Enc}(\mathsf{k}_1, \mathrm{P})$. Este conjunto é armazenado na tabela C';

$$\mathrm{C}' = (\mathsf{Enc}_{\mathsf{k}}(\mathrm{P}), \mathsf{k}) : \mathsf{k} \in \{0,1\}^{\mathrm{n}}$$

2. Ordenação da tabela de forma crescente pelos criptogramas;
3. Progressão por todo o espaço de chaves k_2
   (a) Obtenção do texto limpo $P' = \mathsf{Dec}'(\mathsf{k}_2, \mathrm{C})$
   (b) Pesquisa binária pela entrada na tabela C' tal que $\mathrm{P}' = \mathrm{C}'_{\mathrm{i}}$

É necessário ressalvar que para o exercício em questão seria possível substituir o ponto 3 do algoritmo por:

3. Construção do conjunto de todos os textos limpos gerados por $\mathsf{Dec}'(\mathsf{k}_2, \mathrm{C})$. Este conjunto é armazenado na tabela P'.

$$\mathrm{P}' = (\mathsf{Dec}_{\mathsf{k}}(\mathrm{C}), \mathsf{k}) : \mathsf{k} \in \{0,1\}^{\mathrm{m}}$$

4. Interseção dos conjuntos produzidos. O resultado da interseção contém o par de chaves correta $(\mathsf{k}_1, \mathsf{k}_2)$.

Caso exista mais do que um par de chaves candidata, estas são testadas com outros pares de texto limpo-criptograma (P,C). Normalmente nesta situação é necessário apenas mais um par (P,C).

Sublinha-se que, embora no caso em análise não exista essa necessidade, os dados a armazenar devem dizer respeito à componente da equação mais fácil de calcular (e.g. 3DES). Além disso quando a memória disponível é inferior à necessária para a realização do algoritmo o conjunto C' pode ser framentado em porções da memória disponível.[2]

Este ataque requer $\mathcal{O}(2^n + 2^m)$ operações de cifra para compor os conjuntos C' e P', ao contrário do que é necessário no exercício da alínea anterior $\mathcal{O}(2^{n+m})$. Uma vez que n = m são necessárias $\mathcal{O}(2^{n+1})$ operações de cifra, o dobro do que seria necessário para quebrar o S-DES. Conclui-se portanto que o S2DES promove uma segurança consideravelmente inferior à quantidade de bits nas chaves pelo esquema da cifra.

Demonstração:

```
> python skeleton.py --plaintext 'xxxbeira' -s2des --key 104 --key 692 --mim
K1: 104  K2: 692
```

3) Implement S-DES in mode CBC (Basically use the sdes function).
You will need to alter the original program to create the function

char * cbcsdesCBC(char *message, ... key, ... mode)

Note that in mode CBC the first character of the message is the IV!
Solution: **Demonstração:**

```
> python skeleton.py --plaintext 'xxxbeira' --cbc --key 1021
plaintext:  xxxbeira
ciphertext: 785f6cffbf409fc6
```

4) Hand In by E-Mail

- Report describing the work done (in PT or EN)
- Programs written

References

[1] W. Diffie and M. E. Hellman. Special feature exhaustive cryptanalysis of the nbs data encryption standard. *Computer*, 10(6):74–84, June 1977.

[2] S. Even and O. Goldreich. On the power of cascade ciphers. *ACM Trans. Comput. Syst.*, 3(2):108–116, May 1985.

[3] fvicente. sdes. `https://github.com/fvicente/sdes/blob/master/sdes.c`, 2009.

[4] Alfred J. Menezes, Scott A. Vanstone, and Paul C. Van Oorschot. *Handbook of Applied Cryptography*. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 1996.

[5] Edward F. Schaefer. A simplified data encryption standard algorithm. *Cryptologia*, 20(1):77–84, 1996.