SFWR 4C03 – Assignment 1 Report

Andrew Lutz
0664122
lutza
February 7, 2011

DES Implementation in Python

I solved this assignment by breaking up the algorithm into 4 main sections, the first section was the transformation of a user inputted string into a 64-bit representation. I accomplished this by getting the ascii values of each character and using the handy built in bitwise operators of python to form a list of bits, of length 64. The second part consisted of generating the 16 sub-keys used in the plaintext ciphering. This was accomplished by rotating and permuting the key 16 times. The third section was the ciphering of the plaintext. This consisted of a loop that executed 16 times, each time feeding the right 32 bits into f function, permuting the 32 bits into a 48 bit string, xor'ing the result with the respective sub-key and feeding the result into the sbox function, which returned a 32 bit value. The final value was then concatenated with the left 32 bits and permutated once more. Resulting in a 64 bit cipher of the original user input.

***Notes on program. Execute with ./des.py . My program has two modes, normal and info. To run in –info mode, enter the –info flag when executing the program.

```
Ex.   ./des.py –info
```

This will display data being processed at each stage.

Testing

Test 1:
Contents of hello.txt: 12345678

```
Andrew–Lutzs–MacBook–Pro:~ Andrew$ openssl des–ecb –nosalt –nopad –K
3132333435363738 –iv 2011 –in hello.txt | xxd –b
0000000: 10010110 11010000 00000010 10001000 01111000 11010101  ....x.
0000006: 10001100 10001001
```

```
The above bit string in hex is:     96 d0 02 88 78 d5 8c 89
```

My program:

```
Andrew-Lutzs-MacBook-Pro:desktop Andrew$ ./des.py
Enter 64-bit string (plaintext):12345678
Enter 64-bit string (key):12345678


Your ciphertext in hex is...
                        96 d0 02 88 78 d5 8c 89
```

-Results match, therefore correct!

Test 2

Contents of hello.txt: asdfghjk

```
Andrew-Lutzs-MacBook-Pro:~ Andrew$ openssl des-ecb -nosalt -nopad -K
7177657274797569 -iv 2011 -in hello.txt | xxd -b
0000000: 00100101 00011101 10000011 01000000 00000001 10001001  %..@..
0000006: 01000011 01111000

The above bit string in hex is:     25 1d 83 40 01 89 43 78
```

My program:

```
Andrew-Lutzs-MacBook-Pro:desktop Andrew$ ./des.py
Enter 64-bit string (plaintext):asdfghjk
Enter 64-bit string (key):qwertyui


Your ciphertext in hex is...
                        25 1d 83 40 01 89 43 78
```

-Results match, therefore correct!

For this assignment I followed tutorials on the following websites:

http://people.eku.edu/styere/Encrypt/JS-DES.html#ffunc
http://www.eventid.net/docs/desexample.asp
http://www.comms.scitech.susx.ac.uk/fft/crypto/des_algorithm_details.txt

I used SBox and permutation tables found in the publicly licensed pyDes implementation.

http://sourceforge.net/projects/pydes/