Homework 3

Total Points: 180 **Due Date:** Friday, November 13, 1998

1. (*100 points*) Please implement the Vigenère, Beaufort, and Variant Beaufort ciphers described in class. You are to write two programs. The first, *myencrypt*, enciphers a message. The second, *mydecrypt*, deciphers a message. Each accepts command line parameters, as follows:

   myencrypt *mode key plaintext*

   mydecrypt *mode key ciphertext*

   Here, *key*, *plaintext*, and *ciphertext* are all CAPITALIZED words, and *mode* is **–v** for the Vigenère cipher, **–b** for the Beaufort cipher, and **–r** for the Variant Beaufort cipher (also called the Reverse Beaufort cipher). If any command line argument is malformed, or an error occurs in your code, print an appropriate error message and exit with an exit code of 1. Otherwise, exit with an error code of 0. Please write your code robustly, and comment it!

   Submitting your programs. If you write your programs in a compilable language (like C), you ***must*** submit a makefile to compile your source code. A single *make* should generate both *myencrypt* and *mydecrypt* executables. ***Do not*** submit *myencrypt* or *mydecrypt* executables unless they are written in a scripted language (such as shell script or Perl). Your program should run correctly on any of the CSIF machines (that is, on the DECs, the HPs, and the SGIs).

2. (*50 points*) A *weak key* is a key that is its own inverse. For example, for the Cæsar cipher, the letter M (13) is a weak key, as a message may be enciphered and then deciphered using the same key. The DES has four *weak keys*. Prove that 0x0000000000000000 (the key of all 0 bits) and 0xffffffffffffff (the key of all 1 bits) are both weak keys.

3. (*30 points*) Consider the RSA cipher with $p = 5$ and $q = 7$. Show that $d = e$ for all choices of public key d and private key e.

Extra credit

4. (*50 points*) Find the DES's other two weak keys. Note that we ignore the *parity bit* for the keys when determining if keys are different; that is, 0x8080808080808080 and 0x7f7f7f7f7f7f7f7f don't count (as these are 0x0000000000000000 and 0xffffffffffffffff, respectively, with the parity bit changed).