Homework 2

Due: February 5, 2016 **Points:** 100

1. (*12 points*) Classify each of the following as an example of a mandatory, discretionary, or originator controlled policy, or a combination thereof. Justify your answers.

   (a) The file access control mechanisms of the UNIX operating system

   (b) A system in which no memorandum can be distributed without the creator's consent

   (c) A military facility in which only generals can enter a particular room

   (d) A university registrar's office, in which a faculty member can see the grades of a particular student provided that the student has given written permission for the faculty member to see them.

2. (*15 points*) Given the security levels TOP SECRET, SECRET, CONFIDENTIAL, and UNCLASSIFIED (ordered from highest to lowest), and the categories A, B, and C, specify what type of access (read, write, both, or neither) is allowed in each of the following situations. Assume that discretionary access controls allow anyone access unless otherwise specified.

   (a) Paul, cleared for (TOP SECRET, { A, C }), wants to access a document classified (SECRET, { B, C }).

   (b) Anna, cleared for (CONFIDENTIAL, { C }), wants to access a document classified (CONFIDENTIAL, { B }).

   (c) Jesse, cleared for (SECRET, { C }), wants to access a document classified (CONFIDENTIAL, { C }).

   (d) Sammi, cleared for (TOP SECRET, { A, C }), wants to access a document classified (CONFIDENTIAL, { A }).

   (e) Robin, who has no clearances (and so works at the UNCLASSIFIED level), wants to access a document classified (CONFIDENTIAL, { B }).

3. (*20 points*) Use the character frequencies from Figure 10–1 to decipher the following ciphertext, which was enciphered using the Caesar cipher: TEBKFKQEBZLROPBLCERJXKBSBKQP.

4. (*20 points*) Alice and Bob are creating RSA public keys. They select different moduli n_{Alice} and n_{Bob}. Unknown to both, n_{Alice} and n_{Bob} have a common factor.

   (a) How could Eve determine that n_{Alice} and n_{Bob} have a common factor without factoring those moduli?

   (b) Having determined that factor, show how Eve can now obtain the private keys of both Alice and Bob.

5. (*33 points*) This problem asks you to implement a buffer overflow attack on a program. In the Resources area of SmartSite (or the Homework area of the nob.cs.ucdavis.edu class web site) is a program *bad.c* (also see below). This program contains a buffer overflow vulnerability; see the call to *gets*(3) at line 13. Your job is to exploit the overflow by providing input to the running process that will cause the program to invoke the function `trap` (which, you may notice, is not called anywhere else). You will know you've succeeded when you run the program, give it your input, and it prints "`Gotcha!`"

   The following questions will help guide you. Please turn in your answers to them, a hex dump of the input you use to call `trap`, and a typescript or screen shot of you running the program *bad*, giving it your input, and showing its output.

   (a) What is the address of the function `trap`()? How did you determine this?
   (b) What is the address on the stack that your input must overwrite (please give both the address of the memory location(s), and their contents)? How did you locate this address?
   (c) What is the address of `buf`?
   (d) The *sled* is the input you give to alter the return address stored on the stack. What is the minimum length your sled must be?

   This is a listing of *bad.c*.

```
#include <stdio.h>
#include <stdlib.h>

int trap(void)
{
        printf("Gotcha!\n");
        exit(0);
}

int getstr(void)
{
        char buf[12];
        gets(buf);
        return(1);
}

int main(void)
{
        getstr();
        printf("Overflow failed\n");
        return(1);
}
```

Extra Credit

6. (*20 points*) Euler's generalization of Fermat's Little Theorem says that, for integers a and n such that a and n are relatively prime, $a^{\phi(n)} \bmod n = 1$. Use this to show that deciphering of an enciphered message produces the original message with the RSA cryptosystem. Does enciphering of a deciphered message produce the original message also?

   Hint: You must prove both the case where the message m and n are relatively prime, and where they are *not* relatively prime.