Homework 3

Due: February 22, 2016 **Points:** 100

1. (*40 points*) Consider the Otway-Rees protocol. Assume that each enciphered message is simply the bits corresponding to the components of the message concatenated together. So, for example, in the first message, one must know the names "Alice" and "Bob", and the length of the random numbers r_1 and n, to be able to parse the portion of the first message that is enciphered with k_{Alice}. The separate parts of the enciphered message have no indicators; the recipient is expected to determine them.

   (a) Consider Alice when all 4 steps of the protocol have been completed. How does Alice know that steps 2 and 3 have taken place?

   (b) Massicotte asks us to assume that an adversary Edgar is impersonating Bob, and has sufficient control over the exchange so that he receives the messages intended for Bob. Bob never sees them. What components of the protocol does Edgar know — that is, does he know r_1, r_2, n, or $k_{session}$, or the names of "Alice" and "Bob"? How?

   (c) Given this, in step 4 of the protocol, how might Edgar provide Alice with a session key that he knows?

   (d) How might someone fix this?

2. (*30 points*) Revisit the example for x := y + z in Section 17.1.1. Assume that x does not exist in state s. Confirm that information flows from y and z to x by computing $H(y_s \mid |x_t)$, $H(y_s)$, $H(z_s \mid x_t)$, and $H(z_s)$ and showing that $H(y_s \mid x_t) < H(y_s)$ and $H(z_s \mid x_t) < H(z_s)$.

3. (*30 points*) This problem asks you to extend the buffer overflow attack from the last homework assignment. In the Resources area of SmartSite (or the Homework area of the nob.cs.ucdavis.edu class web site) is a program *realbad.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 `runcom` and cause the *system*(3) function to be executed with a command embedded in the input you have given. You must pass in a parameter that is a Linux command, which the program will then execute. (I recommend the command *id*(1).)

   Please turn in the following:

   (a) A hex dump of the input you use. Please also show where the parameter to `trap`() is in your input.
   (b) A screenshot of the program's output for that input.

   realbad.c

   This is a listing of *realbad.c*.

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  void runcom(char *cmd)
5  {
6          system(cmd);
7          exit(0);
8  }
9
10 int getstr(void)
11 {
12         char buf[12];
13         gets(buf);
14         return(1);
15 }
16
```

```
17  int main(void)
18  {
19          getstr();
20          runcom("echo Overflow failed");
21          return(1);
22  }
```

Extra Credit

4. (*20 points*) Consider the RSA cryptosystem. Show that the ciphertexts corresponding to the messages 0, 1 and $n-1$ are the messages themselves. Are there other messages that produce the same ciphertext as plaintext?