Homework 3

Due: November 8, 2016 (note extension!) **Points:** 100

Questions

Remember to justify your answers where appropriate.

1. (*20 points*) The relations *certified* (see ER1) and *allowed* (see ER2) can be collapsed into a single relation. Please do so and state the new relation. Why doesn't the Clark-Wilson model do this?

2. (*28 points*) Consider the RSA cipher with $p = 5$ and $q = 7$. Show that $d = e$ for all choices of public key e and private key d.

3. (*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.

4. (*32 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?

Extra Credit

5. (*30 points*) Assume that a cryptographic checksum function computes hashes of 128 bits. Prove that the probability is 0.5 that at least one collision will occur after hashing (2^{64}) randomly selected messages.