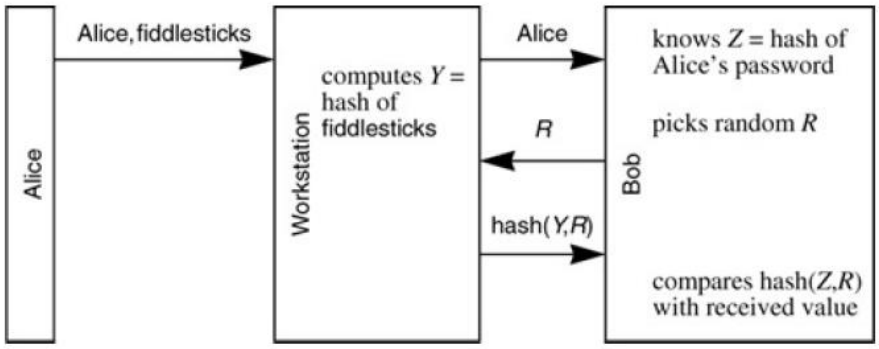
Homework 3

Due: 7:00PM, Tuesday, 11/24/2015

Name: ID:

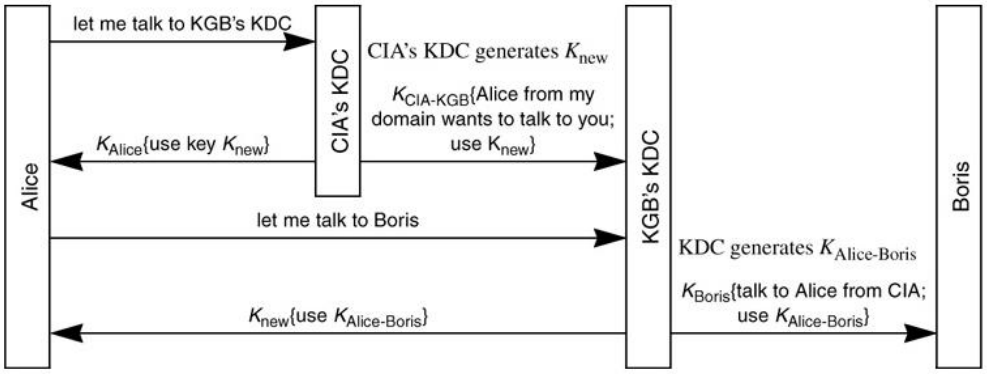
1. In class, we asserted that it is extremely difficult, without public key cryptography, to have an authentication scheme which protects against both eavesdropping and server database disclosure. Consider the following authentication protocol. Alice knows a password. Bob, a server that will authenticate Alice, stores a hash of Alice's password. Alice types her password (say fiddlesticks) to her workstation. The following exchange takes place:



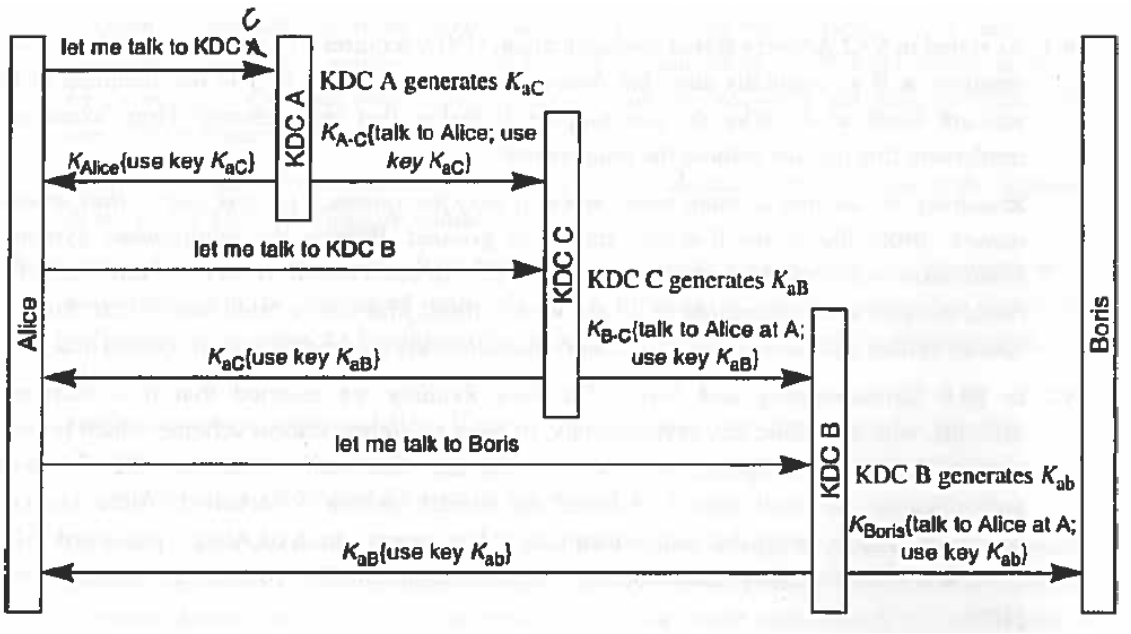
Is this an example of an authentication scheme that isn't based on public key cryptography and yet guards against both eavesdropping and server database disclosure? (Here, we do not consider eavesdropping between Alice and the workstation) (10 Points)

No. Knowledge of the hash of Alice's password, which is stored in the server database, is sufficient to impersonate Alice's workstation to Bob.

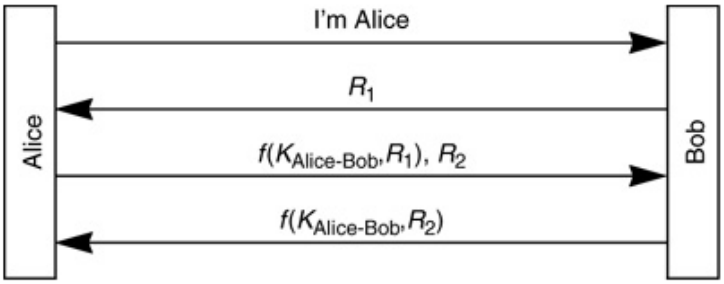
2. Assume that Alice wants to talk to Boris through a chain of three KDCs (Alice's KDC, a KDC that has shared keys with both Alice's KDC and Boris's KDC, and finally, Boris's KDC). Give the sequence of events necessary to establish communication. Hint: below is how Alice and Boris establish communication through two KDCs (Alice's KDC and Boris's KDC). (10 Points)



The protocol looks something like this:

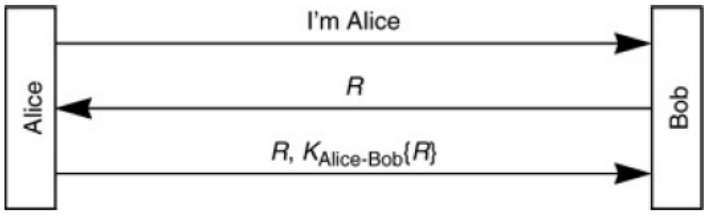


3. Is the following authentication protocol secure against reflection attack? Why? (10 Points)



Yes. Because here Alice is always the first to prove its identity. Even Alice opens another connection with Bob, she still has to compute $f(K_{Alice\text{-}Bob}, R)$ first.

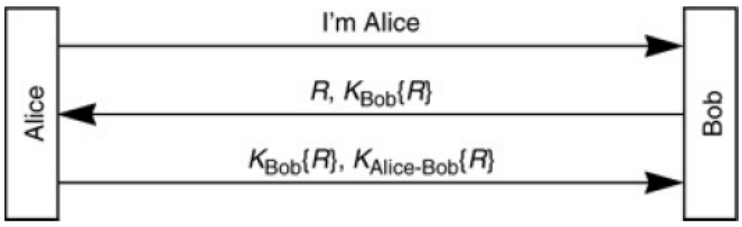
4. Suppose we are using a three-message mutual authentication protocol, and Alice initiates contact with Bob. Suppose we wish Bob to be a stateless server, and therefore it is inconvenient to require him to remember the challenge he sent to Alice. Let's modify the exchange so that Alice sends the challenge back to Bob, along with the encrypted challenge. So the protocol is:



Is this protocol secure? (10 Points)

No. An eavesdropper can replay Alice's message at any time. If Bob cannot remember his current challenge, he won't know that the challenge response is to a previous challenge.

5. Let's modify the protocol from the previous problem so that Bob sends both a challenge, and a challenge encrypted with a key that only he knows, to Alice:



Is this protocol secure? (10 Points)

No, for the same reason.

6. Design a two-message authentication protocol, assuming that Alice and Bob know each other's public keys, which accomplishes both mutual authentication and establishment of a session key. (10 points)

Alice picks a session key *K* and sends along a timestamp *T*. She encrypts *K* with Bob's public key and signs the entire message. Bob responds with the timestamp encrypted with *K*.

Bob knows it is Alice from the signature and timestamp. Alice know it is Bob because only he can decrypt *K*.

7. Some old cellular phones are vulnerable to a fraud known as "cloning". The protocol cellular phones use is that a phone transmits its telephone number followed by a cleartext password. The phone company checks its database of phone number/password to make sure the phone is legitimate before allowing the call to go through. The phone number is the one billed. The problem is, anyone can eavesdrop on cellular phone transmissions and clone such a phone by using the overhead phone number/password. Suggest a design based on public key cryptography and one based on secret key cryptography to remove the vulnerability. Can you guard against the phone company database being stolen? (20 points)

   Instead of transmitting a cleartext password, the phone should receive a challenge from the phone company and respond with the encrypted challenge. With public key technology, the phone encrypts the challenge with its private key and the phone company database contains only the public key, so a cloner gains nothing by stealing the database. With secret key technology, the phone encrypts the challenge with the secret. In this case, theft of the database enables the cloning of phones.

8. With CBC, if one ciphertext block is lost, how many plaintext blocks are affected? With PCBC, why do things get back in sync if c_n and c_{n+1} are switched? How about if a ciphertext block is lost? How about any permutation of the first n blocks? (20 Points)

   In CBC decryption, each ciphertext block affects two plaintext blocks, one through decryption and one through XOR.

   In PCBC decryption, each ciphertext block affects the corresponding plaintext block by XOR its decryption, while it affects all following plaintext blocks by XOR the XOR result of it and its decryption. Thus, a set of ciphertext blocks affects the following plaintext blocks in a manner independent of the order of ciphertext blocks within the set. The effect is just an XOR of the XOR of all the ciphertext blocks and their decryption.