CS 554 Homework #4

Due: Thursday, April 14

Alexander Powell

1. No, because while this authentication scheme is not based on public key cryptography, it still fails to guard against eavesdropping. For example, a third person like Trudy, could still look over Alice's shoulder and watch her type the password "fiddlesticks" into the computer.

2. To establish a connection between Alice and Boris using a chain of three KDCs, it would look somewhat similar to the figure in the book. First, Alice would send a message to her KDC, with the intent to communicate with Boris. At this point KDC_{Alice} would send a message encrypted with Alice's key, K_{Alice} to the shared $KDC_{\text{Alice and Boris}}$. Next, $KDC_{\text{Alice and Boris}}$ would send Alice's message to KDC_{Boris} encrypted with Boris's key, K_{Boris}. Further communication would follow this pattern until the end of the conversation.

3. There is an advantage because if the user is required to enter several passwords every time they login, it will take more time for the eavesdropped to acquire enough valid password to successfully impersonate the user. This means that both the user and the system can use the list of passwords for longer than usual, although it's not as convenient for the user. However, if the eavesdropper is persistent and successfully overhears enough passwords to login, this opens up a security risk because there's a longer amount of time before the passwords are changed. All in all, there are both pros and cons to this method.

4.
   - $A \oplus R$ is not secure for a session key, because if someone discovers it they also discover A.
   - $\{R + A\}_A$ is secure for a session key.
   - $\{A\}_A$ is not secure for a session key because it is the same for all sessions.
   - $\{R\}_{R+A}$ is secure for a session key.

5. No, this protocol is not secure because it is susceptible to the replay attack. That is, if there's some eavesdropper listening, they can replay Alice's message at any time they want. If bob remembers his current challenge, he won't know the response is to a previous challenge.

6. This protocol is still not secure from eavesdropping because someone could listen to the response from Bob, which contains the encrypted challenge with Bob's key, and well as the one encrypted with their shared key.

7. The two-message authentication protocol that achieves both mutual authentication and establishment of a session key is descibed as follows: First, Alice has to choose a session key, K, and sends a message signed with that key and encrypted by Bob's public key. Additionally, a timestamp must be placed on the signature of the message. Finally, Bob responds to Alice with the timestamp, encrypted with the session key, K.

8. The Expanded Needham-Schroeder Protocol can be shortened to a 6 message protocol be removing the 7th message, because by the 7th message, Bob already knows that he is talking to Alice. This is because when Alice performs a simple operation on the nonce, and then re-encrypts it, she has already verified her true identity as Alice, so there's no need for the last operation of $K_{AB}\{N_3 - 1\}$.

9. For protocol 11-18, the Needham Schroeder protocol, N_1 must be unpredictable. If not, then a third party like Trudy could impersonate the KDC by sending Alice an old ticket that Trudy had stolen previously.

For protocol 11-19, the Expanded Needham Schroeder protocol, the nonce N_1 must be unpredictable.

For protocol 11-20, the Otway-Rees protocol, the nonce N_C must be unpredictable.

Finally, for protocol 11-21, Kerberos, the nonce N_1 must be unpredictable.

10. Even if the message was encrypted, if the phone was being wiretapped (meaning someone is eavesdropping), that person can see both the message being sent as well as the ecrypted message, which gives them sufficient information to mount a dictionary attack to gain the key used to ecrypt the message. At this point, the entire conversation is open to the person doing the wiretapping.

11. If Bob crashes before recieving Alice's reply, Bob's database should be overwritten just after he sends Alice the value n she uses to take the hash of the password. In this way, an impersonator like Trudy wouldn't be able to impersonate Alice. This is one of the advantages of adding salt to pad the password.

12. In protocol 12-3, Alice can be assured that the other side has the information stored at Bob because Bob recieves $2^a mod p$ from Alice and can compute the hash of $(2^{ab} mod p, 2^{bW} mod p)$, because he knows b, p, $2^a mod p$ and $2^W mod p$. Also, someone who has stolen Bob's database cannot impersonate Alice to Bob because Alice can compute the hash$'(2^{ab} mod p, 2^{bW} mod p)$.

13. To compute K in the SRP protocol, Alice must first begin by choosing some a, and then sending her first message to Bob containing $g^a mod p$. Alice also computes W from the password. On the other side, Bob stores Alice's $g^W mod p$ along with her first message, ans chooses some b, a challenge c_1, and a 32 bit number u. After he has done this, he sends $g^b + g^W mod p$, along with u and c_1 back to Alice. At this point, both Alice and Bob have enough information to compute the K value. They both compute K using the following formula:

$$K = g^{b(a+uW)} mod p.$$

14. To compute K, first Alice must choose some value a, and send Bob her message along with $g^a mod p$, much like in the SRP protocol. On the other side, Bob stores Alice's message along with $g^W mod p$. Next he chooses a value b and a challenge c_1, and then sends Alice $g^b mod p$ and c_1. Once Alice has recieved this from Bob, she can compute W from the password. At this point Alice and Bob can both compute K by taking the hash of $g^{ab} mod p$ and $g^{Wb} mod p$. This protocol is insecure because someone impersonating Bob could implement a dictionary attack. Because W is a function of Alice's password, be could feed a list of possible passwords in until one is valid.

15. If Alice is the user, then Alice's TGT is $K_{Alice}\{Alice, S_A\}$, and Alice's workstation has created a session key S_A. This is a problem for KDC when it tries to get the session key from TGT. To cope with this, Alice should always pair her name with the TGT when sending messages. There is no difference in security between the two schemes except when Alice changes her password, and therefore her key. This is because knowledge of KDC is all that is required for mutual authentication of Alice and the KDC. Using this new scheme, if Alice thinks that someone has figured out her password so she changes it, the TGT would be invalidated immediately. However, tickets could still be used after the password is changed. Because the TGT is invalidated after her password is changed, Alice would have to send a message containing the old TGT and the new key encrypted with the same S_A. After this is done, a new TGT can be computed.

16. The only thing the bad guy would need to do is to replace Alice's encrypted key with his own encrypted key.

17. This is because the authenticator is used to prove knowledge of the session key. However, without knowing the session key, the person asking for the ticket is unable to decrypt the credentials field to get the ticket. Also, the authenticator is used to prove knowledge of the shared key. This is important because further messages during the session may not be encrypted, which would not leave an opportunity for encryption.

18. It was necessary for Bob to increment the value before re-encrypting and sending back because if he didn't increment it and just encrypted the same value, the message sent back would be the same that Alice sent, so an attacker could respond correctly without knowing how to decrypt it. This is not necessary in Kerberos V5 because in AP_REP, the encrypted section is encrypted with the key inside the ticket from the AP_REQ, unless a SUBKEY field is included in the AUTHENTICATOR from the AP_REQ, in which case it is encrypted with the subkey. No, it would not have been as secure for Bob to send back the contents of the checksum field encrypted and not incremented.

19. In order for B/Y/Z/A/C to find a path to A/C/Y, it must trust B/Y/Z/A and A/C. By following that path, it has made a connection with A/C/Y.

20. When a certificate is expired, its revocation status is no longer published. That is, the certificte may have been revoked a long time ago, but it will no longer be included in the CRL. Certificate expiration date is the cut-off date for CRL inclusion. So, it's important to give certificates an expiration date to keep CRL size bounded. Also, you can't trust an expired certificate because you can't check its revocation status.