Sam, Antonio, Omar, René

Problem 1

In the protocol, Alice sends a fresh nonce N_A to Bob encrypted with key K, Bob sends a fresh nonce N_B encrypted with key K as well as N_A as a response, and Alice sends back N_A to Bob. A problem arises in the last step of the protocol, where Alice sends back N_A to Bob, an unencrypted nonce is sent, leaving the message vulnerable for tampering, as this would go undetected due to the lack of encryption. If the attacker intercepts and changes the message here at the last step, Bob has no way of verifying the authenticity of this message when it arrives, potentially leading to incorrect conclusions about the shared key. Encrypting the message in the last step would prevent this, not letting the attacker tamper with this message undetected. The way the protocol works as it currently does leads to a lack of mutual authentication. On the basis of these points, one can conclude that A and/or B can reach the wrong conclusion if attackers are present.

Problem 2

We think the protocol provides a secure shared key. K_AB is safely transmitted, encrypted with the individual keys A and B each shared with the KDC. This guarantees that only A and B have access to K_AB. Additionally, the combination of nonces along with the encryption process ensures confidentiality and authentication, which are important for maintaining a secure communication channel. The nonce manipulation, involving incrementing and decrementing by A and B respectively, reinforces this security by providing a dynamic element that safeguards against replay attacks. Overall, this design maintains the reliability and trustworthiness of the shared key for confidential communication.

Problem 3

Assuming only Dolov-Yao attackers mode, unique keys, and that the principals A, B, and S ignore any message they receive that contains a timestamp that is more than 5 seconds old here is how this attack can be shown:

1. Client A to Server:

A --> S: A, {T_A, B, K_AB}K_AS   where T_A is current time.

2. Interception of the Original Message:

The intruder T intercepts the initial message from Client A to the server, it can be done because the Dolev-Yao attacker mode allows the intruder T to intercept messages by controlling the network and the intruder T can capture the message sent from Client A without needing to break any encryption, simply by monitoring the data surfing through the network.

3. Storage, Modification and Reintroduction to the network:

- The intruder T holds onto this message instead of letting it proceed immediately.
- The intruder T can now modify the message, if this is his objective, modifying parts of the message that are not protected by encryption or are not crucial for the message's integrity check.
- The intruder T waits for the right moment, in this case just before the 5-seconds time when the message will be ignored so this ensures the server processes the message and forwards it to the Client B with a timestamp that is considered fresh by Bob's standards, even though K_AB suggested by Client A (and intercepts by the intruder T) is not newly generated at that moment, and it is sent into the network, so the server receives the message from the intruder T and not the original one.

4. Server to Client B:

When the server receives the replayed message from the intruder T (thinking it's from the client A), generates a new timestamp "T_S" and forwards the message to Bob:

S --> B: {T_S, A, K_AB}K_BS where T_S is current time.

Since Client B checks the timestamp T_S from the server and finds it fresh, he accepts K_AB as the new shared key with Client A. However, the instance of K_AB is actually old, chosen by the intruder T at the time of the initial interception.