Attack

Key idea : undefined behavior when decrypting with a symmetric key.
As per what the teacher said : when decrypting with a symmetric key that is not the right one, the decryption might not fail.

Attack :
Bruteforce with a random symmetric key “K_IB” the message that B should receive in the third step in order to make B believe that A sent it a secret that is not the right key.

Exact attack scenario :

- A initiates the scenario as usual, sending {|A, N_A|}_K_AB
- *I* copies it and stores it for later
- *I* blocks the communication between A and S.
- Now *I* will act as S. Bruteforce starts by sending the following repeatedly until B emits {|<gibberish> + 1|}_<gibberish_key> :
    - I -> B : {|A, N_A|}_K_AB
    - {|<random value at each iteration coded on the same amount of bits as an identity>, <whatever>, <whatever>, <whatever>|}_K_IB

Like this, since there is no bruteforce protection, here is what will be happening :

- No size (in bytes) is given as to how much it should be for an identity. It is safe to assume that it is a small value (since identities are characters (1 byte)). It may be more in this scenario but still a low value.
- By repeatedly sending the two steps described before, B will think that A is trying to communicate something (first step). Then it will try to decrypt the message that uses K_IB as a symmetric key. This will lead to “correct” decryption failure (because B will use K_BS as a key to decrypt this message). In usual decryption scenarios, no error is raised if the wrong key is used. Therefore it suffices that the first few bytes end up decrypting as “A” and B will think that it successfully decoded the message from A. **THIS IS A VERY LIKELY SCENARIO** because identities are easily bruteforce-able because they are very small bitstrings. So it will send back to A {|<gibberish that it thinks is the nonce> + 1|}_<gibberish_key> (hence this is the end of bruteforce condition).

In this scenario, we managed to make B think that it received the secret <gibberish_key> from A even though it was not the case. *I* does not learn anything but manages to trick B into thinking that it received something from A. That is authentication failure.

How to fix (multiple approaches) :

- Enforce long bitstrings to code identities
- Have some brute forcing protection
- Make it so that A will send the key K_AB only if it is sure that B knows it is talking with A through the use of nonces AND identities (making it harder to bruteforce)
- Use a symmetric algorithm that fails when wrong key is used