1 Problem Statement

There is a server that runs in two phases:

- In the first phase (registration phase), users are registered to the server and obtain symmetric shared keys for communicating with the server later. The server issues to each user a unique user identifier that it links to the shared key.
- In the second phase (online phase), a subset of users start an interaction (e.g. a conference, a game, or a voting), where all communication is mediated by the server, using the symmetric keys shared in the registration phase. They are sending messages that will be interpreted by the server in a certain way (which is not important for the model).

Using Proverif, model the protocol above with at least three honest parties (Alice, Bob, and Chris). The protocol should satisfy the following properties:

1. If an honest user finishes registration with an identity UID linked to a shared key K, then so does the server.
2. Ensure that the previous property is injective, i.e. any (UID,K) pair issued by the server will be accepted by at most one honest user.
3. Any party participating in the online phase must have been pre-registered.
4. If the server thinks that an honest party A has sent a message M in the online phase, then A must have indeed sent M.

2 Solution Steps (A Possible Approach)

2.1 The initial key exchange protocol

We start from a simple key exchange protocol, mentioned in the previous lectures. The initial version of the protocol (let us denote it P) is given below. The Proverif source of this protocol can be found at https://courses.cs.ut.ee/MTAT.07.014/2020_fall/uploads/Main/basicexample_notypeflaws.
2.2 The combined key exchange protocol

Each honest user initiates an instance of protocol P with the server. While the server’s public key may come from a trusted source, the users’ public keys are not linked to any identities. The message $M$ that the participants exchange in the original protocol can be instantiated with a fresh UID that the server generates for a user. Verify that the properties (1) and (2) hold.
2.3 Adding the online phase

Create another server process for the online phase. The keys generated in the registration phase can be stored e.g. in a table for future reference. For simplicity, the server may select the set of active participants himself. First, let us try to let the parties use the pre-shared key to encrypt their message.

Introducing more encrypted messages can introduce collisions, where the attacker can use the new messages to break properties (1) and (2). This can be solved by adding special labels to messages to prevent from messing them up, or introducing phases. After fixing the problem, verify that the property (3) holds as well.

2.4 Enhancing the online phase

Using pre-shared keys to encrypt the messages is still not enough to prove liveness of a participant, since the attacker may apply replay attacks and reuse messages that have been sent in the previous rounds. To avoid that, the server can present to each user a challenge that can be answered only knowing the shared key.