Part (a):

If user A and all users *B1 - Bn* share a single secret key *k*, each recipient can use this key *k* to verify the MAC tags on the messages received from A. However, this approach is insecure because any user *Bi* can create a forged message and calculate a valid MAC using the same key *k*.

As a result, for example, *B1* cannot tell if a message with a valid MAC was genuinely sent by A or if it was forged by another user *Bj*. This lack of differentiation undermines the trustworthiness of the messages, as any user can impersonate A by simply using the shared key to generate a valid MAC. Thus, this system does not provide adequate security or integrity for the messages being transmitted.

Part (b):

To prevent the problem mentioned in part (a), each user *Bi* needs to have only a portion of the keys from a larger set of keys. This means that each user will receive a specific subset *Si* of keys.

The main goal of this arrangement is to ensure that no single user *Bi* has access to all the keys in the complete set *S*. If one user were to possess all the keys, they could potentially create or forge a valid Message Authentication Code (MAC) independently. This would compromise the integrity of the messages sent by A.

By limiting each user's access to only a part of the key set, it becomes impossible for any individual user to generate a valid MAC on their own. Instead, multiple users must collaborate and share their keys to forge a MAC. This setup enhances the security of the system by preventing unauthorized message creation, thus safeguarding the integrity of communications from A.

Part (c):

To satisfy the condition when *n* = 10 with *m* = 5 (keys), we need to construct subsets such that each user can verify the MACs they receive, but no single user has all 5 keys. One possible construction is to assign each subset *Si* such that each user shares a unique combination of 3 out of the 5 keys. For example:
S1={k1,k2,k3}
S2={k1,k2,k4}
S3={k1,k2,k5}
S4={k1,k3,k4}
S5={k1,k3,k5}
S6={k1,k4,k5}
S7={k2,k3,k4}
S8={k2,k3,k5}
S9={k2,k4,k5}
S10={k3,k4,k5}
Each user has a different combination of 3 keys, and it requires access to all subsets to reconstruct the full set of keys.

Part (d):

The scheme described in part (c) becomes vulnerable if users are allowed to collude. For example, if users B1 and B2, who each have different subsets of keys, decide to share their keys, they can combine their access to form a larger set.

Let's say:

- User *B1* has the keys S1={k1,k2,k3}
- User *B2* has the keys S2={k1,k4,k5}

If *B1* and *B2* collude and share their keys, they could collectively access:

S={k1,k2,k3,k4}

With this combination of keys, they now have enough to forge valid MAC tags for messages. This means they could create messages that appear to be from user A, effectively impersonating A and compromising the integrity of the system.

This highlights a critical flaw in the scheme: it assumes that users will not cooperate or share their keys. If they do, they can bypass the security measures put in place, undermining the entire system. To maintain security, additional measures must be implemented to prevent such collusion among users.