Entity Authentication

INTRODUCTION

Entity authentication is a technique designed to let one party prove the identity of another party. An entity can be a person, a process, a client, or a server. The entity whose identity needs to be proved is called the claimant; the party that tries to prove the identity of the claimant is called the verifier.

Data-Origin Versus Entity Authentication

There are two differences between message authentication (data-origin authentication), discussed in Chapter 13, and entity authentication, discussed in this chapter.

1) ***Message authentication might not happen in real time; entity authentication does.***
2) ***Message authentication simply authenticates one message; the process needs to be repeated for each new message. Entity authentication authenticates the claimant for the entire duration of a session.***

Verification Categories

Something known

Something possessed

Something inherent

PASSWORDS

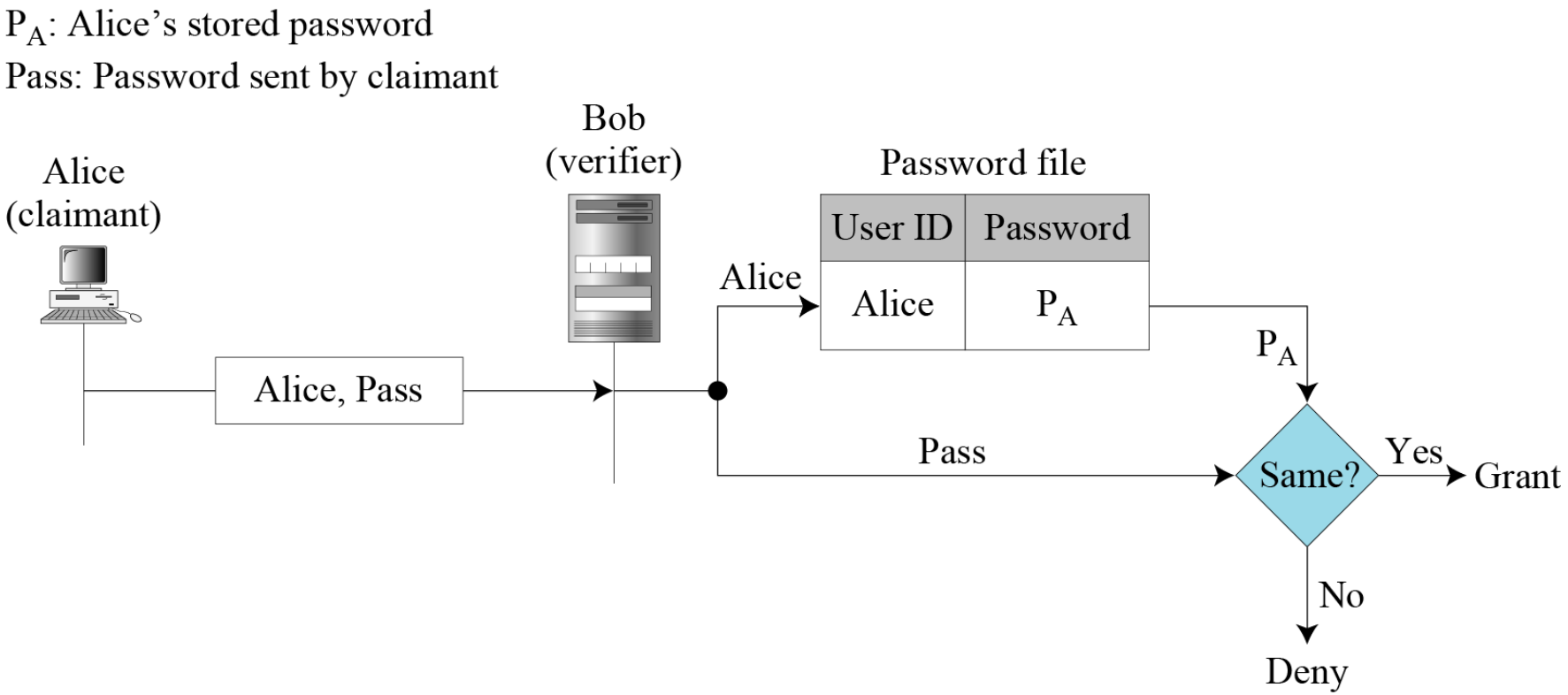
The simplest and oldest method of entity authentication is the password-based authentication, where the password is something that the claimant knows.

1) Fixed Password
2) One-Time Password

Fixed Password

First Approach

Figure *User ID and password file*

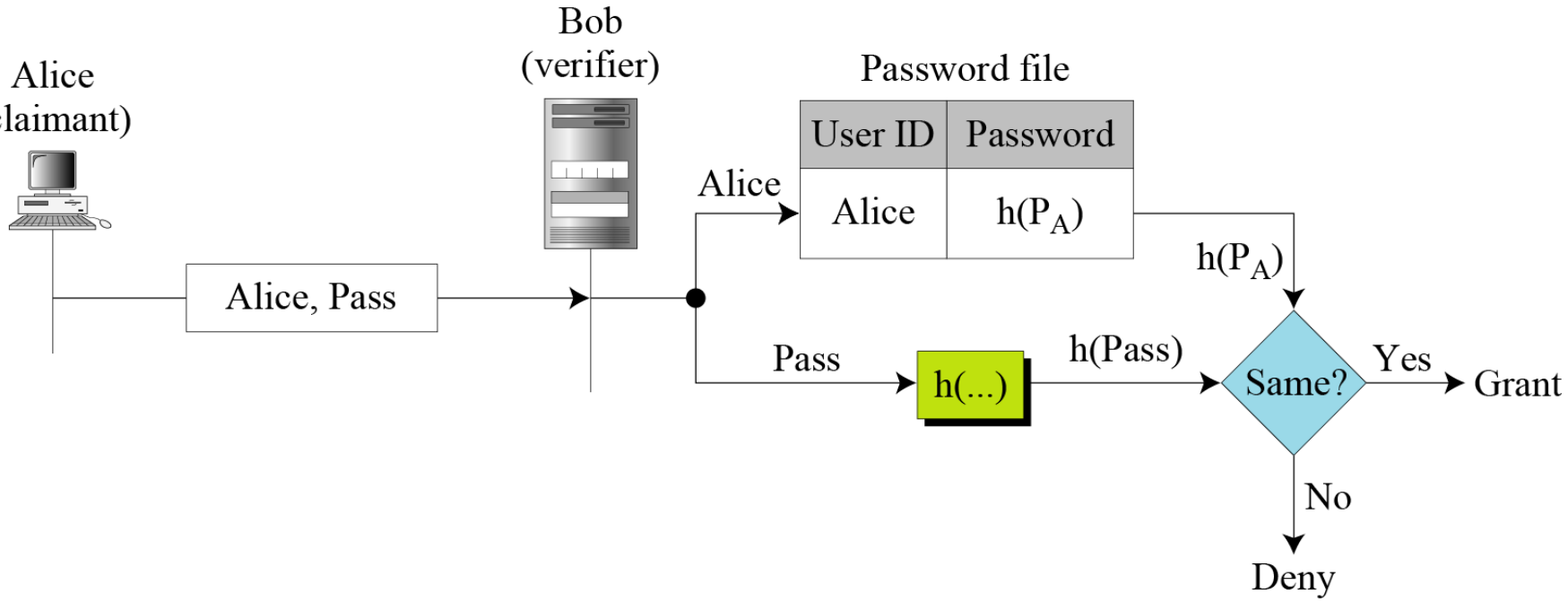


Second Approach

Figure *Hashing the password*

P_A: Alice's stored password

Pass: Password sent by claimant



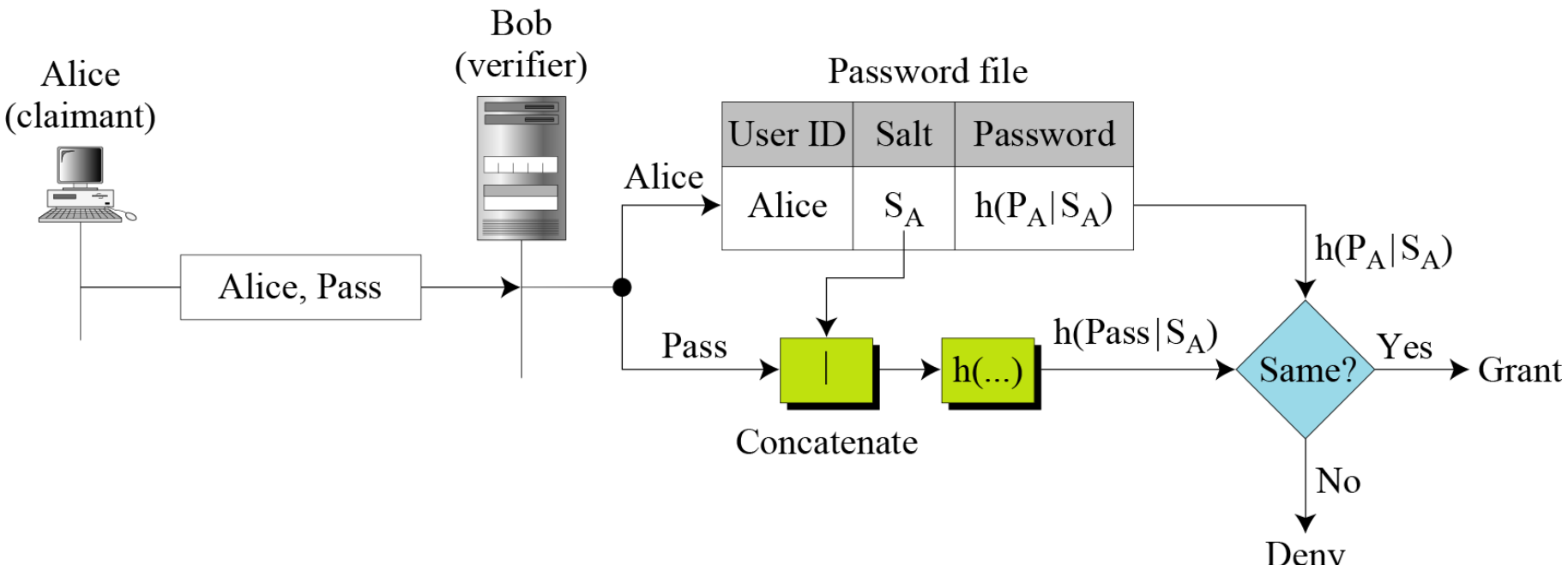
Continued

Third Approach

Figure *Salting the password*

P_A: Alice's password
S_A: Alice's salt
Pass: Password sent by claimant

mein tane aa password apelo hato...
mari website no password e ke nathi(salt)



Continued

Fourth Approach

In the fourth approach, two identification techniques are combined. A good example of this type of authentication is the use of an ATM card with a PIN (personal identification number).

One-Time Password

First Approach

In the first approach, the user and the system agree upon a list of passwords.

Second Approach

In the second approach, the user and the system agree to sequentially update the password.

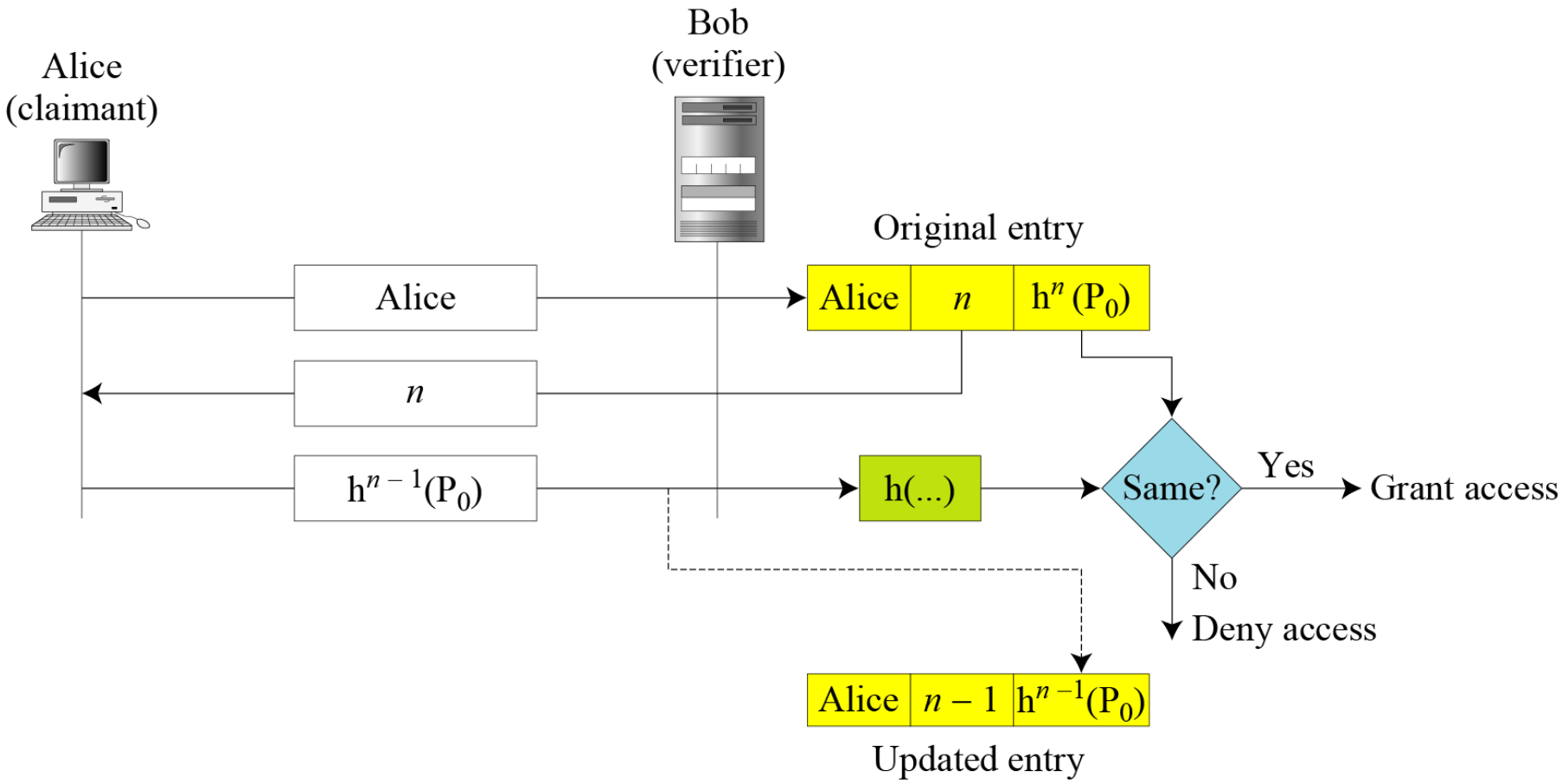
Third Approach

In the third approach, the user and the system create a sequentially updated password using a hash function.

$$h^n(x) = h(h^{n-1}(x)) \quad h^{n-1}(x) = h(h^{n-2}(x)) \quad \ldots \quad h^2(x) = h(h(x)) \quad h^1(x) = h(x)$$

Continued

Figure *Lamport one-time password*



CHALLENGE-RESPONSE

In password authentication, the claimant proves her identity by demonstrating that she knows a secret, the password. In challenge-response authentication, the claimant proves that she knows a secret without sending it.

1 Using a Symmetric-Key Cipher
2 Using Keyed-Hash Functions
3 Using an Asymmetric-Key Cipher
4 Using Digital Signature

Continue

Note

In challenge-response authentication, the claimant proves that she knows a secret without sending it to the verifier.

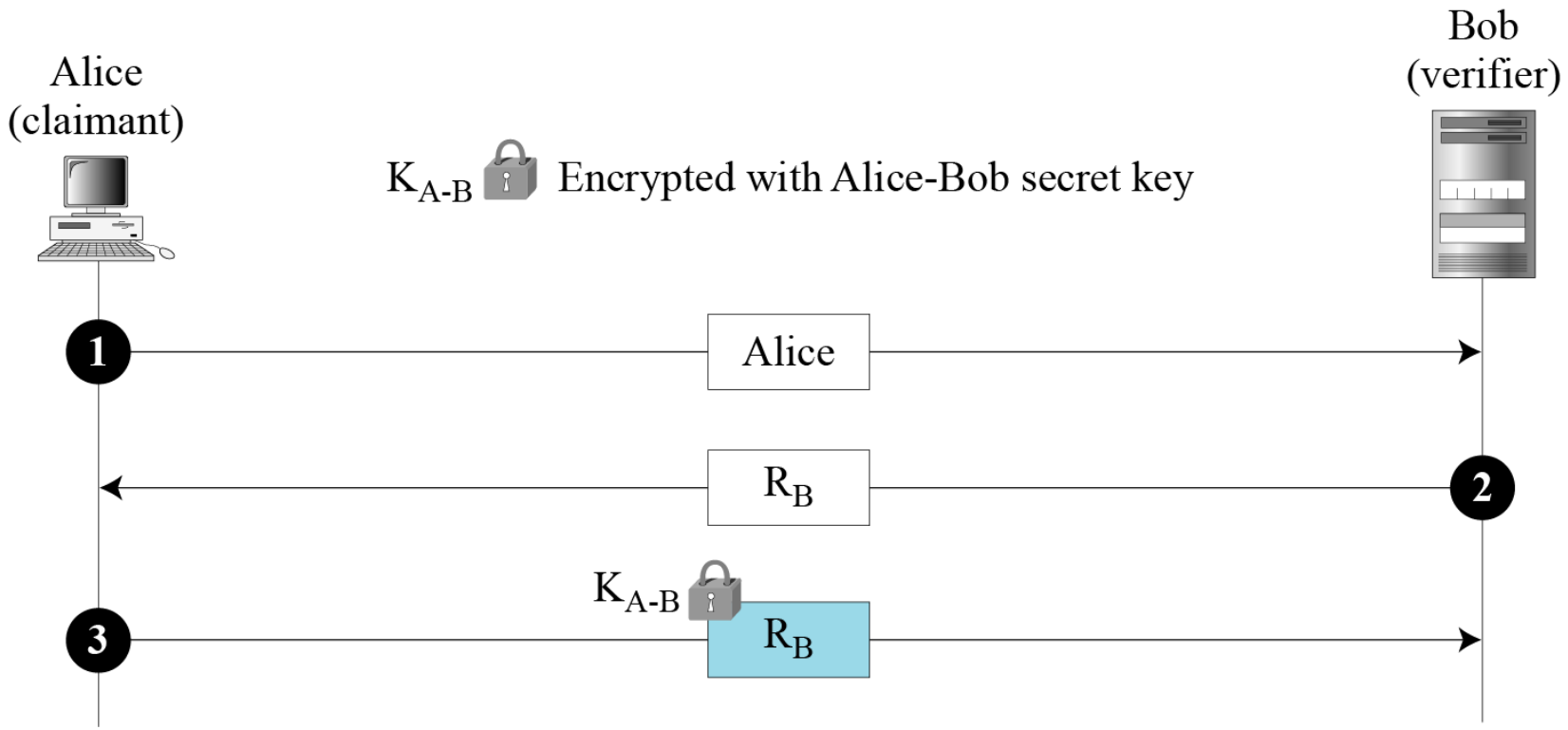
Note

The challenge is a time-varying value sent by the verifier; the response is the result of a function applied on the challenge.

Using a Symmetric-Key Cipher

First Approach

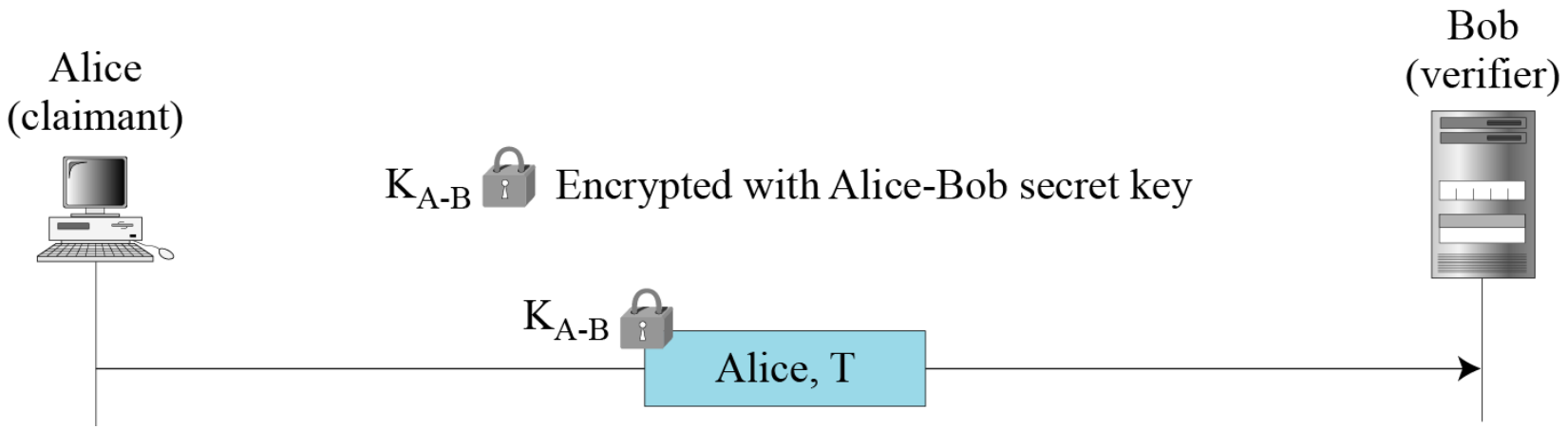
Figure *Nonce challenge*



Continued

Second Approach

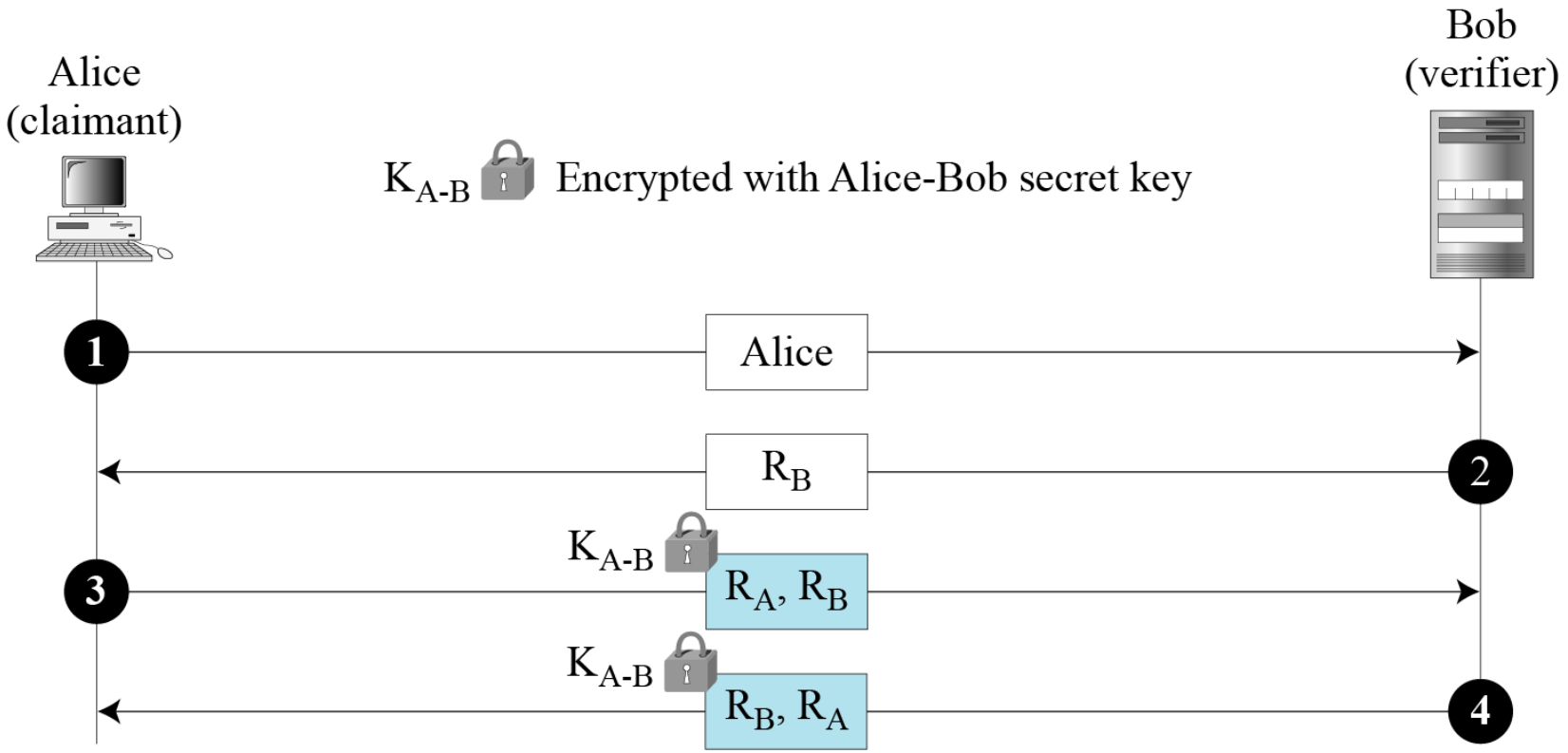
Figure *Timestamp challenge*



Continued

Third Approach.

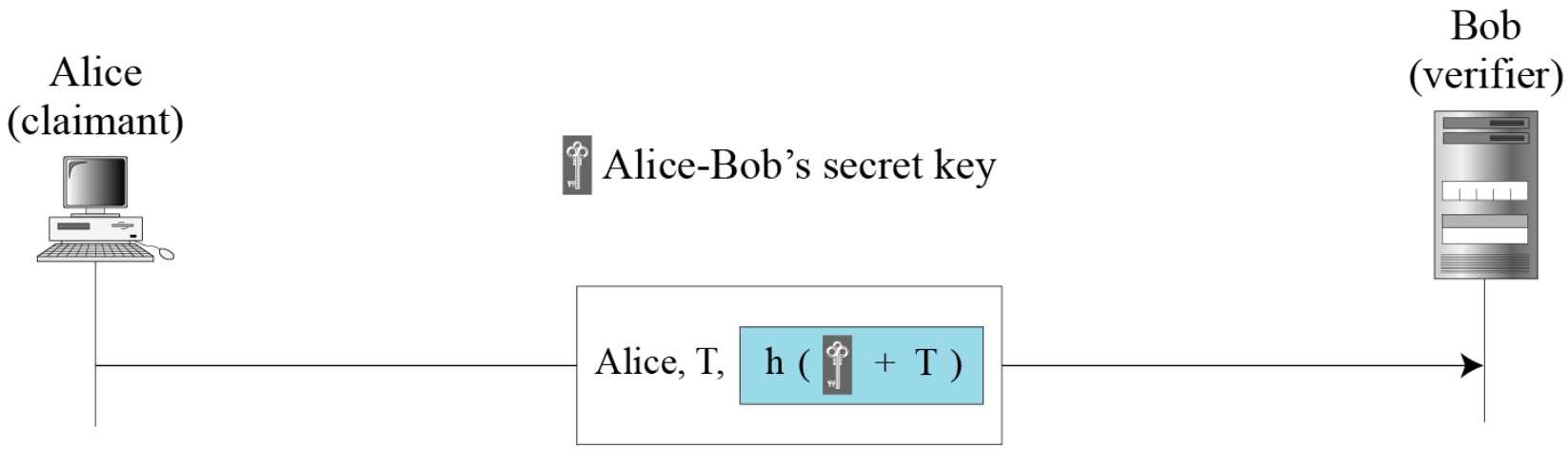
Figure ***Bidirectional authentication***



Using Keyed-Hash Functions

Instead of using encryption/decryption for entity authentication, we can also use a keyed-hash function (MAC).

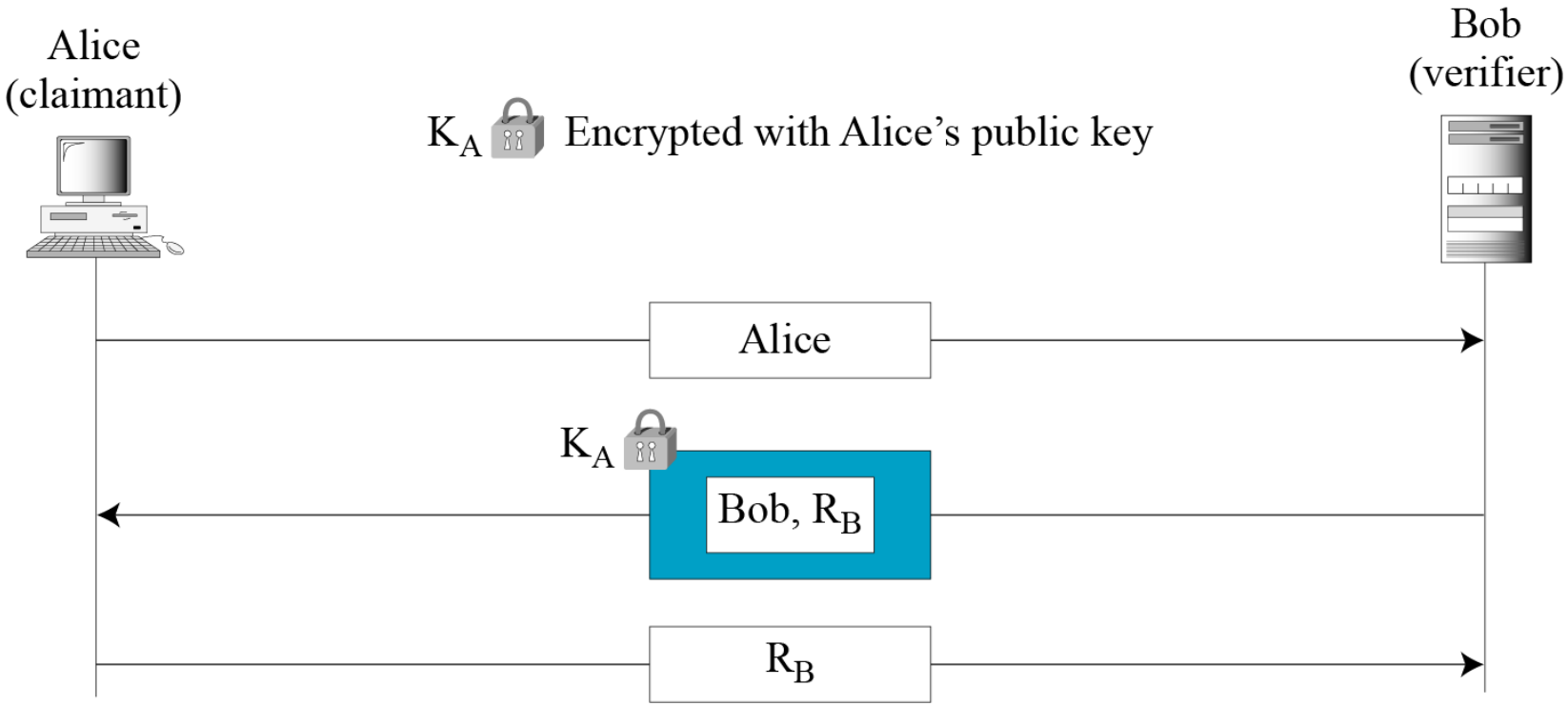
Figure ***Keyed-hash function***



Using an Asymmetric-Key Cipher

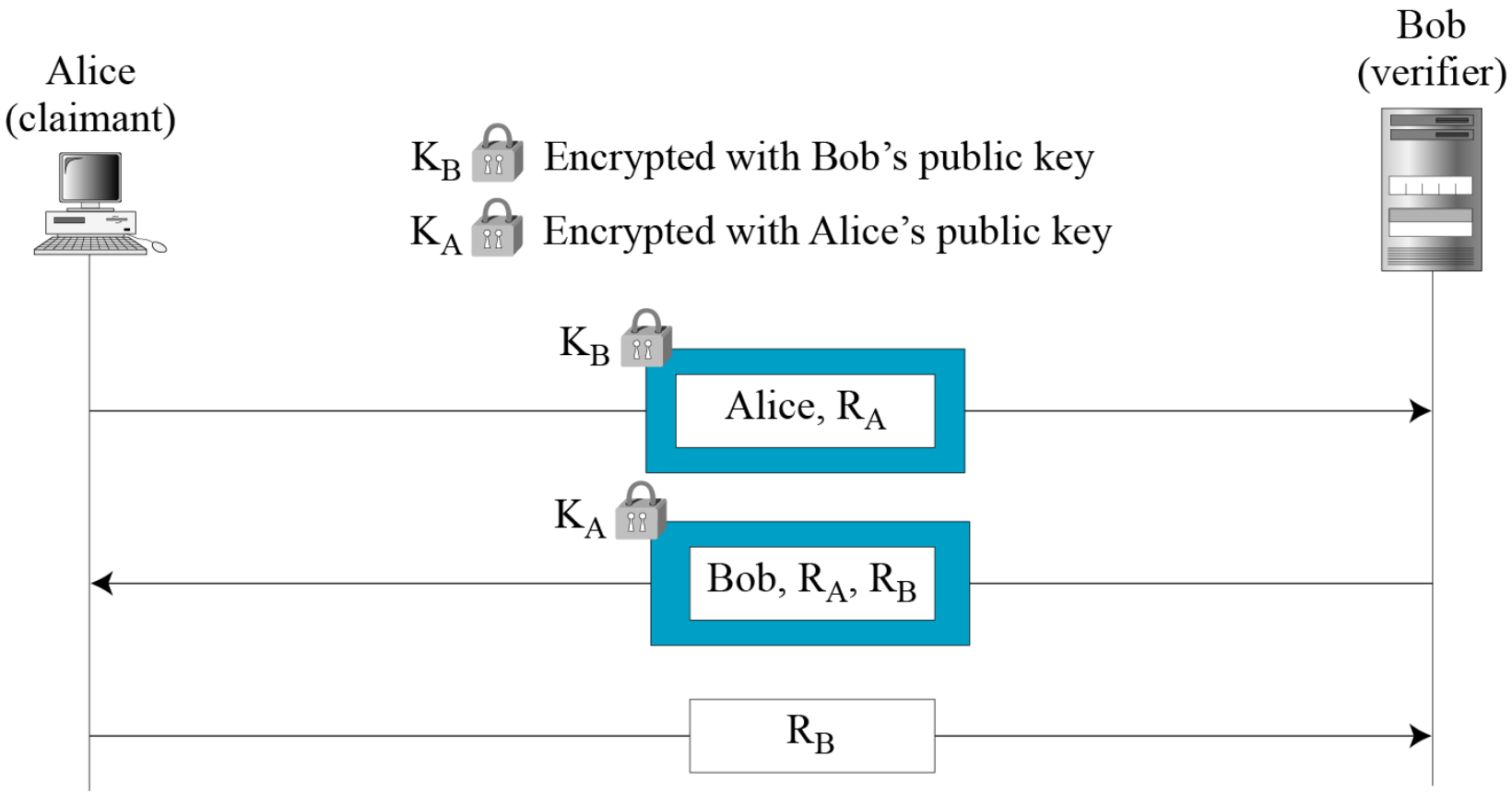
First Approach

Figure ***Unidirectional, asymmetric-key authentication***



Second Approach

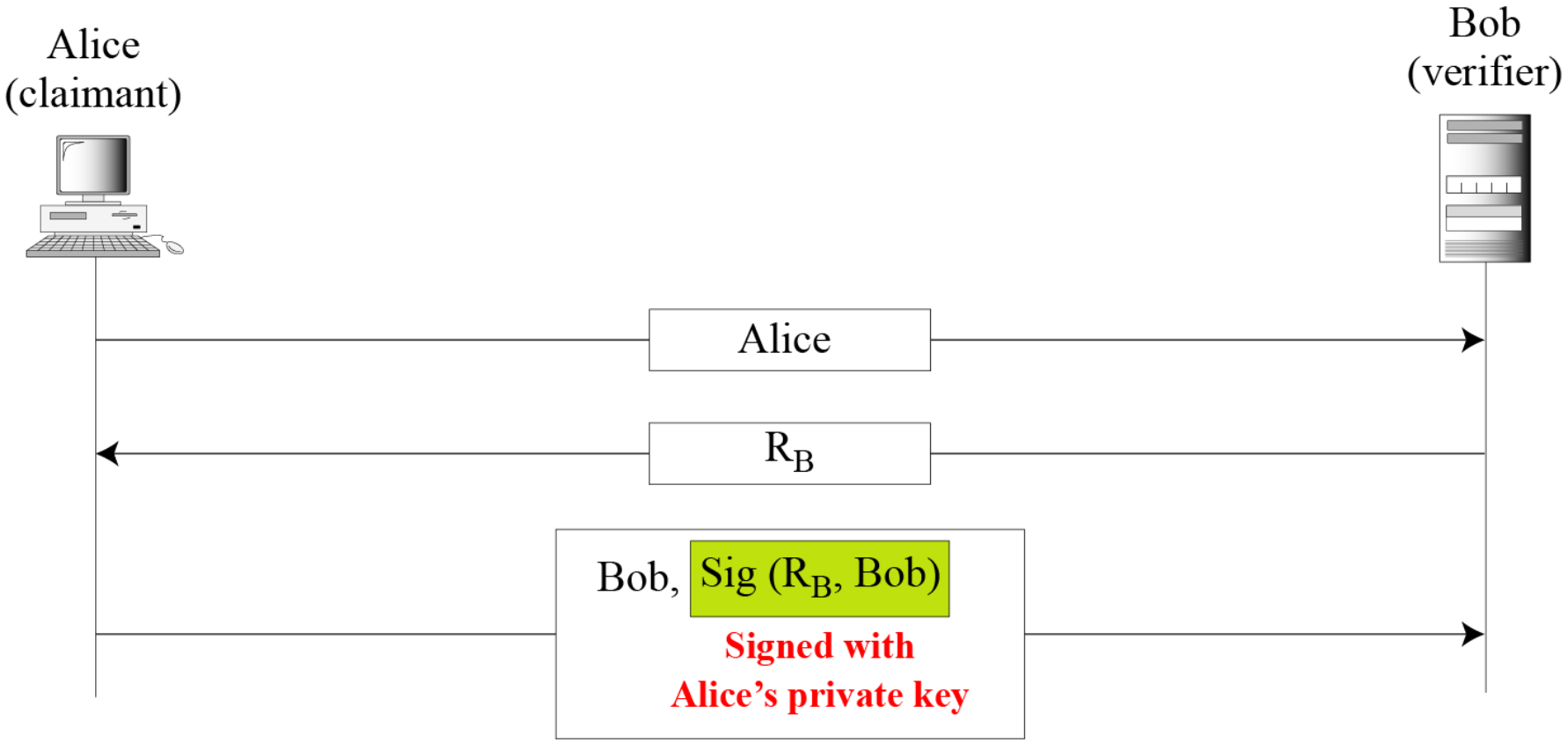
Figure ***Bidirectional, asymmetric-key***



Using Digital Signature

First Approach

Figure ***Digital signature, unidirectional***



Continued

Second Approach

Figure *Digital signature, bidirectional authentication*