Entity Authentication

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***.
- When Bob tries to prove the identity of Alice, Alice is the claimant, and Bob is the verifier

Data-Origin Versus Entity Authentication

- Message authentication (or data-origin authentication)
  - might not happen in real time; entity authentication does.
  - Alice sends a message to Bob. When Bob authenticates the message, Alice may or may not be present in the communication process.
  - required when an email is sent from Alice to Bob
  - message authentication simply authenticates one message; the process needs to be repeated for each new message.

- Entity authentication
  - there is no real message communication involved until Alice is authenticated by Bob.
  - Alice needs to be online and to take part in the process. Only after she is authenticated can messages be communicated between Alice and Bob.
  - required when Alice gets cash from an automatic teller machine.
  - authenticates the claimant for the entire duration of a session

Verification Categories

- In entity authentication, the claimant must identify herself to the verifier.
- done with one of three kinds of witnesses:
  - ***something known-*** secret known only by the claimant that can be checked by the verifier.
    - Examples - password, a PIN, a secret key, and a private key.
  - ***something possessed***-something that can prove the claimant's identity.
    - Examples - passport, a driver's license, an identification card, a credit card, and a smart card.
  - ***something inherent-*** inherent characteristic of the claimant.
    - Examples- conventional signatures, fingerprints, voice, facial characteristics, retinal pattern, and handwriting.

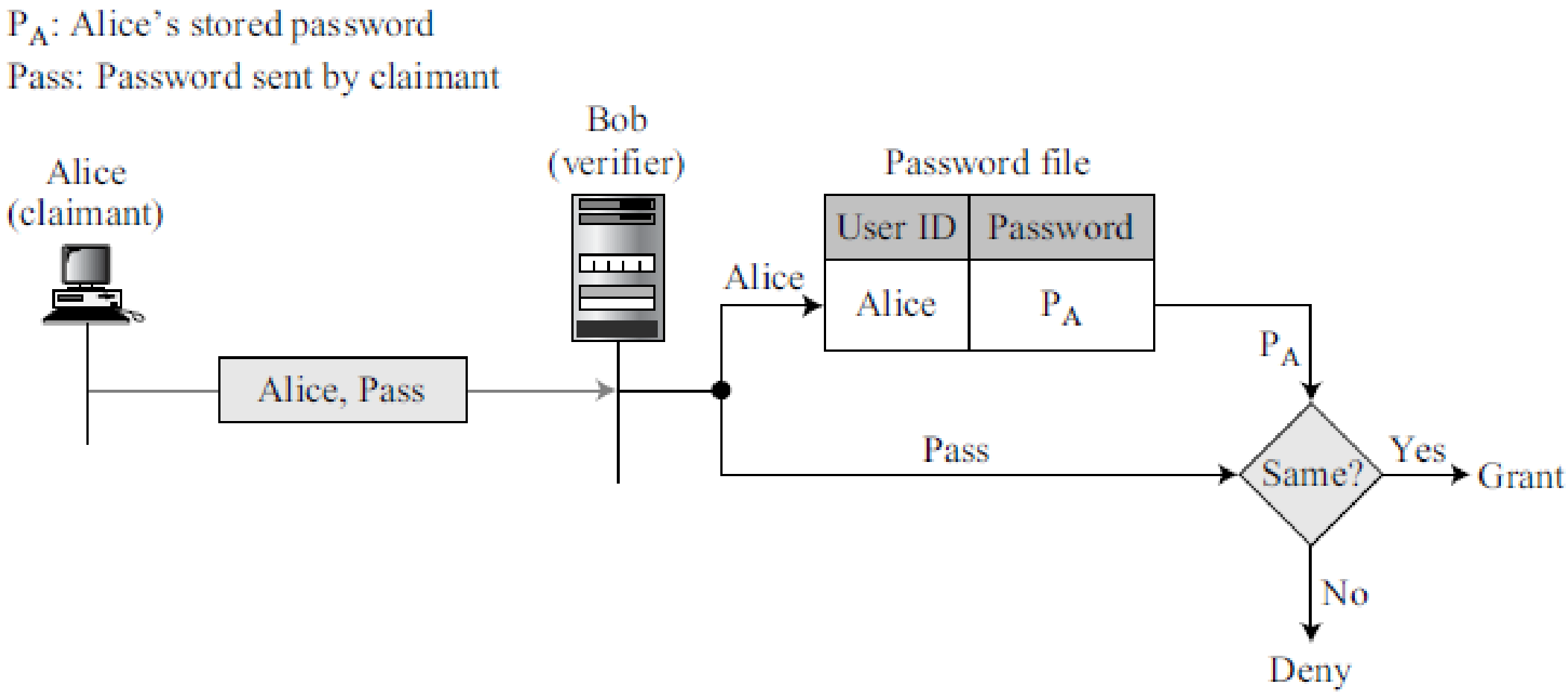
PASSWORDS

- oldest method of entity authentication - password-based authentication,
- password is something that the claimant knows.
- A password-used when a user needs to access a system to use the system's resources (login).
- Each user has a user identification that is public, and a password that is private.
- divide these authentication schemes into two groups:
  - fixed password
  - one-time password.

Fixed Password

- used over and over again for every access.
- Several schemes have been built, one upon the other
- ***First Approach*** system keeps a table (a file) that is sorted by user identification.
- To access the system resources, the user sends her user identification and password, in plaintext, to the system.
- The system uses the identification to find the password in the table.
- If the password sent by the user matches the password in the table, access is granted; otherwise, it is denied.

User ID and password file



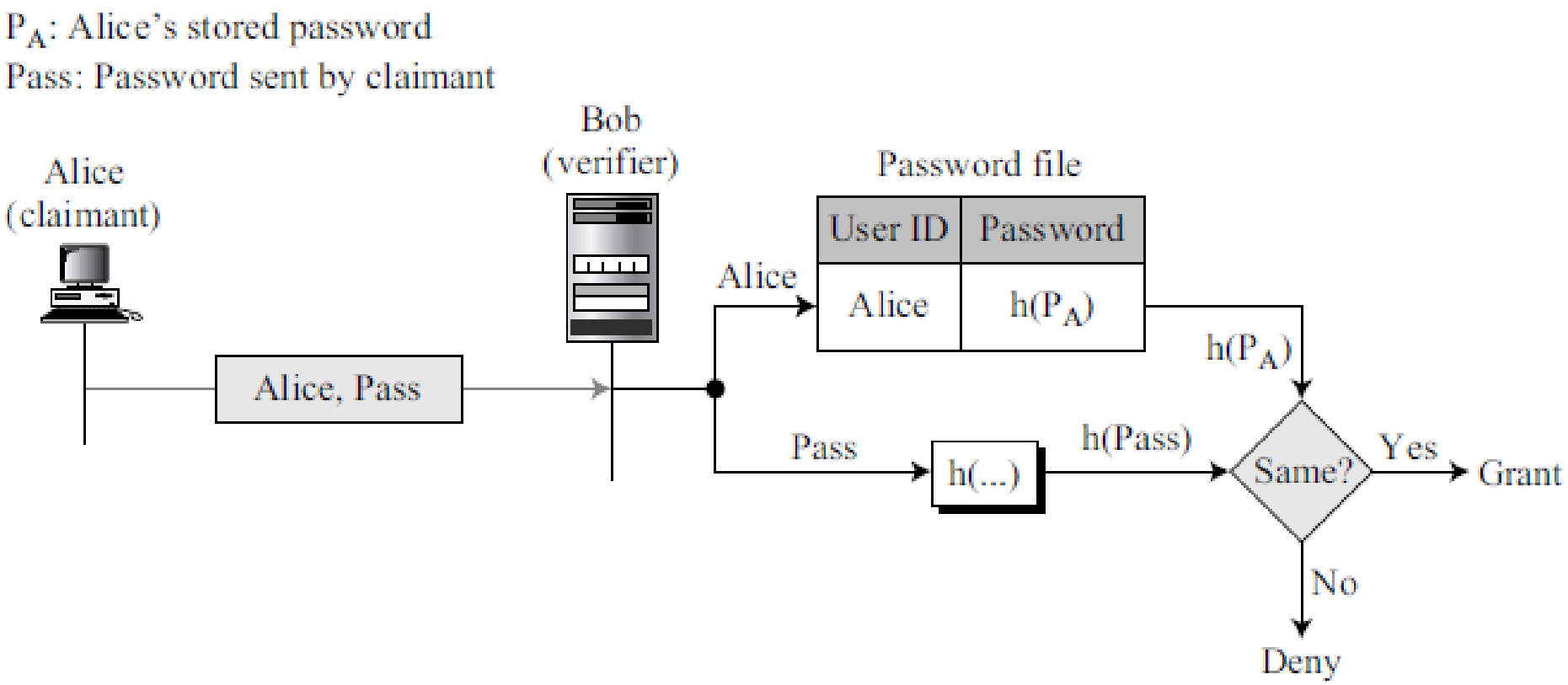
Attacks on the First Approach

- Eavesdropping- watch Alice when she types her password
  - Systems, as a security measure, do not show the characters a user types
  - Eve can listen to the line and intercept the message, thereby capturing the password for her own use
- Stealing a password- Eve tries to physically steal Alice's password.
  - This can be prevented if Alice does not write down the password and instead she just commits it to memory.
- Accessing a password file
  - Eve can hack into the system and get access to the ID/password file.
  - Eve can read the file and find Alice's password or even change it.
  - To prevent this type of attack, the file can be read/write protected.
  - However, most systems need this type of file to be readable by the public
- Guessing

Second Approach-Hashing the password

- store the hash of the password (instead of the plaintext password) in the password file.
- Any user can read the contents of the file, but, because the hash function is a one-way function, it is almost impossible to guess the value of the password
- When the password is created, the system hashes it and stores the hash in the password file.
- When the user sends the ID and the password, the system creates a hash of the password and then compares the hash value with the one stored in the file.
- If there is a match, the user is granted access; otherwise, access is denied.
- file does not need to be read protected.

Second Approach-Hashing the password



Dictionary Attack

- The hash function prevents Eve from gaining access to the system even though she has the password file.
- There is still the possibility of dictionary attack.
- In this attack, Eve is interested in finding one password,regardless of the user ID.
- For example, if the password is 6 digits, Eve can create a list of 6-digit numbers (000000 to 999999), and then apply the hash function to every number; the result is a list of one million hashes.
- She can then get the password file and search the second-column entries to find a match.
- This could be programmed and run offline on Eve's private computer.
- After a match is found, Eve can go online and use the password to access the system.
- The third approach shows how to make this attack more difficult.

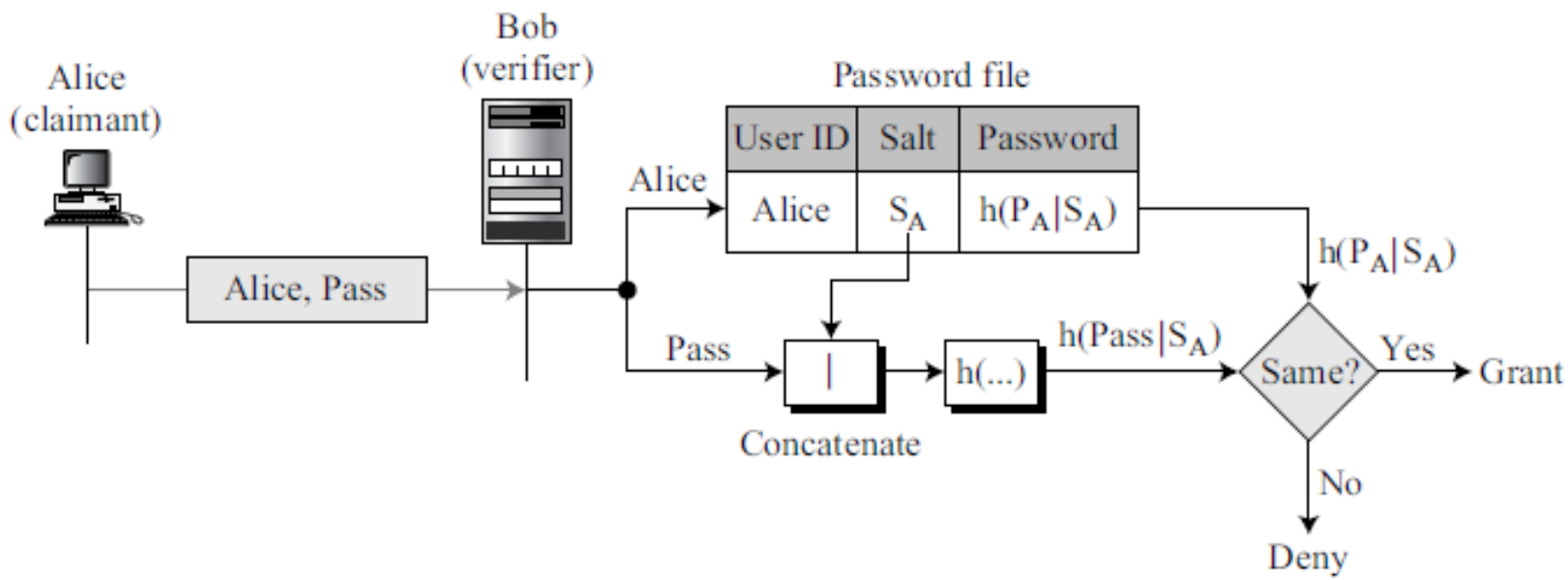
Third Approach-salting the password

- When the password string is created, a random string, called the salt, is concatenated to the password.
- The salted password is then hashed. The ID, the salt, and the hash are then stored in the file.
- Now, when a user asks for access, the system extracts the salt, concatenates it with the received password, makes a hash out of the result, and compares it with the hash stored in the file.
- If there is a match, access is granted; otherwise, it is denied
- Salting makes the dictionary attack more difficult. If the original password is 6 digits and the salt is 4 digits, then hashing is done over a 10-digit value.
- Eve now needs to make a list of 10 million items and create a hash for each of them.
- The list of hashes has 10 million entries, and the comparison takes much longer.
- Salting is very effective if the salt is a very long random number.
- The UNIX operating system uses a variation of this method.

P_A: Alice's password

S_A: Alice's salt

Pass: Password sent by claimant



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).
- The card belongs to the category “something possessed ” and the PIN belongs to the category “something known”.
- The PIN is a password that enhances the security of the card.
- If the card is stolen, it cannot be used unless the PIN is known.
- The PIN number, however, is traditionally very short so it is easily remembered by the owner.
- This makes it vulnerable to the guessing type of attack.

One-Time Password

- password that is used only once.
- This kind of password makes eavesdropping and salting useless

First Approach

- the user and the system agree upon a list of passwords
- Each password on the list can be used only once.
- There are some drawbacks to this approach.
- First, the system and the user must keep a long list of passwords.
- Second, if the user does not use the passwords in sequence, the system needs to perform a long search to find the match.
- This scheme makes eavesdropping and reuse of the password useless.
- The password is valid only once and cannot be used again.

Second Approach

- In the second approach, the user and the system agree to sequentially update the password.
- The user and the system agree on an original password, P1, which is valid only for the first access.
- During the first access, the user generates a new password, P2, and encrypts this password with P1 as the key.
- P2 is the password for the second access.
- During the second access, the user generates a new password, P3, and encrypts it with P2;
- P3 is used for the third access.
- In other words, Pi is used to create Pi+1.
- If Eve can guess the first password (P1), she can find all of the subsequent ones.

Third Approach

- user and the system create a sequentially updated password using a hash function
- In this approach, devised by Leslie Lamport, the user and the system agree upon an original password, P0, and a counter, n
- The system calculates $h^n(P0)$, where h^n means applying a hash function n times

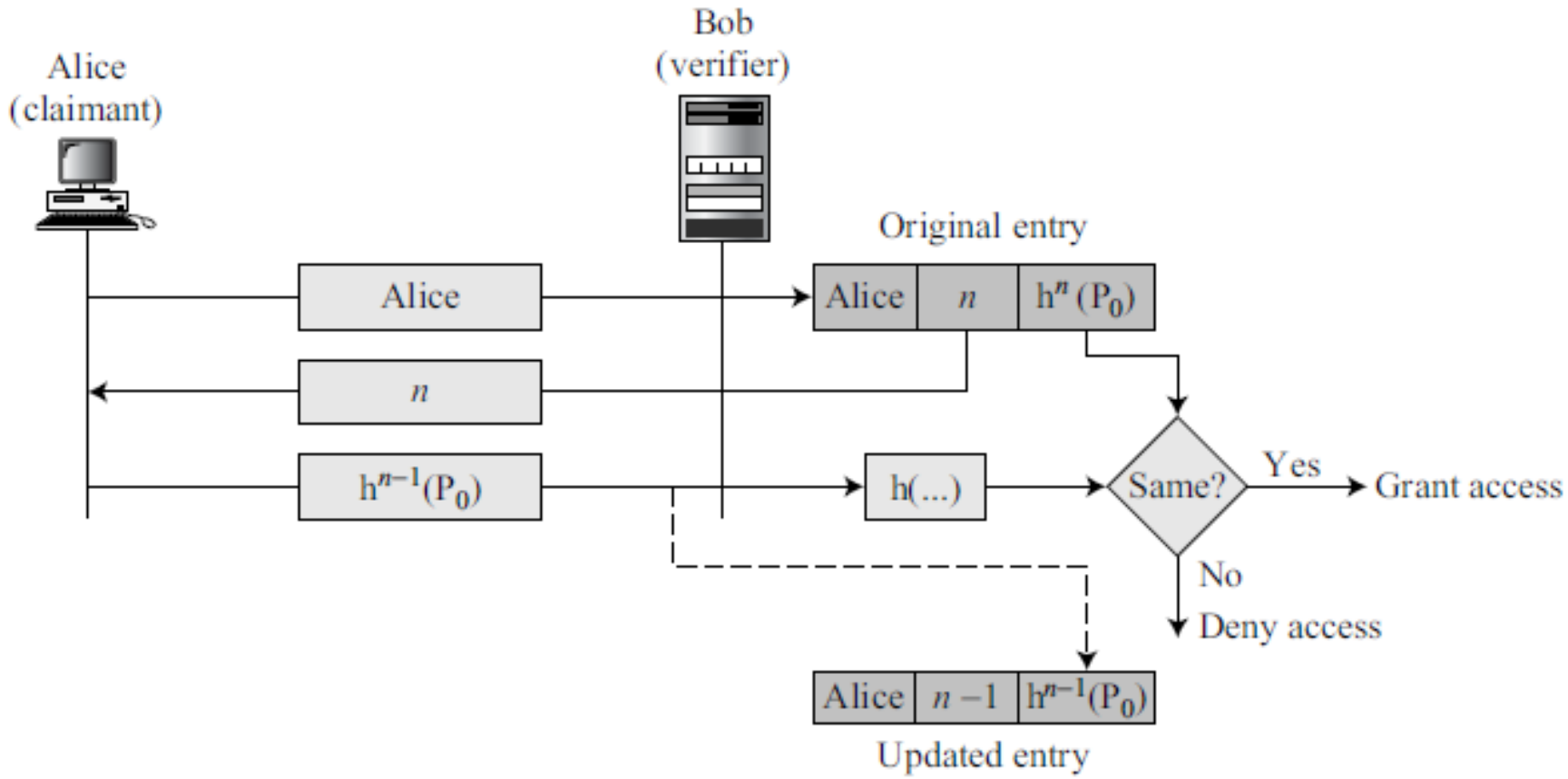
$$h^n(x) = h(h^{n-1}(x)) \quad h^{n-1}(x) = h(h^{n-2}(x)) \quad \ldots \quad h^2(x) = h(h(x)) \quad h^1(x) = h(x)$$

- The system stores the identity of Alice, the value of n, and the value of $h^n(P0)$.

Third Approach Contd..

- When the system receives the response of the user in the third message, it applies the hash function to the value received to see if it matches the value stored in the entry.
- If there is a match, access is granted; otherwise, it is denied.
- The system then decrement the value of n in the entry and replaces the old value of the password $h^n(P0)$ with the new value $h^{n-1}(P0)$
- When the user tries to access the system for the second time, the value of the counter it receives is n − 1.
- The third message from the user is now $h^{n-2}(P0)$
- When the system receives this message, it applies the hash function to get $h^{n-1}(P0)$, which can be compared with the updated entry.
- The value of n in the entry is decremented each time there is an access.
- When the value becomes 0, the user can no longer access the system; everything must be set up again.
- For this reason, the value of n is normally chosen as a large number such as 1000.

Lamport one-time password



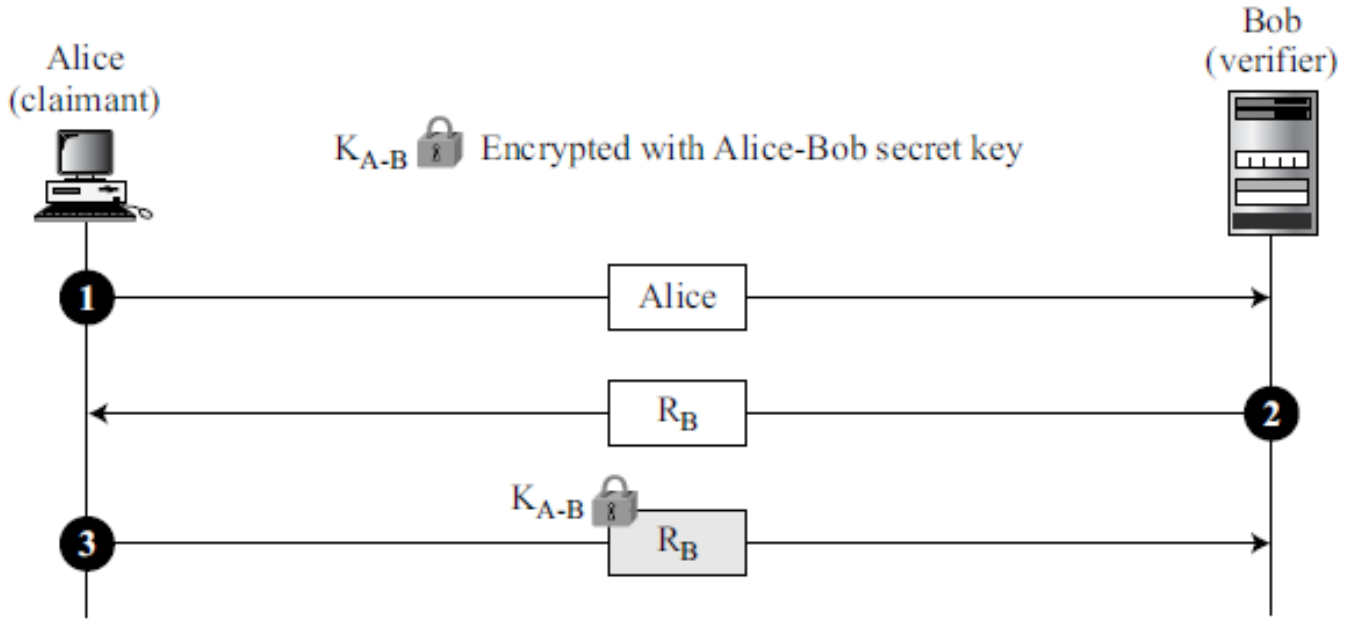
CHALLENGE-RESPONSE

- In password authentication, the claimant proves her identity by demonstrating that she knows a secret, the password.
  - susceptible to interception by the adversary.
- In challenge-response authentication, the claimant proves that she knows a secret without sending it.
  - claimant does not send the secret to the verifier; the verifier either has it or finds it
- The challenge is a time-varying value such as a random number or a timestamp that is sent by the verifier.
- The claimant applies a function to the challenge and sends the result, called a response, to the verifier.
- The response shows that the claimant knows the secret.

Using a Symmetric-Key Cipher

- The secret here is the shared secret key, known by both the claimant and the verifier.
- The function is the encrypting algorithm applied on the challenge.
- First Approach- verifier sends a nonce, a random number used only once, to challenge the claimant.
- A nonce must be time-varying; every time it is created, it is different.
- The claimant responds to the challenge using the secret key shared between the claimant and the verifier

Nonce challenge



The first message is not part of challenge-response, it only informs the verifier that the claimant wants to be challenged.

The 2nd message- challenge. RB is the nonce randomly chosen by the verifier (Bob) to challenge the claimant.

The claimant encrypts the nonce using the shared secret key known only to the claimant and the verifier and sends the result to the verifier.

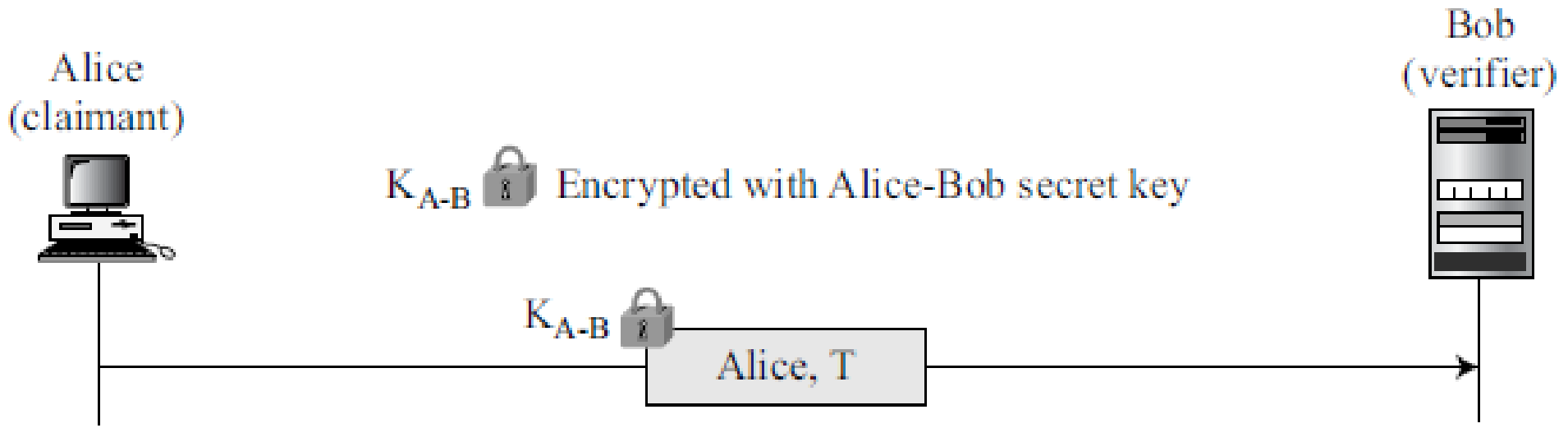
The verifier decrypts the message. If the nonce obtained from decryption is the same as the one sent by the verifier, Alice is granted access.

- the claimant and the verifier need to keep the symmetric key used in the process secret.
- The verifier must also keep the value of the nonce for claimant identification until the response is returned.

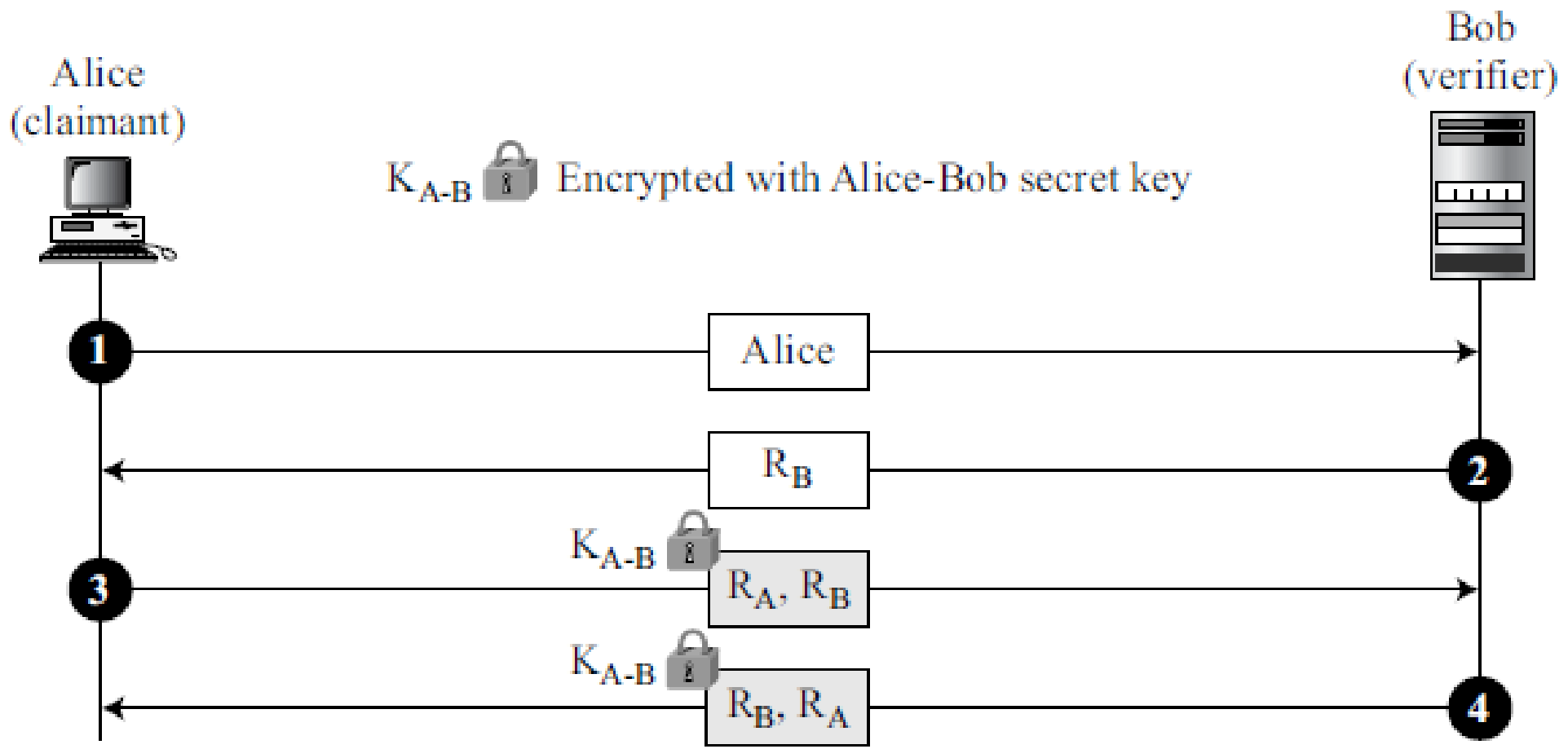
Second Approach

- the time-varying value is a timestamp, changes with time.
- In this approach the challenge message is the current time sent from the verifier to the claimant.
- However, this supposes that the client and the server clocks are synchronized; the claimant knows the current time.
- This means that there is no need for the challenge message.
- The first and third messages can be combined.
- The result is that authentication can be done using one message, the response to an implicit challenge, the current time. Figure 14.6 shows the approach.

Timestamp challenge



Bidirectional authentication

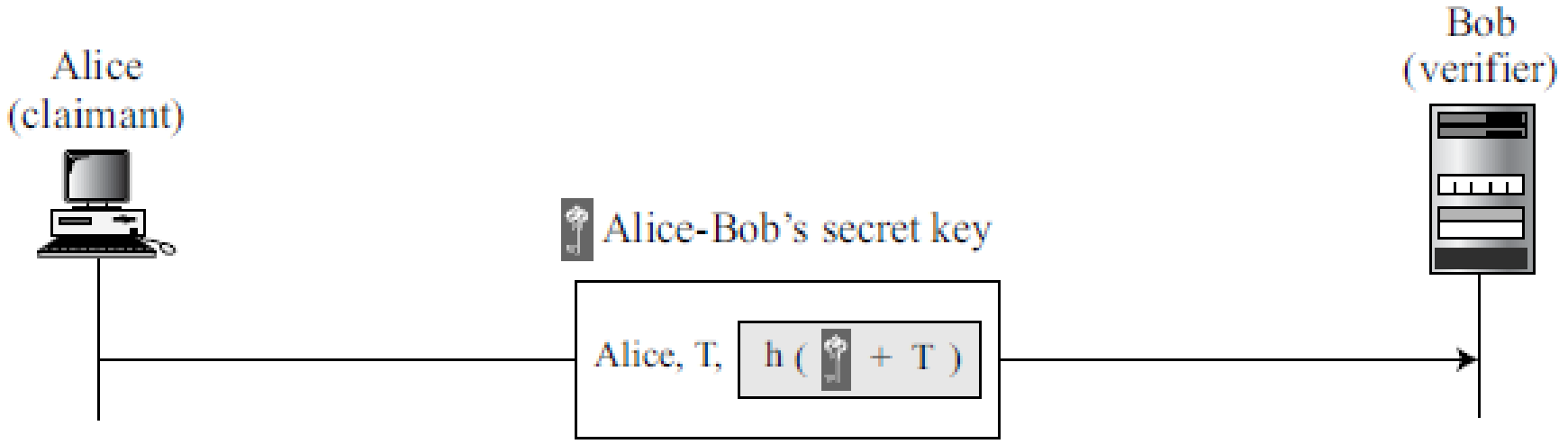


Third Approach

- The first and second approaches are for unidirectional authentication.
- Alice is authenticated to Bob, but not the other way around.
- If Alice also needs to be sure about Bob's identity, we need bidirectional authentication.
- The second message RB is the challenge from Bob to Alice.
- In the third message, Alice responds to Bob's challenge and at the same time, sends her challenge RA to Bob.
- The third message is Bob's response.
- Note that in the fourth message the order of RA and RB are switched to prevent a replay attack of the third message by an adversary

Using Keyed-Hash Functions

- One advantage to the scheme - it preserves the integrity of challenge and response messages and at the same time uses a secret, the key
- can use a keyed-hash function to create a challenge response with a timestamp.
- the timestamp is sent both as plaintext and as text scrambled by the keyed-hash function.
- When Bob receives the message, he takes the plaintext T, applies the keyed-hash function, and then compares his calculation with what he received to determine the authenticity of Alice.

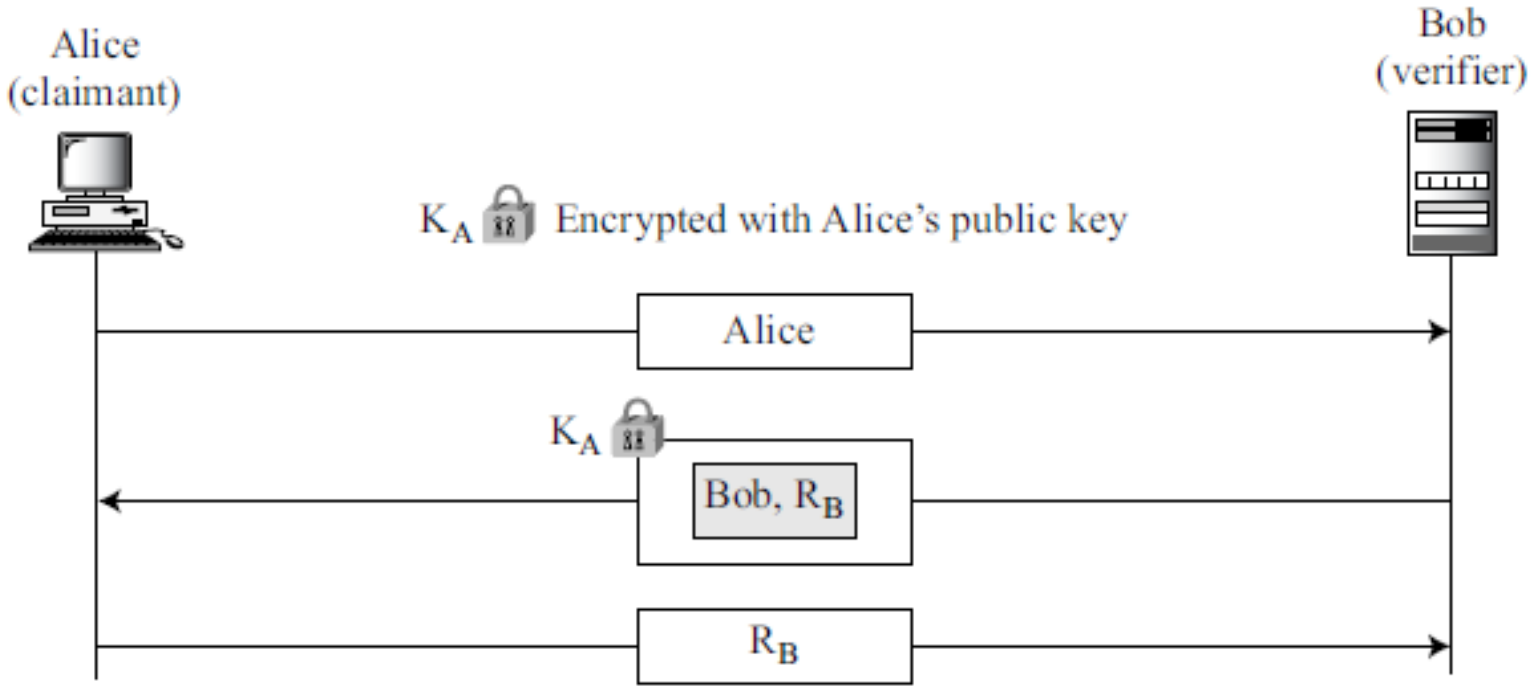


Using an Asymmetric-Key Cipher

- the secret must be the private key of the claimant.
- The claimant must show that she owns the private key related to the public key that is available to everyone.
- This means that the verifier must encrypt the challenge using the public key of the claimant; the claimant then decrypts the message using her private key.
- response to the challenge is the decrypted challenge.
- Following are two approaches:
- one for unidirectional authentication and one for bidirectional authentication.

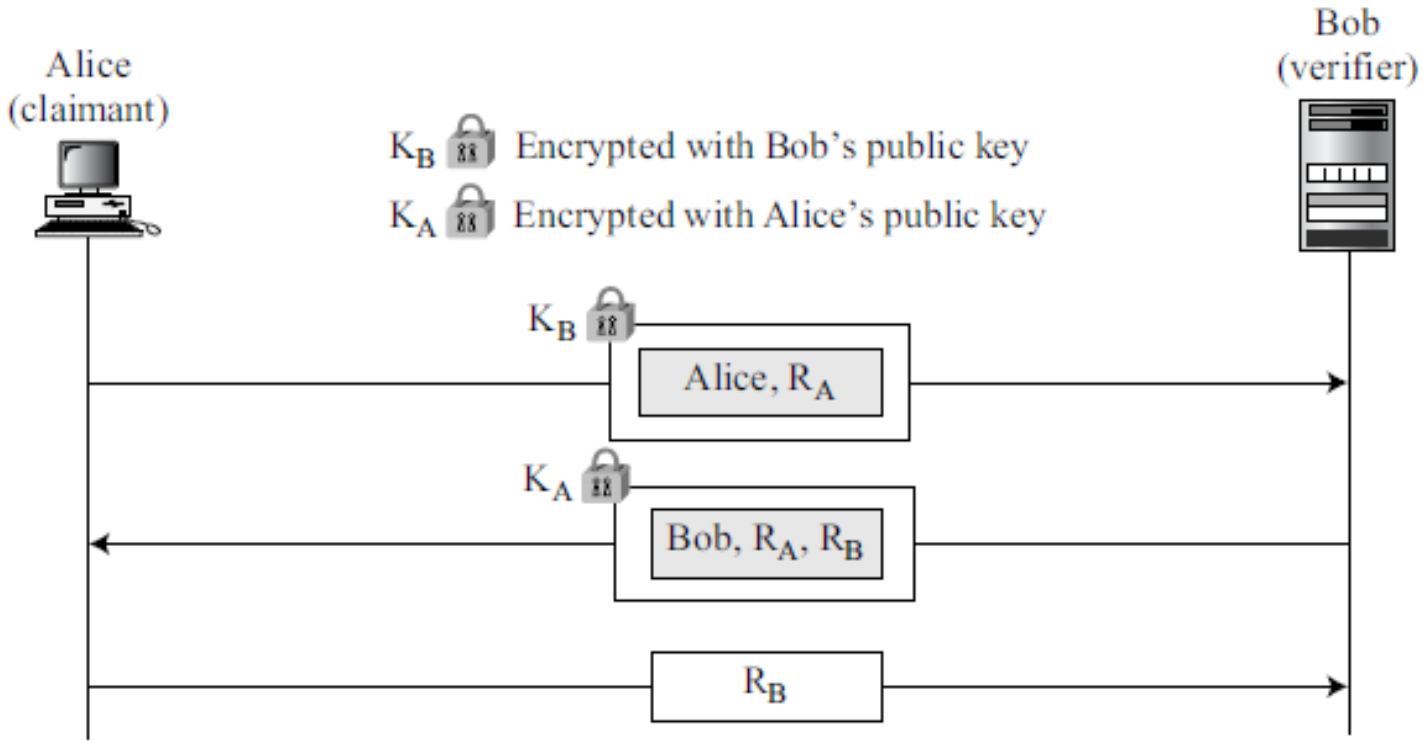
First Approach-Unidirectional, asymmetric-key authentication

- Bob encrypts the challenge using Alice's public key.
- Alice decrypts the message with her private key and sends the nonce to Bob.



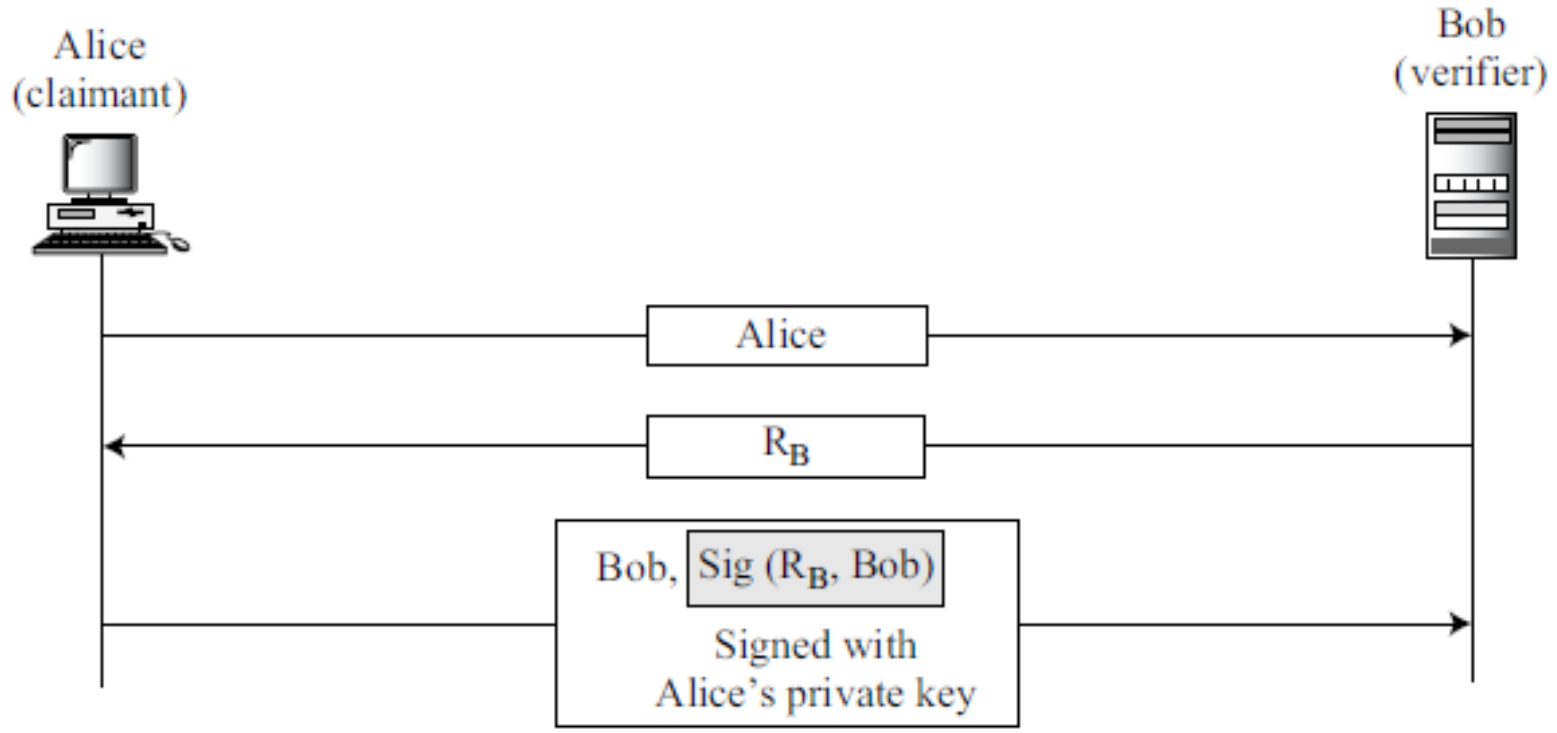
Second Approach-Bidirectional, asymmetric-key

- two public keys are used, one in each direction.
- Alice sends her identity and nonce encrypted with Bob's public key. Bob responds with his nonce encrypted with Alice's public key.
- Finally, Alice, responds with Bob's decrypted nonce.

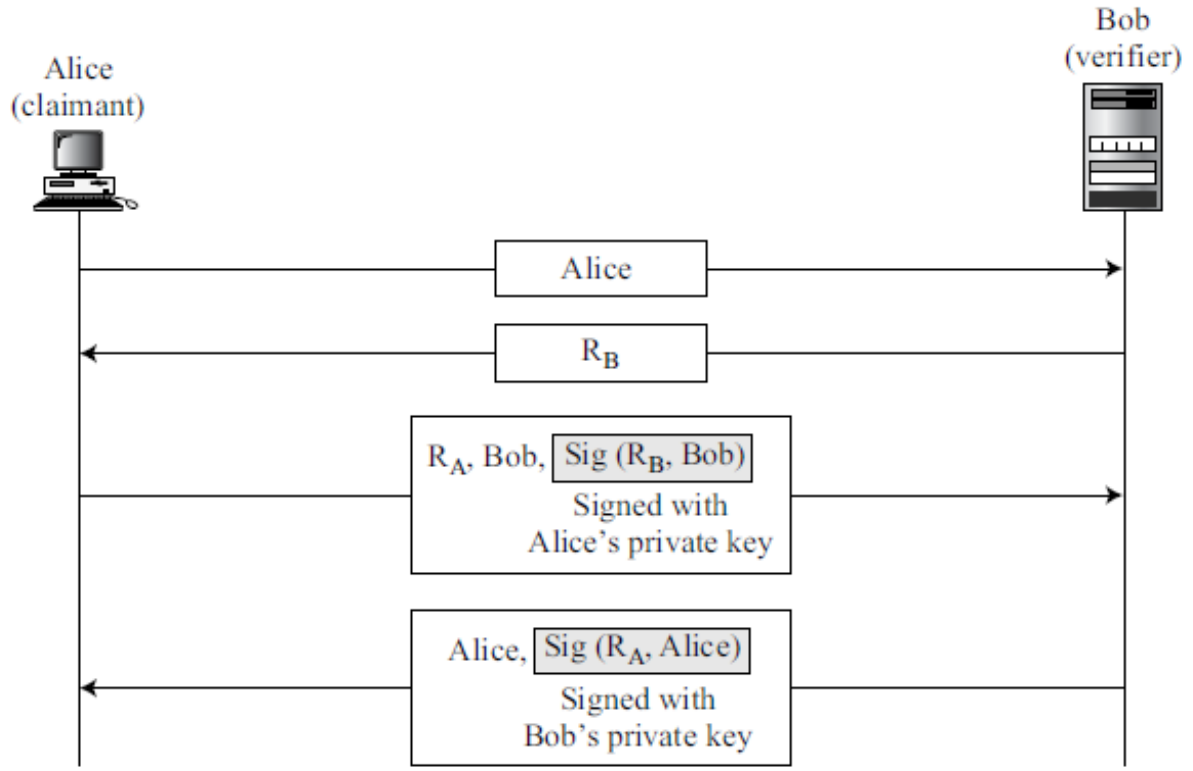


Using Digital Signature

- The claimant uses her private key for signing
- First Approach-Digital signature, unidirectional authentication



Second Approach-Digital signature, bidirectional authentication



ZERO-KNOWLEDGE

- In password authentication, the claimant needs to send her secret (the password) to the verifier-subject to eavesdropping by Eve
- In addition, a dishonest verifier could reveal the password to others or use it to impersonate the claimant.
- In challenge-response entity authentication, the claimant's secret is not sent to the verifier.
- The claimant applies a function on the challenge sent by the verifier that includes her secret.
- In some challenge-response methods, the verifier actually knows the claimant's secret, which could be misused by a dishonest verifier.
- In other methods, the verifier can extract some information about the secret from the claimant by choosing a preplanned set of challenges.

- In zero-knowledge authentication, the claimant does not reveal anything that might endanger the confidentiality of the secret.
- The claimant proves to the verifier that she knows a secret, without revealing it.
- The interactions are so designed that they cannot lead to revealing or guessing the secret.
- After exchanging messages, the verifier only knows that the claimant does or does not have the secret, nothing more.
- The result is a yes/no situation, just a single bit of information.

Fiat-Shamir Protocol

- a trusted third party chooses two large prime numbers p and q to calculate the value of $n = p \times q$.
- The value of n is announced to the public; the values of p and q are kept secret.
- Alice, the claimant, chooses a secret number s between 1 and n − 1 (exclusive). She calculates $v = s^2 \bmod n$.
- She keeps s as her private key and registers v as her public key with the third party.

Verification of Alice by Bob can be done in four steps

1. Alice, the claimant, chooses a random number r between 0 and n − 1 (r is called the commitment). She then calculates the value of $x = r^2 \bmod n$; x is called the witness.

2. Alice sends x to Bob as the witness.

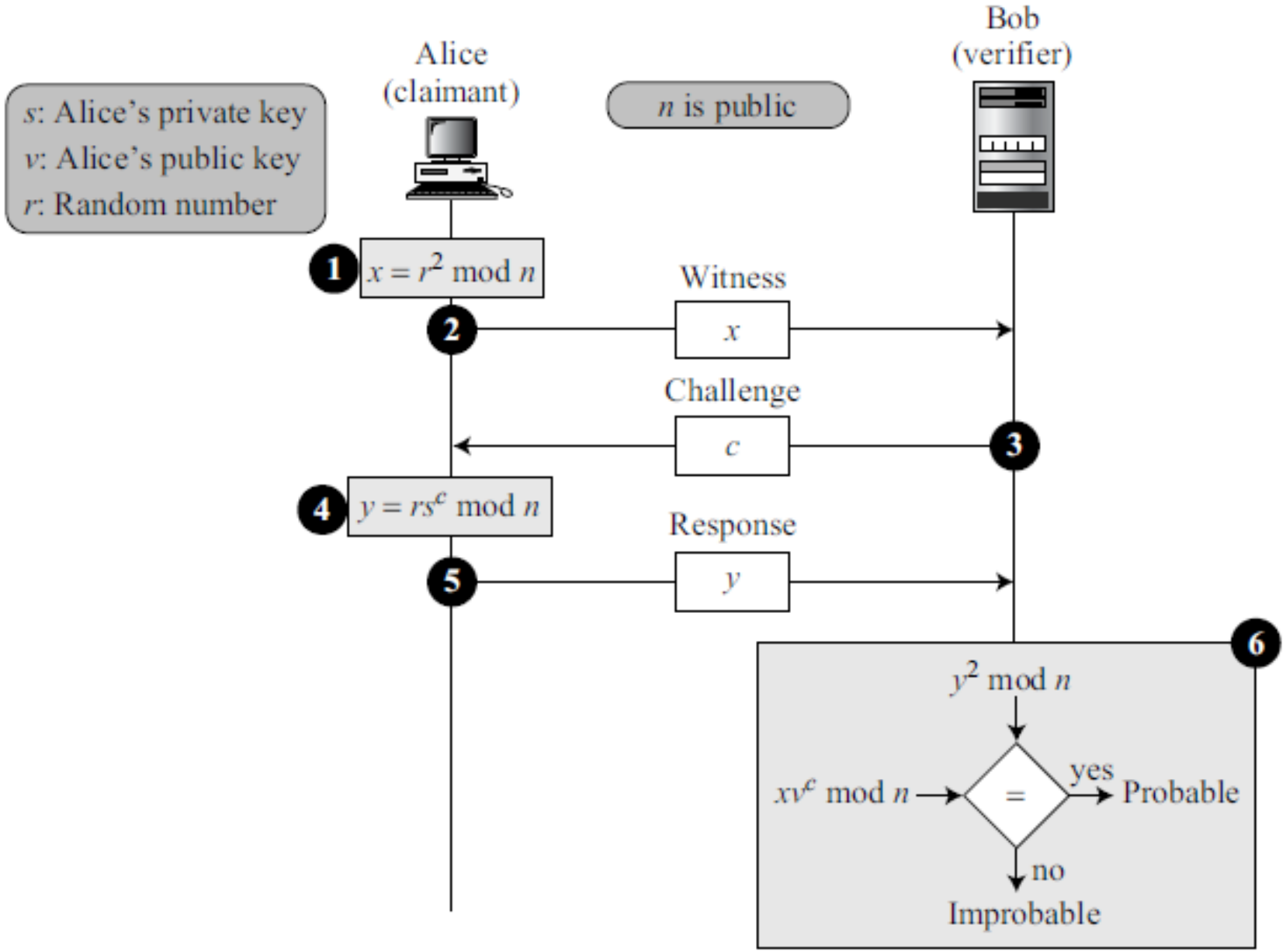
3. Bob, the verifier, sends the challenge c to Alice. The value of c is either 0 or 1.

4. Alice calculates the response $y = rs^c$;r is the random number selected by Alice in the first step, s is her private key, and c is the challenge (0 or 1).

5. Alice sends the response to Bob to show that she knows the value of her private key, s. She claims to be Alice.

6. Bob calculates y^2 and xv^c. If these two values are congruent, then Alice either knows the value of s (she is honest) or she has calculated the value of y in some other ways (dishonest) because we can easily prove that y^2 is the same as xv^c in modulo n arithmetic as shown below:

$$y^2 = (rs^c)^2 = r^2 s^{2c} = r^2 (s^2)^c = xv^c$$


Bob
(verifier)
Alice
(claimant)
s: Alice's private key
v: Alice's public key
r: Random number
n is public
1
x = r^2 mod n
Witness
2
x
Challenge
c
3
4
y = rs^c mod n
Response
5
y
6
y^2 mod n
xv^c mod n
=
yes
Probable
no
Improbable

- The six steps constitute a round; the verification is repeated several times with the value of c equal to 0 or 1 (chosen randomly).
- The claimant must pass the test in each round to be verified.
- If she fails one single round, the process is aborted and she is not authenticated.
- Alice can be honest (knows the value of s) or dishonest (does not know the value of s).
- If she is honest, she passes each round. If she is not, she still can pass a round by predicting the value of challenge correctly.
- Two situations can happen:

Alice guesses that the value of c (the challenge) will be 1 (a prediction).

She calculates $x = r^2/v$ and sends x as the witness.

a. If her guess is correct (c turned out to be 1), she sends y = r as the response. We can see that she passes the test ($y^2 = xv^c$).

b. If her guess is wrong (c turned out to be 0), she cannot find a value of y that passes the test.

She probably quits or sends a value that does not pass the test and Bob will abort the process.

Alice guesses that the value of c (challenge) will be 0.

She calculates $x = r^2$ and sends x as the witness.

c. If her guess is correct (c turned out to be 0), she sends y = r as the response.

We can see that she passes the test ($y^2 = xv^c$).

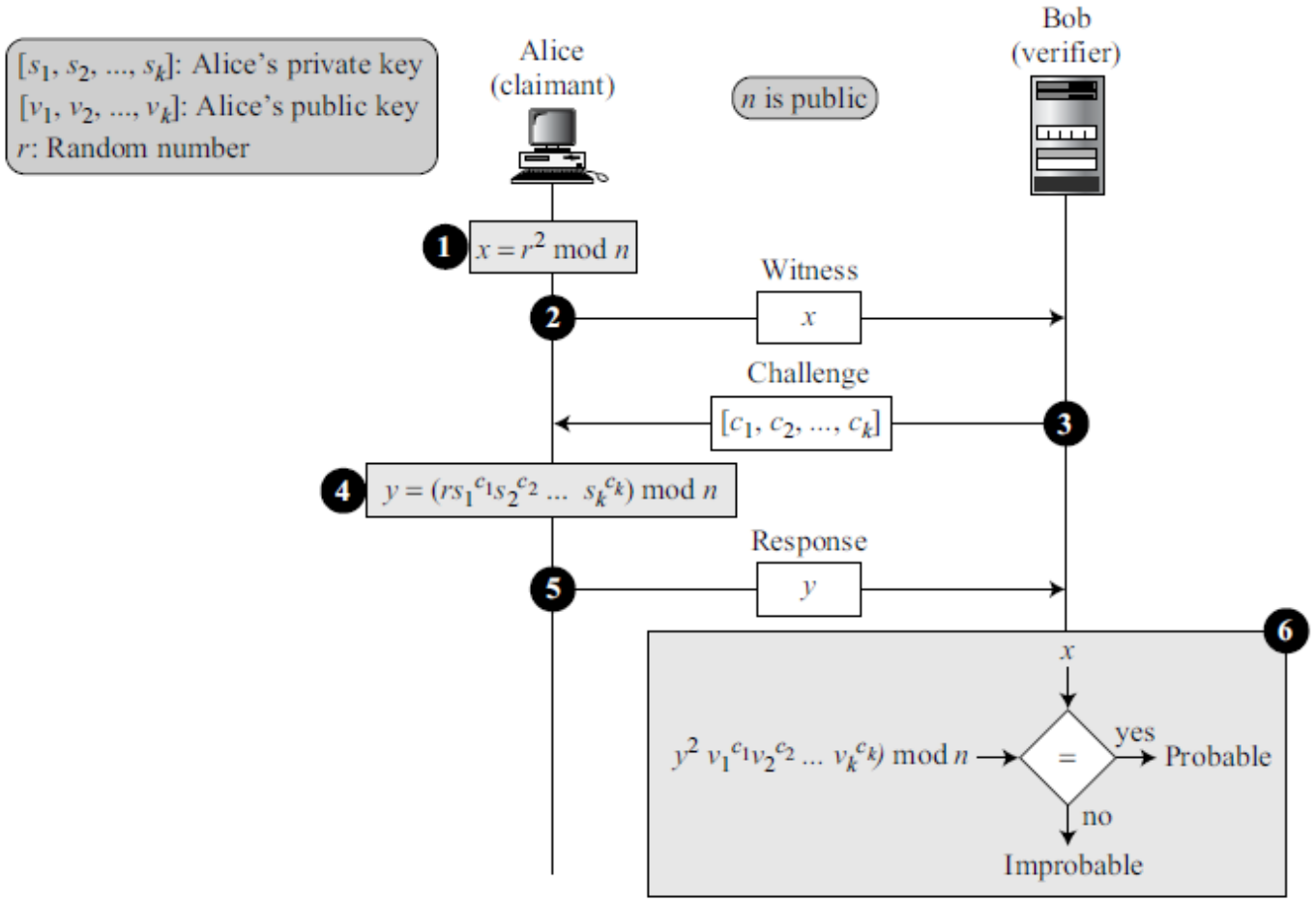
d. If her guess is wrong (c turned out to be 1), she cannot find a value of y that passes the rest.

She probably quits or sends a value that does not pass the test and Bob will abort the process.

- dishonest claimant has a 50 percent chance of fooling the verifier and passing the test (by predicting the value of the challenge).
- In other words, Bob assigns a probability of 1/2 to each round of the test. If the process is repeated 20 times, the probability decreases to $(1/2)^{20}$ or 9.54×10^{-7}.
- It is highly improbable that Alice can guess correctly 20 times.

Feige-Fiat-Shamir Protocol

- The Feige-Fiat-Shamir protocol uses a vector of private keys [s1, s2, …, sk], a vector of public keys [v1, v2, …, vk], and a vector of challenges (c1, c2, …, ck).
- The private keys are chosen randomly, but they must be relatively prime to n.
- The public keys are chosen such that $v_i = (s_i^2)^{-1} \bmod n$



$y^2 v_1{}^{c_1} v_2{}^{c_2} \ldots v_k{}^{c_k}$ is the same as x:

$$
\begin{aligned}
y^2 v_1{}^{c_1} v_2{}^{c_2} \ldots v_k{}^{c_k} &= r^2 (s_1{}^{c_1})^2 (s_2{}^{c_2})^2 \ldots (s_k{}^{c_k})^2 \, v_1{}^{c_1} v_2{}^{c_2} \ldots v_k{}^{c_k} \\
&= x \, (s_1{}^2)^{c_1} (v_1{}^{c_1}) \, (s_2{}^2)^{c_2} (v_2{}^{c_2}) \ldots (s_2{}^2)^{c_2} (v_2{}^{c_k}) \\
&= x \, (s_1{}^2 v_1)^{c_1} \, (s_2{}^2 v_2)^{c_2} \ldots (s_k{}^2 v_k)^{c_k} = x \, (1)^{c_1} \, (1)^{c_2} \ldots (1)^{c_k} = x
\end{aligned}
$$

- The three exchanges constitute a round; verification is repeated several times with the value of c's equal to 0 or 1 (chosen randomly).
- The claimant must pass the test in each round to be verified.
- If she fails a single round, the process is aborted and she is not authenticated.

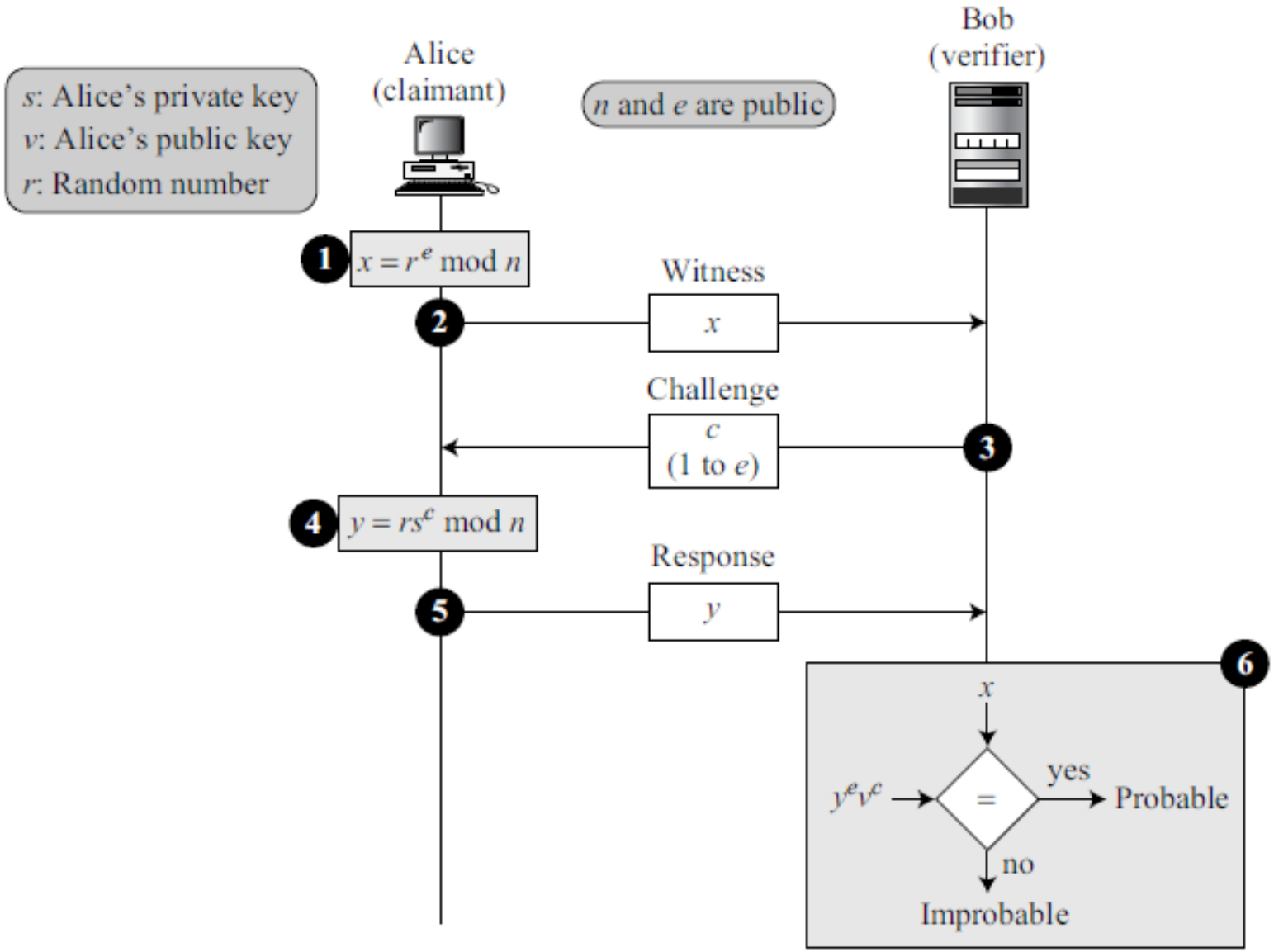
Guillou-Quisquater Protocol

- extension of the Fiat-Shamir protocol in which fewer number of rounds can be used to prove the identity of the claimant.
- A trusted third party chooses two large prime numbers p and q to calculate the value of $n = p \times q$.
- The trusted party also chooses an exponent, e, which is coprime with ϕ, where $\phi = (p - 1)(q - 1)$.
- The values of n and e are announced to the public; the values of p and q are kept secret.
- trusted party chooses two numbers for each entity, v which is public and s which is secret.
- Relationship between v and s is different: $s^e \times v = 1 \bmod n$.

- The three exchanges constitute a round; verification is repeated several times with a random value of c (challenge) between 1 and e.
- The claimant must pass the test in each round to be verified. If she fails a single round, the process is aborted and she is not authenticated.

$$y^e \times v^c = (r \times s^c)^e \times v^c = r^e \times s^{ce} \times v^c = r^e \times (s^e \times v)^c = x \times 1^c = x$$

Guillou-Quisquater protocol



BIOMETRICS

- Biometrics is the measurement of physiological or behavioral features that identify a person (authentication by something inherent).
- Biometrics measures features that cannot be guessed, stolen, or shared.

Components

- capturing devices, processors, and storage devices.
- Capturing devices such as readers (or sensors) measure biometrics features.
- Processors change the measured features to the type of data appropriate for saving.
- Storage devices save the result of processing for authentication

Enrollment

- Before using any biometric techniques for authentication, the corresponding feature of each person in the community should be available in the database.
- This is referred to as enrollment

Authentication

- Authentication is done by verification or identification.

Verification

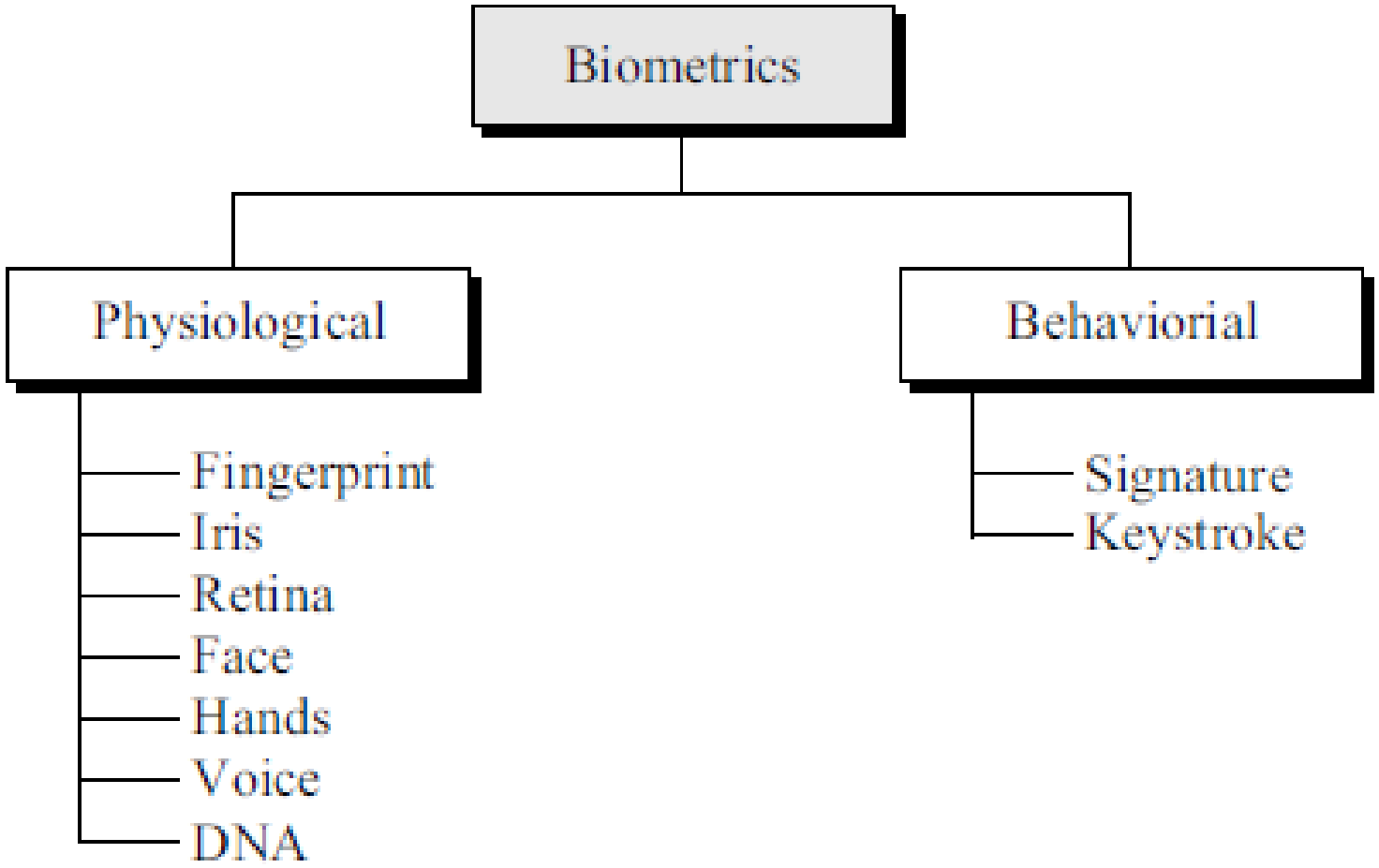
- In verification, a person's feature is matched against a single record in the database (one-to-one matching) to find if she is who she is claiming to be.
- example, when a bank needs to verify a customer's signature on a check

Identification

- In identification, a person's feature is matched against all records in the database (oneto-many matching) to find if she has a record in the database.
- example, when a company needs to allow access to the building only to employees.

Biometrics Techniques

- divided into two broad categories: physiological and behavioral.



Fingerprint

- minutiae-based
  - system creates a graph based on where individual ridges start/stop or branch.
- Image based
  - system creates an image of the fingertip and finds similarities to the image in the database.
- been used for a long time.
- show a high level of accuracy and support verification and identification.
- can be altered by aging, injury, or diseases

Iris

- measures the pattern within the iris that is unique for each person.
- normally requires a laser beam (infrared).
- very accurate and stable over a person’s life.
- also support verification and identification.
- some eye diseases, such as cataracts, can alter the iris pattern.

Retina

- The devices for this purpose examine the blood vessels in the back of the eyes.
- these devices are expensive and not common yet.

Face

- This technique analyzes the geometry of the face based on the distance between facial features such as the nose, mouth, and eyes.
- Some technologies combine geometric features with skin texture.
- Standard video cameras and this technique support both verification and identification.
- accuracy can be affected by eyeglasses, growing facial hair, and aging.

Hands

- This technique measures the dimension of hands, including the shape and length of the fingers.
- This technique can be used indoors and outdoors.
- However, it is better suited to verification rather than identification.

Voice

- Voice recognition measures pitch, cadence, and tone in the voice.
- can be used locally (microphone) or remotely (audio channel).
- mostly used for verification.
- accuracy can be diminished by background noise, illness, or age.

DNA

- DNA is the chemical found in the nucleus of all cells of humans and most other organisms.
- The pattern is persistent throughout life and even after death.
- Extremely accurate.
- can be used for both verification and identification.
- The only problem is that identical twins may share the same DNA.

Behavioral Techniques

- measure some human behavior traits.
- behavioral techniques need to be monitored to ensure the claimant behaves normally and does not attempt to impersonate someone else

Signature

- human experts today determine whether a signature on a check or a document is the same as a signature on file.
- Biometric approaches use signature tablets and special pens to identify the person.
- These devices not only compare the final product, the signature, they also measure some other behavioral traits, such as the timing needed to write the signature.
- Signatures are mostly used for verification.

Keystroke

- keystrokes (typing rhythm) technique measures the behavior of a person related to working with a keyboard.
- can measure the duration of key depression, the time between keystrokes, number and frequency of errors, the pressure on the keys, and so on.
- inexpensive because it does not require new equipment.
- it is not very accurate because the trait can change with time (people become faster or slower typists).
- also text dependent.

- Accuracy
- Accuracy of biometric techniques is measured using two parameters:
- False rejection rate (FRR) –
  - measures how often a person, who should be recognized, is not recognized by the system.
  - FRR is measured as the ratio of false rejection to the total number of attempts (in percentage).

- false acceptance rate (FAR)-
  - measures how often a person, who should be recognized, is not recognized by the system.
  - FRR is measured as the ratio of false rejection to the total number of attempts (in percentage).

References

- Behrouz A. Forouzan and Debdeep Mukhopadhyay – “Cryptography and Network Security”, McGraw Hill, 2nd Edition, 2008.