Sanjivani Rural Education Society's

Sanjivani College of Engineering, Kopargaon-423603

(An Autonomous Institute Affiliated to Savitribai Phule Pune University, Pune)

NAAC 'A' Grade Accredited, ISO 9001:2015 Certified

Department of Information Technology

(NBA Accredited)

Cryptography and Cyber Security [IT311]

Mrs. Kanchan D. Patil

Assistant Professor

Unit 3: Message Digest & Key Management

- Hash Algorithms: SHA-1, MD5, Key Management: Introduction, Key Management: Generations, Distribution, Updation, Digital Certificate, Digital Signature, Kerberos 5.0.

Kerberos

- It is an **authentication protocol**.
- The basis for Kerberos is another protocol, called as **Needham Shroeder.**
- It was designed at MIT to allow the workstations to **network resources in a secure manner**
- The name Kerberos signifies a **multi-headed dog** in the Gr mythology (apparently used to keep outsiders away).

Kerberos : Working

- There are four parties involved in the Kerberos protocol:

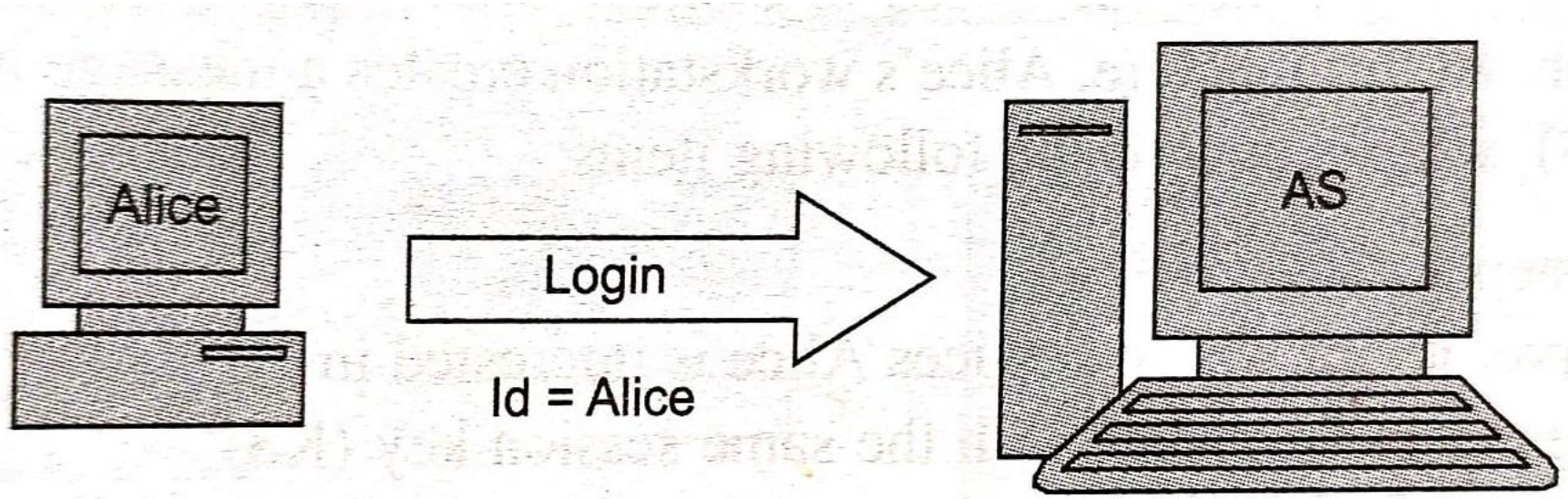
- **Alice :** The client workstation
- **Authentication Server (AS):** Verifies (authenticates) the user during login
- **Ticket Granting Server (TGS):** Issues tickets to certify proof of identity
- **Bob:** The server offering services such as network printing, file sharing or an application program

Kerberos : Working

- The job of AS is **to authenticate every user** at the login time.
- AS shares a **unique secret password** with every user.
- The job of TGS is **to certify to the servers** in the network that a **user** is really what she claims to be.
- For proving this, the **mechanism of tickets** (which allow entry into a server, just as a ticket allow parking a car or entering a music concert) is used.

Kerberos : Working

- There are three primary steps in the Kerberos protocol.
- **Step 1: Login**
  - To start with, Alice, the user, sits down at an arbitrary public workstation and enters her name.
  - The work station sends her name in plain text to the AS, as shown in Figure (Alice Sends login request to AS)

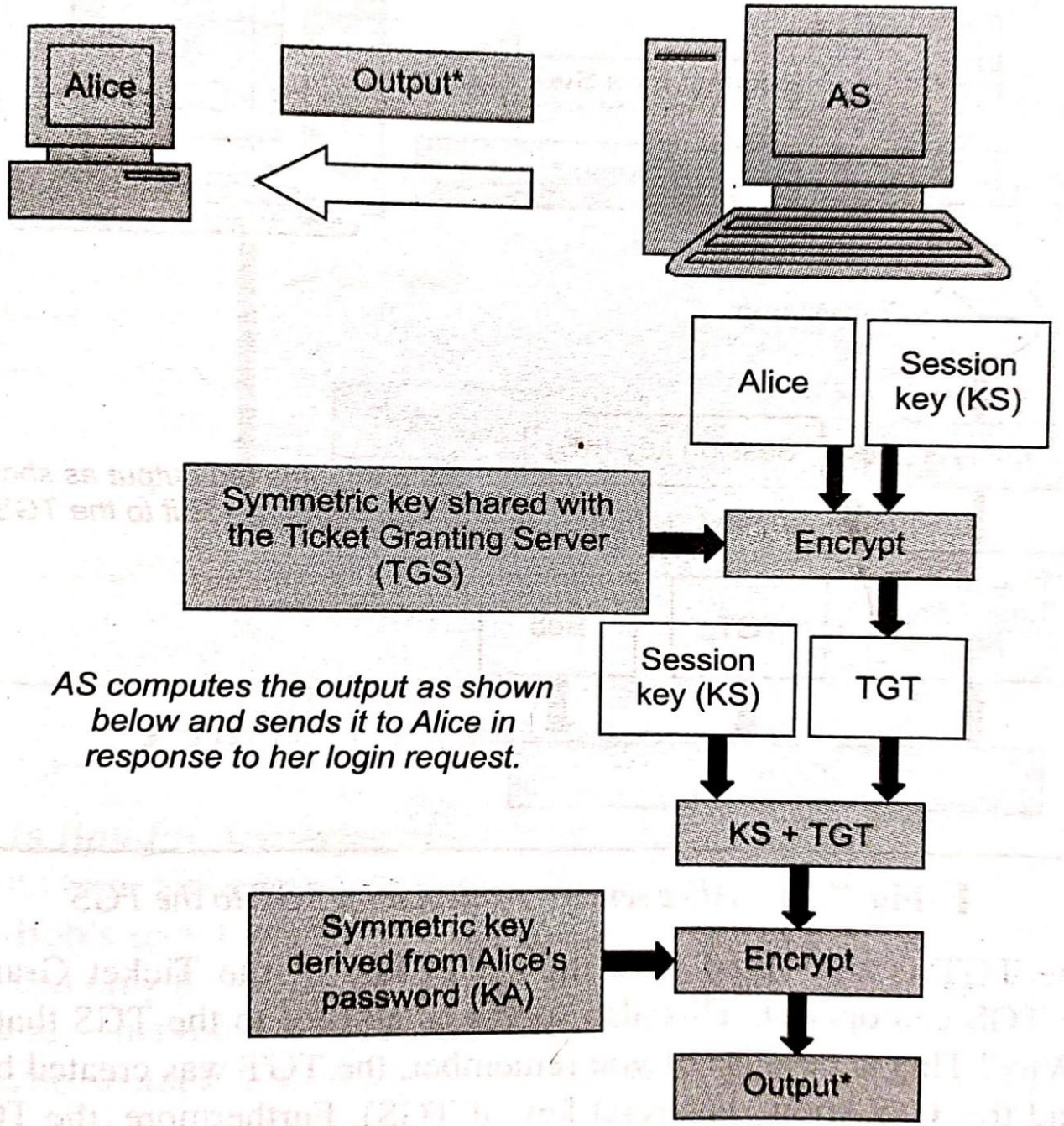


Kerberos : Working

- **Step 1: Login**
  - In response, the AS performs several actions.
  - It first creates a **package of the user name** (Alice) and a randomly generated **session key** (KS).
  - It encrypts this package with the **symmetric key** that the **AS shares** with the **Ticket Granting Server (TGS).**
  - The output of this step is called as the **Ticket Granting Ticket** (TGT). **Note that the TGT can be opened only by the TGS,** since only it possesses the corresponding **symmetric key for decryption.**
  - The AS then combines the TGT with the session key (KS), and encrypts the two together using a **symmetric key** derived from the password of Alice (KA). **Note that the final output can, therefore, be opened only by Alice.**

Kerberos : Working

- Step 1: AS sends back encrypted session key and TGT to Alice



Kerberos : Working

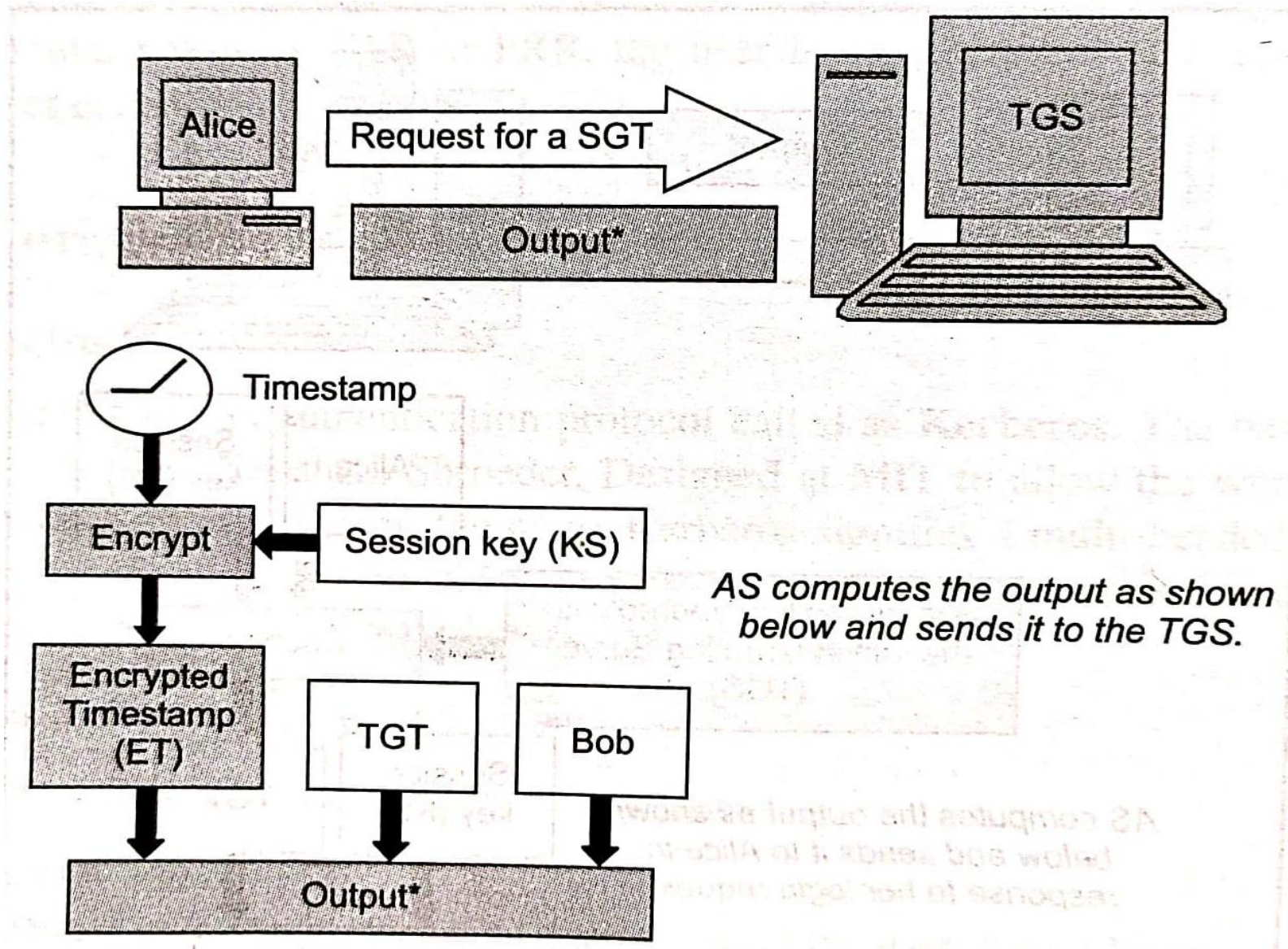
- **Step 1: Login**
  - After this message is received, Alice's workstation asks her for the password.
  - When Alice enters it, the workstation **generates the symmetric key** (KA) derived from the password (in the same manner as AS would have done earlier) and **uses that key to extract the session key (KS) and the Ticket Granting Ticket (TGT).**
  - The workstation destroys the password of Alice form its memory immediately, to prevent an attacker from stealing it.
  - **Note that Alice cannot open the TGT, as it is encrypted with the key of the TGS.**

Kerberos : Working

- **Step 2: Obtaining a service granting ticket (SGT)**
  - Now, let us assume that after a successful login, Alice wants to make use of Bob - the email server, for some email communication.
  - For this, Alice would inform her workstation that she needs to contact Bob.
  - Therefore, Alice needs a ticket to communicate with Bob.
  - Alice's workstation creates a message intended for the Ticket Granting Server (TGS), which contains the following items:
    - The TGT as in step 1
    - The id of the server (Bob) whose services Alice is interested in
    - The current timestamp, encrypted with the same session key

Kerberos : Working

- Step 2: Alice sends a request for a SGT to the TGS

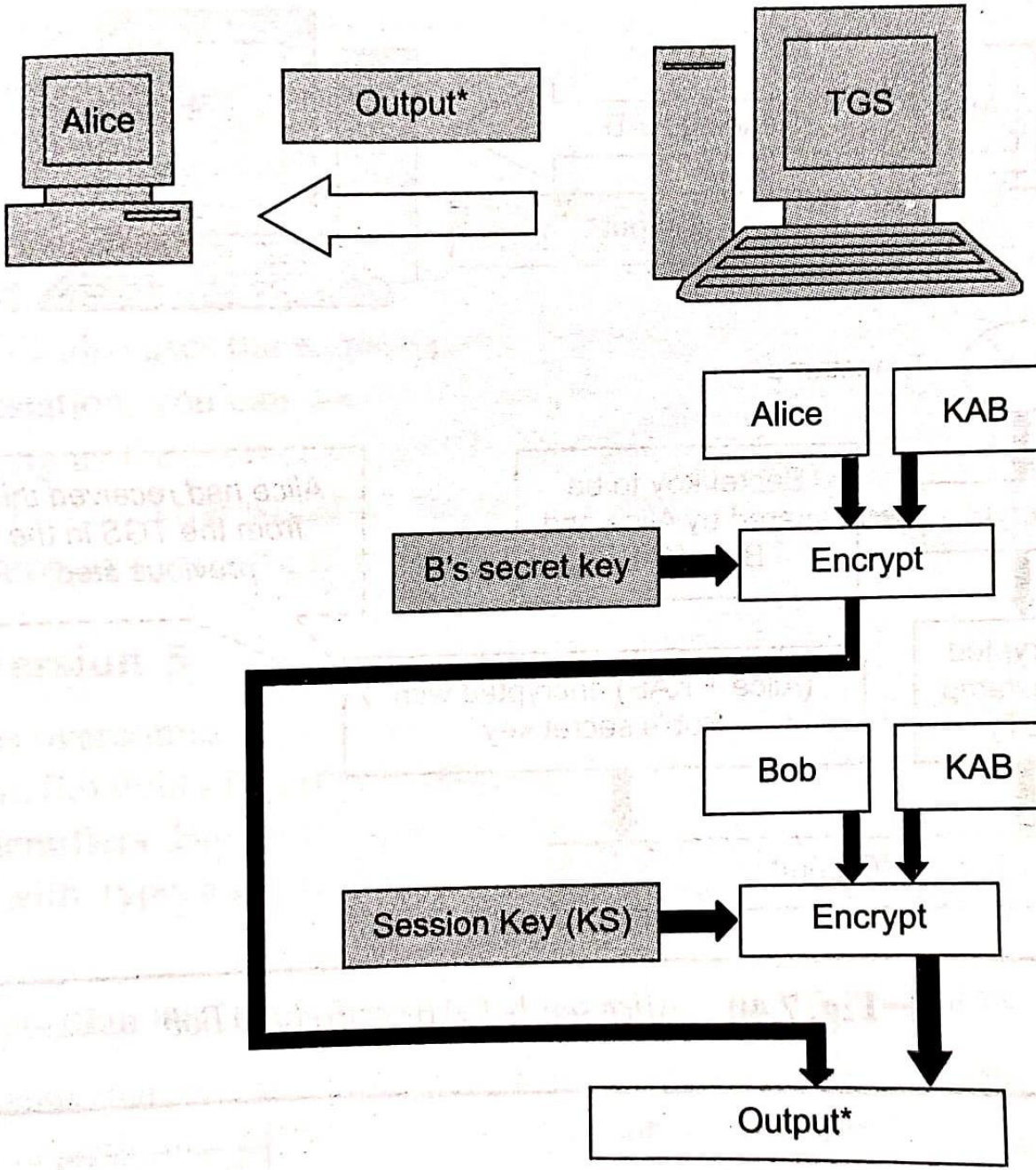


Kerberos : Working

- **Step 2: Obtaining a service granting ticket (SGT)**
  - TGT is encrypted with the secret key of the Ticket Granting Server (TGS) so, only the TGS can open it.
  - This also serves as a proof to the TGS that the message indeed came from Alice.
  - Once the TGS is satisfied of the credentials of Alice, the TGS creates a session key KAB, for Alice to have secure communication with Bob.
  - **TGS sends session key twice to Alice:**
    - Once combined with Bob's id (Bob) and encrypted with the session key (KS)
    - Second time, combined with Alice's id (Alice) and encrypted with Bob's secret key (KB).

Kerberos : Working

- Step 2: TGS sends response back to Alice.



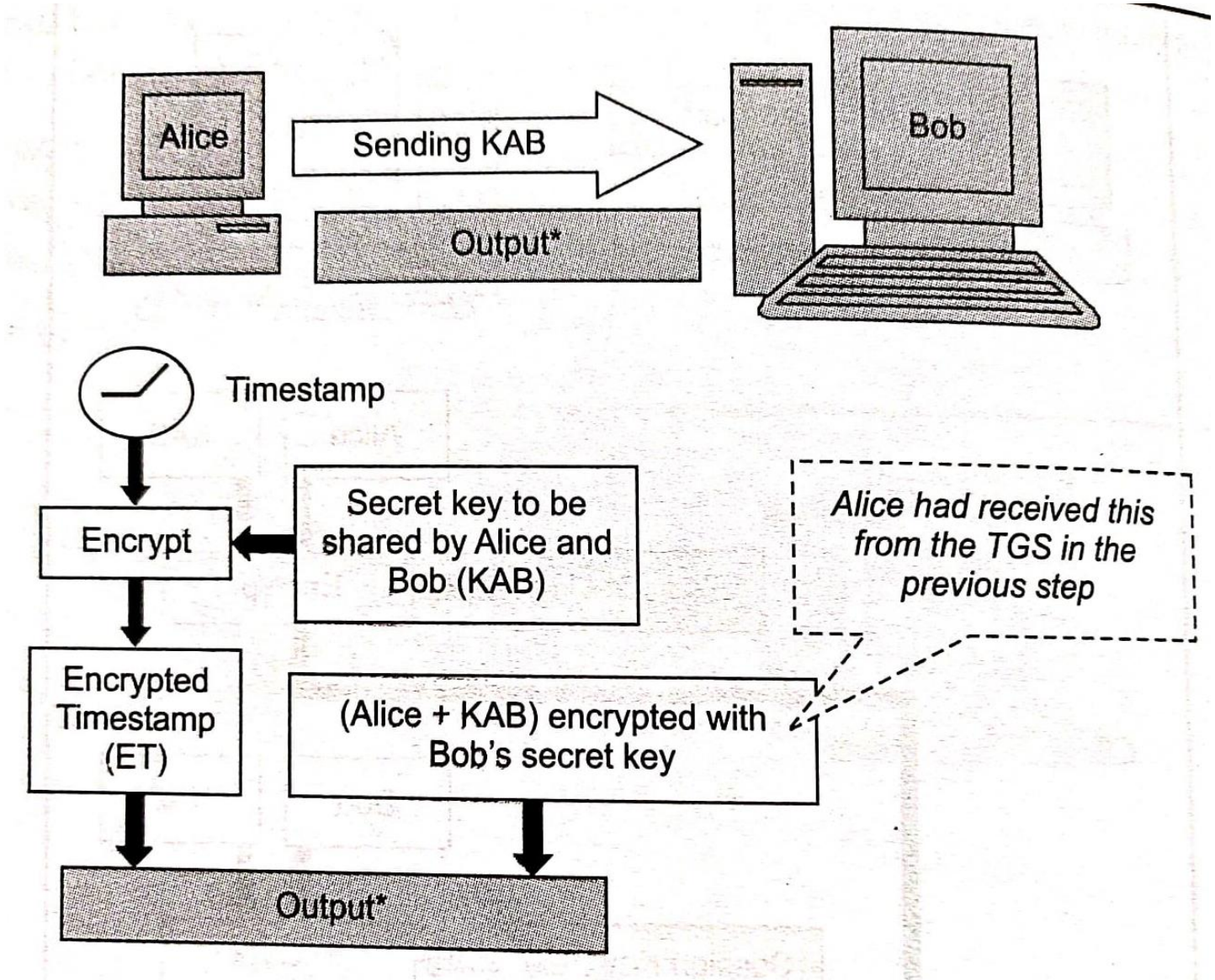
Kerberos : Working

- **Step 3: User contacts Bob for accessing the server**

  - Alice can now send KAB to Bob in order to enter into a session with him.
  - Since this exchange is also desired to be secure, Alice can simply forward KAB encrypted with Bob's secret key to Bob (which she had received from the TGS in the previous step).
  - This will ensure that only Bob can access KAB.
  - To guard against replay attacks, Alice also sends the timestamp, encrypted with KAB to Bob.

Kerberos : Working

- Step 3: Alice sends KAB securely to BOB

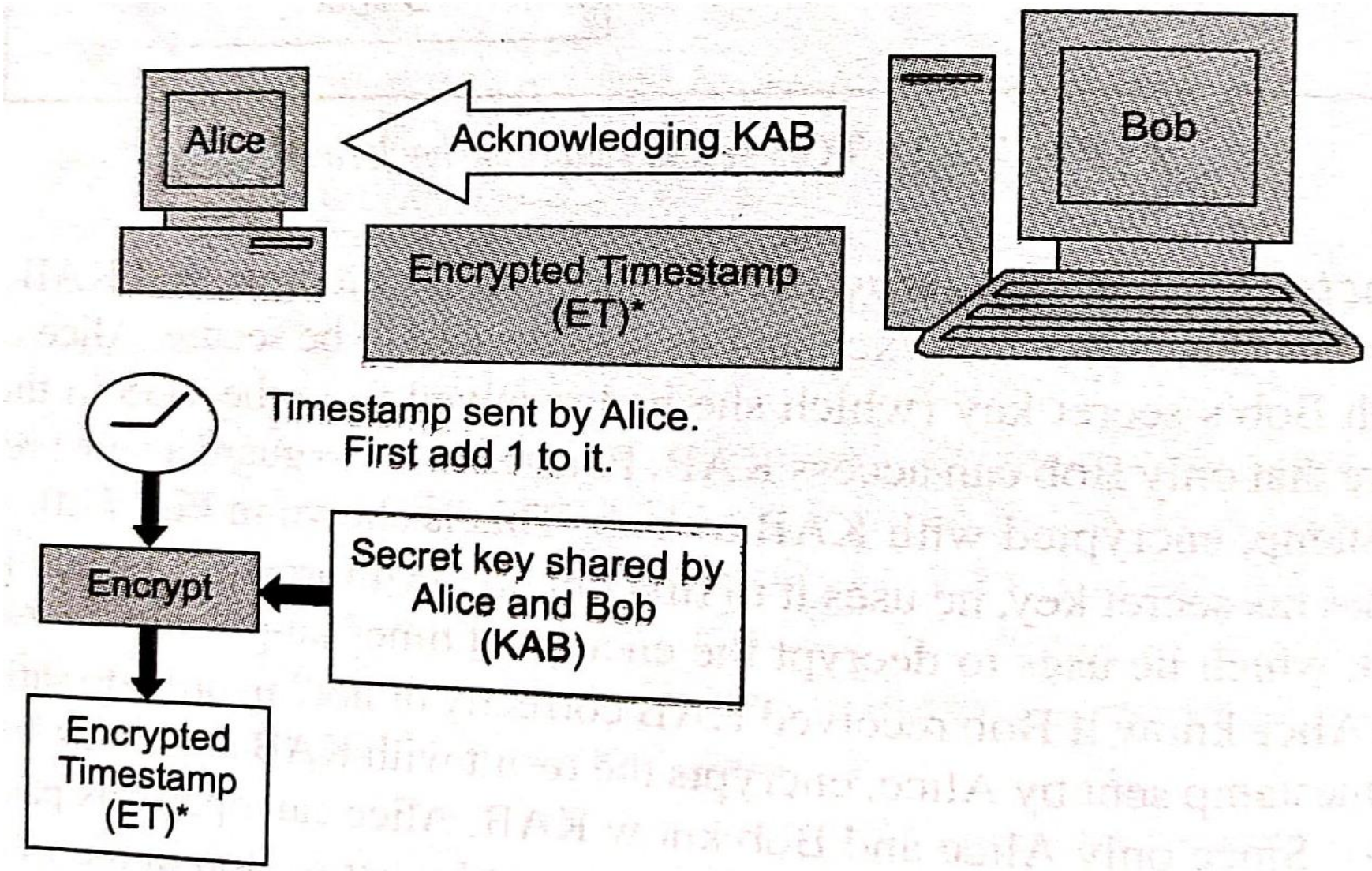


Kerberos : Working

- **Step 3: User contacts Bob for accessing the server**
  - Only Bob has his secret key, he uses it to first obtain the information (Alice + KAB).
  - From this, it gets the key KAB, which he uses to decrypt the encrypted timestamp value.
  - To confirm that Bob received KAB correctly, Bob now adds 1 to the timestamp sent by Alice, encrypts the result with KAB and sends it back to Alice.
  - Since only Alice and Bob know KAB, Alice can open this packet and verify that the timestamp incremented by Bob was indeed the one sent by her to Bob in the first place.
  - Now, Alice and Bob can communicate securely with each other.

Kerberos : Working

- Step 3: BOB acknowledge the receipt of KAB



Kerberos Version 5

- Version 5 of Kerberos overcomes some of the shortcomings of Version 4.
- Version 4 demands the use of DES.
- Version 5 allows flexibility in terms of allowing the choice of other algorithms.
- Version 4 depends on IP addresses as identifiers.
- However, Version 5 allows the use of other types as well (for this, it tags network addresses with type and length).

Kerberos Version 5

- **Following are the key differences between Kerberos Versions 4 and 5.**

- The key salt algorithm has been changed to use the entire principal name.
  - This means that the same password will not result in the same encryption key in different realms or with two different principals in the same realm.
- The network protocol has been completely redone and now uses ASN.I encoding everywhere.
- There is now support for replay caches, so authenticators are not vulnerable to replay.
- There is support for transitive cross-realm authentication.

Kerberos Version 5

- Kerberos tickets can now contain multiple IP addresses and addresses for different types of networking protocols.
- A generic crypto interface module is now used, so other encryption algorithms beside DES can be used.

- There is now support for forwardable, renewable and postdatable tickets.
  - **Forwardable:**
    - The user can use this ticket to request a new ticket, but with a different IP address.
    - Thus, a user can use their current credentials to get credentials valid on another machine.

Kerberos Version 5

- There is now support for forwardable, renewable and postdatable tickets.
  - **Renewable:**
    - A renewable ticket can be renewed by asking the KDC for a new ticket with an extended lifetime.
    - However, the ticket itself has to be valid (in other words, we cannot renew a ticket that has expired; we have to renew it before it expires).
    - A renewable ticket can be renewed up until the maximum renewable ticket lifetime.
  - **Postdatabale:**
    - These are tickets which are initially invalid and have a starting time some time in the future.
    - To use a postdatable ticket, the user must send it back to the KDC to have it validated during the ticket's valid lifetime.

References:

- Atul Kahate,”Cryptography and Network Security”, second edition, Tata McGraw Hill