# Lecture4: Authentication and Password

# Introduction to Authentication

- **Authentication** is a fundamental aspect of software security, ensuring that only authorized users or systems can access the software and its resources. It involves verifying the identity of a user, device, or other entity before granting access to the system.

The most common method of authentication is a unique login and password, but as cybersecurity threats have increased in recent years, most organizations use and recommend additional authentication factors for layered security.

- **History of Authentication:**

Digital authentication goes back to the 1960s when modern computers became available at large research institutes and universities. Back then, computers were massive—often taking up entire rooms—and a scarce resource. Most universities that had a computer only had one. That meant students and researchers had to share it. But this also meant that users could access other users' files without limitation. When Fernando Corbato, a student at MIT, noticed this weakness, he created a basic password program that prompted the user to enter their password and saved it within a plaintext file in the file system. From there, digital authentication was born.

# Introduction to Authentication

- **Authentication Use Cases:**

Today, authentication is common practice not only among IT professionals and scientists, but for non-technical users as well. Whether that's logging in to Facebook with a username and password or opening a phone with Touch ID or a unique PIN, most people have used authentication to access their private information and devices at home and at work. Of course, as technology has advanced and hackers have become more adept and widespread, new methods of authentication are gaining traction to better secure personal, business, and government resources from unauthorized access.

# Authentication mechanisms

Authentication mechanisms use any of three qualities to confirm a user's identity.

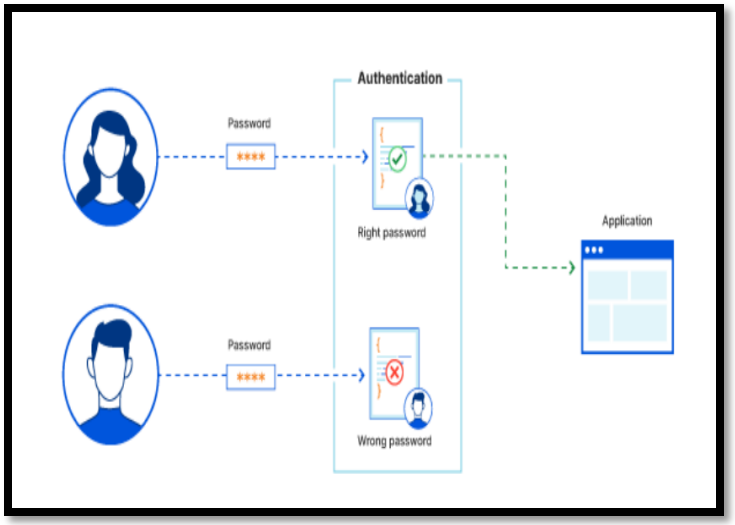
- **Something known:** indicates that we authenticate based on some information that only the user has knowledge of. the Passwords or PIN numbers.
- **Something possessed:** can be combined with the first quality or used alone. Verifying a user’s identity based on this quality means that we authenticate using something the user has in his possession, such as passport, a driver’s license, an identification card, a credit card, and a smart
- **Something inherent:** refers to the biometric properties of the user. If the properties are unique each user, it is possible to authenticate based on them. For example, conventional signatures, fingerprints, voice, facial characteristics, retinal pattern, and handwriting.

# Password-based authentication

Passwords are the most common methods of authentication. Passwords can be in the form of a string of letters, numbers, or special characters. To protect ourselves we need to create strong passwords that include a combination of all possible options.

Most people use simple passwords instead of creating reliable passwords because they are easier to remember. An average person has about 25 different online accounts, but only 54% of users use different passwords across their accounts.

However, passwords are prone to several attacks include eavesdropping, stealing a password, accessing the password file, guessing, and the dictionary attack.



# Password-based authentication

**Characteristics of Strong Passwords:**

There are many characteristics that make a password easy to crack using exhaustive attacks. Some of the characteristics are:

- Passwords are easily guessable.
- Passwords are relatively short in length.
- Passwords have some sorts of the pattern which is easy to deduce. e.g. abc123, XYZ, 46824682 etc.
- Passwords have been repeating characters like abab11 or xy12x. People use the passwords with repeating characters so that they can remember

Having passwords with a random sequence of numbers with at least one capital letter, number, a Special character from large key space makes password very strong

# Password-based authentication

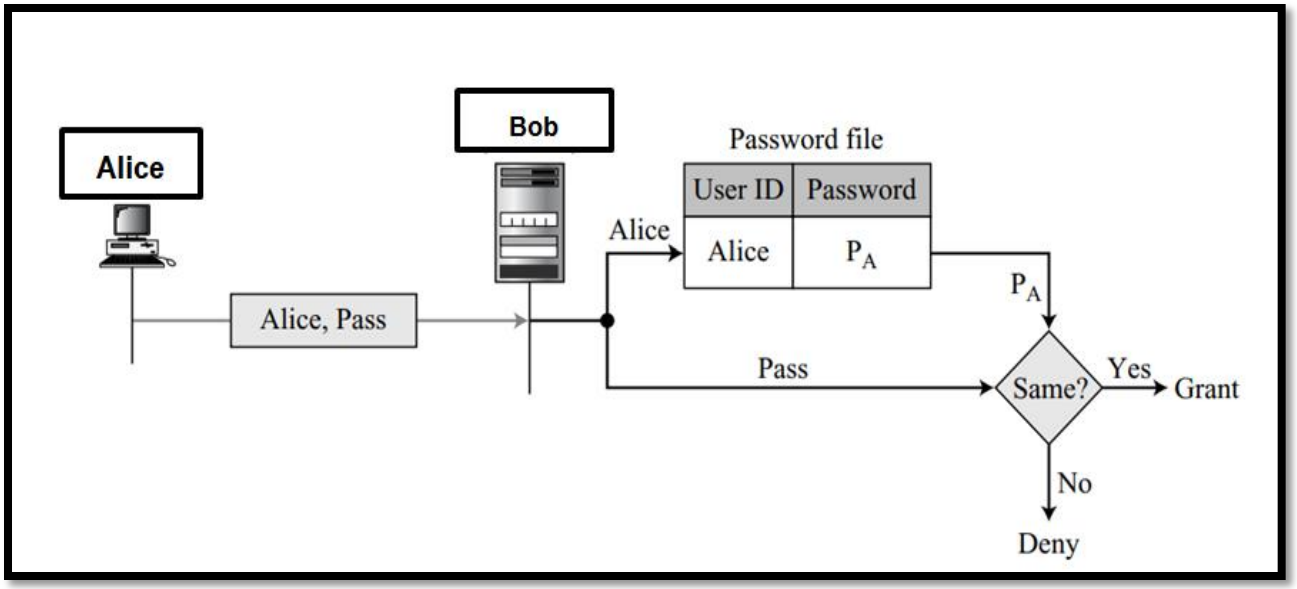
Password-based authentication can be divided into two broad categories:

- Fixed Password: is a password that is used over and over again for every access.
- one-time Password: is a password that is used only once. This kind of password makes eavesdropping and salting useless.

# Password-based authentication

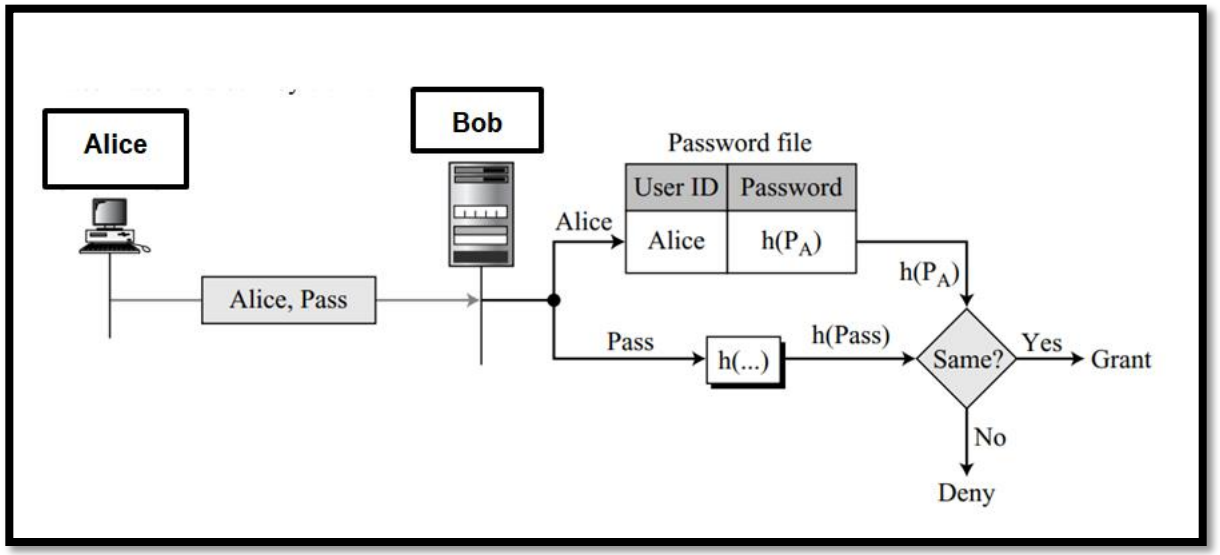
- **Fixed Password:** is a password that is used over and over again for every access.

- In the very elementary approach, the system keeps a table (a file) that is sorted by user identification. To access the system resources, the user sends his 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. This approach is subject to several kinds of attack: Eavesdropping, Stealing a password, Accessing a password file, Guessing.



# Fixed Password

- A more secure approach is to 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. In this case, the file does not need to be read protected. This approach is subject to dictionary Attack.



# Fixed Password

- The third approach is called 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