Access Control Systems

COMP 4670 - Network Security

Lesson 04

Outlines

- Introduction to Access Control
- Authentication Methods
- Authentication Design Approaches
- Network Application Authentication Protocols

Access Control

- An access control system is the first line of defense against unauthorized access to computer and networks assets.
- In fact, controlling access to assets is one of the central themes of computer and network security.
- An **asset** could be data sources, software, devices, personnel, and other information
- An access control system simply controls access to valuable resources by granting and denying access privileges.
- An access control system will provide identification, authentication, authorization, and accountability.

Subject and Object

What is the Subject?

- A subject is an **active** entity that accesses a passive object to receive information from, or data about, an object. Subjects can be users, programs, processes, computers, or anything else that can access a resource.

What is the Object?

- An object is a **passive** entity that provides information to active subjects. Some examples of objects include files, databases, computers, programs, processes, printers, and storage media.

Access Control System: Basic Functions

An access control is any hardware, software, security policy or instructions that help the system to carry on the following functions:

1. Identify and authenticate users or other subjects attempting to access resources.
2. Determine whether the access is authorized.
3. Grant or restrict access based on the subject's identity.
4. Monitor and record access attempts.

Access Control Methods

Preventive Access Control:

- Focus on stopping unwanted or unauthorized activity from occurring.
- **Examples:** authentication systems, firewalls, fences, security training, security policies

Detective Access Control:

- Operate after the fact and can discover the activity only after it has occurred.
- **Examples:** intrusion detection systems, vulnerability assessment.

Access Control Methods

Deterrent Access Control:

- Discourage security policy violations by subjects. Deterrent and preventive controls are similar, but deterrent controls often depend.
- **Examples:** security policy awareness, security logs, and security camera.

Corrective Access Control:

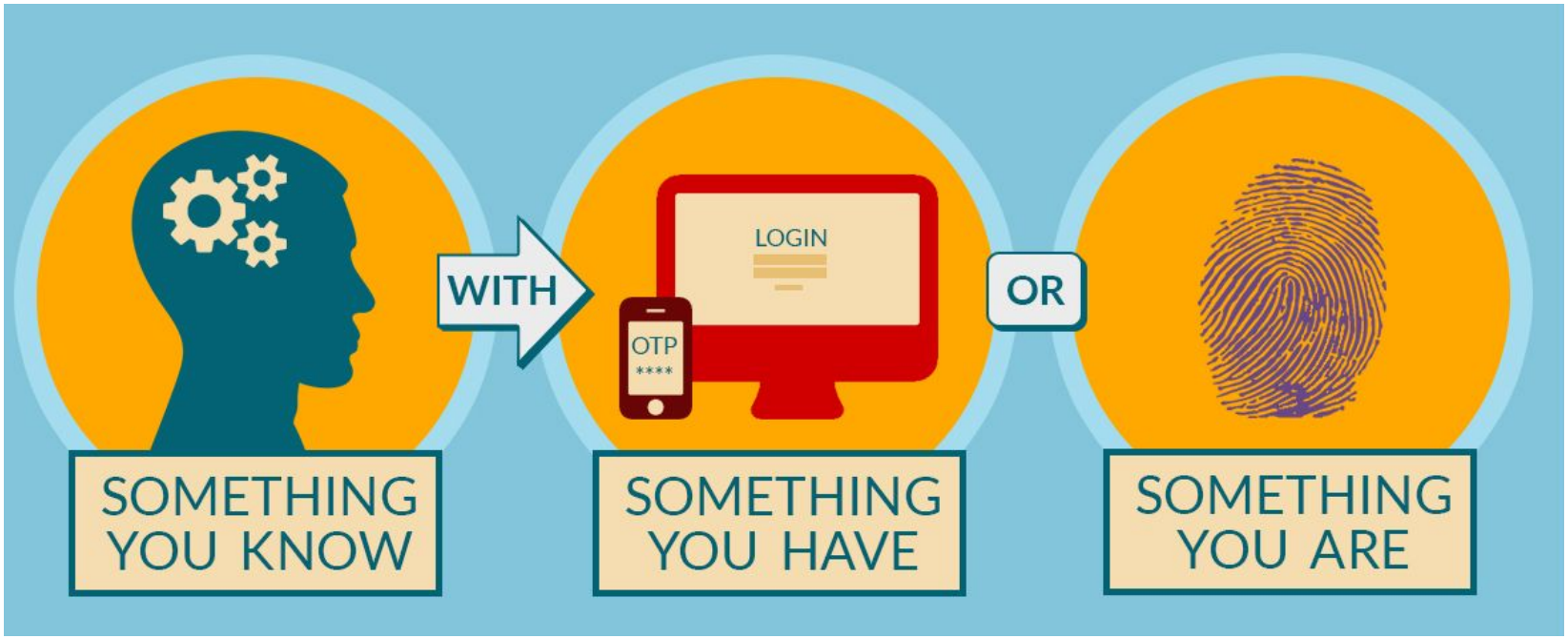
- A corrective control modifies the environment to return systems to normal after an unwanted or unauthorized activity has occurred.
- **Examples:** antivirus, backup and restore plans, system recovery plans.

Authentication Systems

- Authentication plays a major role in any access control system.
- Authentication requires two systems, namely, **identification** and **verification**.
- Identification is the process of a subject claiming an identity.
- Verification verifies the identity of the subject by comparing one or more factors against a database of valid identities.
- Identification and authentication always occur together as a single two-step process.

Authentication Methods

- There are different authentication techniques, but in general, they are categorized into three main methods.



Authentication Methods

There are three main methods of authentication, also known as authentication factors.

Authentication Factors:

- **Knowledge Factors:** all the information, the subject (user) must know to have access to the object (e.g., user account).
- **Token Factors:** any physical or digital token that subject must possess to have access to the object
- **Inherence Factors:** physical or behavioral characteristics of the subject that are uniques and sufficient to give access to the object

Knowledge Based Authentication

Knowledge-based authentication relies on a shared secret known only to the authorized subjects and could be verified by the authentication system.

Examples:

- Password
- PIN Code
- Pass Code
- Passphrase
- Visual Password Pattern Lock
- API Key (eg. for web services clients, or mobile apps)

Knowledge Based Authentication

Password:

- There is no **minimum password length** everyone agrees on.
- A good password should be minimum of 12 to 14 characters in length (NO NOT 8 characters).
- Should **not be guessable** (e.g. dictionary password or semantically linked to the subject).
- Mix of alphanumeric characters, special characters, upper and lower case alphabetic characters.

The reason for 8 character length password ??

Take five chimpanzees. Put them in a big cage. Suspend some bananas from the roof of the cage. Provide the chimpanzees with a stepladder. BUT also add a proximity detector to the bananas, so that when a chimp goes near the banana, water hoses are triggered and the whole cage is thoroughly soaked.

Soon, the chimps learn that the bananas and the stepladder are best ignored.

Now, remove one chimp, and replace it with a fresh one. That chimp knows nothing of the hoses. He sees the banana, notices the stepladder, and because he is a smart primate, he envisions himself stepping on the stepladder to reach the bananas. He then deftly grabs the stepladder... and the four other chimps spring on him and beat him squarely. He soon learns to ignore the stepladder.

Then, remove another chimp and replace it with a fresh one. The scenario occurs again; when he grabs the stepladder, he gets mauled by the *four* other chimps -- yes, including the previous "fresh" chimp. He has integrated the notion of "thou shallt not touch the stepladder".

Iterate. After some operations, you have five chimps who are ready to punch any chimp who would dare touch the stepladder -- and none of them knows why.

Originally, some developer, somewhere, was working on an old Unix system from the previous century, which used the old DES-based "crypt", actually a password *hashing* function derived from the DES block cipher. In that hashing function, only the first eight characters of the password are used (and only the low 7 bits of each character, as well). Subsequent characters are ignored. That's the banana.

The Internet is full of chimpanzees.

Password or Pattern lock

- Given **9 dots grid** where we need to connect at least four dots and do not repeat any of them.
- We could have a most **389,112** unique pattern locks
- A password of **8 digits** (only 0-9) length have **100,000,000**

Is this a permutation or combination problem?

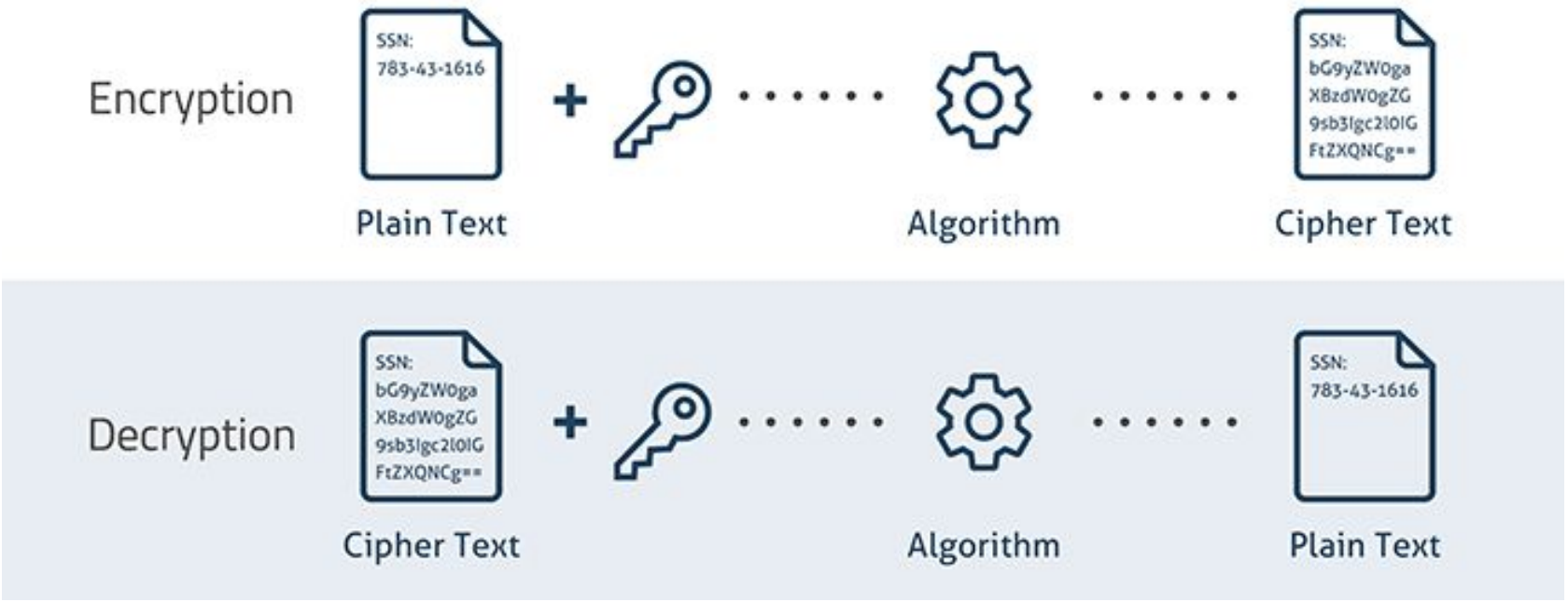
Why using Pattern lock over password?

Password Attacks and Security

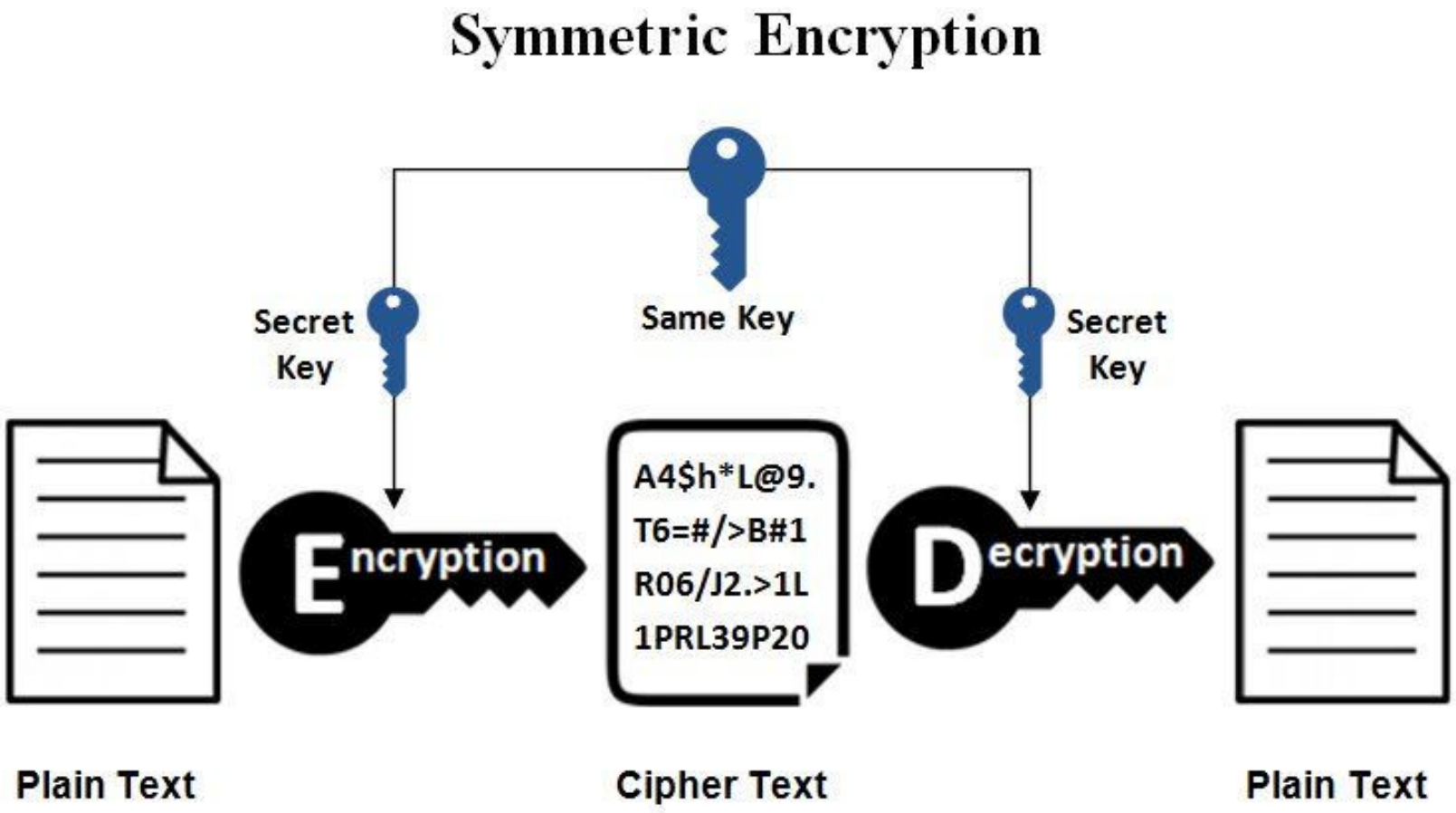
- Storing the password in secure storage over the network at the server side is very important.
- If the password is stored in plain text (human-readable) then if the password file obtained an attacker could easily attack the system and bypass the authentication step.
- The best way to protect the password is to use secure hash to hash the password and only store the message digest (the output of the hashing)
- If the attacker obtained access the password files or database he/she can not recover the password from the hash.

Cryptographic 101

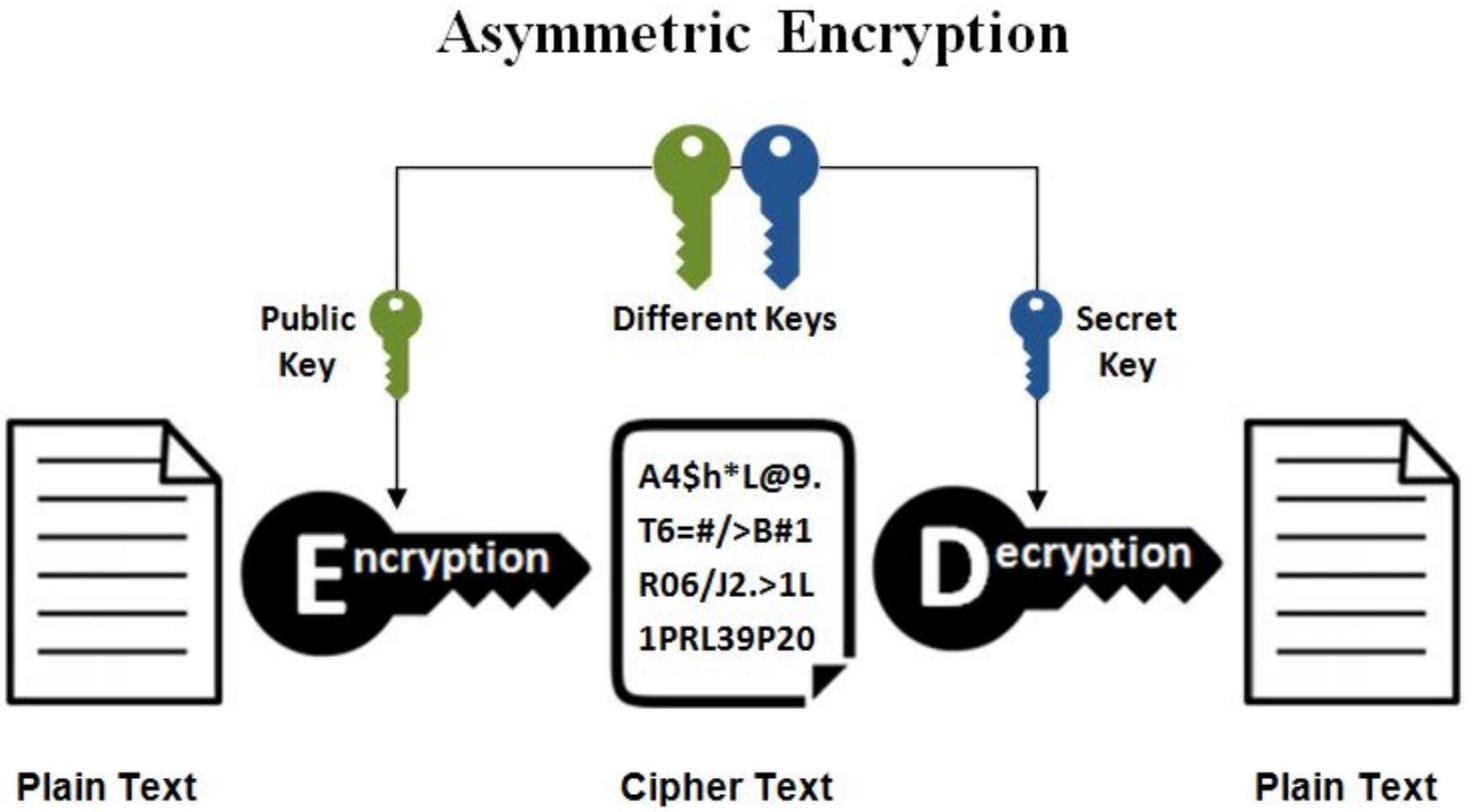
The science of coding and decoding messages so as to keep these messages secure.



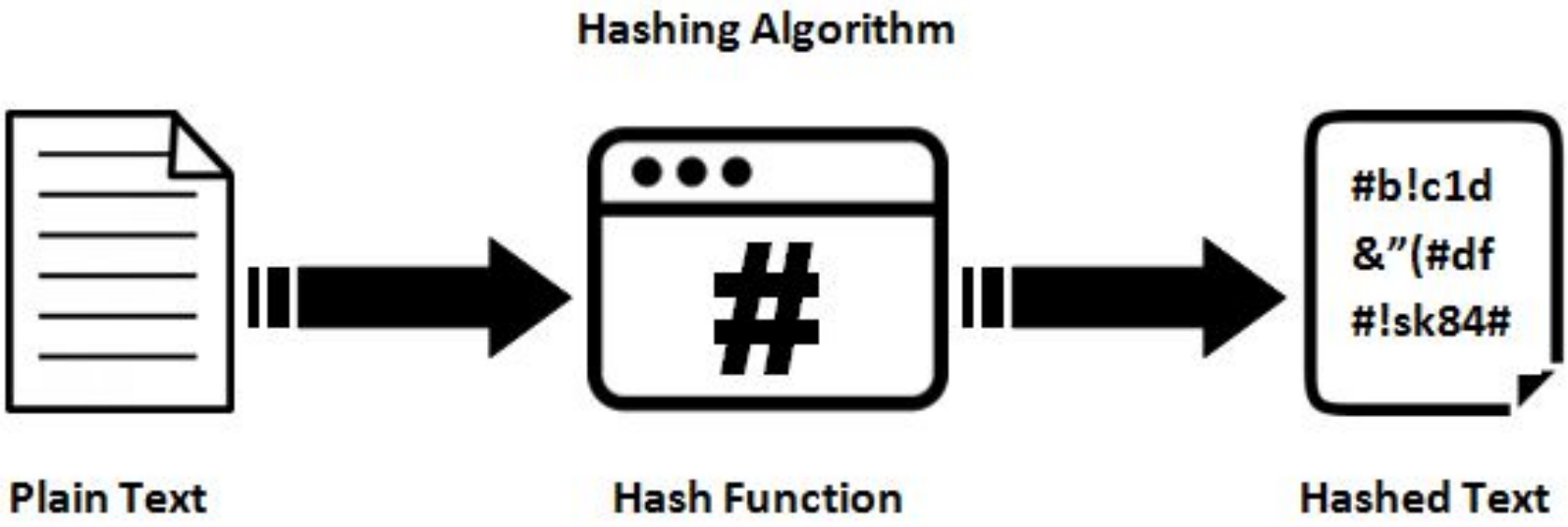
Symmetric Encryption



Asymmetric Encryption | Public Key Encryption



Secure Hashing



Password Attacks and Security

1. Why we do hash the password and not encrypt them?
2. What are the advantages of using hashing vs encryption?
3. How do we know that the system is hashing or encrypting the password?
4. When we send the password over the network from the client to the server should we send it hashed or encrypted?

Password Attack - Dictionary Attack

- A dictionary attack is an attempt to discover passwords by using every possible password in a predefined database or list of common or expected passwords.
- The attacker starts with a database of words commonly found in a dictionary.
- Dictionary attack databases also include character combinations commonly used as passwords.
- Dictionary attack could use a dictionary of top common passwords, foreign password.
- Examples for common password:
    - password1, passXword, r00t1, admin123, 321nimda

Password Attack - Bruteforce

- A **brute-force** attack is an attempt to discover passwords for user accounts by systematically attempting all possible combinations of letters, numbers, and symbols.
- **Defence Techniques:**
  - **Lock out accounts** after a defined number of incorrect password attempts. Account lockouts can last a specific duration, such as one hour, or the accounts could remain locked until manually unlocked by an administrator.
  - Use **adaptive hashing** functions (e.g **bcrypt**)
  - **Block incoming login** attempts from **specific IP address**, or device fingerprint after a defined number of incorrect login attempts

Password Attack - Birthday Attack

- Focuses on finding collisions in the password hashing method.
- Its name comes from a **statistical phenomenon** known as the birthday paradox.
- The birthday paradox states that if there are **23 people** in a room, there is a **50 percent** chance that any two of them will have the same birthday.
- If you know that the hash of the admin account password is **AE6RGSG3**, some tools can identify a password that will create the same hash of **AE6RGSG3**, even if it is not the same password of the admin.
- Defence against this attack is by using a hashing algorithms with a sufficient number of bits to make collisions computationally infeasible

Password Attack - Rainbow Table

- Rainbow attack assumes the attacker has access to hashed password files or database.
- To reduce the time required by the attacker to generate possible passwords hash them and compare them to the hashed password in the password file by using large databases of precomputed hashes.
- This means the attacker does need to generate a password and hash them he/she can simply lookup the hashed passwords from the password file in the Rainbow table if the match exists then he/she discovered the password.

Password Attack - Rainbow Table

Is "8193ccb7h400cefahdwoj2jgflo89asm" exist?

Password Files

Password	MD5 Hash
123456	e10adc3949ba59abbe56e057f20f883e
password	5f4dcc3b5aa765d61d8327deb882cf99
12345	827ccb0eea8a706c4c34a16891f84e7b
12345678	25d55ad283aa400af464c76d713c07ad
qwerty	d8578edf8458ce06fbc5bb76a58c5ca4
123456789	25f9e794323b453885f5181f1b624d0b
1234	81dc9bdb52d04dc20036dbd8313ed055
baseball	276f8db0b86edaa7fc805516c852c889
dragon	8621ffdbc5698829397d97767ac13db3
football	37b4e2d82900d5e94b8da524fbeb33c0

Rainbow Table

Secure Password Storage

The most severe attack against password-based authentication is probably when the password files or database is obtained by the attacker.

How to handle the users' passwords securely?

1. The user enters credentials such as a username and password.
2. The user's system encrypt the password and sends to the authenticating system.
3. The authenticating system decrypt the password , and calculate the hashed of the received password compares this hash to the hash stored in the password database file. If it matches, it indicates the user entered the correct password.

Salt, Hash and Store

- The passwords are vulnerable to rainbow attack if we only hash and store the password.
- To mitigate the risk of rainbow attack we need to salt the password.
- Passwording salting is the process of **adding a random string**, called salt, to the password before hashing it.
- A good salting technique would make the rainbow attack infeasible.
- What is a good password salting technique??

Password Salting

- Do not use a **fixed salt**, the attacker can still implement a rainbow table approach. Therefore a salt reuse is a bad idea.
- Do not use a **short salt**, because the attacker could build a rainbow table for every possible salt.
- Use **adaptive hashing function** like **bcrypt**, because with today computational power a hashing function such as SHA1 or SHA256 could be executed on passwords at ar ate of **100M per second**.

```
hash("LetMeIn!")                    = 2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73043362938b9824
hash("LetMeIn!" + "QxLUF1bgIAdeQX") = 9e209040c863f84a31e719795b2577523954739fe5ed3b58a75cff2127075ed1
hash("LetMeIn!" + "bv5PehSMfV11Cd") = d1d3ec2e6f20fd420d50e2642992841d8338a314b8ea157c9e18477aaef226ab
hash("LetMeIn!" + "YYLmfY6IehjZMQ") = a49670c3c18b9e079b9cfaf51634f563dc8ae3070db2c4a8544305df1b60f007
```

Token Based Authentication

- A **hardware** or **software** token allow the user to login into the system.
- The most common approach for token based authentication is **OTP token** that generates a one time password.
- One-time passwords are dynamic passwords that change every time they are used.
- There are **two types** of OTP tokens:
  - **Synchronous** Dynamic Password Tokens
  - **Asynchronous** Dynamic Password Tokens

Token-Based Authentication

- Hardware Token Limitations
  - Involves additional costs, such as the cost of the token and any replacement fees.
  - Users always need to carry the token with them
- Software Token Limitations
  - Vulnerable to software attacks
  - Less challenging to clone and forgery

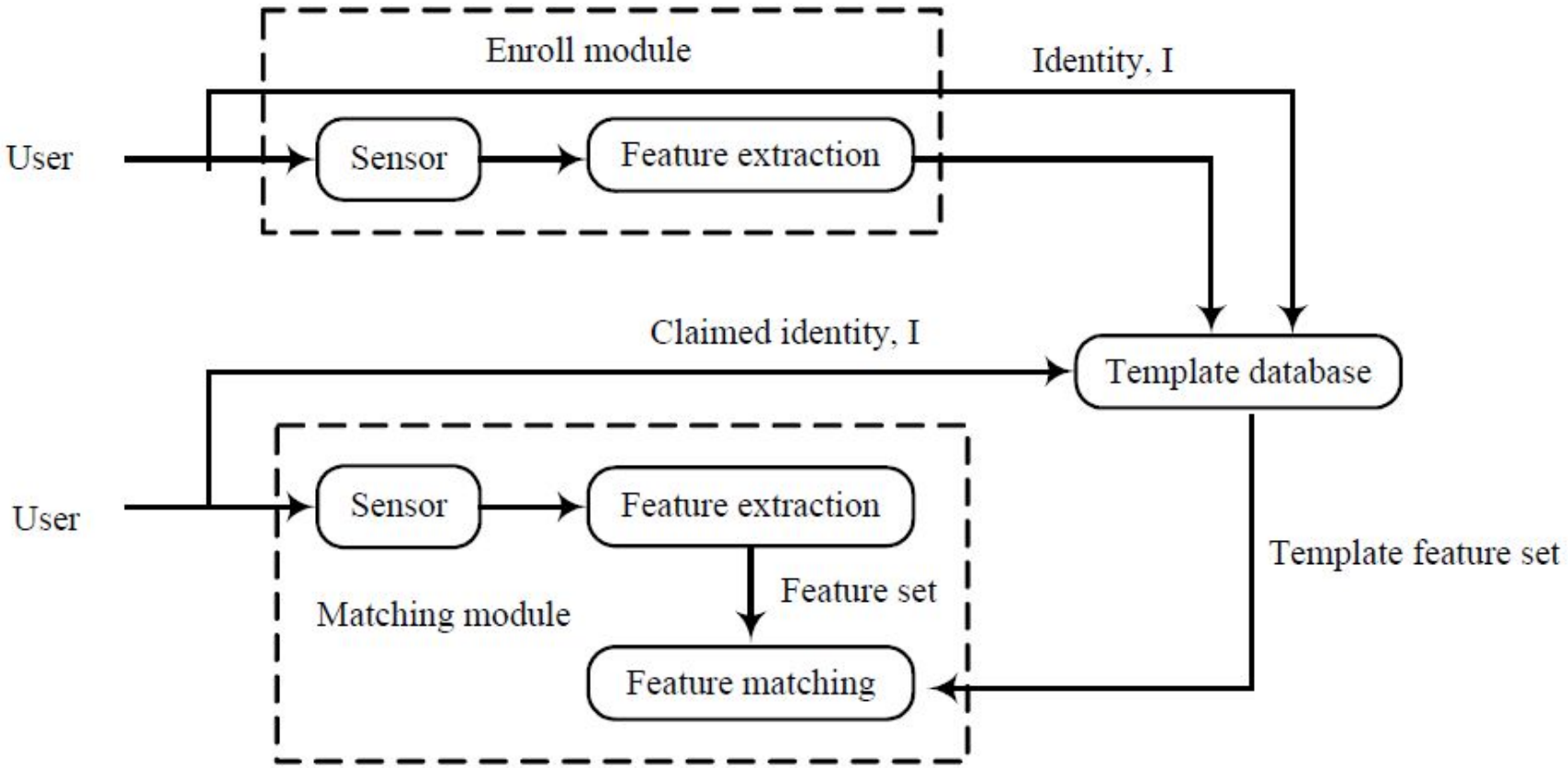
Biometric based Authentication

- Biometrics is the measurement and **statistical analysis** of people's unique physical and behavioral characteristics.
- The idea behind biometric is that every person could be uniquely identified using some physical and behavioral characteristics
- Biometric is mainly used for identification and access control, or for identifying individuals who are under surveillance.
- Examples of physical and behavioral characteristics currently used for automatic identification include **fingerprints**, **voice**, **iris**, **retina**, **hand**, **face**, **handwriting**, **keystroke**, **finger shape**, **gait**, **ear shape**.

The “Best” Biometric Characteristic

- The ideal biometric characteristic has five qualities: robustness, distinctiveness, availability, accessibility, and acceptability
  - **Robust**: unchanging on an individual over time.
  - **Distinctive**: has great variation over the population.
  - **Available**: all the individuals in the population have it
  - **Accessible**: inexpensive to measure or obtain
  - **Acceptable**: people do not object using it.

How Biometric System Work



Biometric Authentication Attacks

- Biometrics system is subject to security attacks like any other software system. There several attacks that target biometric systems, such as
- Fake Biometric Trait (attack against the sensor)
- Capture and Replay Attack
- Attack on the biometric database
- Attack on the matching algorithm (adversarial machine learning)

Why we can not secure biometric trait as we secure password?

Design Approaches

There are many design options when implementing authentication some of the most common design options are:

1. Multifactor Authentication
2. Step-up Authentication
3. Out-of-Band Authentication

Multifactor Authentication

- Multifactor authentication is any authentication using two or more factors. Two-factor authentication requires two different elements to provide authentication.
- When two or more different factors are employed, two or more different methods of attack must succeed to collect all relevant authentication elements.
- **Examples:**
  - Using ATM card (require the card and the card PIN code)
  - Using PIN code and fingerprint scan to unlock your phone
  - Using password and facial recognition to login into a machine

Step-up Authentication

- Step-up authentication requires the user to **re-authenticate** his identity after an initial authentication step. (using a different authentication factor)
- This is normally required when the user tries to access area within the system that contains sensitive information or when the user tries to perform an action that is considered a high-risk action.
- **For example**, user login into his bank account using an account ID and a password. Then, the user attempt to transfer a large amount of funds overseas, at this point the system will ask the user to reauthenticate by calling the bank, or by entering a OTP token.

Out-of-Band authentication

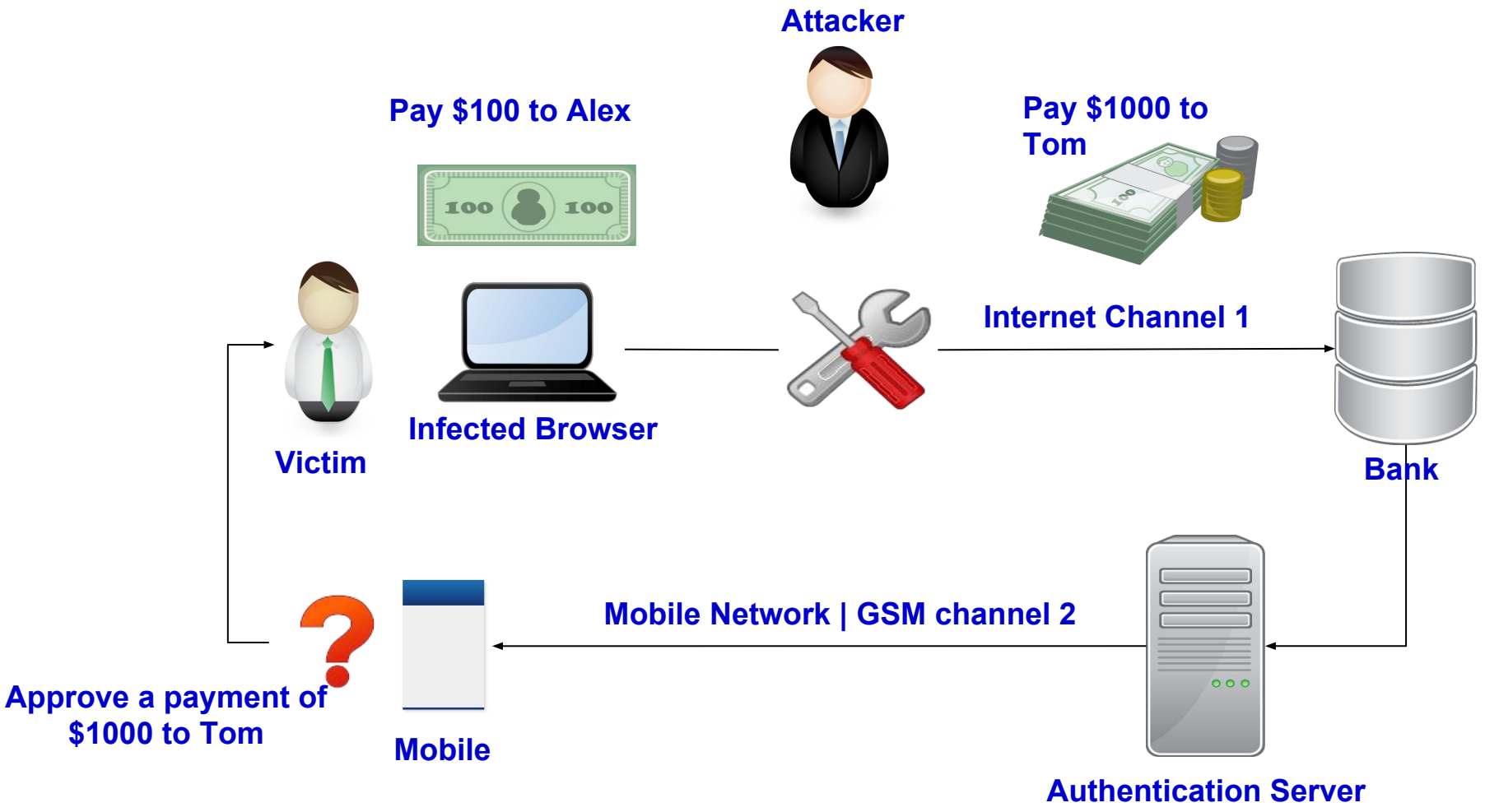
- Used in **financial institutions** and other organizations with high security requirements.
- Makes hacking an account more difficult because two separate and unconnected authentication channels would have to be compromised for an attacker to gain access.
- Defensive technique for **RAT** (**Remote Access Trojan**) and man in the browser attack, Rat in the Browser attack (RitB)
- **Rat in the Browser attack** (RitB) is a trojan that infect web browsers.

Out-of-Band authentication

How Rat in the Browser attack (RitB) works?

1. Trojan is injected into the browser.
2. The trojan becomes an invisible middleman to a web browsing session.
3. Commonly used in bank account hacking.
4. Rat in the Browser session is very hard to detect as the web browsing session looks normal and doesn't raise red flags.
5. RitB attack is altering web pages in real time, making the site look normal.

Out-of-Band authentication



Authentication Protocols

- There are many authentication protocols that provide authentication and authorization in computer networks
- Common Network Authentication Protocols
  a. OAUTH
  b. Single Sign ON (SSO)
  c. Radius
  d. Kerberos

OAUTH 2.0

What is OAUTH 2.0?

- OAuth 2 is an authorization framework that enables applications to obtain limited access to user accounts on an HTTP service

- **Context:**
  - A third party application is attempting to get access to the user's account.
  - It needs to get permission from the user before it can do so.

OAUTH 2.0 | Example

1. The user accesses a game web application.
2. The game web application asks the user to login to the game via Facebook.
3. The user logs into Facebook and is sent back to the game.
4. The game can now access the user's data in Facebook, and call functions in Facebook on behalf of the user (e.g., posting status updates).

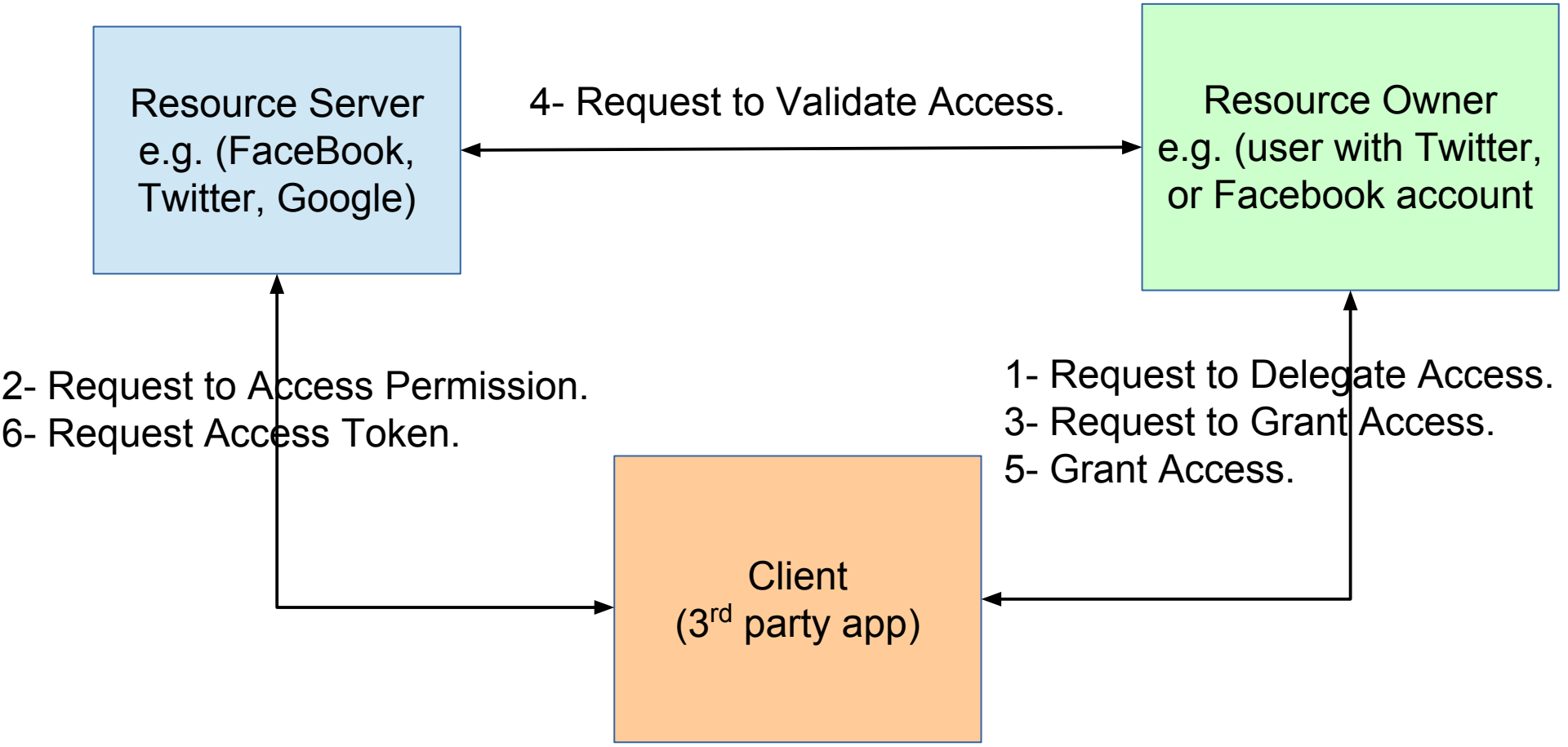
OAUTH Roles (Key Players)

- Resource Owner:
  - The user who authorizes an application to access his account
  - The application's access to the user's account is limited to the "scope" of the authorization granted (e.g. read or write access).
- Client:
  - The application that wants to access the user's account.
  - must be authorized by the user, and the authorization must be validated by the Resource Server.

OAUTH Roles (Key Players)

- Resource Server:
  - Hosts the protected user accounts or resources
- Authorization Server:
  - Verifies the identity of the user then issues access tokens to the application.

OAUTH How it works?



OAUTH How it works?

- **Step 01:** The user shows intent
  - I want to use PhotoMaker to edit my photos on Google Drive.
  - PhotoMaker “OK” Let me go ask for permission
- **Step 02:** The consumer gets permission
  - PhotoMaker: Hello Google Service provider. I have a user that would like me to edit his photo on Google Drive. Can I have a request token?”
  - Google (Service Provider): “Sure. Here’s a **token** and a **secret**.”

OAUTH How it works?

- **Step 03:** The user is redirected to the service provider
  - PhotoMaker: “OK, Alex. I’m sending you over to Google so you can approve my request. Take this token with you.”
  - Alex “OK”
  - PhotoMaker directs Alex to Google for authorization
- **Step 04:** The user gives permission
  - Alex: “Google, I’d like to authorize this request token that PhotoMaker gave me.”
  - OK, just to be sure, you want to authorize PhotoMaker to do X, Y, and Z with your Google account?”
  - Alex “Yes”
  - Google: “OK, you can go back to PhotoMaker and tell them they have permission to use their request token.”

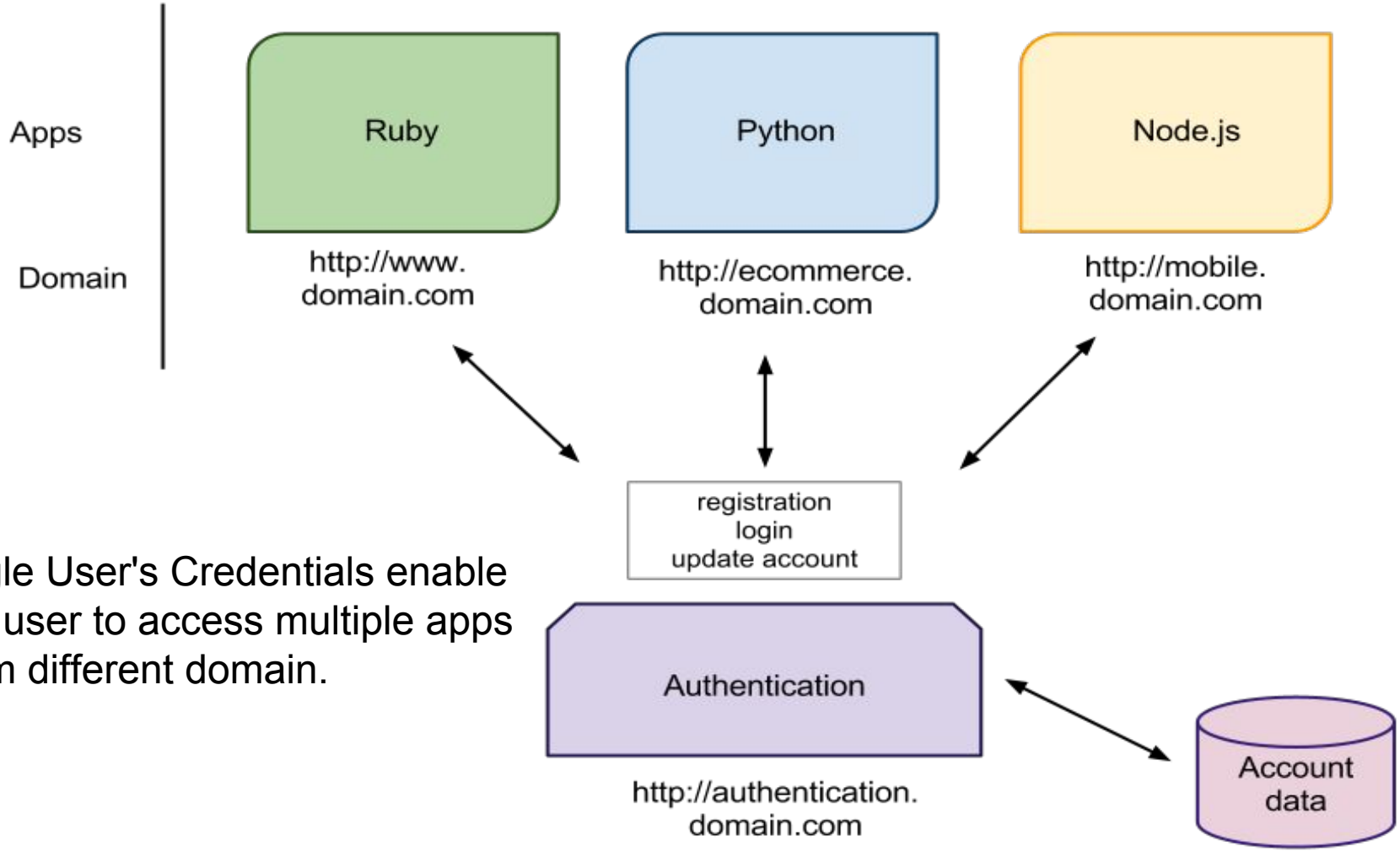
OAUTH How it works?

- **Step 05:** The consumer obtains an access token
  - PhotoMaker: “Google, can I exchange this request token for an access token?”
  - Google: “Sure. Here’s your access token and secret.”
- **Step 6:** The consumer accesses the protected resource
  - PhotoMaker: “I’d like to Access Alex's Photos uploaded in the last 10 days. Here’s my access token!”
  - Google: “Here is a list of the Photos from the last 10 days”

Single Sign ON (SSO)

- Is an authentication service that permits a user to **use one set of login credentials** to access multiple applications from different domains
- The service authenticates the end user for all the applications the user has been given rights to and eliminates further prompts when the user switches applications during the same session.
- Login Once and Access Many.
- Commonly applied in cloud environment for **Software-as-a-Service (SaaS)**

Single Sign ON (SSO)

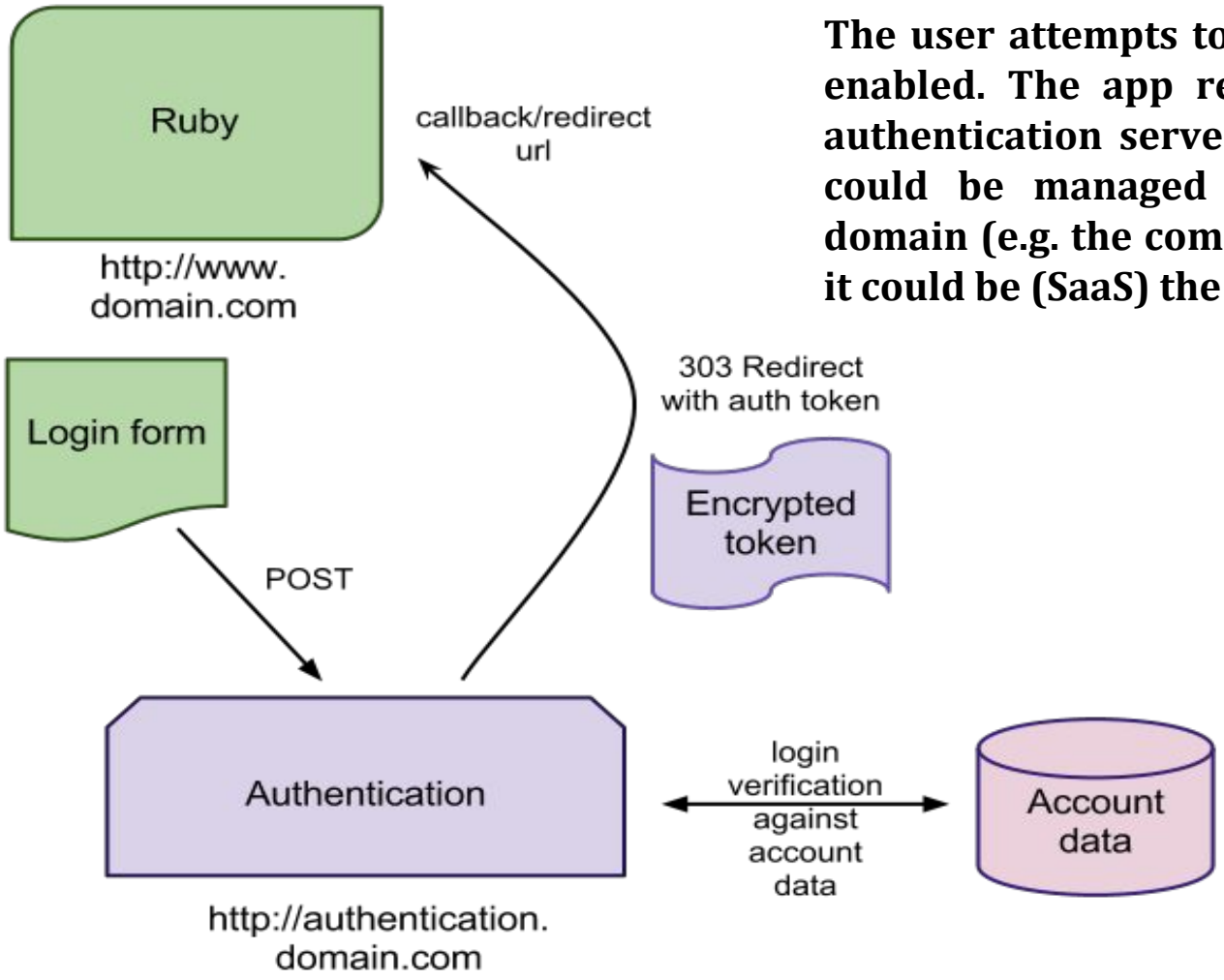


Single User's Credentials enable
The user to access multiple apps
From different domain.

Single Sign On (SSO) Key Players

- **User | Client:**
  - a user or an application that needs to access multiple resources to accomplish one or more business goal.
- **Identity Provider (IdP)**
  - IdP is the system that asserts information about a subject
- **Service Provider (SeP)**
  - SeP is the system that relies on the information supplied to it by the identity provider.

SSO | How it works?



The user attempts to access an app where SSO is enabled. The app redirect the user to the SSO authentication server. The authentication server could be managed and hosted by the user's domain (e.g. the company the user work for it) or it could be (SaaS) the user's company uses.

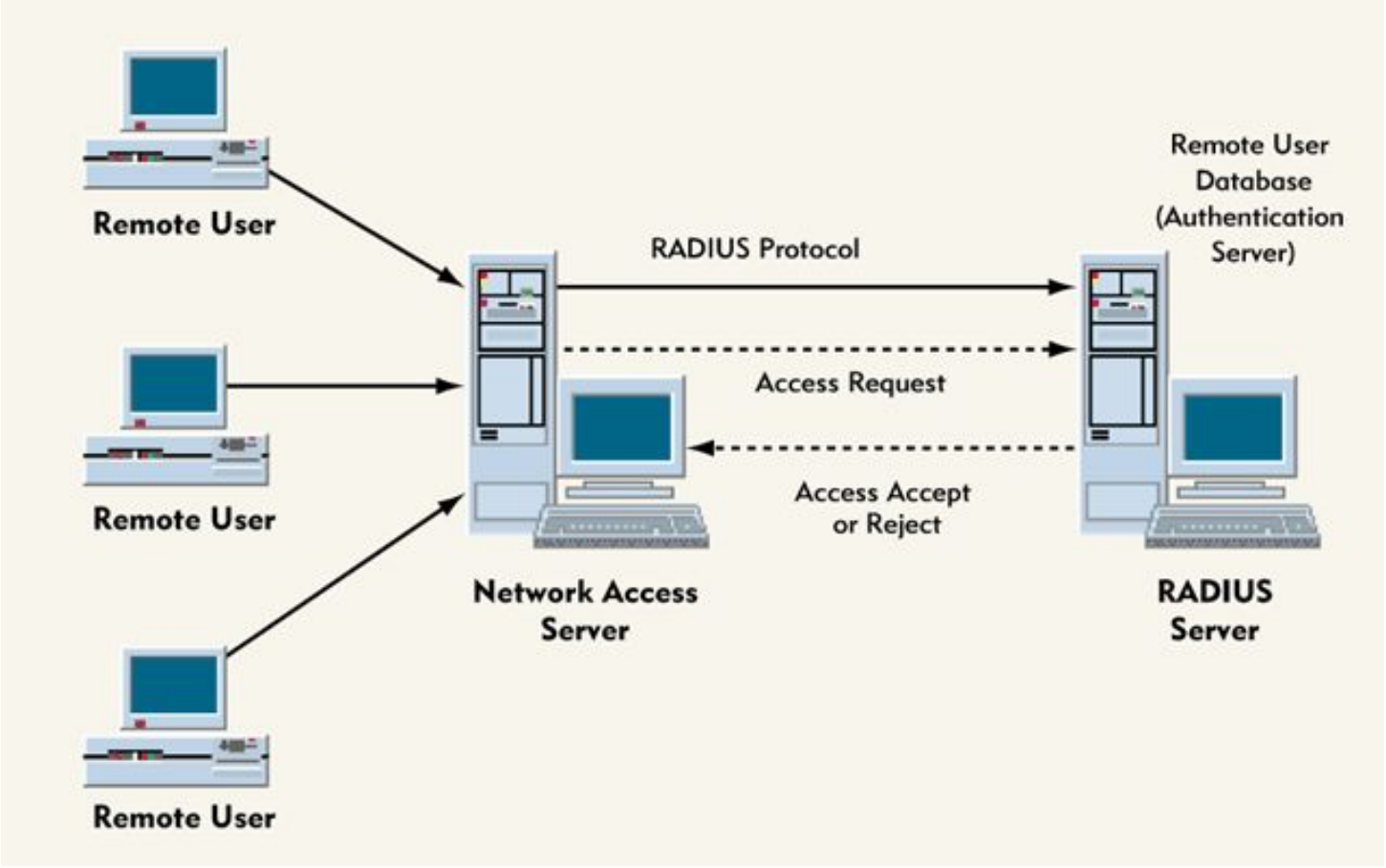
SSO | How it works?

1. The user attempts to access the service provided.
2. The service provider redirects the users to the identity provider.
3. The user claims identity and provides suitable credentials.
4. The identity provider verifies the user's credentials and create an access token and redirect the users back to the service provider.
5. The service provider validates the access token.

RADIUS Protocol

- RADIUS is an **AAA protocol** that manages network access. The AAA stands for authentication, authorization, and accounting.
- RADIUS is a **client/server protocol** that runs in the application layer, and can use either TCP or UDP
- RADIUS uses the User Datagram Protocol (UDP) by default and encrypts only the exchange of the password.
- It doesn't encrypt the entire session, but additional protocols can be used to encrypt the data session.

RADIUS Protocol



Kerberos

- Ticket authentication is a mechanism that employs a third-party entity to prove identification and provide authentication.
- Kerberos offers a **single sign-on** solution for users and protects login credentials.
- Kerberos uses **symmetric key encryption** and hashing for securing the authentication server and maintains its integrity.
- In Kerberos, the user's credential is not shared with any of the resource providers.
- Kerberos is a **centralized authentication** system that is commonly used **over LAN**

Questions