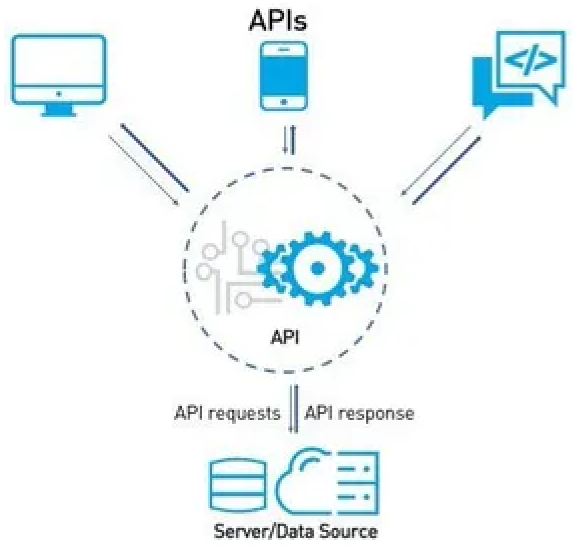
Application Programming Interface (API)

What is an API?

Application Programming Interface (API) can be defined as a set of instructions or rules that allows two systems (which include software applications) to interact with each other. Basically, API will act as an medium in between the application, ie, in between the frontend and backend.

Example:

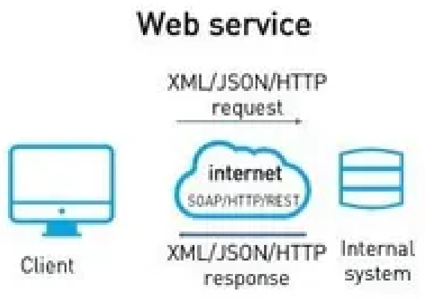
One common API that we are using can be **Social Media APIs** which includes the facebook APIs, Instagram APIs and manymore. In the case of the facebook,the app allows users to log in with their Facebook account using Facebook's API and post their photos directly to Facebook.



Submitted by: Nanda Krishnan V , Msc Cybersecurity

How APIs make apps or websites more useful

We all know that API will connect the software applications, in the case of the websites or applications we can add more features to the software with the help of the API, as we have different APIs available, to make them unique and powerful.
To be more specific, if you use a map app to find a nearby restaurant, the app might use several APIs to display maps, show restaurant listings, and even provide reviews.



Google Pay - API Usage

Payment Gateway API:

- **Purpose**: To securely process transactions between banks, credit cards, and digital wallets.
- **How It Helps**: When you send money, Google Pay uses a payment gateway API to connect with your bank or card provider, securely transferring funds without exposing sensitive details like your bank account number. This API also handles refunds and payment verifications.

Banking API:

- **Purpose**: To connect with different banks for checking balances, making transactions, and receiving updates.
- **How It Helps**: Google Pay connects with your bank through a secure banking API to check your account balance in real time. This API enables features like sending money directly from your bank account, getting notified of deposits, and managing account information.

QR Code API:

- **Purpose**: To generate and read QR codes for payments.
- **How It Helps**: Google Pay can generate QR codes for your account so others can scan it to pay you directly. When you want to pay a merchant, the QR Code API allows the app to scan the merchant's code quickly and complete the payment securely.

Authentication API (e.g., OAuth or Google Sign-In):

- **Purpose**: To verify the user's identity before performing any action.
- **How It Helps**: Google Pay uses authentication APIs to confirm your identity, whether through a fingerprint, Face ID, or PIN. This ensures that only you can authorize payments and access your financial information.

Notification API:

- **Purpose**: To send real-time alerts and updates.
- **How It Helps**: After you make a payment or receive money, Google Pay uses a notification API to instantly send you alerts, letting you know that the transaction went through. This is helpful for keeping track of spending and making sure payments are successful.

Security - API

As we all know security is very important in the modern world, everything we do must be with enough care, otherwise our personal informationcould be stealed by the unknown users. One common technique that we use to transfer data through API is that ,Tokens.

API Tokens

Tokens in the API act as the digital key ie, if a user is logging in with his credentials, they are given some unique token. These token will be consisting of their session and permissions.This token is then sent along with every API request to verify the user's identity and access level.

Advantages of Token	Disadvantages of Token
● Enhanced Security ● Cross platform compatibility ● User Convinence ● Access controll	● Security Vulnerabilities if Exposed ● Increased server load ● Management complexity ● Token expiration

Challenges faced by API

1. **Security**:
    a. Security is an integral part of the system, we must take enough measure to protect the CIA triad. As part of a continuous improvement process, there should be a place for security considerations and tasks which cover the minimum threats, and industry standards like OWASP's Top 10 Most Critical Web Application Security Risks can be applied.
2. **Authentication & Authorization:**
    a. With APIs connecting multiple systems, managing who has access to what becomes complex, especially when applications grow and interact with many third-party services. Ensuring that every API request is properly authenticated, managing permissions, and keeping access tokens secure are all critical but challenging tasks.
3. **Preventing MITM Attack:**
    a. When data travels between clients and servers, it is vulnerable to interception by attackers if not properly encrypted. APIs, especially those handling sensitive information, need strong encryption and secure protocols to keep data safe in transit. These can be occurred in the following scenarios:
        i. **Lack of HTTPS/SSL Encryption:**
        ii. **Weak Encryption Algorithms:**
        iii. **Insufficient Authentication of API Clients:**

Explain why it could be risky if an API allowed access to personal data without proper permission. Give a simple example of what could go wrong.

We all know that API is a set of instruction and act as a middlemen between frontend and backend. So if we dont have a proper security measure in API integration, these could be easly stealed by the unknown person. These includes our credentials including the personal datas. So this may cause

- privacy breaches
- identity theft
- financial losses.

Example

Consider the example **Google Pay**,

Use Case - Unauthorized Transaction History Access

1. **Scenario** - An external service uses this API without strict permission controls so that anyone using the service can access the credentials and information including the transaction details and many more of the Google Pay users.
2. **Risk** - They external service or the unauthorized access could expose the user credentials which include how many amount have been spent by the user, contact information and more personal details. The risk factor is that the unauthorized users can exploit the users data , can attempt financial fraud by impersonating the user.
3. **Consequences** -
    a. **Privacy Violation -** The external service could see the personal information about the users and these may violate the privacy of the original user.
    b. **Financial Risk -** After knowing how the user is spending the money , the unauthorized user can easily get the pattern. They can lead to the planned or targeted attack to the user. Attackers may try to impersonate the user or trick them into providing even more sensitive data.
4. **Legal effects -** Companies like Google Pay must comply with regulations like GDPR (General Data Protection Regulation) , Failure to protect personal data can lead to hefty fines and legal penalties.