API Keys and their security

After having so much discussion about API Keys and OAuth Credential, in this article, we will focus on which one is better from security and privacy point of view. But before proceeding further let's first recall what does *Authentication* and *Authorization* means?

Authentication means to validate your identity by using credentials like User Id, Username, Password. It is concerned with determining whether you are what you say you are using credentials presented by you. A most general example is Login Form that we encounter almost on all the websites.

Authorization, on the other hand, aims at determining whether you are authorized to access the resources or not. After your identity is authenticated by the system, the next step comes to give you access to the resources based on your identity.

For example, If we take example of a Laptop. Suppose there are 2 users in a Laptop, One is Admin and Another one is ABC. ABC is not authorized to delete a file. ABC can only create a file, read a file, modify a file.
If a user has entered credential of Admin. The System will first match the credentials with the one in database to determine which user is this or for a case whether it is a legit user or not. After authenticating the credentials, System gets to know that it is Admin, so it will see the set of permissions for the Admin and grant only the privileges that are meant for Admin. Like Admin can delete a file also. But if it is user ABC, then it will only be able to create a file, read a file and modify a file, deletion will not be allowed. This is known is Authorization.

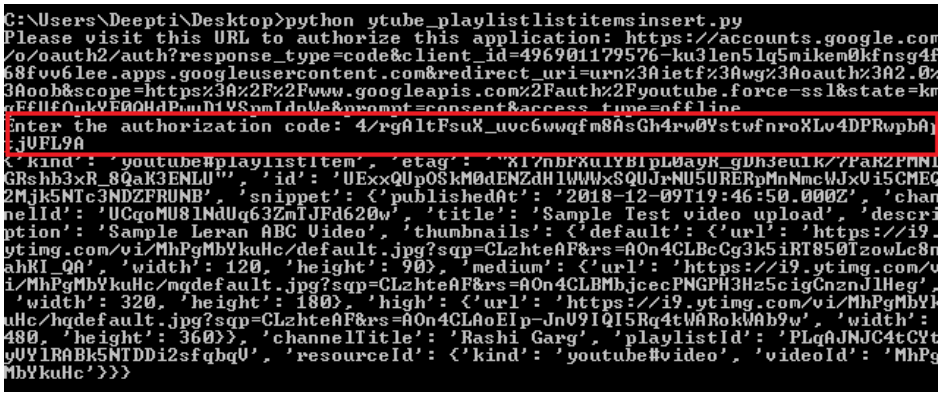
After going through these differences we can easily understand the difference between API Key and OAuth. There are ***three types*** of security mechanism for an API –

1. **HTTP Basic Authentication:** In this mechanism HTTP User Agent provides a Username and Password. Since this method depends only on HTTP Header and entire authentication data is transmitted on insecure lines, Thus, it is prone to Man-In-The-Middle Attack where a user can simply capture the HTTP Header and login using copy-cat Header and a malicious packet. Due to enforced SSL, this scheme is very slow. HTTP Basic Authentication can be used in situations like Internal Network where speed is not an issue.
2. **API Keys:** API Keys came into picture due to slow speed and highly vulnerable nature of HTTP Basic Authentication. API Key is the code that is assigned to the user upon API Registration or Account Creation. API Keys are generated using the specific set of rules laid down by the authorities involved in API Development. This piece of code is required to pass whenever the entity (Developer, user or a specific program) makes a call to the API. Despite easy usage and fast speed, they are highly insecure.

   Question still remains, WHY ??
   The problem is, API Key is a method of Authentication, not Authorization. They are like username and password, Thus providing entry into the system. In general, API Keys are placed at the following places: Authorization Header, Basic Auth, Body Data, Custom Header, Query String.
   Anytime while making a request we need to send an API Key by placing it in any of the above places. Thus if at any point of time network is compromised, then the entire network gets exposed and API Key can be easily extracted. Once an API Key is stolen, it can be used for indefinite amount of time. Unless and until the project owner revokes t API Key and generate a new one.

3. **OAuth:** OAuth is not only a method of Authentication or Authorization, but it's also a mixture of both the methods. Whenever an API is called using OAuth credential, user logs into the system, generating a token. Remember this token is active for one session only after which user has to generate a new token by logging again into the system. After submitting this token to the Server, User is authorized to the roles based on the credentials.
   Now if take an example from Youtube Data API, First the user will authenticate itself by submitting credentials like username and password and then Submit the generated token to the server and authorize itself for the role.
   Images below shows how OAuth Credential works:

   After successful login, a token is generated. This token when presented to the server decides tha appropriate rights for the calling user and generates the results accordingly. The highlighted portion in the image represent the Authorization Token that was generated.



References:

1. https://nordicapis.com/3-common-methods-api-authentication-explained/
2. https://zapier.com/engineering/apikey-oauth-jwt/
3. https://apifriends.com/api-security/api-keys-oauth/
4. https://cloud.google.com/endpoints/docs/openapi/when-why-api-key
5. https://nordicapis.com/why-api-keys-are-not-enough/

Recommended Posts:

Difference between Cyber Security and Information Security
Need Of Information Security
System Security
What is Information Security?
Is SSL enough for Cloud Security?
Security Threats to IoT Devices
Meltdown Security Vulnerability
Spectre Security Vulnerability
Threats to Information Security
Top 5 Information Security Breaches
Security Management System
Information Security | Confidentiality
Information Security | Integrity
Information System and Security
Digital Forensics in Information Security

RashiGarg
Check out this Author's contributed articles.

If you like GeeksforGeeks and would like to contribute, you can also write an article using contribute.geeksforgeeks.org or mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please Improve this article if you find anything incorrect by clicking on the "Improve Article" button below.

Article Tags : GBlog Information-Security

Be the First to upvote.

0

To-do Done

No votes yet.

Feedback/ Suggest Improvement | Add Notes | Improve Article

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments