Authenticating to a REST API

4.5.1

REST API Authentication

Merriam-Webster defines authentication as “...an act, process, or method of showing something (such as an identity) to be real, true, or genuine.” For security reasons, most REST APIs require authentication so that random users cannot create, update or delete information incorrectly or maliciously, or access information that shouldn't be public. Without authentication, a REST API permits anyone to access the features and system services that have been exposed through the interface. Requiring authentication is the same concept as requiring a username/password credential to access the administration page of an application.

Some APIs don't require authentication. These APIs are usually read-only and don't contain critical or confidential information.

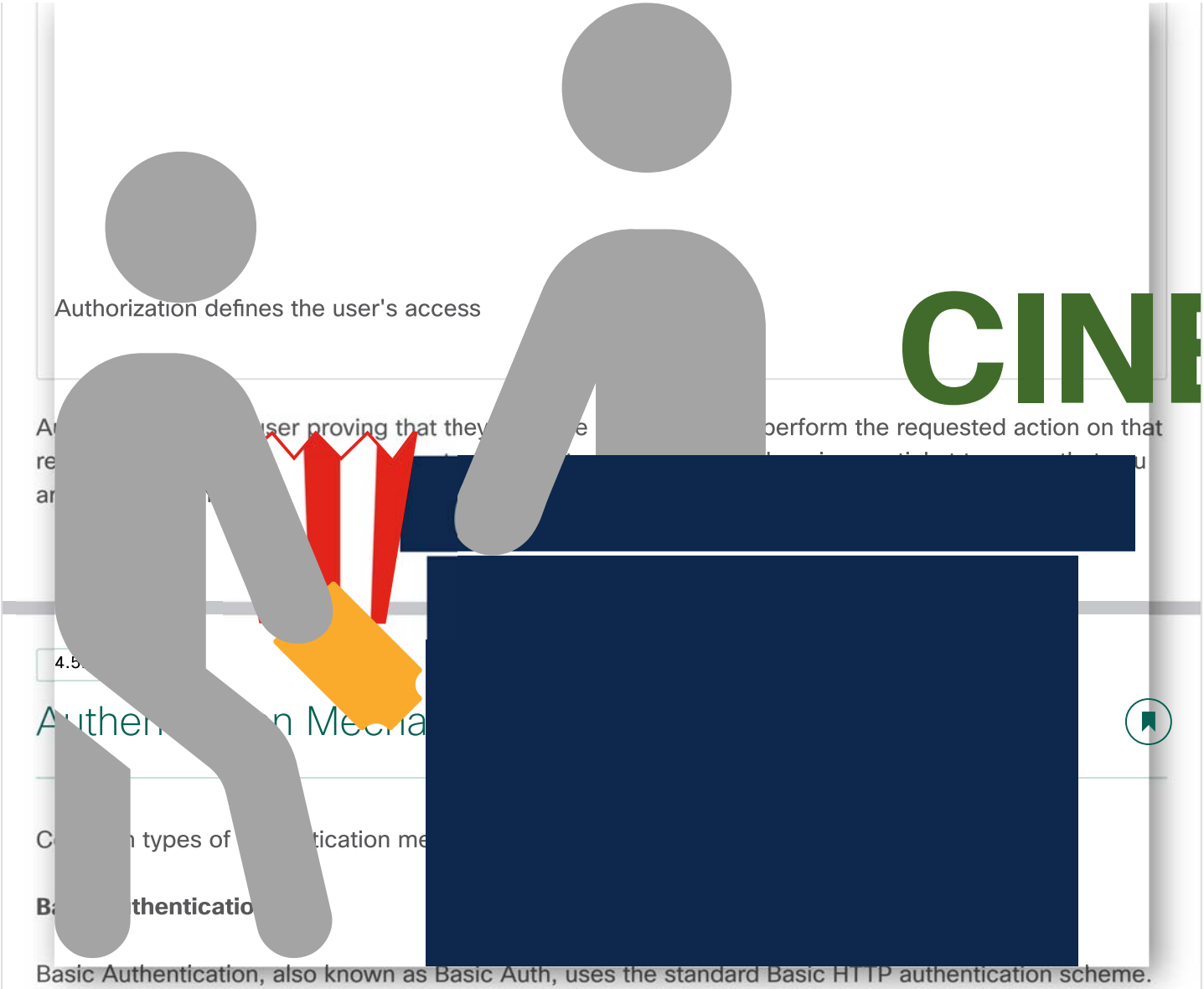
It's important to understand the difference between authentication and authorization when working with REST APIs, because the two terms are often used interchangeably, or incorrectly. Knowing the difference will help you troubleshoot any issues regarding the security surrounding your REST API request.

Authentication

Authentication proves the user's identity

A verifying th ay
th en you go to
id etrics to prove you are the person you claim to be.

Authorization



Basic Authentication

Basic Authentication, also known as Basic Auth, uses the standard Basic HTTP authentication scheme. Basic Auth transmits credentials as username/password pairs separated with a colon (`:`) and encoded using Base64.

In a REST API request, the Basic Auth information will be provided in the header:

```
Authorization: Basic <username>:<password>
```

Basic Auth is the simplest authentication mechanism. It is extremely insecure unless it is paired with requests using HTTPS rather than HTTP. Although the credentials are encoded, they are not encrypted. It is simple to decode the credentials and get the username/password pair.

Bearer authentication

Bearer Authentication, also known as Token Authentication, uses the standard Bearer HTTP authentication scheme. It is more secure than Basic Authentication and is typically used with OAuth (discussed later) and Single Sign-On (SSO). Bearer Authentication uses a bearer token, which is a string generated by an authentication server such as an Identity Service (IdS).

In a REST API request, the Bearer Auth information will be provided in the header:

```
Authorization: Bearer <bearer token>
```

Just like Basic Authentication, Bearer Authentication should be used with HTTPS.

API key

An API key, also referred to as an API Token, is a unique alphanumeric string generated by the server and assigned to a user. To obtain a unique API key, the user typically logs into a portal using their credentials. This key is usually assigned one time and will not be regenerated. All REST API requests for this user must provide the assigned API key as the form of authentication.

Just as with the other types of authentication, API keys are only secure when used with HTTPS.

API keys are intended to be an authentication mechanism, but are commonly misused as an authorization mechanism.

The two types of API keys are public and private.

A public API key can be shared and enables that user to access a subset of data and APIs. Do not share a private key, because it is similar to your username and password. Most API keys do not expire, and unless the key can be revoked or regenerated, if it is distributed or compromised, anyone with that key can indefinitely access the system as you.

A REST API request can provide an API key in a few different ways:

- Query string: Only recommended for public API keys
- Header: Uses the Authorization key or a custom key `Authorization: <API Key>` or `Authorization: APIkey <API Key>` or `APIkey: <API Key>`
- Body data: Uses a unique key as the identifier `Content-Type: application/json`
- Cookie: Uses a unique key as the identifier `Cookie: API_KEY=<API Key>`

4.5.4

Authorization Mechanisms

Authorization mechanisms

Open Authorization, also known as OAuth, combines authentication with authorization. OAuth was developed as a solution to insecure authentication mechanisms. With increased security compared to the other options, it is usually the recommended form of authentication/authorization for REST APIs.

There are two versions of OAuth, simply named OAuth 1.0 and OAuth 2.0. Most of today's REST APIs implement OAuth 2.0. Note that OAuth 2.0 is not backwards compatible.

Defined in the OAuth 2.0 Authorization Framework, “OAuth 2.0 authorization framework enables a third-party application to obtain limited access to an HTTP service, either on behalf of a resource owner by orchestrating an approval interaction between the resource owner and the HTTP service, or by allowing the third-party application to obtain access on its own behalf.”

Essentially, OAuth 2.0 enables pre-registered applications to get authorization to perform REST API requests on a user's behalf without the user needing to share their credentials with the application itself. OAuth lets the user provide credentials directly to the authorization server, typically an Identity Provider (IdP) or an Identity Service (IdS), to obtain an access token that can be shared with the application.

This process of obtaining the token is called a flow. The application then uses this token in the REST API as a Bearer Authentication. The web service for the REST API then checks the Authorization server to make sure that the token is valid, and that the requester is authorized to perform the request.

4.5.5

Lab - Explore REST APIs with API Simulator and Postman

In this lab, you will learn how to use the School Library API simulator to make API calls to list, add and delete books. Later, you will use Postman to make these same API calls. Postman is a useful tool when an API developer web site is not available while providing the ability to easily save, organize and reuse APIs.

You will complete the following objectives:

- Part 1: Launch the DEVASC VM
- Part 2: Explore API Documentation Using the API Simulator
- Part 3: Use Postman to Make API Calls to the API Simulator
- Part 4: Use Python to Add 100 Books to the API Simulator

Explore REST APIs with API Simulator and Postman

4.4 Introduction to REST APIs

4.6 API Rate Limits