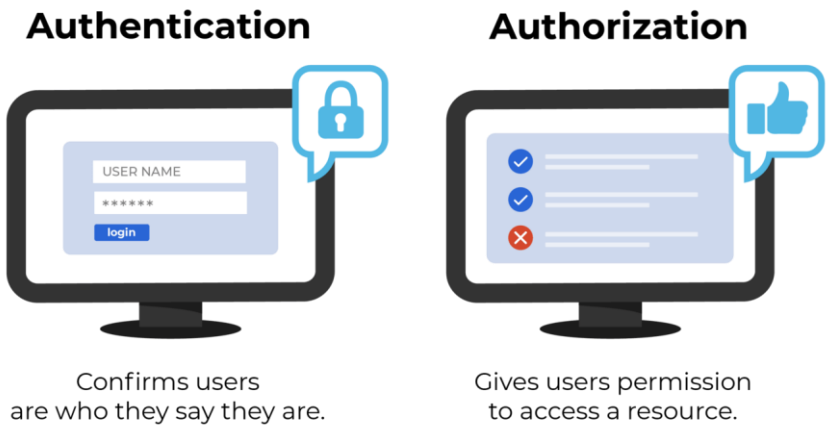
Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Istruzione e del Merito
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA
ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE

Autenticazione e Autorizzazioni in API RESTful

Sicurezza nelle API RESTful



Nella progettazione di API RESTful, l'autenticazione e le autorizzazioni sono due aspetti fondamentali per garantire la sicurezza e il controllo dell'accesso alle risorse. In questa lezione, esploreremo i concetti di autenticazione e autorizzazioni in relazione alle API RESTful e discuteremo le diverse strategie e meccanismi utilizzati per implementarli.

Autenticazione

L'autenticazione è il processo di verifica dell'identità di un utente o applicazione.

Metodi comuni di autenticazione:

- 1. HTTP Basic Authentication
- 2. Token-based Authentication
- 3. OAuth 2.0
- 4. OpenID Connect

HTTP Basic Authentication

- Un metodo semplice per autenticare, ma non sicuro su HTTP non criptato.
- Esempio in Java:

```
HttpURLConnection connection = (HttpURLConnection) new
URL("https://api.example.com/resource").openConnection();
String basicAuth = "Basic " +
Base64.getEncoder().encodeToString("username:password".getBytes());
connection.setRequestProperty("Authorization", basicAuth);
```

Token-based Authentication (JWT)

Dopo l'autenticazione, il server genera un token (es. JWT) per le richieste successive.

Esempio di generazione JWT in Java:

```
String jwtToken = Jwts.builder()
        .setSubject("user")
        .setIssuedAt(new Date())
        .setExpiration(new Date(System.currentTimeMillis() + 86400000))
        .signWith(SignatureAlgorithm.HS256, "secretKey")
        .compact();
```

OAuth 2.0

Framework di autorizzazione che consente a un'applicazione di eseguire azioni per conto di un utente.

Esempio di richiesta token OAuth 2.0 in Java:

```
String tokenEndpoint = "https://auth.example.com/oauth/token";
String urlParameters =
"grant_type=authorization_code&code=authCode&redirect_uri=https://yourapp.com/c
allback";
HttpURLConnection connection = (HttpURLConnection) new
URL(tokenEndpoint).openConnection();
connection.setRequestMethod("POST");
connection.setRequestProperty("Content-Type", "application/x-www-form-
urlencoded");
```

Autorizzazioni

Autorizzazioni

Le autorizzazioni determinano ciò che un utente autenticato può fare nell'applicazione o API.

Strategie comuni:

- 1. Controllo basato sui ruoli (RBAC)
- 2. Controllo basato sulle risorse
- 3. Controllo basato su attributi (ABAC)

Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Istruzione e del Merito
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA
ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE

Controllo basato sui ruoli

Gli utenti vengono associati a uno o più ruoli predefiniti (es. amministratore, utente). Le azioni consentite dipendono dal ruolo assegnato.

Esempio in Java:

```java
String userRole = getUserRole(userId);

if (userRole.equals("ADMIN")) {
    // Permetti tutte le operazioni
} else if (userRole.equals("USER")) {
    // Permetti solo operazioni di lettura
}
```

Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Istruzione e del Merito
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA
ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE

Controllo basato sulle risorse

Gli utenti possono accedere solo alle risorse di cui sono proprietari o alle quali hanno un permesso specifico. Ad esempio, solo il proprietario di un account può modificarlo.

```
if (resourceOwnerId.equals(authenticatedUserId)) {
    // Permetti l'accesso
} else {
    throw new UnauthorizedException("Accesso negato alla risorsa");
}
```

Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Istruzione e del Merito
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA
ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE

Controllo basato su attributi

Gli accessi sono determinati in base ad attributi degli utenti e delle risorse. Ad esempio, solo gli utenti del dipartimento "Marketing" possono accedere a specifiche risorse.

```
String userDepartment = getUserDepartment(userId);

if (userDepartment.equals("Marketing")) {
    // Consenti l'accesso alla risorsa
} else {
    throw new UnauthorizedException("Accesso negato alla risorsa");
}
```

Considerazioni sulla Sicurezza(1)

Protezione delle credenziali:

- Utilizza sempre HTTPS per garantire la sicurezza delle credenziali in transito.
- Non memorizzare mai le password in chiaro, ma utilizza algoritmi di hash sicuri (es. bcrypt, PBKDF2).

Considerazioni sulla Sicurezza(2)

Validazione e sanitizzazione delle richieste:

- Valida sempre i dati in ingresso per prevenire attacchi come SQL injection o XSS.

Considerazioni sulla Sicurezza(3)

Prevenzione contro attacchi comuni:

- Limita il numero di tentativi di accesso per prevenire attacchi di forza bruta.
- Utilizza meccanismi di protezione come CSRF e XSS.

Conclusione

- L'autenticazione e le autorizzazioni sono fondamentali per garantire la sicurezza e il controllo dell'accesso alle risorse delle API RESTful. La scelta del metodo di autenticazione e delle strategie di autorizzazione dipende dalle esigenze specifiche dell'applicazione. È importante considerare anche le pratiche di sicurezza aggiuntive per proteggere le credenziali e prevenire attacchi comuni.