



UNICA

UNIVERSITÀ  
DEGLI STUDI  
DI CAGLIARI



sAifer Lab

Joint lab on Safety and Security of AI

# Autenticazione e Autorizzazione

Maura Pintor

maura.pintor@unica.it

# Introduzione

Nella progettazione di sistemi web, sicurezza e controllo degli accessi sono fondamentali per proteggere dati e risorse

Due concetti chiave sono:

- **Autenticazione** - verifica dell'identità di un utente
- **Autorizzazione** - definizione dei permessi di un utente autenticato

Questi sono da gestire nel contesto dei nostri servizi web per fare in modo che solo gli utenti autorizzati accedano alle risorse messe a disposizione (per esempio, a dati confidenziali)

# Autenticazione

L'autenticazione assicura che chi accede a un sistema sia chi dichiara di essere. Può avvenire con diversi metodi:

- Basata su credenziali, ovvero username e password
- Autenticazione a più fattori (MFA), quindi combinazione di *almeno due* metodi (es. password + codice One Time Password - OTP)
- Autenticazione basata su token, JWT (JSON Web Token) o sessioni
- OAuth e OpenID Connect, autenticazione federata tramite provider esterni (Google, Facebook, ecc.)

# Flusso tipico dell'autenticazione

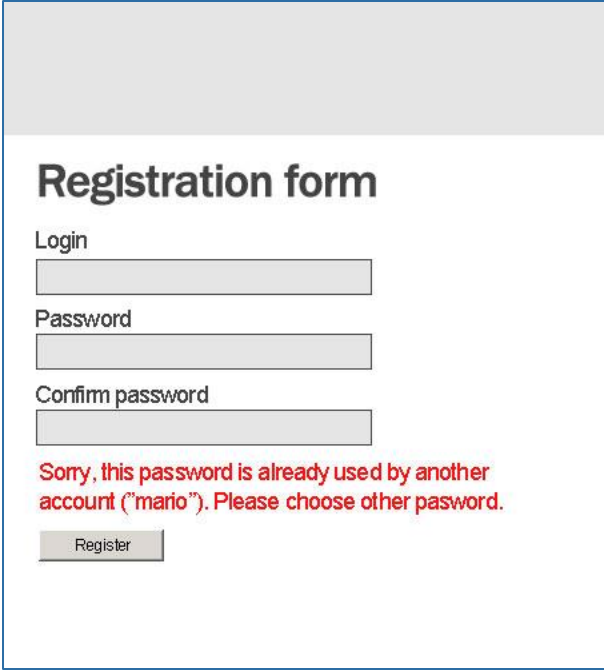
1. L'utente inserisce le credenziali
2. Il sistema verifica le credenziali e, se valide, genera un token o una sessione
3. Il token/sessione viene usato per autenticare le richieste successive



# Autenticazione basata su credenziali

Username e password sono il metodo più comune, ma soggetto a vulnerabilità come attacchi brute-force e phishing

Best practice: uso di **hashing** con algoritmi sicuri per la cifratura, l'archiviazione e la verifica e obbligo di password complesse richieste agli utenti



A mockup of a web registration form. It features a light gray header bar. Below it, the title "Registration form" is displayed in a bold, dark font. The form contains three input fields: "Login", "Password", and "Confirm password", each with a corresponding label to its left. Below the "Confirm password" field, a red error message is shown: "Sorry, this password is already used by another account ('mario'). Please choose other pasword." (Note the typo 'pasword' in the original image). At the bottom of the form is a "Register" button.

**Registration form**

Login

Password

Confirm password

Sorry, this password is already used by another account ("mario"). Please choose other pasword.

Register

# Multi Factor Authentication (MFA)

Per essere autenticato, l'utente può superare diverse prove (due per la two-factor authentication, 2FA). L'idea è che solo se l'utente è chi dice di essere può superare i vari livelli di autenticazione

Si scelgono due o più metodi distinti (possibilmente di tipo diverso) tra:

- Qualcosa che l'utente sa (password, PIN)
- Qualcosa che l'utente ha (OTP via SMS, email o app come Google Authenticator)
- Qualcosa che l'utente è (biometria: impronte digitali, riconoscimento facciale)

Vantaggi: aumenta la sicurezza riducendo il rischio di accessi non autorizzati anche in caso di credenziali compromesse



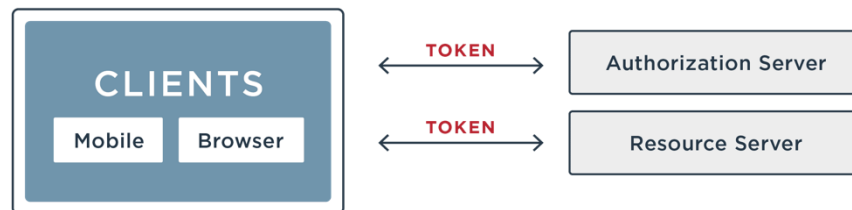
# Autorizzazione basata su token

Gli utenti si autenticano, e ricevono un token di accesso univoco che useranno per le successive autenticazioni (di solito con una scadenza)

- Session Token, generato dal server e memorizzato lato client (cookie o local storage)
- JSON Web Token (JWT), token firmati digitalmente che contengono informazioni sull'utente e sui suoi permessi, evitando la necessità di sessioni lato server

Vantaggi: scalabilità, facilità di utilizzo nelle API RESTful

Rischi: la gestione errata dei token (es. memorizzazione in localStorage senza protezioni) può esporre il sistema a attacchi di cross-side scripting (XSS)



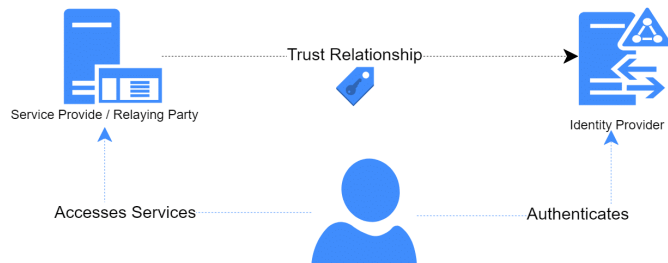
# Autenticazione federata

OAuth 2.0 e OpenID Connect: permettono agli utenti di autenticarsi tramite **provider** esterni come Google, Facebook, Microsoft

SSO (Single Sign-On): un'unica autenticazione permette l'accesso a più servizi senza dover reinserire le credenziali

Vantaggi: riduzione del numero di password da gestire, maggiore sicurezza e praticità

Rischi: se l'account del provider viene compromesso, tutti i servizi collegati sono a rischio





# Autorizzazione

L'autorizzazione determina quali azioni un utente autenticato può eseguire

Modelli di autorizzazione:

- Role-Based Access Control (RBAC), gli utenti hanno ruoli con permessi specifici
- Attribute-Based Access Control (ABAC), i permessi dipendono da attributi dell'utente e del contesto
- Access Control List (ACL), liste di permessi assegnate a utenti o gruppi su specifiche risorse

# Esempio di flusso di autorizzazione

1. Un utente autenticato richiede l'accesso a una risorsa
2. Il sistema verifica se l'utente ha i permessi necessari
3. Se i permessi sono validi, l'accesso viene concesso; altrimenti, viene negato

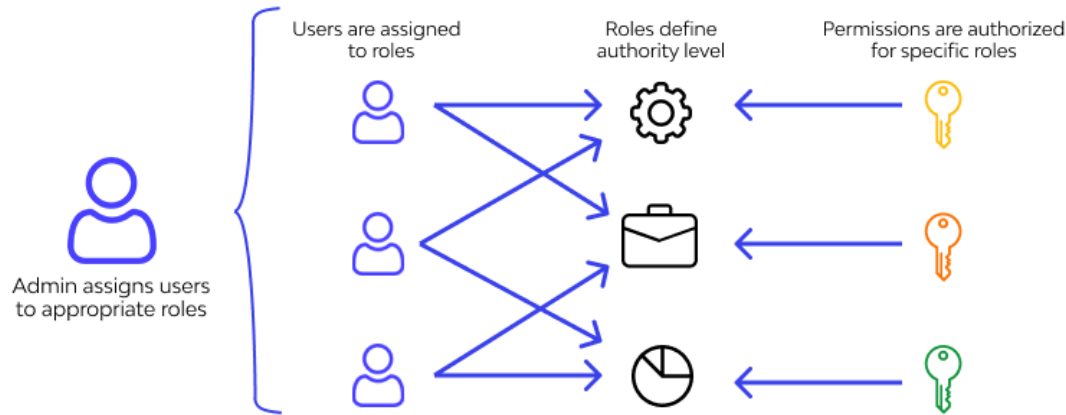
# Role-Based Access Control (RBAC)

Gli utenti sono assegnati a ruoli specifici (es. amministratore, moderatore, utente standard)

I ruoli determinano i permessi per l'accesso alle risorse

Facilita la gestione centralizzata dei permessi, ma può essere rigido in contesti dinamici

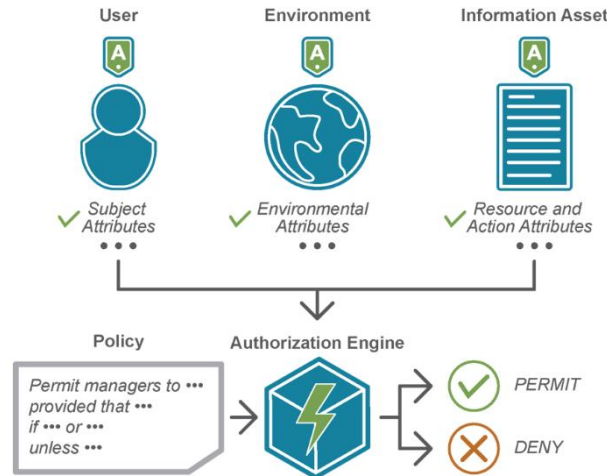
- per esempio per dare temporaneamente accesso a una singola risorsa a un solo utente



# Attribute-Based Access Control (ABAC)

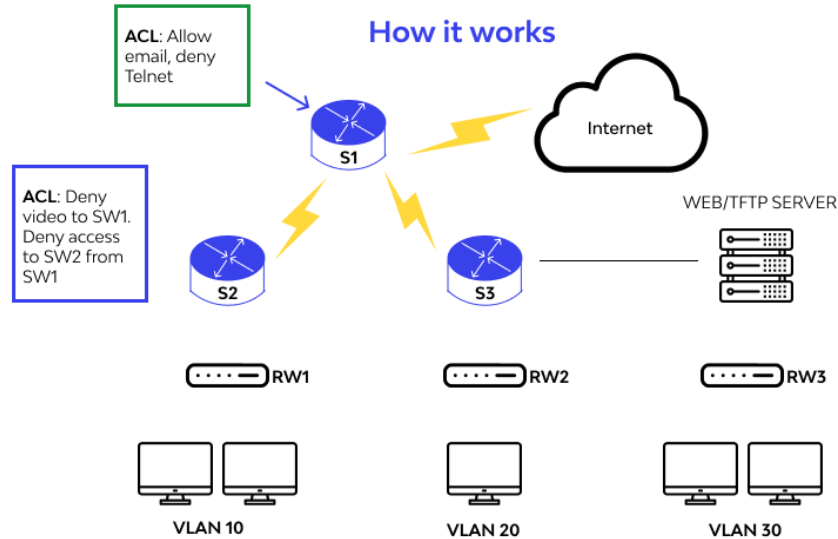
I permessi dipendono da attributi dell'utente (es. età, dipartimento, orario di accesso) e del contesto (es. dispositivo usato, posizione geografica)

- Offre maggiore flessibilità rispetto a RBAC
- Può risultare più complesso da implementare e gestire



# Access Control List (ACL)

Le risorse hanno liste di controllo accessi che specificano chi può accedere e con quali permessi  
Permette una gestione dettagliata dei permessi a livello di singola risorsa  
Può diventare difficile da amministrare in sistemi con molte risorse e utenti



# Sicurezza e best practice per autenticazione e autorizzazione

- Protezione delle credenziali: utilizzo di hashing e salting per le password
- Uso di HTTPS per la protezione delle comunicazioni
- Limitazione della durata delle sessioni per ridurre i rischi di attacchi
- Logging e monitoraggio per rilevare accessi sospetti
- Principio del privilegio minimo: concedere solo i permessi strettamente necessari
- Revisione periodica dei permessi: evitare che utenti ex-dipendenti o con ruoli cambiati abbiano ancora accesso a dati sensibili
- Implementazione di audit trail: tracciare tutte le modifiche ai permessi e gli accessi effettuati