Sessions

- A user's interaction with an application over a period of time is known as a session. Sessions help applications track authenticated users and maintain state across multiple requests.
- **Types**
  - Application Sessions
    - Session Identifiers
      - Unique IDs representing user sessions
      - Stored in cookies (web apps) or client-side storage (SPAs, native apps)
    - Session Data Storage
      - Server-side: memory, filesystem, database, shared services (e.g., Redis)
      - Client-side: in-memory or local persistent storage (careful with sensitive data)
    - Session Timeout
      - Idle timeout
        - Session expires after a period of inactivity
        - Protects against abandoned sessions being misused
      - Maximum session duration
        - Session expires after a fixed period, regardless of activity
    - Session Renewal
      - Applications may renew sessions upon user interaction
      - Proactive prompts to users before session expiration
    - User Experience Considerations
      - Balance between security and usability
      - Strategies to prevent loss of user work upon session expiration
  - Identity Provider Sessions
    - Purpose
      - Maintain user authentication state across multiple applications
      - Facilitate Single Sign-On (SSO)
    - Mechanism
      - Identity Provider (IdP) creates a session with session data
      - Session information stored in cookies set by the IdP
    - Session Management
      - IdP uses cookies to recognize authenticated users
      - Applications redirect users to IdP for authentication or session checks
    - Session Renewal
      - Applications can redirect users to IdP to renew tokens or sessions
      - Use of parameters like prompt and max_age in OIDC to control reauthentication
- **Multiple Sessions**
  - Existence at Different Layers
    - Users may have sessions with applications and identity providers
    - Possible layers:
      - Application Session
      - Authentication Broker Session
      - Remote Identity Provider Session
  - Interaction Between Sessions
    - Application session expiration may not affect IdP session
    - IdP session termination may not immediately end application sessions
  - Session Termination Scenarios
    - User logout
    - Session timeout
    - Administrator-initiated termination
    - Cookie deletion or server restarts
  - Considerations
    - Define impact of session termination at each layer
    - Plan for consistent user experience
- **Session Duration**
  - Idle Timeout
    - Based on inactivity period
    - Resets upon user activity in the application
  - Maximum Session Duration
    - Fixed maximum time for a session
    - Enhances security by requiring periodic reauthentication
  - User Experience
    - Warnings before session expiration
    - Options to extend session
    - Balancing security needs with usability
  - Factors Influencing Duration
    - Sensitivity of application and data
    - Device type (desktop vs. mobile)
    - User context (consumer vs. enterprise
- **Session Renewal**
  - Mechanisms
    - Redirect to Identity Provider
      - Application redirects user to IdP for session renewal
      - IdP may reauthenticate user or issue new tokens based on existing session
    - Silent Authentication
      - Use of hidden iframes to check session status (limited by browser policies)
    - Use of Authentication Parameters
      - OIDC prompt Parameter
        - Controls whether to force reauthentication (prompt=login) or suppress prompts (prompt=none)
      - OIDC max_age Parameter
        - Specifies maximum time since last authentication
        - IdP may prompt user to reauthenticate if exceeded
    - Alternative Methods
      - Some IdPs offer APIs to check session status without redirects
- **Token Renewal**
  - Need for Renewal
    - Access tokens and ID tokens have expiration times
    - Applications may need fresh tokens to continue operations
  - Methods
    - Refresh Tokens
      - Confidential clients can use refresh tokens to obtain new access tokens
      - Public clients face challenges due to secure storage limitations
    - Redirect to Identity Provider
      - Applications redirect users to IdP to obtain new tokens
      - If IdP session is valid, new tokens are issued without reauthentication
    - Silent Authentication Challenges
      - modern browsers implement policies like SameSite cookies and block third-party cookies, affecting iframe-based silent authentication.
      - May not be reliable for token renewal
    - Considerations for SPAs
      - Avoid storing long-lived tokens in browser storage
      - Use short-lived access tokens and redirect for renewal
- **Reconstituted Sessions**
  - Improving User Experience
    - Allow users to restore session state after expiration
    - Reduce disruption caused by session timeouts
  - Techniques
    - Session State Retention
      - Store session data securely even after session expiration
      - Restore data upon user reauthentication
    - User Reauthentication Flow
      - Prompt user to reauthenticate when session expires
      - After successful authentication, restore session state
- **Set limits on how long dormant session data is retained**