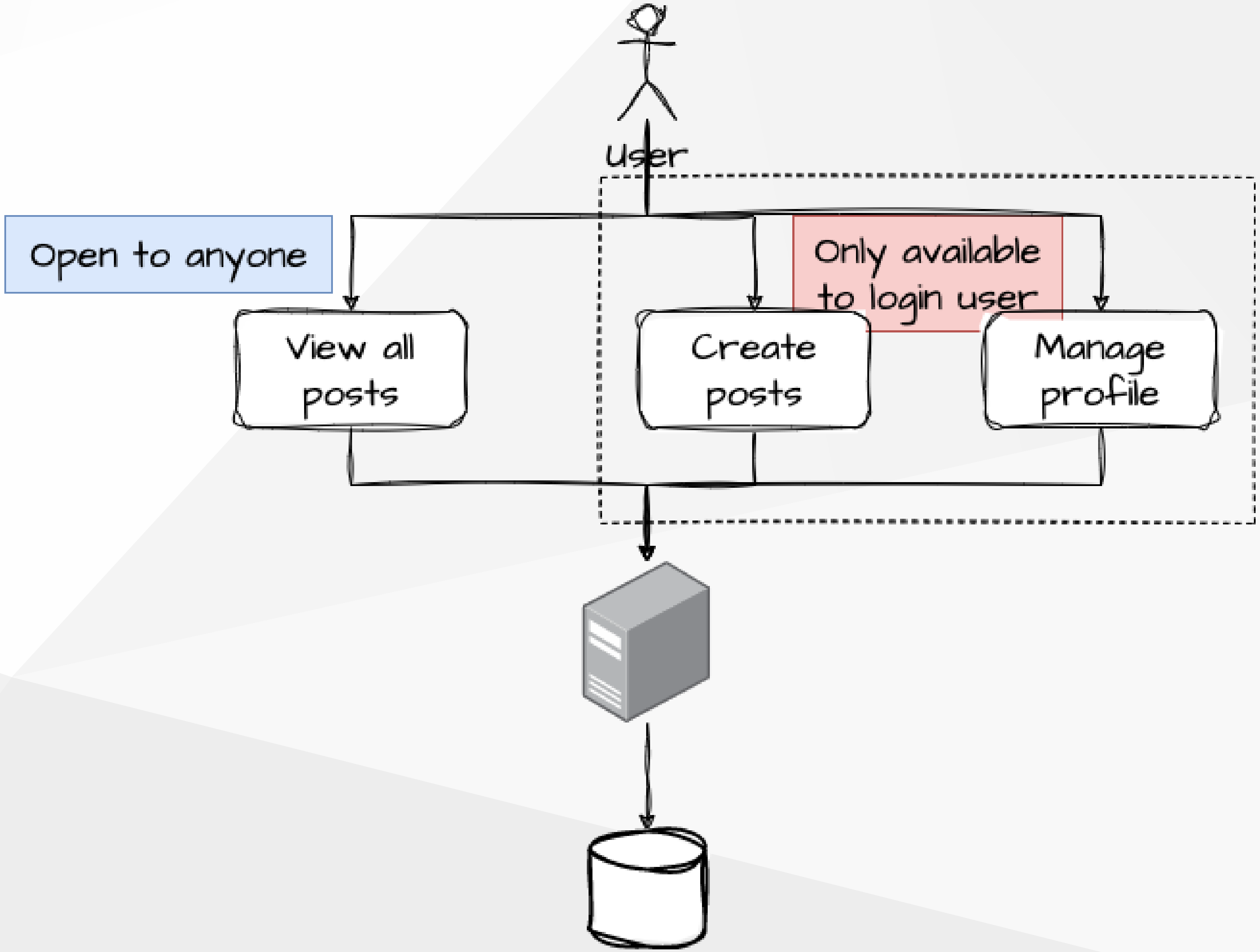
Authentication & Authorization

- Authentication is the process of verifying the identity of a user.
- Authorization is the process of verifying that a user has access to a resource.

Authentication

Question:

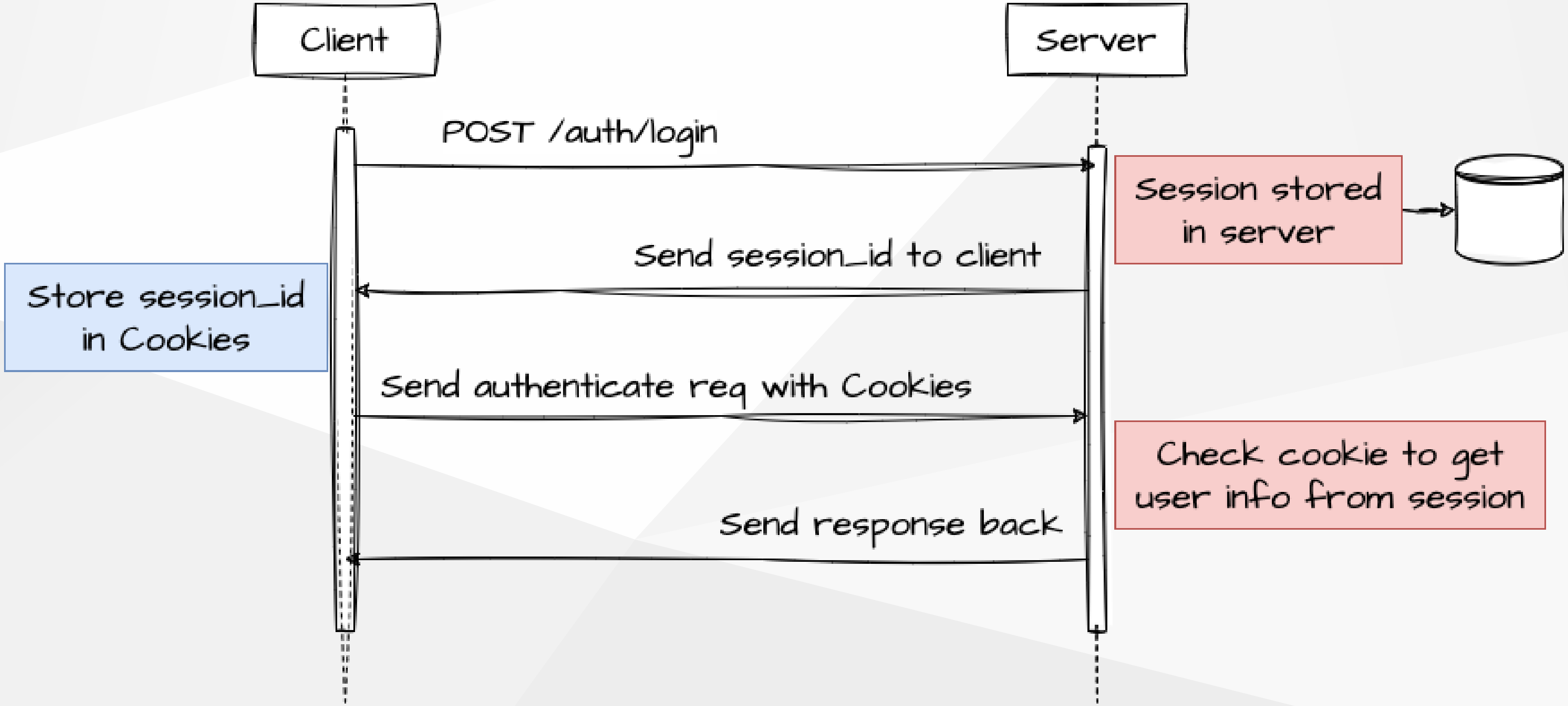
- How to verify the identity of a user?
- How to keep the identity of a user?



Cookies and Sessions

- Cookies are small pieces of data stored in the browser of a user.
- Sessions are a server-side storage of information that is desired to persist throughout the user's interaction with the web site or web application.
- Sessions and cookies work together to authenticate users and store information about them.
- Sessions are more secure than cookies as it is stored in server, whereas cookies are stored in browser.

How it works?


Client
Server
POST /auth/login
Session stored in server
Send session_id to client
Store session_id in Cookies
Send authenticate req with Cookies
Check cookie to get user info from session
Send response back

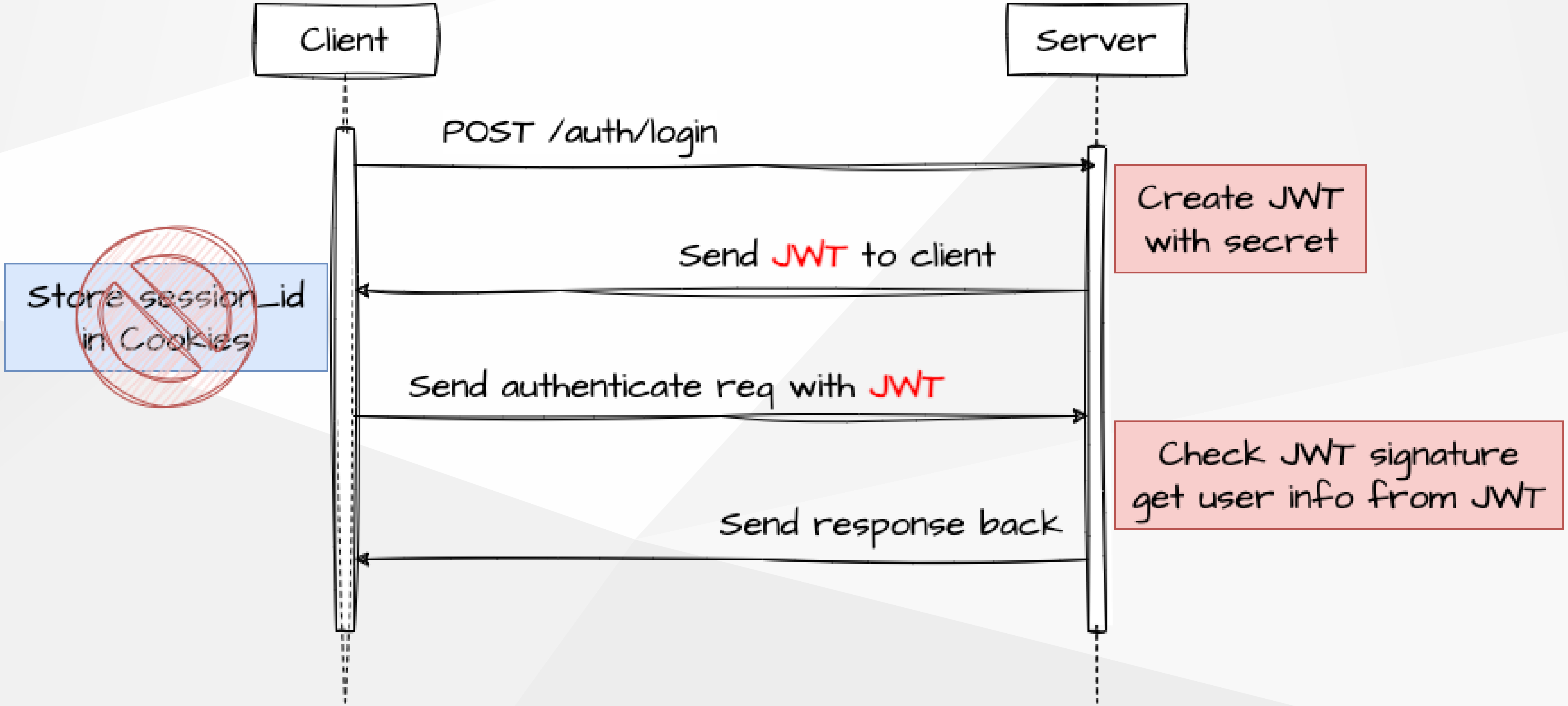
Issues with Cookies

- Cookies are vulnerable to Cross-Site Scripting (XSS) attacks.
- Cookies are vulnerable to Cross-Site Request Forgery (CSRF) attacks.
- Session need to be stored in server, which is not scalable.
- Mobile apps might not use cookies.

JSON Web Tokens (JWT)

- JWT is an open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object.
- This information can be verified and trusted because it is digitally signed.

How it works?

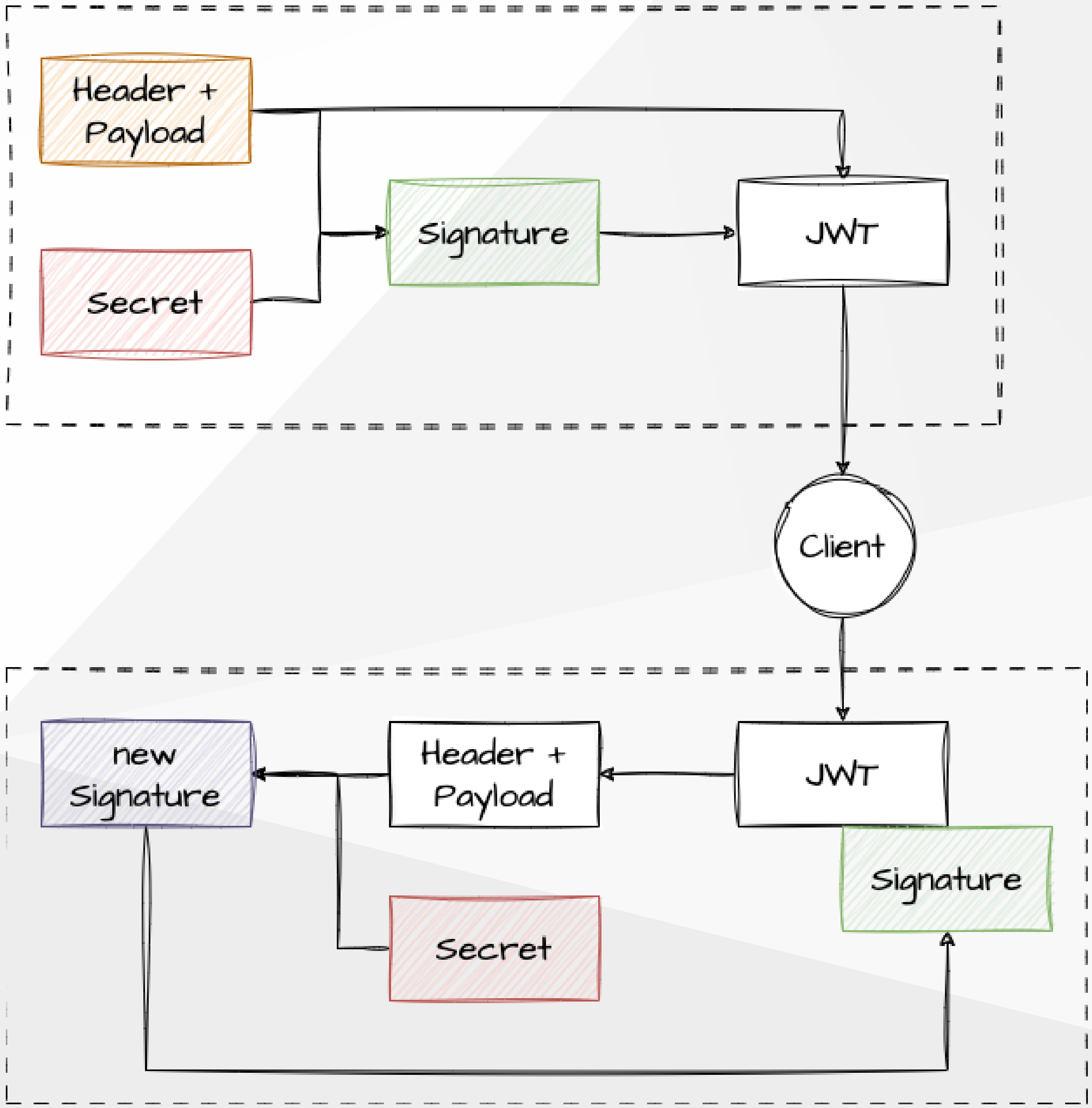

Client
Server
POST /auth/login
Create JWT with secret
Send JWT to client
Store session_id in Cookies
Send authenticate req with JWT
Check JWT signature get user info from JWT
Send response back

JWT Structure

- JWT
- Header
  - Algorithm, token type, etc.
- Payload
  - Claims, e.g. user id, expiration time, etc.
  - No sensitive information
- Signature
  - Header + Payload + **Secret**

Signing and Verifying

- The secret is only known to the server.



JWT vs Session & Cookies

- JWT is stateless, while session is stateful.
- JWT is easy for scalability.
- JWT avoid CORS issues.

Advantages of JWT

- Compact: JWT is compact, which is easy to pass through URL, POST parameter, or inside HTTP header.
- Self-contained: JWT contains all the information needed to verify the token.
- Secure: JWT can be signed using a secret or a public/private key pair.
- Cross-domain: JWT can be used across different domains.
- Cross-platform: JWT can be used across different platforms.

Disadvantages of JWT

- Compromised secret: If the secret is compromised, all the tokens are compromised.
- Data visibility: JWT is not encrypted, so the payload can be read by anyone.
- Revoking: JWT cannot be revoked, unless you maintain a blacklist.
- Data overhead: JWT is larger than session. Should be careful what need to be passed within JWT in URL.

JWT in Node.js

- jsonwebtoken
- passport-jwt