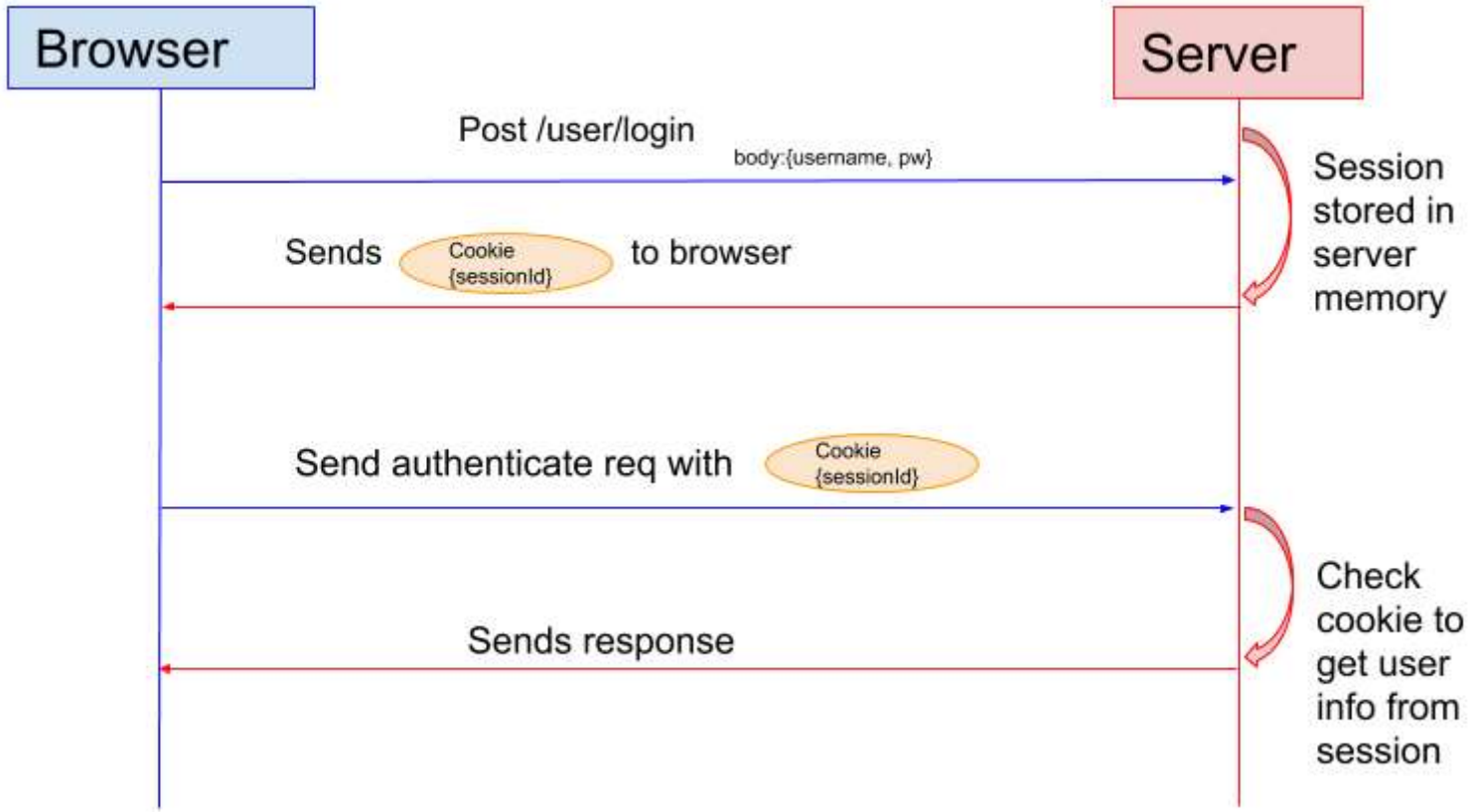
Kentico

Authentication, JWT, Security & Stuff

Slavomir Moroz

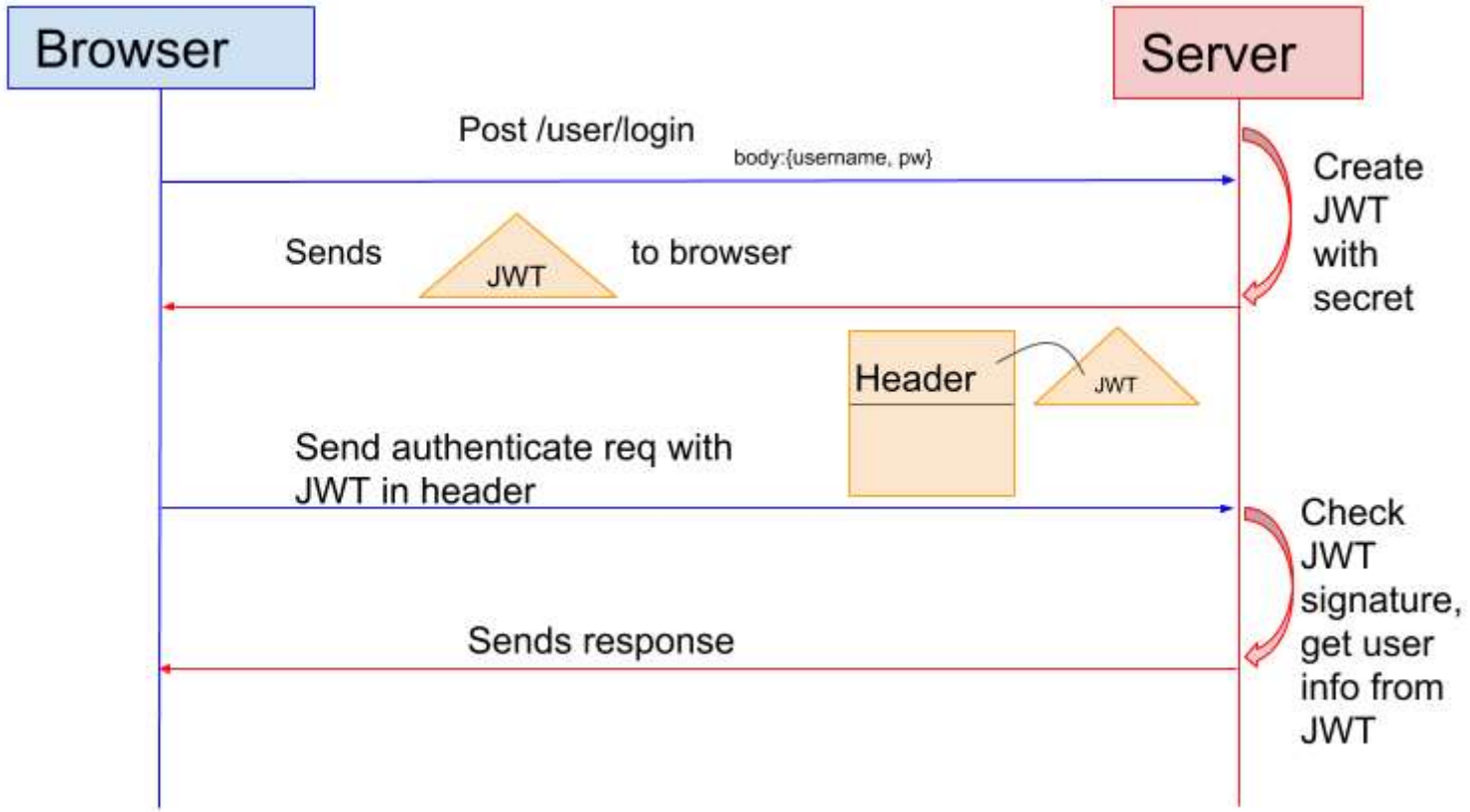
Kentico

Session



Kentico

Token



Kentico

What's a token?

- Most commonly used format is JSON Web Token (JWT)
- Claims
  - are statements about an entity (typically, the user) and additional data.
  - e.g. name, email, permissions, expiration, issuer
- JWT consists of
  - Header – token type information and hashing algorithm used by token
  - Pyload – collection of claims
  - Signature – used to verify that content hasn't changed and that token is legitimate
    - (works same way as digital signature)
- https://jwt.io/introduction/

Kentico

Session vs. Token

- Statefull vs. Stateless

- The biggest difference here is that the user's state is not stored on the server, as the state is stored inside the token on the client side instead.

- Scalability:
  - **Sessions** need to store user data in server's memory. Scaling becomes an issue with huge number of users.
  - **Tokens** – no issue, data are stored on the client side
- Multiple APIs
  - **Session** - cookies normally work on a single domain or subdomains. It poses issues when APIs are served from a different domains.
  - **Tokens** – no issues

Kentico

Where to store tokens?

https://stormpath.com/blog/where-to-store-your-jwts-cookies-vs-html5-web-storage

TLDR:

- Cookie
  - Sent with each request automatically. Vulnerable to CSRF
  - Secure flag – only sent via HTTPS
  - HttpOnly flag – not accessible from JavaScript. Protects you from XSS.
  - Works only with single domain. Complicated with multiple APIs (domains).

- Local storage
  - Is not attached to requests automatically. You can choose whether you wish to send JWT. Protects from CSRF.
  - No secure transfer standards. You must ensure that JWT is sent over HTTPS and never HTTP.
  - Accessible from JavaScript. Vulnerable to XSS.

Kentico

Threats

Cross-Site Scripting (XSS) ranks #7 in OWASP TOP TEN 2017 (Top 10 security issues)

Cross-Site Request Forgery (CSRF) is nowadays less of a threat.

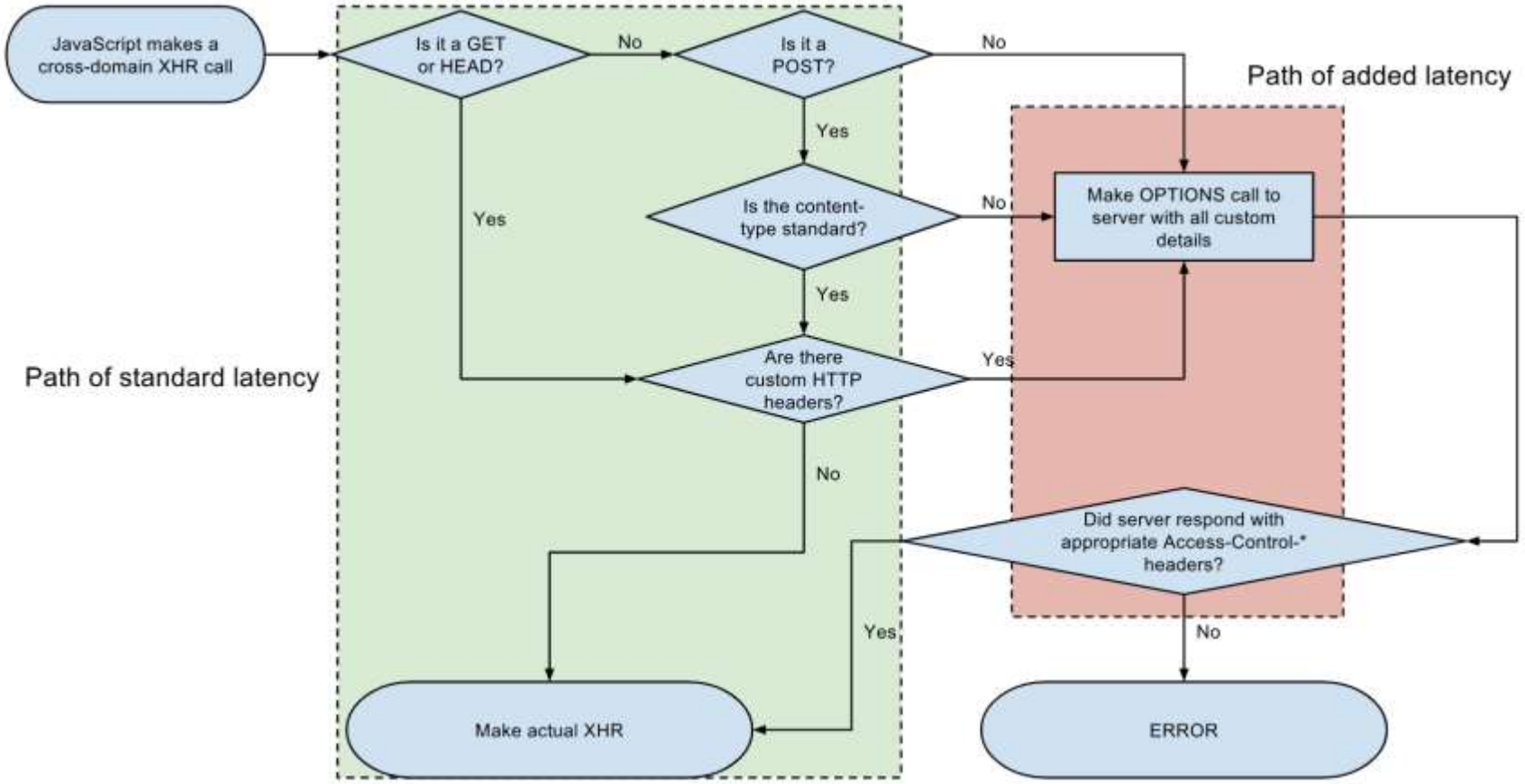
Even if you don't implement any CSRF protection, you can eliminate the thread by properly configuring your website.

- Disable the option the embed your site as an iframe via HTTP header.
  - X-Frame-Options: DENY
- Configure CORS to only allow requests from your trusted domains.

Defense mechanisms mentioned above only works with modern browsers with proper security headers and CORS support.

Kentico

Cross-origin resource sharing (CORS)



Kentico

So what should I use?

Cookies are safer

BUT

They are “impossible” to use when client communicates with server devices with multiple domains.

Kentico

Slightly better multiple domain solution

Long lived REFRESH token provided by auth server when provided with valid credentials and stored as cookie.

Short lived ACCESS token provided by auth server when provided with refresh token and stored in local storage or application memory.

REFRESH token only purpose is to request new ACCESS token.
ACCESS is then used by client to authenticate against API servers.