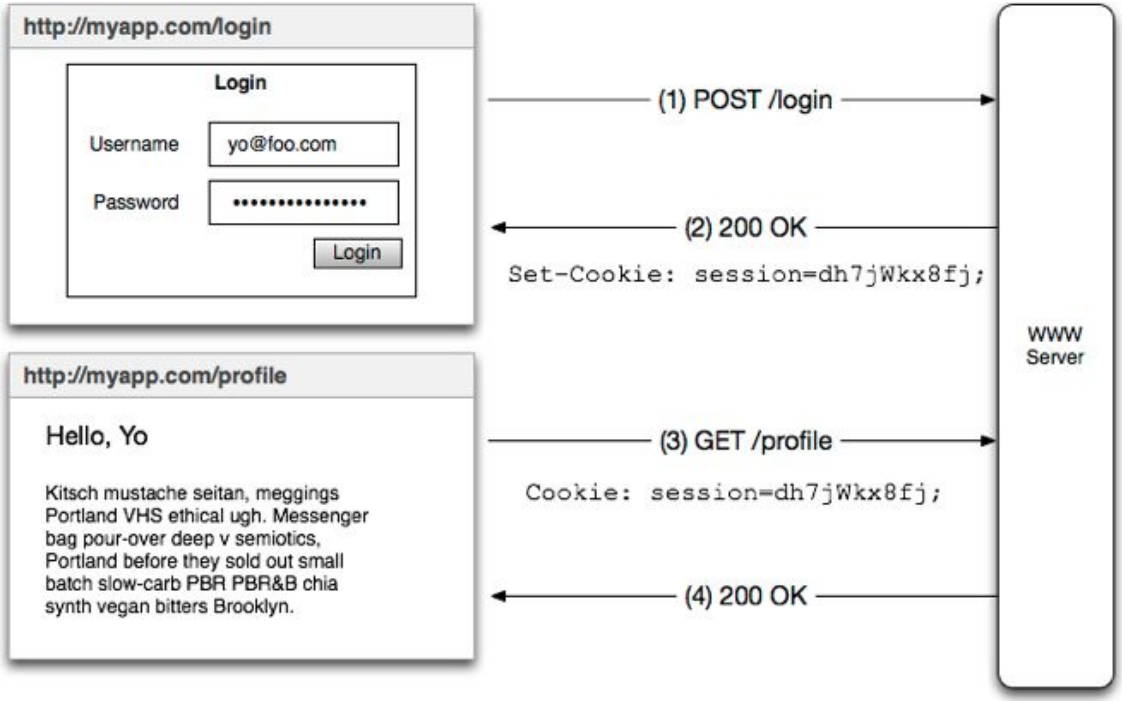
JWT

JSON Web Tokens

Classic authentication with sessions



- They're opaque and have no meaning (they're just pointers)
- Database heavy: session ID lookup on every request
- Cookies need to be secured to prevent session hijacking

Definitions

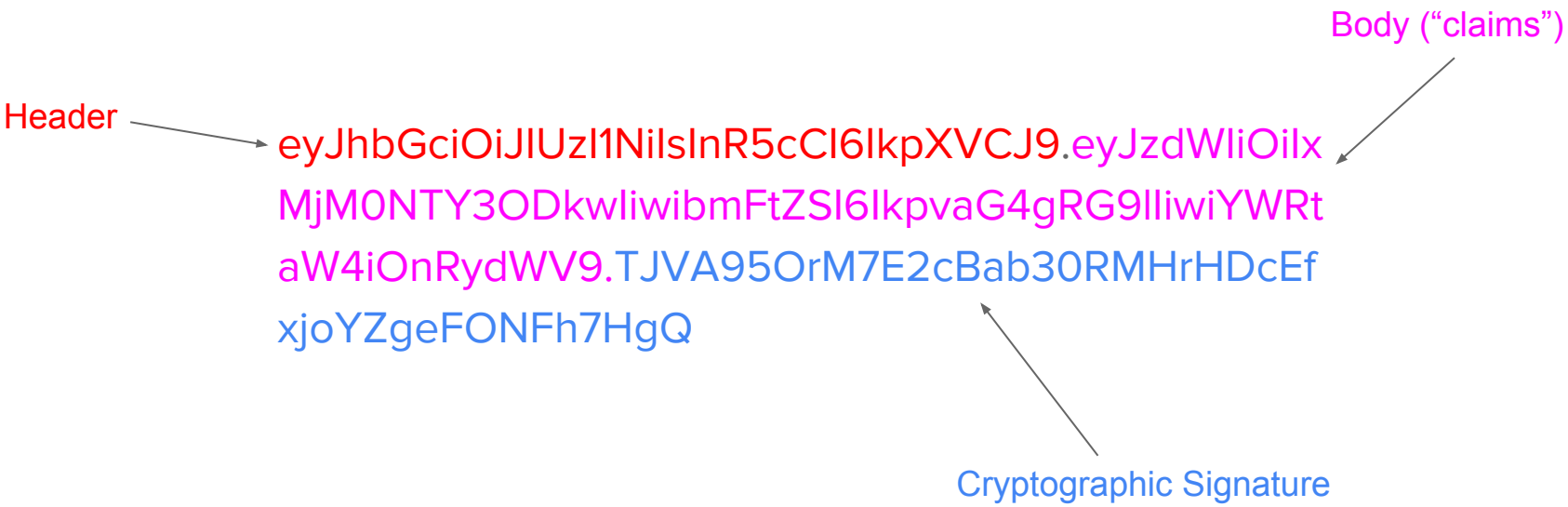
Authentication is proving who you are.

Authorization is being granted access to resources.

Tokens are used to persist authentication and get authorization.

JWT is a token format.

Example


Body (“claims”)
Header
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIx
MjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiYWRt
aW4iOnRydWV9.TJVA95OrM7E2cBab30RMHrHDcEf
xjoYZgeFONFh7HgQ
Cryptographic Signature

Issuing JWTs

1. User has to present credentials to get a token (password, api keys, etc.)
2. Tokens are issued by your server, and signed with a secret key that is private
3. The client stores the tokens, and uses them to authenticate requests

JWT Creation

```
 header = {
          "alg":"HS256"
}

claims = {
    "isAdmin": true,
    "userName": "Arik"
}

signed = HMACSHA256(
    base64UrlEncode(header)+"."+base64UrlEncode(claims), "secret"
)

token = base64UrlEncode(header)+"."+base64UrlEncode(claims)+"."+signed
```

Sending

POST http://mywebsite.com/secretPage

Headers:

Authorization: **BEARER** eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiYWRtaW4iOnRydWV.
TJVA95OrM7E2cBab30RMHrHDcEfxjoYZgeFONFh7HgQ