M&I SYSTEMS GROUP

VOJVODINA ICT CLUSTER

Autentikacija i Autorizacija

GO4Code

Marko Cvijanović
Mihailo Bubnjević

Autentikacija

- Dokazivanje da je neka činjenica ili dokument ispravan
- U sferi softvera se podrazumeva da se to odnosi na korisnike i verifikaciju njihovih identiteta
- Proces verifikacije korisničkog identiteta pre nego što mu dopustimo da koristi naš sistem ili naše resurse
- Autentikacija se obavlja u 3 koraka:
    - Identifikacija - Ko si ti?
    - Autentikacija - Dokaži da si to ti.
    - Autorizacija - Da li imaš dozvolu za ovo?

Autentikacija

- Načini autentikacije:
  - Ono što korisnik zna (username + šifra, passphrase)
  - Ono što korisnik ima (token, fizički uređaj)
  - Ono što korisnik jeste (biometrika, otisak prsta, facial recognition)
- Često se koriste kombinacije dva ili više pristupa pri autentikaciji radi jače sigurnosti
  - MFA / 2FA
  - Google Authenticator

Autorizacija

- Autorizacija je proces provere da li korisnik ima dozvolu da koristi deo sistema kojem pokušava da pristupi
- Korisniku je predefinisan nivo pristupa na osnovu pravila u sistemu
- Kontrola pristupa je često definisana određenim ulogama koje se dodeljuju korisnicima
- Uloga ili rola definiše podskup funkcija kojim korisnici sa tom rolom mogu da pristupe

Autentikacija

- Basic Authentication
  - Osnovni i najlakši način autentikacije preko HTTP
  - Retko se koristi
  - Klijent u request header stavlja **username:password**
  - Koristi **Base64** encoding

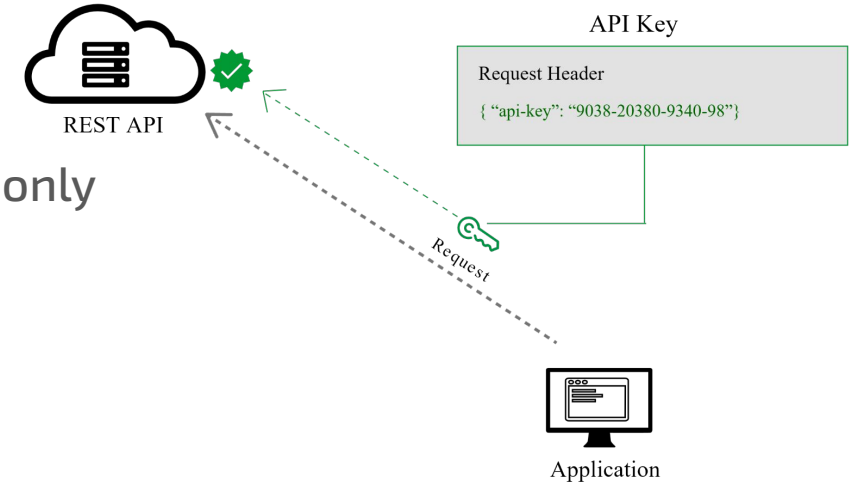
```
Authorization: Basic ZGVtbzpwQDU1dzByZA==
```

Autentikacija

- Bearer Authentication
  - Šema za autentikaciju takođe zvana Token Authentication
  - “give access to the bearer of this token”
  - Kriptovani string koji generiše server kada se korisnik uloguje
  - Klijent ga šalje u **Authorization** Header-u
  - Sigurna samo preko HTTPS (SSL)

Autentikacija

- API KEYS
  - Jednostavno rešenje
  - Unapređenje naspram Basic Auth
  - Generiše se unikatna vrednost koju korisnik dobije od sistema i koristi za svaki sledeći pristup sistemu
  - Šalje se u Authorization Headeru
  - Dobro rešenje samo u slučaju Readonly sistema



HTTPS

- S u HTTPS znači SECURE
- Klijent i server se rukuju (Handshake)
- HTTPS koristi TLS za enkripciju normalnih HTTP zahteva i odgovora
- TLS koristi i simetričnu i asimetričnu ekripciju
- Website mora da sadrži TLS sertifikat da bi koristio ovu vrstu enkripcije
- Sertifikat mora da bude verifikovan od strane **Trusted Root Certification Authority**

HTTPS

1. Klijent kontaktira server da započne komunikaciju
2. Server mu šalje svoj sertifikat i javni ključ
3. Klijent proverava sertifikat kod Trusted Root Certification Authority-ja
4. Klijent i server se dogovaraju koji je najjači tip enkripcije koji obe strane mogu da podrže
5. Klijent enkriptuje *session key* koristeći serverov javni ključ i šalje ga serveru
6. Server dekriptuje klijentov ključ sesije sa svojim privatnim ključem
7. Sesija je uspostavljena
8. Za buduću komunikaciju se koristi ključ sesije za enkripciju i dekripciju podataka koji server i klijent razmenjuju

Autentikacija

- OAuth (2.0)
  - Standard za token-based autentikaciju preko javnih mreža
  - OAuth dopušta third-party servisima kao što je Facebook i Google da koriste korisničke informacije bez otkrivanja korisničkih kredencijala
  - Sa korisničkom dozvolom on daje *access token* third-party servisima da dele korisničke informacije
  - “sign in with Google” je primer OAuth implementacije

JWT (Json Web Token)

- Standard u industriji za razmenu informacija između dva entiteta
- JWT sadrži JSON objekat sa informacijama koje se razmenjuju
- Najčešće se koristi za autorizaciju nakon što se neki korisnik ulogovao
- Dopušta korisniku da pristupi rutama, servisima i resursima bez dodatnog logovanja koristeći sesije
- Server generiše JWT i šalje ga korisniku
- Korisnik koristi taj token za sve naredne zahteve ka tom serveru sve dok token ne istekne

Struktura JWT-a

- **Header**
  - sadrži tip tokena i algoritam koji koristi za potpisivanje (Npr. SHA256)

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

Struktura JWT-a

- **Payload**
  - Sadrži tvrdnje (*Claims*)
  - Claim je neka informacija o entitetu (najčešće korisniku) i neke dodatne podatke
  - Postoje 3 vrste Claim-ova:
    - Registrovane (Registered claims)
    - Javne (Public Claims)
    - Privatne (Private Claims)

Struktura JWT-a - Payload

- Registered claims
    - predefinisane tvrdnje koje nisu obavezne ali su preporučene (issuer, expiration time, subject, audience..)
- Public claims
    - definiše ih korisnik tog JWT-a po potrebi
- Private claims
    - tvrdnje koje znaju samo onaj ko je napravio token (server) i onaj koji koristi token(klijent)

Struktura JWT-a

- **Signature**
  - Da bi se napravio potpisni deo JWT-a potrebno je uzeti enkodovane header, payload, secret i algoritam koji je naveden u headeru, i potpisati ga.

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  secret)
```