NONE hashing algorithm

Problem nastaje kada napadač preuzme token i promeni mu heš algoritam na “none”. Neke biblioteke ovakav token tretiraju kao validan pa je tako napadaču omogućen pristup.

Nase resenje je otporno na ovu ranjivost proverom potpisa.
Ne postoji mogucnost za probijanje ovog tipa jer se svakom tokenu proverava potpis i ukoliko ne postoji ili je neispravan ne propusta se dalje u funkcionalnosti (dobijamo exception “UnsupportedJwtException”)

Konretno 82. Linija u TokenUtils (.parseClaimsJws(token)) resave problem.
Test koji pokazuje otpornost resenja.
(https://github.com/jwtk/jjwt/commit/060666a32e903a08a5eef43ed9c5af75ea05dba5)

Token sidejacking

Problem nastaje kada napadač ukrade token i koristi ga kako bi dobio dozvolu za određene funkcionalnosti umesto korisnika.

Rešenje se štiti od ovog napada tako ubacuje korisnički kontekst u token.
Koraci ove zaštite su:

1. Random string se generiše prilikom autentifikacije i uključuje se u token.
2. Klijentu se string šalje kao cookie.
3. Heš stringa se čuva u tokenu kako bi se zaštitili od krađe stringa iz tokena i setovanja cookie-a istom vrednošću.
4. Tokom token validacije, ako prijvaćeni token ne sadrži dobar kontekst dolazi do odbijanja.

Token explicit revocation by the user

Problem nastaje jer je token nevalidan samo kada istekne. Ne postoji način za eksplicitno povlačenje tokena. Tako da ne postoji zaštita kada bi se ustanovilo da napadač poseduje token.

Rešenje bi obuhvatilo token blacklist u koju bi se smestio token prilikom logout-a.

Token information disclosure

Problem nastaje kada napadač preuzme token i pristupi podacima iz njega kako bi došao do informacija vezanih za sistem (role, login format).

Zaštita bi bila šifrovanje tokena.