# OWASP TOP 10

## Injection

Od injection napada smo se zaštitili validacijom svih podataka koji se šalju na bek. Sve dto-ove u našoj aplikaciji smo validirali koristeći anotacije iz javax.validation.constraints paketa. Iznad svih obaveznih polja u dto-u smo stavili *@NotBlank* anotaciju koja proverava da li je uneta vrednost za dato polje. Takođe, iznad svih polja je dodata *@Size* anotacija koja određuje minimalnu i maksimalnu dužinu ukoliko se nalazi iznad String-a, ili ukoliko se nalazi iznad kolekcije minimalan i maksimalan broj elemenata u kolekciji. Najvažnija anotacija je *@Pattern* kojoj se prosleđuje regex koji dato polje mora da zadovolji. Zahvaljujući regex-u onemogućeno je slanje neželjenih karaktera na back i zaštićeni smo od injection napada.

Na isti način smo se zaštitili od injection napada preko *RequestParams* i *PathVariable*

Zaštitu od injectiona na frontu smo implementirali pomoću *Validators* modula iz @angular/forms.

## Broken-Authentication

Help

## Sensitive Data Exposure

Prilikom rada sa pacijentima u našoj bolničkoj aplikaciji smatrali smo da je njihov medicinski karton osetljiva informacija kojoj ne bi trebao svako da ima pristup. Svaki novi pacijent koji je dodat u bazu je prethodno šifrovan, i time je i prilikom potencijalnog napada na bazu aplikacije i krađe podataka obezbeđena tajnost njegovih ličnih podataka. Jedino doktori aplikacije imaju pristup pacijentima – doktor pacijent poverenje. Zaštitu podataka o pacijentima u tranzitu obezbeđuje https.

Keycloak obezbeđuje sigurno rukovanje korisnicima u sistemu – doktor, admin i super admin.

## XML External Entities

Ne koristimo XML format nigde u aplikaciji.

## Broken Access Control

Svi resursi za koje postoji mogućnost pristupa tokom korišćenja naše aplikacije su zaštićeni pomoću keycloak-a. Kompletan RBAC sistem je implementiram korišćenjem rola i permisija vezanih za te role.

## Security Misconfiguration

Valjda je sve dobro konfigurisano…

## Cross-Site Scripting XSS

U WebSecurityConfiguration konfiguracionoj klasi je reimplementirana *configure* metoda i dodata je zaštita od xss napada:

http.headers().xssProtection()  
.and().contentSecurityPolicy("script-src 'self'");

## Insecure Deserialization

Svaka deserijalizacija objekta je odrađenja u *try catch* bloku i ukoliko dođe do neispravne deserijalizacije to je bilo zabeleženo.

## Using Components with Known Vulnerabilities

Ne radimo to.

## Insufficient Logging & Monitoring

Svaki izlaz iz aplikacije je pokriven logovima koristeći *Logger* klasu iz org.slf4j paketa. U sistemu postoje 3 tipa logova: informacijoni logovi – info, logovi upozorenja – warn i logovi grešaka aplikacije – error. Većina logova u aplikaciji je bila informaciona, oni su standardni tokovi koji beleže normalan tok korišćenja aplikacije. Logovi upozorenja predstavljaju neku grešku sa klijentove strane – pogrešni kredencijali prilikom logovanja na aplikaciju, pokušaj pristupa resursu po nepostojećem id..Logovi greške su interne serverske greške – pogrešna putanja do direktorijuma, loš ključ prilikom šifrovanja/dešifrovanja podataka, neuspela konekcija ka bazi..

Svi logovi sadrže informaciju o:

* Datumu – tačan datum kada se događaj desion
* Vremenu – tačno vreme kada se događaj desio
* Izvor događaja – koji program, komponenta ili korisnički nalog je prouzrokovao događaj
* Tip događaja – info, warning, error
* Opis događaja – opis koji bliže opisuje konkretan događaj
* Status kod/rezultat – status kod koji je prouzrokovao izvršen događaj
* Opis status koda – poruka koja bliže opisuje razlog koji je doveo do status koda

Logovi obezbeđuju neporecivost.

Primeri logova:

* 1. Info log

logger.info(String.*format*("%s called method %s with status code %s: %s",  
 "Medical device", "receivePatientStatus", HttpStatus.*OK*, "patient status received"));

1. Warn log

logger.warn(String.*format*("%s called method %s with status code %s: %s",  
 "Medical device", "receivePatientStatus", HttpStatus.*NOT\_FOUND*, "non existent patient id"));

1. Error log

logger.error(String.*format*("%s called method %s with status code %s: %s",  
 "Medical device", "receivePatientStatus", HttpStatus.*BAD\_REQUEST*, "untrusted certificate"));

Admin bolničke aplikacije i super admin adminske aplikacije imaju pregled logova i takođe imaju pregled alarma koje su ti logovi izazvali i mogu blagovremeno da reguju ukoliko dođe do napada na aplikaciju.