OWASP Top 10

A1 Injection

**Opis**

Injection napad predstavlja porodicu napada koji se ostvaruju slanjem podataka koji nisu validirani. Podaci koji stignu mogu prouzrokovati nepredvidjene akcije.

Najcesci napad jeste SQL Injection, proces injektovanja koda u bazu podataka. Izazvan direktnim prosledjivanjem ulaznih parametara u SQL upite, pri cemu ukoliko se koristi konkatenacija stringova za upit maliciozni korisnik moze uspesno izbrisati bazu, neautorizovano pristupiti podacima I cak I izmeniti iste.

**Resenje**

Validacijom unosa na back-u I na front-u delimicno sprecavamo napade. Validacije sadrze ogranicenja (pre svega specijalnih karaktera) I zahteve (mora sadrzati bar 2 karaktera, nema brojeva..) za svaki unos.

SQL Injection se resava parametrizovanim upitima ka bazi podataka. Kao primer imamo pronalazak korisnika po username-u, gde smo formirali upit tako da izbegnemo potencijalni napad. Na ovaj nacin napadac ne moze da promeni namenu upita.

Safe

repo.Database.First(&user, "username = ?", userName)

Not safe

repo.Database.First(&user, "username = " + userName)

U okviru PKI sistema koristili smo JPA koji obezbedjuje sigurne unose pomocu prepared statement-a.

Log injection se resava koriscenjem proverene logrus biblioteke za Nistagram sistem, a za PKI sistem smo koristili slf4j proverenu biblioteku.

A2 Broken Authentication

**Opis**

Pokusaj menjanja lozinke drugog korisnika predstavlja ozbiljan napad. U slucaju uspeha on pristupa sadrzaju koji ne sme da vidi ili menja, a takodje i sprecava pravog korisnika da on pristupi svojem sadrzaju.

Drugi nacini za maliciozan ulaz u nalog nekog korisnika bi bili brute-force I dictionary napad, ali I samo ostavljanje ulogovanog naloga bez nadleznosti (odlazak od uredjaja na neko vreme I potencijalni pristup neke druge osobe tom uredjaju).

**Resenje**

Korisnik koji je zaboravio svoju lozinku ima pravo da kreira novu i za to je potrebno da ukuca svoj email kako bi verifikacioni mejl stigao na odgovarajuce mesto. Mejl ce biti poslat samo ako postoji registrovani korisnik sa tim email-om. Takodje ce se kreirati token koji ce isteci 5 minuta nakon slanja mejla. Kada mejl stigne potrebno je da korisnik klikne na poslati link i potvrdi da to jeste njegov nalog. U slucaju da je token istekao a korisnik nije kliknuo na link, potrebno je da se opet posalje zahtev za izmenu lozinke. Bez poznavanja email-a I pristupa mejlu on ne moze izvrsiti napad.

Brute-force I dictionary napad uspesno se savladavaju uz pomoc konkatenacije lozinke sa salt-om i hesiranjem, kao I validacijom jer se njom zahteva od korisnika da kreira dovoljno jaku lozinku.

Nakon sto se korisnik uloguje kreira se jwt token. Nakon sat vremena token istice I neophodno je ponovno logovanje korisnika. Sve do momenta odjave korisnika vrsi se redovna provera statusa tokena. Ovo sprecava svako duze nekoriscenje aplikacije, za slucaj da se korisnik ne odjavi.

Uz pomoc autentifikacije korisnik ne moze tek tako da pristupi podacima nekog korisnika, kao i da manipulise istim. TLS protocol I REST komunikacija (koja je stateless) dodatno su doprineli sigurnosti aplikacije.

A3 Sensitive Data Exposure

**Opis**

Slabo zasticeni podaci unutar aplikacije podlezni su napadima. Napadi koji mogu da se pojave su kradja kljuca I middle-man koji krade podatke (dok se oni transportuju ili iz brauzera korisnika).

**Resenje**

Kako bi smo izbegli ove probleme stitimo se sa:

* koriscenjem TLS/SSL protokola za sigurnu (HTTPS) komunikaciju izmedju klijenta I servera. Podaci u transportu se cuvaju upravo zbog najnovije verzije TLS-a 1.2.
* izbegavanjem cuvanja osetljivih podataka bez enkripcije. Lozinke su enkriptovane koriscenjem bcrypt algoritma za hesiranje (pre hesiranja se na lozinku nalepi slucajno generisan salt).
* podsticanjem korisnika da kreira jaku sifru. Potrebno je bar 12 karaktera, bar jedno veliko slovo, bar jedno malo slovo, bar jedan broj i bar jedan specijalan karakter. Validacija je formirana uz pomoc regex izraza I na back-u I na front-u.

A4 XML External Entities (XXE)

**Opis**

XXE je napad na aplikaciju koja parsira XML unos. Kada taj unos sadrzi reference na eksterni entitet I obradjen je sa slabom konfiguracijom XML parsera, dolazi do velike opasnosti. Ovo moze dovesti do otkrivanja poverljivih podataka I prekida usluga.

**Resenje**

U okviru nase aplikacije ne postoje xml fajlovi koji bi bili podlozni ovom napadu, ali nacin na koji bi pristupili problemu bi bio koriscenje libxml2. Ova biblioteka kreira XML documenta uz pomoc DOM nalik interfejsa I nudi XSD validaciju.

A5 Broken Access Control

**Opis**

Prilikom neautorizovanog koriscenja funkcionalnosti I manipulisanja podacima, svi podaci podlezu velikoj opasnosti. Jedan od cestih napada koji se javlja jeste pokusaj pristupanja stranici, kojoj napadac nema pravo pristupa, putem URL-a.

**Resenje**

Neovlasceni pristup resursima aplikacije resavamo redovnom proverom pristupa korisnika. Role-based access control nam je omogucio postavljanje uloga I permisija za autentifikovane korisnike aplikacije. RBAC ogranicava mrezni pristup zasnovan na ulozi osobe u sistemu, a uloge se odnose na nivoe pristupa koje korisnici imaju.

U PKI sistemu smo ovaj problem resili Spring Security-jem I dodeljivanjem uloga I permisija.

Prilikom pokusaja malicioznog subjekta da uz pomoc URL-a dospe do stranice za koju nije ovlascen, on ce biti redirektovan na stop stranicu (ona ga obavestava da ne moze da vidi stranicu koju taj URL gadja).

A6 Security Misconfiguration

**Opis**

Lose konfigurisana bezbednost najcesce je rezultat nebezbedne podrazumevane konfiguracije ili nedovrsene konfiguracije. Svaki operativni sistem, biblioteka I aplikacija moraju biti bezbedno konfigurisani.

**Resenje**

Sve biblioteke su podignute na najnovije verzije kako bi izbegle sve pretnje koje se javljaju u prethodnim verzijama. Pored toga nisu uvezani celi paketi sa svim funkcionalnostima vec delovi neophodni za akciju koju planiramo da izvrsimo. Koristimo najnovije verzije softvera koji ukljucuju operativni system (za potrebe koriscenja docker-a smo koristili windows 10) i sistem za upravljanje bazom podataka.

Log fajlovi ne sadrze osetljive detaljne informacije o greskama kao ni osetljive podatke korisnika. Takodje, nije omogucen pristup gledanja nikom drugom sem sistem adminu.

U aplikaciji ne postoje default-no napravljeni korisnici, te smo izbegli mogucnost napada na osetljive podatke korisnika iz fajla. Nakon sto se korisnik uloguje kreira se jwt token koji se koristi za sigurni transfer informacija izmedju endpoint-ova.

Dobra podela na mikroservise u okviru Nistagram sistema je omogucila da komponente budu razdvojene, te se jednim napadom ne moze napasti veliki broj komponenti istovremeno.

TLS protocol I REST komunikacija dodatno su doprineli sigurnosti aplikacije.

A7 Cross-Site Scripting (XSS)

**Opis**

XSS napad baziran je na ubacivanju skripte na web stranicu koju pregledaju korisnici aplikacije. Napad se javlja i na klijentskoj I na serverskoj strani. Njime je moguce preuzeti podatke svih korisnika koji pristupaju sajtu ali I ubaciti maliciozan sadrzaj, preuzeti sesiju korisniku i/ili redirektovati korisnika na malicioznu stranicu.

**Resenje**

Izbegavanje napada na:

1. klijentskoj strani
   * uz pomoc validacije na front-u
   * Vue framework podrzava {{varijabla}} izraze za postavljanje vrednosti varijabli (sa njima se vrednost varijable ne moze izvrsiti u pretrazivacu)
   * nisu koriscene v-hmtl directive kao I druge forme osetljive na XSS
2. serverskoj strani

* uz pomoc validacije na back-u
* pravilnim escape-ovanjem uz pomoc text/template biblioteke koja sadrzi HTMLEscapeString I JSEscapeString u Nistagram sistemu, a u PKI sistemu smo koristili XSS protection header I contentSecurityPolicy (“script-src `self`”)

A8 Insecure Deserialization

**Opis**

Prilikom bilo koje serijalizacije I deserijalizacije podataka u/iz fajla moguca je pojava ovog napada.

**Resenje**

Cuvanje svih podataka sem logova obavlja se u bazi. To znaci da se za te informacije ne vrsi serijalizacija i deserijalizacija. Dok logrus I slf4j biblioteke za cuvanje logova su proverene I read-only pristup logovima ima samo admin sistema.

A9 Using Components with Known Vulnerabilities

**Opis**

Za pomoc prilikom implementiranja funkcionalnosti koriscene su razne biblioteke. Svaka od njih donosi sa sobom I rizik, sto moze rezultovati narusavanjem CIA trijade (poverljivost, integritet I dostupnost).

**Resenje**

Za izbegavanje napada su koriscene najnovije verzije biblioteka. One prevazilaze probleme ranijih verzija.

U okviru Nistagram sistema za samu proveru mogucih napada I ranjivosti koriscena je National Vulnerability Database (NVD). Svaka opasnost koja je evidentirana uz pomoc NVD, uspesno je resena podizanjem verzije biblioteke (od koje opasnost proizilazi) I dodavanjem novih preporucenih biblioteka.

U okviru PKI sistema za samu proveru mogucih napada I ranjivosti koriscen je Dependency Check. Generisan izvestaj (moze se videti u dependency check report folderu BSEP i XML projekta) nam je pomogao da evidentiramo opasnosti I ranjivosti koje smo resili podizanjem verzije biblioteke (od koje opasnost proizilazi) I dodavanjem novih preporucenih biblioteka.

A10 Insufficient Logging & Monitoring

**Opis**

Sva nezeljena ponasanja poput neuspesnog logovanja ili promene lozinke bitno je zabeleziti. Svaka akcija je potencijalno maliciozni napad.

**Resenje**

Za resavanje ovog problema koristimo logger iz logrus biblioteke za Nistagram sistem. Postoje info I error logovi za svaki mikroservis zasebno. Info logovi prate sve uspesno sprovedene akcije, a error logovi sve neuspesne akcije. Nakon odredjenog vremena vrsi se arhiviranje I kreiranje novog log file-a (za nova upisivanja). U log fajl ne upisuju se osetljivi podaci (email ili lozinka), vec samo podaci koji su neophodni za jasne poruke I nisu podlozni novom napadu (vreme kreiranja, sifra aktivnosti, tekst greske, status uspesnosti, ime pozvane funkcije). Logove nije moguce menjati, vec samo videti I to moze samo admin sistema. Uz pomoc evidencije napada moguce je primetiti novu ranjivost koju do sada admin sistema nije uvideo.

U okviru PKI sistema ovaj problem smo resili uz pomoc slf4j biblioteke po istom principu.

Svaka aktivnost koju korisnik obavi I pokusa da argumentuje da nije obavio, lako se moze dokazati citanjem log fajla.