UNIVERZITET U BEOGRADU
ELEKTROTEHNIČKI FAKULTET

SONARQUBE IZVEŠTAJ

Projektni zadatak iz predmeta Razvoj bezbednog softvera

Predmetni profesor, asistent i saradnik:

Žarko Stanisavljević, prof. dr

Danko Miladinović, as. ms

Petar Vuković, Zühlke Serbia

Student:

Lazar Vulić 2022/3162

Beograd, školska godina 2022/2023.

SADRŽAJ

1. UVOD

Slike će se nalaziti u okviru foldera SLIKE i nazivane su po principu SlikaX, gde će X označavati redni broj slike. Struktura foldera SonarQube - Izvestaj je po sledećem principu:

- SonarQube Izvestaj:
  - SLIKE:
    - Slika1.1.png, Slika1.2.png
    - …
    - Slika55.png
  - SonarQubeIzvestaj.docx
  - SonarQubeIzvestaj.pdf

2. GENERALNI PREGLED APLIKACIJE I BAGOVA

SLIKA	PROBLEM	OPIS PROBLEMA
Slika1.1	-	Inicijalni pregled (Overview) projekta u SonarQube aplikaciji.
Slika1.2	-	Inicijalni pregled (Overview) projekta u SonarQube aplikaciji nakon ponovnog pravljenja SonarQube podešavanja. Sada je projekat Pass-ovao. Prethodno je Fail-ovao iz nekog nepoznatog razloga (Slika 1.1).*
Slika2	-	Generalni pregled spiska bagova koji postoje u projektu a koje prepoznaje SonarQube aplikacija. Bagovi mogu biti namerno ili slučajno napravljeni. SonarQube nudi objašnjenje do čega može dovesti pojedini bag, kao i način na koji se može razrešiti posmatrani bag.
Slika3	setID() ne prima parametar.	Seter metoda za ID u klasi Persons.java nije pravilno napisana. SonarQube nam sugeriše da je potrebno svega 3 minuta kako da to ispravimo (3min effort), pokazuje nam mesto u kodu gde je propust napravljen, kao i način da to ispravimo.
Slika4	get() metode mogu da vrate null.	Geteri unutar klase MovieRepository mogu biti null. Potrebno je pozvati metode isPresent() pre pristupa vrednostima.
Slika5	PreparedStatement() nije zatvoren.	PreparedStatement() nije zatvoren. Mogao je eksplicitno da se zatvori, ili implicitno tako što bi PreparedStatement() premestili unutara () iza try.
Slika6	PreparedStatement() nije zatvoren.	PreparedStatement() nije zatvoren. Mogao je eksplicitno da se zatvori, ili implicitno tako što bi PreparedStatement() premestili unutara () iza try.
Slika7	CSRF zaštita onemogućena. True Positive	Neko je namerno isključio CSRF. Rešenje je ili da se uključi, ili da se radi sa tokenima: token se kreira na početku sesije korisnika pomoću CSPRNG, uskladišti se zatim u podatke sesije korisnika. Generiše se pri kreiranju sesije korisnika nasumice string, koji predstavlja CSRF token. Taj token se zatim dodaje u sesiju korisnika i pri svakom HTTP odgovoru token se šalje pretraživaču korisnika. Pri svakoj operaciji korisnika se šalje zahtev sa ugrađenim tokenom u vidu skrivenog polja forme ili HTTP Request Header-a. Na serveru se proverava da li primljeni token odgovora uskladištenom tokenu iz sesije korisnika.

3. SQL Injection propusti u Security Hotspots odeljku

SLIKA	PROBLEM	OZNAKA	OPIS PROBLEMA
Slika8	SQL Injection - Komentarisanje filma	True Positive	True Positive iz razloga što se radi o SQL Injection napadu, ne konkateniraju se samo Integeri, već i String koji predstavlja komentar. Problem za aplikaciju!
Slika9	SQL Injection – Dohvatanje svih komentara za odabrani film	True Positive	U uskom smislu statičke analize True Positive, jer se konkatenira String, ali ovaj String predstavlja ID filma za koji se dohvataju komentari, koji je svakako broj, pa bi moglo da bude i False Positive. Napadač bi ipak mogao da podmetne neki svoj String, tako da, smatram da ovo ipak predstavlja sigurnosni propust i da je SQL Injection moguć, jer se radi konkatenacija Stringa, nebitno što taj String predstavlja broj. Svakako je lepše rešenje da se odradi sve sa PreparedStatement.
Slika10	SQL Injection – Pronalazak heširanog korisnika po username	True Positive	Ovo je svakako True Positive iz razloga što se konkatenira String. Rešenje za ovakav propust je svakako PreparedStatement.
Slika11	SQL Injection – Pretraga filmova	True Positive	Ovo je True Positive, bukvalno školski primer za SQL Injection i to baš na primeru na kome se zahtevalo da izvedemo SQLi napad u tekstu projekta.
Slika12	SQL Injection – Dohvatanje filma po ID	False Positive	Ovo je False Positive. Radi se konkatenacija broja (integera) a ne Stringa, pa u uskom smislu statičke analize ovo nije sigurnosni propust, ali se svakako moglo mnogo lepše uraditi preko PreparedStatement-a.
Slika13	SQL Injection – Dohvatanje filma po ID i njegovih žanrova	False Positive	Ovo je False Positive iz istog razloga kao što je navedeno za red iznad, Slika12.
Slika14	SQL Injection – Brisanje filma iz baze filmova po ID.	False Positive	Ovo je takodje False Positive iz istog razloga kao što je spomenuto za prethodna 2 primera, ali bih ja ovo svakako radio sa PreparedStatement-om iz razloga što je DELETE iskaz u SQL jeziku potencijalno opasan, pa radi dodatne zaštite bih se osigurao.
Slika15	SQL Injection – Brisanje komentara iz baze za film sa zadatim ID.	False Positive	Isto objašnjenje kao za 3 reda iznad, Slika14, Slika13 i Slika12.

Slika16	SQL Injection – Brisanje rejtinga za film sa zadatim ID.	False Positive	Isto objašnjenje kao za 4 reda iznad, Slika15, Slika14, Slika13 i Slika12.
Slika17	SQL Injection – Brisanje iz tabele movies_to_genres za film sa zadatim ID.	False Positive	Isto objašnjenje kao za 5 reda iznad, Slika16, Slika15, Slika14, Slika13 i Slika12.
Slika18	SQL Injection – Dohvatanje liste permisija na osnovu zadatog roleId.	False Positive	False Positive iz razloga što je roleId svakako Integer a ne String. Suštinski ne predstavlja sigurnosni propust, ali je svakako lepše uraditi sa PreparedStatement-om.
Slika19	SQL Injection – Pretraga osoba po zadatom parametru	True Positive	True Positive iz razloga što je parametar searchTerm String a ne Integer, a radi se obična konkatenacija parametra na prethodno formirani kostur SQL upita. Rešenje je da se uradi PreparedStatement.
Slika20	SQL Injection – Dohvatanje osobe po zadatom ID.	True Positive	True Positive iz razloga što je parametar tipa String. O ovome treba dodatno razmisliti zato što ako se uđe u smisao parametra, vidimo da je to identifikator osobe, tj. broj (integer) koji je konvertovan u String. Iz tog razloga mogao je da bude označen i kao False Positive, ali sam ja odabrao da je True Positive. Svakako, najbolje bi bilo da se i ovde radi sa PreparedStatement-om, jer tekuće rešenje nije baš najsrećnije.
Slika21	SQL Injection – Brisanje osobe po zadatom ID.	False Positive	False Positive. Parametar je integer, tj. broj a ne String. Nije moguće konverzijama broja stvoriti takav String koji bi predstavljao opasnost po ovaj DELETE SQL iskaz. Svakako možemo radi unapređene bezbednosti i ovde odraditi PreparedStatement, koji bi nam razrešio svaku buduću dilemu..
Slika22	SQL Injection – Ažuriranje osobe.	True Positive	True Positive. Ovde je moguć SQL Injection napad jer parametar Person personUpdate sadrži polja koja su po tipu String, pa je moguće izvesti SQLi napad pri ažuriranju osobe. Ovde je već rađeno sa PreparedStamenet-om, samo treba koristiti ? i onda setovati redom dinamičke delove SQL upita.
Slika23	SQL Injection – Kreiranje novog ili ažuriranje postojećeg rejtinga filma.	False Positive	False Positive. Ovde nije moguć SQL Injection napad jer je parametar tipa Rating, koji u sebi ne sadrži polja tipa String, vec isključivo celobrojna polja. Svakako moglo je da bude rađeno sa PreparedStatement-om.
Slika24	SQL Injection – Dohvatanje liste ocena	True Positive	True Positive. Stavljeno da je True Positive iz razloga što je parametar String. Ali ako se dublje

	za film sa prosleđenim movieId.		uđe u suštinu parametra vidimo da je to identifikator filma koji je sam po sebi ceo broj, pa bi moglo da bude i False Positive. Svakako možemo se zaštiti dodatno za svaki slučaj sa PreparedStatement-om, što bi razrešilo svaku buduću dilemu po pitanju sigurnosti ovog upita.
Slika25	SQL Injection – Dohvatanje liste rola za korisnika sa prosleđenim roleId.	False Positive	False Positive. Parametar je ceo broj (integer) a ne String, pa nema bojazni i realno opasnosti od SQL Injection-a.
Slika26	SQL Injection – Dohvatanje korisnika po prosleđenom korisničkom imenu.	True Positive	True Positive. Parametar posmatrane metode je String, a ne ceo broj (integer), pa je podložno SQL Injection napadu. Rešenje za zaštitu je da koristimo PreparedStatement.
Slika27	SQL Injection – Validiranje kredencijala (username i password)	True Positive	True Positive. Parametri su String-ovi, a ne celi brojevi, tj. integeri. Rešenje za obezbeđivanje potrebnog nivoa sigurnosti je svakako da se koristiti PreparedStatement.
Slika28	SQL Injection – Brisanje korisnika iz tabele users na osnovu prosleđenog userId.	False Positive	False Positive. Parametar je ceo broj, a ne String pa iz tog razloga nije moguc SQL napad. Svakako, lepse resenje je koristiti PreparedStatement.

4. INSECURE CONFIGURATION U SECURITY HOTSPOTS ODELJKU

SLIKA	PROBLEM	OZNAKA	OPIS PROBLEMA
Slika29 … Slika49	e.printStackTrace()	True Positive	Proglašeno da je True Positive, jer se ispisivanjem steka greške može saznati nešto o detaljima implementacije sistema, poput imena tabela, imena polja unutar tabela, konvencije imenovanja u bazi podataka koju koristimo, detalje poput klauzula u SQL upitima i tehnologijama koje koristimo i njihovim verzijama. Umesto toga, treba koristiti LOG klasu koja nam stoji na raspolaganju i njene metode.

5. OSTATAK SLIKA NAKON ANOTIRANJA PRETHODNO VIĐENIH PROPUSTA

SLIKA	OPIS	OBJAŠNJENJE SLIKE
Slika50	No more security hotspots	Poruka da aplikacija više nema neproverenih sigurnosnih tačaka, nakon što smo pregledali propuste i označili ih kao True Positive (Acknowledge) i False Positive (Safe).
Slika51	False Positive označeni propusti	Spisak propusta koji su označeni kao False Positive propusti.
Slika52	True Positive propusti, part 1	Prvi deo sigurnosnih propusta označenih kao True Positive.
Slika53	True Positive propusti, part 2	Drugi deo bezbednosnih propusta u aplikaciji označenih kao True Positive.
Slika54	True Positive propusti, part 3	Treći deo bezbednosnih propusta u aplikaciji koji su označeni kao True Positive.
Slika55	Bagovi su ostali nakon anotiranja sigurnosnih propusta u aplikaciji.	Nakon popravke sigurnosnih propusta, ostali su samo bagovi u aplikaciji. SonarQube je vrlo moćan alat koji pronalazi bagove, objašnjava posledice do kojih bagovi mogu dovesti i nudi način za popravku pronađenih bagova i sigurnosnih propusta. Ovde se vidi da je Quality Gate status FAILED, iz razloga sto je anotacija sigurnosnih propusta rađena kada je prvobitno projekat bio označen kao FAILED. Ponovnim pokretanjem aplikacije bi se videlo da je PASS.*

*Ovo se verovatno desilo desilo iz razloga što je SonarQube nad ovogidišnjim projektom bio pokrenut pomoću tokena i pod istim nazivom kao projekat koji je korišćen na vežbama. Kada sam obrisao prethodni projekat (Administration -> Projects -> Projects Management -> Select the Project(s) you want to DELETE -> Click on Delete button at top right corner) i promenio token koji koristim a zatim pokrenuo i SonarQube Analysis Run konfiguraciju, projekat je PASS-ovao. Suština pri određivanju svakako ostaje ista što se tiče toga da li je nešto True Positive ili False Negative.