# Univerzitet u Beogradu

# Elektrotehnički fakultet



## “Improper Neutralization of Special Elements used in an SQL Command”

## Seminarski rad iz predmeta Razvoj bezbednog softvera

|  |
| --- |
| Student: |
| Predrag Mitrović 2019/3173 |

Beograd, Maj 2020.

# Uvod u SQL Injection

SQL Injection je jedan od najčešćih napada na softverske sisteme, tipično web aplikacije. Svaka aplikacija koja koristi SQL bazu podataka za skladištenje informacija i ima propuste koji omogućavaju napadaču da iskoristi sigurnosne propuste može biti uspešno napadnuta. Napad se zasniva na podmetanju delova ili čitavih SQL komandi koje imaju za cilj da manipulišu bazom podataka. Napadač bi mogao da pokuša da dođe do podataka iz baze kojima ne bi smeo da ima pristup, da vrši izmenu podataka, da obriše podatke, tabele ili čak čitavu bazu.

# Tipovi napada

Postoji nekoliko tipova SQL Injection napada:

* In band SQLi

Napadač koristi isti kanal komunikacije kako bi izvršio napad i kako bi prikupio rezultate napada. Ukoliko sistem ne rukuje pravilno sa izuzecima, napadač može izvršiti akcije koje dovode do greške pri izvršenju upita u bazi podataka i na taj način doći do informacija koje će mu biti od koristi u budućnosti. Takođe, može koristiti i UNION operator kako bi došao do bitnih informacija.

* Blind SQLi

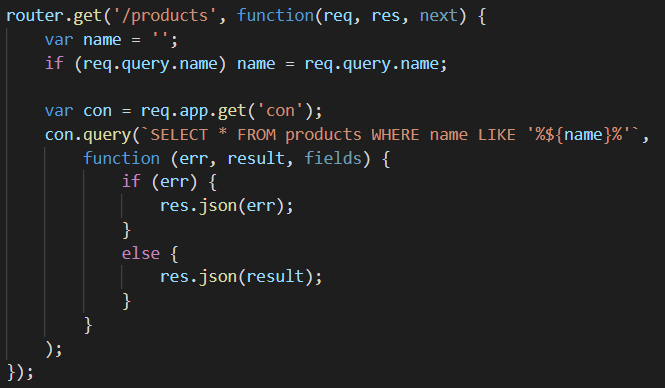
Napadač šalje podatke ka serveru i analizira odgovor koji je dobio kako bi došao do zaključaka o strukturi baze podataka. Server ne šalje podatke iz baze podataka ka napadaču, ali svojim ponašanjem implicitno daje odgovore na pitanja od interesa. Napadač može kreirati HTTP zahtev ka serveru koji u sebi sadrži određeni upit. Odgovor koji dobije od servera će sadržati tražene podatke koje će napadač zanemariti, ali u zavisnosti od odgovora napadač može dobiti odgovor (tačno ili netačno) na upit koji je poslao. Moguće je i da upit sadrži funkciju koja čini da baza odloži odgovor nekoliko sekundi u zavisnosti od toga da li je upit tačan ili netačan. Napadač može doći do zaključaka na osnovu vremena koje je proteklo od kada je poslao zahtev.

* Out of band SQLi

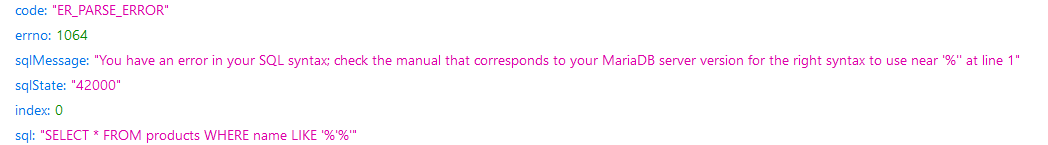
Napadač može izvršiti ovu vrstu napada samo ako su određene funkcionalnosti omogućene na serveru baze podataka. Izvodi se u slučajevima kada je server previše spor ili nestabilan. Ova tehnika se oslanja na kapacitet servera da kreira DNS ili HTTP zahteve za prenos podataka ka napadaču.

# Primer napada

Pretpostavimo da vebsajt ima get metodu za dohvatanje svih proizvoda. Kod je dat u nastavku.



Kako nije izvršena sanitizacija korisničkog unosa i unos je direktno ugrađen u sql upit, postoje preduslovi za uspešan napad. Sve što je neophodno za početak napada jeste da se pozove get metoda na zadatoj putanji i da joj se kao query parametar zada karakter ’. Dobijeni odgovor je dat u nastavku.



Iz odgovora možemo zaključiti da je u pitanju MySQL baza podataka i možemo videti koji upit je izazvao grešku. Pažljivim odabirom query parametra koji na kraju ima oznaku za komentar -- moguće je izvršiti upite u nastavku i doći do bitnih informacija:

1. Otkrivanje naziva baze podataka

SELECT \* FROM products WHERE name LIKE '%%' order by 5;

SELECT \* FROM products WHERE name LIKE '%%' union select 1,2,3,4,5 from users;

SELECT \* FROM products WHERE name LIKE '%%' union select user(),version(),database(),@@GLOBAL.have\_ssl,5 from users;

1. Otkrivanje naziva tabela koje se nalaze u bazi

SELECT \* FROM products WHERE name LIKE '%%'; select \* from information\_schema.tables where table\_schema='sqli’;

1. Otkrivanje poverljivih informacija o narudžbinama

SELECT \* FROM products WHERE name LIKE '%%'; select \* from orders;

# Posledice napada

Ova vrsta napada može imati negativne posledice u vidu otkrivanja osetljivih podataka, kreiranja lažnih podataka, izmene i brisanja postojećih podataka. Može doći i do zaobilaženja kontrole pristupa u smislu pristupa aplikaciji bez logovanja na sistem ili pristupa delovima sajta za koje korisnik nije autorizovan.

# Sprečavanje i ublažavanje posledica

Neke od preporuka za sprečavanje i ublažavanje posledica su:

* Koristiti softvere koji detektuju potencijalne slabosti sistema na SQL Injection kako bi propusti bili uočeni i otklonjeni.
* Ukoliko su dostupni, preporuka je koristiti mehanizme koji automatski forsiraju razdvajanje između podataka i koda. Nikada ne bi trebalo dinamički konstruisati upite konkatenacijom stringova već bi trebalo koristiti parametrizovane upite i uskladištene procedure koje primaju parametre ili varijable i omogućavaju strogu tipizaciju. Moguće je koristiti bibiloteke i radne okvire koje pružaju jaku zaštitu od ovog napada ukoliko se pravilno koriste kao što su Hibernate ili Entity Framework.
* Pretpostaviti da je svaki korisnički unos potencijalno maliciozan. Umesto uvođenja ograničenja šta korisnik ne sme da unese, bolje je prihvatati samo unos koji sadrži karaktere koje je dozvoljeno da unese. Odbiti sav unos koji odudara od specifikacije. Koristiti šablone tamo gde se traži specifičan tip podataka (datum, broj telefona…).
* Obezbediti pravilno rukovanje izuzecima. Vršiti obradu grešaka na serveru i vratiti odgovor koji ne otkriva informacije o internim greškama korisniku. Prosečnom korisniku biznis greške ne znače mnogo, a potencijalnom napadaču svaka informacija koju dobije može biti od značaja.
* Ograničiti prava pristupa bazi podataka. Obezbediti samo neophodne dozvole korisnicima baze podataka.