Pregledni rad
https://doi.org/10.31784/zvr.8.1.10
Datum primitka: 25. 11. 2019.
Datum prihvaćanja: 8. 1. 2020.

NAPADI UBACIVANJEM SQL IZRAZA – PREVENCIJA I OBRANA

Nikolina Ljubičić
Univ. bacc. inf, studentica, 51 000 Rijeka, Hrvatska; *e-mail*: ljubicic.nikolina@gmail.com

Danijela Jakšić
Dr. sc., docentica, Odjel za informatiku, Sveučilište u Rijeci, Radmile Matejčić 2, 51 000 Rijeka, Hrvatska; *e-mail*: danijela.jaksic@inf.uniri.hr

Patrizia Poščić
Dr. sc., redovita profesorica, Odjel za informatiku, Sveučilište u Rijeci, Radmile Matejčić 2, 51 000 Rijeka, Hrvatska; *e-mail*: patrizia@inf.uniri.hr

SAŽETAK

Napad ubacivanjem SQL izraza jedna je od najozbiljnijih prijetnji sigurnosti aplikacija zasnovanih nad bazom podataka. Ona omogućuje napadaču da stekne kontrolu nad bazom podataka aplikacije, čime napadač dobiva mogućnost izmjene podataka. Mnoga se istraživanja usmjeravaju na ovaj problem. Brojni su autori predložili različite pristupe za otkrivanje i sprječavanje ove ranjivosti, no koliko je ovaj problem i dan danas prisutan govori i činjenica da ni ti pristupi ne predstavljaju u potpunosti uspješno rješenje. U ovom radu opisano je nekoliko vrsta napada ubacivanjem SQL izraza te različiti alati i metode koji mogu pružiti određenu prevenciju i obranu od tih napada. Predložene su različite metode za rješavanje problema napada ubacivanjem SQL izraza u obliku opsežnih pregleda varijacija napada ubacivanjem SQL izraza te njihovih opisa i primjera, kako bi se bolje spoznao njihov štetan utjecaj i krajnje posljedice. Štoviše, u radu je dan i kratak pregled prijedloga pojedinih autora vezanih uz prevenciju i obranu od napada ubacivanjem SQL izraza. Na kraju je iznesen zaključak uz objektivan pregled i analizu cjelokupnog istraživanja. Glavni doprinos rada je pregled dosadašnjih istraživanja i pristupa vezanih uz: a) prevenciju napada ubacivanjem SQL izraza i b) obranu od napada ubacivanjem SQL izraza.

Ključne riječi: *napad ubacivanjem SQL izraza, prevencija, obrana, napad, tehnike*

1. UVOD

U današnje vrijeme su Internet i web tehnologije postale vrlo bitne stavke našeg društva, čime se javlja potreba za njihovim stalnim rastom i razvitkom. Samim time povećava se broj poslovnih

aplikacija koje sadrže osobne, financijske i ostale vrste podataka, a povezane su na Internet, ili ga koriste kao temelj svojeg svakodnevnog rada. Sukladno tome povećao se i broj napada ubacivanjem SQL izraza, odnosno dovodi se u pitanje sigurnost njihove baze podataka. Napad ubacivanjem SQL izraza predstavlja napad u kojem se zlonamjeran kod umeće u nizove koji se kasnije prosljeđuju SQL poslužitelju za raščlanjivanje i izvršavanje. Ako pri izradi aplikacije postoji nedostatak odgovarajućeg znanja ili nerazumijevanje određenih segmenata ona time postaje podložnija napadima ubacivanjem SQL izraza. Određeno potencijalno rješenje može biti učinkovito u sadašnjoj upotrebi, ali se zbog promjene tehnologije pojavljuju novi rizici i izazovi. Štoviše, moraju se kombinirati različita rješenja učinkovitosti protiv različitih vrsta napada i sigurnosti te zbog toga sustav treba biti pod stalnim nadzorom.

Postoji više načina da programer, odnosno administrator sustava, spriječi napade na svoje sustave. Pri tome se koriste brojne tehnike u sustavima koji imaju različite primjene, kao što su: provjera unosa, najmanje privilegije, prilagođena poruka o pogrešci i slično. Iako ove tehnike predstavljaju najbolji način za prevenciju i obranu od ranjivosti napada ubacivanjem SQL izraza, njihova primjena je problematična u praksi. Te su tehnike sklone ljudskim pogreškama te stoga nisu jednako učinkovite kao automatizirane tehnike, unatoč trudu programera da primjene defanzivno kodiranje prema svim pravilima - uvijek postoji mogućnost da se potkradu greške. Stoga su istraživači predložili niz tehnika i pristupa da pomognu razvojnim inženjerima nadoknaditi nedostatke prilikom kodiranja.

2. NAPAD UBACIVANJEM SQL IZRAZA

Napad ubacivanjem SQL izraza odnosi se na slabost ili nedostatak u sustavu, koji se iskorištava na način da neovlašteni korisnik pristupa pohranjenim podacima. Napad općenito predstavlja ilegalni pristup koji je stečen dobro izrađenim mehanizmima. Napad ubacivanjem SQL izraza je vrsta napada prilikom kojeg napadač dodaje zlonamjerne ključne riječi ili operatore u SQL upite. To napadaču omogućuje ilegalan i neograničen pristup pohranjenim podacima u bazi podataka (Kindy i sur., 2012). Napadači su razvili široku lepezu sofisticiranih tehnika napada koje se koriste prilikom iskorištavanja SQL ranjivosti (Halfond, Orso, 2008).

SQL napadi se najčešće događaju kada programer web aplikacije nije osigurao da se primljene vrijednosti iz web obrasca, kolačića, ulaznog parametra i slično zaštite. Parametre je potrebno provjeriti prije prosljeđivanja SQL upitima koji će se kasnije izvršiti na poslužiteljima baze podataka (Clarck, 2012).

U svrhu izbjegavanja napada potrebno je dobro poznavanje temeljne baze podataka s kojom su programeri u interakciji te dubinsko razumijevanje i svijest o mogućim sigurnosnim pitanjima. U suprotnome programeri aplikacija često mogu razviti inherentno nesigurne aplikacije koje su osjetljive na napade ubacivanjem SQL izraza. Svaka aplikacija je različita pa samim time treba biti svjestan da je na taj način i svaka SQL točka ubrizgavanja jedinstvena (Clarck, 2012).

2. 1 Primjeri napada ubacivanjem SQL izraza

Teško je prikupiti točne podatke o tome koliko je organizacija bilo ugroženo i napadnuto putem napada ubacivanjem SQL izraza kroz povijest. Povrede sigurnosti i uspješnost napada izvršenih od strane zlonamjernih napadača su zasigurno omiljena tema svjetskih medija. Štoviše, ova tema je postala jako publicirana. Čak i najmanji slučajevi napada neće proći nezapaženo široj javnosti, dok se nekad napadi nisu toliko objavljivali (Clarck, 2012). Međutim, poznato je kako će velike tvrtke zasigurno taj problem pokušati, pa na kraju čak i uspjeti, sakriti od javnosti.

Ovdje su prikazani samo neki od javno dostupnih izvora gdje su injekcijski napadi prouzročili probleme:

- U prosincu 2006. godine američki diskontni trgovac TJX uspješno je napadnut i napadači su ukrali milijune podataka o platnim karticama iz TJX baza podataka
- U kolovozu 2007. godine napadnuto je web-mjesto Ujedinjenih naroda umetanjem SQL izraza kako bi se prikazale anti-američke poruke
- U srpnju 2008. godine napadnuta je Malezijska stranica antivirusnog alata „Kaspersky" umetanjem SQL koda
- U lipnju 2011. godine napadnuta je web stranica pjevačice Lady Gage umetanjem SQL koda. Napadom su prikupljeni podaci o obožavateljima kao što su: njihove e-mail adrese, brojevi telefon te adrese mjesta stanovanja. Smatra se kako su ti podaci iskorišteni za daljnje napade
- U ožujku 2011. godine napadnuta je web stranica „mysql.com" umetanjem SQL koda (Clarck, 2012)

Postoji veliki broj slučaja gdje napadači umeću SQL izraze kako bi došlo do napada na bazu podataka, ovdje su navedeni neki najčešći primjeri.

Napadači imaju cilj zaobići izvršavanje pravilne prijave. Oni su dužni upisati svoje korisničko ime i lozinku, ali to rade na način da u polje, na primjer, unose izraz 'OR '1' = '1, čime se stvara sljedeći SQL kod:

SELECT Ime FROM Korisnik

WHERE Ime = ' ' OR '1' = '1'

AND Lozinka = ' ' OR '1' = '1'

Nakon takvog upita, više se ne uspoređuju podaci koje je predao korisnik sa onima u bazi podataka, već upit uspoređuje samo istinitost relacije, odnosno provjerava je li tvrdnja '1' = '1' istinita. Rezultat toga je uvijek istinit. Samim time svi uvjeti u WHERE dijelu postaju zadovoljeni te se vraća prvi redak u određenoj tablici.

Postoje i drukčiji načini da napadač unese sličan niz znakova, no učinak za napadača i posljedica za bazu je ista. Moguće je umetnuti sljedeći niz znakova: ' OR 'x' = 'x , ' OR' ' = ' i OR 1=1. Valja napomenuti kako je uz spomenute nizove moguće napraviti još nekoliko dodatnih varijanti. Ovaj napad je utemeljen na tautologiji.

Postoji mogućnost da napadač zagasi bazu podataka naredbom SHUTDOWN. Kako bi napadač u tome uspio u određeno polje za unos korisničkog imena i lozinke napadač upisuje: *'; SHUTDOWN – –* . Nakon toga dobiva sljedeći SQL kod:

SELECT Ime FROM Korisnik

WHERE Ime = ' ';

SHUTDOWN – –

'AND Lozinka = 'password'

Također, postoji i naredba koja briše cijelu bazu podataka sa svim podacima. U slučaju kada napadač pronađe način da ubaci naredbu DROP TABLE tada dolazi do uništenja cijele baze podataka. Kako bi to uspio napadač unosi u polje sljedeće podatke: ' ; DROP TABLE Korisnik – – . Nakon čega se dobiva sljedeći SQL kod koji je skoro pa identičan u odnosu na prethodno spomenuti kod:

SELECT Ime FROM Korisnik

WHERE Ime = ' ';

DROP TABLE Korisnik – –

'AND Lozinka = 'password'

Ovdje su prikazani primjeri samo nekih najčešćih oblika napada. (Clarke, 2012).

Iz navedenih primjera se može primijetiti kako niti jedna stranica nije u potpunosti zaštićena od SQL napada.

2. 2 Vrste SQL napada

Nije lako odrediti i kategorizirati sve vrste SQL napada. Isti napadi mogu se u određenim slučajevima zvati različitim nazivom, ovisno o scenariju sustava. Postoje različite metode napada koje ponajviše ovise o cilju napadača. U ovome poglavlju predstavit će se samo najčešće. U sljedećoj tablici prikazan je kratak opis određene vrste te njezina svrha.

Tablica 1. Vrste SQL napada

Vrsta SQL napada	**Opis**	**Svrha**
Tautologija	Ubrizgavaju se u jednu ili više uvjetnih naredbi te se uvijek ocjenjuju kao istiniti (Halfond i Orso, 2008)	Određivanje injekcijskih parametara, zaobilaženje autentifikacija te izdvajanje podataka (John i sur., 2012)

Union upit	Ova vrsta napada se izvršava umetanjem ključnog parametra „UNION" (Parameswari, Kavitha, 2018)	Zaobilaženje provjere autentičnosti i izdvajanje podataka (Parameswari, Kavitha, 2018)
Piggy backed upit	Kod ovog tipa napada zlonamjerni napadač ubacuje dodatne upite u izvorni upit (Fouad, Elshazly, 2013)	Izdvajanje podataka te izmjenjivanje skupa podataka (Fouad, Elshazly, 2013)
Logički neispravan upit	Ovaj napad se smatra preliminarnim korakom za daljnje napade. Šalju se pogrešni upiti u bazu podataka nakon čega poslužitelj baze podataka vraća poruke o pogrešci koje napadač iskorištava za daljnje napade (Fouad, Elshazly, 2013)	Određivanje injekcijskih parametara, identificiranje baze podataka te izdvajanje podataka (Fouad, Elshazly, 2013)

Svim napadima je zajednički cilj prikupiti što više osobnih informacija kako bi se ti podaci mogli dalje iskorištavati.

3. PREVENCIJA

SQL napadi mogu prouzročiti iznimne štete. Stoga je potrebno poduzeti pravodobne mjere zaštite prije nego što do napada uopće dođe. To se postiže prevencijom koja predstavlja skup mjera i postupaka kojim se sprječava potencijalni napad. Prevencija traži način na koji bi se izbjegao potencijalni napad.

Prevencija se postiže provjeravanjem ulaznih podataka što ujedno predstavlja najjednostavniji način prevencije. Programer može onemogućiti ulaz određenih znakova i na taj način se aplikacije mogu zaštiti od mogućih napada. Međutim, mnogi programeri ignoriraju ovu vrstu provjere (Zhang i sur., 2011).

Poseban problem koji se pojavljuje je unošenje meta znakova, jer ih napadači zlonamjerno iskorištavaju. Zabrana takvih znakova nije nužno rješenje. Bolje rješenje bilo bi korištenje funkcija koje kodiraju cjelinu tako da se svi meta znakovi posebno šifriraju i interpretiraju od strane baze podataka. Programeri nisu uvijek u mogućnosti provjeravati sve nevažeće unose pa je s te strane potrebno odrediti one ulaze koji su zakoniti upravo uz pomoć tih funkcija. Ovakav pristup u kojem dolazi do provjere valjanosti unosa naziva se pozitivno podudaranje uzoraka (Halfond i sur., 2006).

Funkcije i procedure koje će zasigurno pomoći pri prevenciji od napada ubacivanjem SQL izraza navedene su u nastavku.

QUOTENAME čini funkciju koja se koristi za izbjegavanje napada ubacivanjem SQL izraza na način da se varijable pročišćuju od neželjenih znakova te na taj način štite podatke izložene napadima. Ova funkcija predstavlja ugrađenu funkciju SQL Servera te sadrži neka ograničenja kao što je primjerice to da ulazni niz znakova funkcije QUOTENAME može imati samo 128 znakova, a ukoliko je dužina znakova veća od 128 znakova vraća se NULL (Khan, 2005).

Proširena pohranjena procedura *sp_executesql* također može smanjiti broj napada ubacivanjem SQL izraza, odnosno zaštititi od napada. Spomenuta procedura uzima tip podataka u obliku niza (engl. *string*) konstante ili varijable kao SQL izraz za izvršavanje. Također, *sp_executesql* nudi još jednu prednost, a to je da može odrediti svoje parametre odvojeno od izjave. Bitno je istaknuti kako *sp_executesql* parametrizacija na strani klijenta štiti od napada i pruža prednosti performanse u pred memoriranju upita pri ponovnom izvođenju (Coles, 2007).

Postoji mogućnost da do napada dođe čak i nakon primjene obrambenih praksi kodiranja. Prije svega to se događa jer je teško obuhvatiti sve izvore ulaza (engl. *input*) u obrambenim praksama kodiranja. Takvo kodiranje sklono je ljudskoj pogrešci pa samim time nije precizno i potpuno kao primjenjivanje nekih automatiziranih tehnika (Mavromoustakos, 2016).

Napadi ubacivanjem SQL izraza su ozbiljna prijetnja za web baze podataka, a posebno u današnje vrijeme kad računarstvo, pohrana i aplikacije u oblaku postaju sve popularnije i korištenije. Kako bi se riješio ovaj problem, u radu (Haixia, Zhihong, 2009) prikazana je shema testiranja sigurnosti baze podataka. Ona proučava kako otkriti potencijalne ulazne točke napada ubacivanjem SQL izraza, automatski generirati testne slučajeve i pronaći ranjivost baza podataka.

Autori (Haixia, Zhihong, 2009) predložili su i nekoliko metoda, kao što su: otkrivanje mogućih ulaznih točaka napada ubacivanjem SQL izraza, automatsko generiranje testnih slučajeva te konačno pronalaženje ranjivosti na bazama podataka pokretanjem testnih slučajeva za simulaciju napada na web-aplikaciju. Predložena shema se pokazala kao učinkovita jer točno prepoznaje ulazne točke injekcije u očekivanom vremenu.

Međutim, nakon analize sheme, otkriveno je kako pristup ne daje potpuno rješenje, već su potrebna dodatna poboljšanja u dva glavna aspekta: sposobnost otkrivanja i razvoj pravila napada.

4. TEHNIKE I MEHANIZMI PREVENCIJE

Postoje razne metode, tehnike i algoritmi koji se primjenjuju kako bi se spriječilo da do napada uopće dođe. Sukladno tome koriste se određeni mehanizmi prevencije koji su opisani u ovom poglavlju.

4. 1 Usporedba triju algoritama

U ovom djelu analizirana su tri najpoznatija algoritma:

Algoritam podudaranja uzoraka: U ovom algoritmu uobičajene SQL naredbe za ubrizgavanje se sastoje od posebnih znakova kao što su: {}, [], (), &, +, ‘, =, <>, < = >. U tu skupinu spadaju i ključne riječi kao što su: ažuriranje, brisanje, ispuštanje, sjedinjavanje te Boolean znakovi kao što su:

AND i OR. Korisnički unos se zatim uspoređuje s iznad navedenim propisima kako bi se provjerila ranjivost s obzirom na napade ubacivanjem SQL izraza. Rezultati ove metode nisu uvijek korisni te ponekad može doći do pogrešne procjene ranjivosti (Senthilkumar i sur., 2017).

Algoritam za provjeru valjanosti stabla raščlanjivanja: Stablo raščlanjivanja sadrži različite čvorove za svaki segment upita. Ono je raščlanjeno prikazivanjem SQL upita. Kako bi se izvršila ova operacija potrebno je poznavanje gramatike jezika. Nakon generaliziranja i stvaranja novog upita dobivaju se stabla koja se mogu usporediti kako bi se utvrdilo sadrže li oba upita istu strukturu. Stablo s ispravnim korisničkim unosima ispunjava prazne čvorove na stablu. Ako dođe do neispravnih korisničkih unosa, stablo koje se tada izradilo razlikuje se od onog s ispravnim unosima te se time potvrđuje njegova ranjivost (Senthilkumar i sur., 2017).

Kriptografija: Ovaj algoritam ima za cilj pohranjivanje korisničkih ulaza i ostalih podataka u šifriranom obliku obrasca u bazi podataka. Ovom metodom šifrirani oblik unosa ulazi u upit za usporedbu s bazom podataka i time sprječava injekciju. Čak i ako napadač kojim slučajem uspije pristupiti podacima, oni će biti u šifriranom obliku. Samim time podatke se ne može dešifrirati ako napadač ne poznaje korišteni način šifriranja i ključ za šifriranje (Senthilkumar i sur., 2017).

Hibridni algoritam: Ovaj algoritam sastoji se od najboljih značajki prethodno navedenih algoritama. Korisnički unosi se najprije uspoređuju s nizom uzoraka za provjeru ranjivosti prijetnje. Ako su uzorci prisutni potrebno je raščlaniti SQL upite na stabla, prije i poslije dodavanja ulaza te ih zatim usporediti. Vrijednosti se dešifriraju prije prikazivanja korisniku (Senthilkumar i sur., 2017).

Svaki od prethodno spomenutih algoritama koristi se za otkrivanje i prevenciju SQL ranjivosti na temelju korisničkih ulaza. Spomenuti algoritmi nisu uvijek učinkoviti te su skloni pogreškama. Stoga se preporučuje korištenje hibridnog algoritma. Ovaj algoritam provodi daljnje provjere nakon što jedan algoritam otkrije ranjivost. Zatim obavlja provjeru valjanosti stabla i kriptografije u svrhu poboljšanja učinkovitosti validacije, ako su ulazi osjetljivi na napade ubacivanjem SQL izraza. (Senthilkumar i sur., 2017).

Ispod je dana tablica koja prikazuje prethodno spomenute algoritme. Iz tablice je vidljivo kako se najveća efektivnost postiže upravo Hibridnim algoritmom, a pri čemu je algoritam za podudaranje uzoraka najmanje efikasan (Senthilkumar i sur., 2017).

Tablica 2. Prikaz efikasnosti određenih algoritama

SQLi Tehnike\ Algoritmi	**Podudaranje uzoraka**	**Validacija stabala raščlanjivanja**	**Kriptografija**	**Hibridni Algoritam**
Tautologija	Manje efikasno	Jako efikasno	Efikasno	Jako efikasno
UNION	Efikasno	Efikasno	Efikasno	Jako efikasno
Komentar	Efikasno	Jako efikasno	Efikasno	Jako efikasno

Izvor: Senthilkumar, Reddy(2017)

U nastavku je opisano još nekoliko preventivnih tehnika:

PSIAQOP: Ova metoda služi za sprječavanje napada ubacivanjem SQL izraza temeljenih na procesu optimizacije upita. Tehnika se zasniva na optimizaciji SQL upita koji dobivaju ulazne podatke od korisnika te su ujedno ranjivi u fazi izvođenja. PSIAQOP analizira izvorni kôd koji se koristi u web aplikacijama i identificira točke koje predstavljaju ranjive upite. Nakon toga ranjivi upiti prolaze kroz proces optimizacije koji se temelji na heurističkim pravilima. U ovom slučaju, analizator upita najprije generira inicijalno predstavljanje ranjivih upita koji je uglavnom u oblicima relacijskog izraza algebre kojim upravlja mehanizam optimizacije. Pogon optimizacije zatim generira broj važećih izvedbi u skladu s heurističkim pravilima za odabir optimalnog plana. Konačno, PSIAQOP zamjenjuje svaku točku pristupne točke optimiziranim upitom u kodu web aplikacije. Web aplikacija tada normalno pokreće svoj kôd te sprječava napade ubacivanjem SQL izraza u postizanju štetnih ciljeva (Lawal i sur., 2016).

SQLProb: Arhitektura koja je utemeljena na proxy poslužitelju kao cilj sprječavanja SQL napada. Ova metoda koristi se u operativnoj pozadini kako bi zaštitila back-end i front-end web servise. Tehnika dinamički prepoznaje i uklanja zlonamjerne SQL strukture sa korisničkih ulaza (Liu i sur., 2009).

X-LOG: Tehnika za provjeru autentičnosti kako bi se spriječili napade ubacivanjem SQL izraza. U ovoj se metodi prijetnja napada ubacivanjem SQL izraza provjerava tijekom izvođenja (engl. runtime). Ova metoda nadgledava dinamički stvorene upite s modelom podataka koji je kreiran od strane X-Log generatora tijekom izvođenja. U slučaju kada sličnost podataka nije ista model prikazuje potencijalni SQL napad. Tada se zaustavlja izvršavanje i izvještavanje na bazi podataka (Indrani, Ramaraj, 2011).

ASCII tehnika: Učinkovita tehnika za detekciju i prevenciju od napada ubacivanjem SQL izraza koja koristi ASCII znakove. Tehnika kombinira statičku i dinamičku analizu. Prevencijske metode su podijeljene u nekoliko faza:

a. Zamjenski znak koji je korišten od strane napadača i zaustavljen detektorom za tekst.
b. Postoji ograničenje za prijavu kao i definirani broj znakova koji se može koristiti za prijavu i unošenje lozinke.
c. ASCII vrijednost se stvara u bazi podataka za osjetljive podatke kao što su: korisnički ID ASCII i ASCII lozinka za zaštitu baze podataka (Balasundram, Ramaraj, 2012).

Amnesia: Ova metoda je potpuno automatizirana u sustavu zaustavljanja SQL napada. Prepoznaje upite prije nego što se krenu izvršavati. Oni su prethodno prepoznati pomoću pristupa koji je temeljen na modelu, a sastoji se od statičkog dijela u kojem se automatski gradi model legitimnog upita pomoću analize programa. Drugi dio od kojeg se model sastoji je dinamički dio u kojem se pregledava i provjerava autentičnost upita koji su dinamički stvoreni sa statičkim modelom kao pomoć pri praćenju izvođenja. Blokiraju se oni upiti za koje se misli da su zlonamjerni. Tehnika se sastoji od nekoliko faza: prepoznavanje hotspota, stvaranje SQL-upita, primjena instrumenata, provjera tijekom izvođenja i nadzor (Kumar, Pateriya, 2012), (Johari, Sharma, 2012). U (Junjin, 2009) Junjin predlaže korištenje ove metode za praćenje SQL ulaznih tokova kao i za generiranje testnih slučajeva.

CANDID: Metoda dinamičke ocjene kandidata za automatsko sprječavanje napada ubacivanjem SQL izraza. Ovaj okvir dinamički izdvaja strukture upita te rješava problem vezan uz ručno mijenjanje aplikacije za izradu pripremljenih izjava. Pokazalo se kako je ova metoda učinkovita samo u nekim slučajevima. Na primjer, neučinkovita je kada se suočava sa vanjskim funkcijama i kada se primjenjuje na pogrešnoj razini. Osim toga, metoda ponekad ne uspijeva zbog ograničene sposobnosti sustava. Bisht i suradnici (Bisht i sur., 2010) predlažu metodu CANDID, prvenstveno iz razloga što rješava problem ručne izmjene aplikacije za izradu pripremljenih izjava.

SQL Dom: Ova tehnika koristi sve razine sučelja (CLI) koje se nalaze na sredini aplikacije i njene baze podataka. Ovo rješenje koristi API koji analizira baze podataka. API određuje ispravnu validaciju ulaznih podataka koja se provjerava kod kompilacije (Kumar, Pateriya, 2012), (Tajpour, 2011), (Amirtahmasebi, 2009).

Tehnike se uglavnom fokusiraju na prepoznavanje prepreka u interakciji s bazom podataka putem CLI-a. SQL DOM je predloženo rješenje za njihovo rješavanje sigurnog okuženja za komunikaciju. Kvalitetna procjena ovog pristupa je pokazala mnoge prednosti u smislu pogreške otkrivanja tijekom vremena kompiliranja, pouzdanosti, testiranja i održavanja. Iako se ovaj mehanizam smatra učinkovitim, on svejedno može biti dodatno poboljšan s više naprednih i novijih alata (Kumar, Pateriya, 2012).

SANIA: Sintaktička i semantička analiza za automatizaciju testiranja protiv napada ubacivanjem SQL izraza, odnosno Sania. Sania predstavlja tehniku za otkivanje SQL napada na web aplikacijama koje su u razvoju te posljedično otklanja pogreške pomoću sljedećih pet postupaka:

a. Sania dohvaća SQL upite između web aplikacije i baze. Zatim prikuplja pravilne SQL upite između klijenta i web aplikacije te između web aplikacije i baze podataka pri čemu analizira ranjivost.
b. Automatski generira i razrađuje napade prema sintaksi i semantici potencijalno osjetljivih mjesta u SQL upitima.
c. Nakon napada se s generiranim kodom sakupljaju stvoreni SQL upiti.
d. Sania uspoređuje stabla raščlanjivanja planiranog SQL upita i onih koji su nastali nakon napada kako bi procijenila sigurnost tih mjesta.
e. Konačno, Sania određuje je li napad uspio ili nije.

Analizirajući sintaksu stabla raščlanjivanja SQL upita, moguće je generirati precizne zahtjeve za točnim napadom (Sajjadi, Pour, 2013).

AIIDA-SQL: Ova metoda sugerira hibridni pristup temeljen na adaptivnoj inteligentnoj detekciji za otkrivanje napada ubacivanjem SQL izraza. AIIDA-SQL kombinira prednosti CBR (engl. *Case-Based Reasoning*) sustava utemeljenog na učenju te prilagodbi s mogućnostima kombinacije umjetne neuronske mreže (engl. *Artificial Neural Network*) i stroja za vektorsku potporu (engl. *Support Vector Machine*). Kroz te mehanizme uzimaju se prednosti obje strategije kako bi klasificiranje SQL upita bilo što pouzdanije. Konačno, kako bi se SQL upiti klasificirali kao nepovjerljivi, koristi

se mehanizam virtualizacije koja kombinira tehnike klasteriranja i neuronskih modela koji su bez nadzora (Sajjadi, Pour, 2013).

Kombinatorni pristup za sprečavanje napada ubacivanjem SQL izraza: Ova tehnika koristi inovativan visoko automatiziran pristup za sprečavanje napada ubacivanjem SQL izraza na web aplikacijama. Ovaj se pristup sastoji od tri koraka:

- Analiziranje točaka iz aplikacije
- Zaštita koda web aplikacije statičkom analizom i provjerom tijekom izvođenja koristeći Hirschberg algoritam za otkrivanje napada ubacivanjem SQL izraza

SUBP (sustav za upravljanje bazom podataka) zatim se koristi pri metodi revizije kako bi se otkrile transakcije. Hircbergov algoritam koristi podjelu te smanjuje vrijeme i prostor za otkrivanjem SQL prijetnji ubrizgavanja. Podržava autentifikaciju korisnika kako bi zaustavilo izvršenje napada ubacivanjem SQL izraza nakon analize SUBP-a (Ezumalai, Aghila, 2009).

SAFELI: Ovo je okvir statičke analize dizajna koji se temelji na prepoznavanju ranjivosti napada ubacivanjem SQL izraza tijekom kompilacijskog vremena. Loša strana ove tehnike je da detektira samo napade ubacivanjem SQL izraza koji se provode posebno na Microsoftovim proizvodima (Fu, Qian, 2008).

SWADDLER: Ova tehnika ispituje stanje internih web aplikacija. Na vrlo dojmljiv način pokazuje otpornost na složene napade na web aplikacijama korištenjem jedne ili čak više varijabli. Tehnika u početku opisuje normalne vrijednosti varijable stanja aplikacije u kritičnim točkama komponenti aplikacije. Kasnije, tijekom faze detekcije pristup prati primjenu nad aplikacijama za identificiranje abnormalnih stanja. Nedostatak ovog pristupa je taj da ponekad djelomično otkriva napade ubacivanjem SQL izraza (Cova i sur., 2007).

5. OBRANA

Kad dolazi do napada na sustav ponekad je potrebno koristiti neke druge tehnike i mehanizme koji se ponešto razlikuju od preventivnih tehnika.

Mehanizmi obrane uključuju tehnike kao što su: resetiranje sustava, reorganiziranje različitih elemenata u sustavu, ponovno miješanje baze podataka i slično. To predstavlja shemu oporavka od napada ubacivanjem SQL izraza (Kindy i sur., 2012).

Stephen Tomas i njegovi suradnici predložili su pripremljeni zamjenski algoritam i odgovarajuću automatizaciju za uklanjanje ranjivosti od napada ubacivanjem SQL izraza iz ranjivih SQL izraza, zamjenjujući ih sa sigurnim pripremljenim izjavama. Ovdje se ne pokušava ukloniti ranjivost SQL izraza, već je naglasak na zamjeni SQL izraza sa već sigurnim i pripremljenim izrazima. Kreiran je pripremljeni algoritam za zamjenu izraza pod nazivom PSR algoritam (engl. *Prepared Statement Replacament*) koji prikuplja informacije iz izvornoga kôda koji sadrži maliciozni kôd. Na taj se način generira siguran pripremljeni kôd izraza koji održava funkcionalni integritet. Sukladno tome, kreiran je i PSR generator koji automatizira generiranje pripremljenog kôda na temelju izraza u Javi. PSR generator je napisan u Javi zbog dostupnosti Open source Java projekata koji sadrži SQLIV.

Njihov istraživački rad je proveden korištenjem četiri prijedloga otvorenog izvora: (i) Net-trust, (ii) ITrust, (iii) WebGoat i (iv) Roller. Na temelju analize, njihov pripremljeni kôd SQL izraza bio je u stanju zamijeniti 64% ranjivosti napada ubacivanjem SQL izraza u četiri otvorena prijedloga izvora. Tada je još analiza provedena samo pomoću Jave, s ograničenim brojem prijedloga. Međutim, logika u algoritmu nije ograničena, pa se može koristiti i proširiti kako bi odgovaralo sintaksi bilo kojeg jezika. To je i dalje otvoreno istraživanje koje je potrebno detaljnije istražiti (Thomas i sur., 2007).

Za obranu od napada ubacivanjem SQL izraza koristi se još jedan pristup, a to je: SQL Rand. Ovaj pristup koristi proxy poslužitelj smješten na sredini web poslužitelja i poslužitelja baze podataka za razmijenjene upite između klijenta i poslužitelja baze podataka. Poslužitelj baze podataka prima skup prihvaćenih ključnih riječi iz nasumičnih SQL upita za računanje. Pogreška koju je stvorio poslužitelj baze podataka, a koja se sastoji od nelegitimnih upita skrivena je kako bi se izbjeglo da se hakeru pruži prilika za dobivanje arhitekture tablica i shema baze podataka (Kumar, Pateriya, 2012), (Tajpour i sur., 2011), (Amirtahmasebi, 2009).

Za provedbu su koristili koncept proxy poslužitelja između Web poslužitelja i SQL poslužitelja. Ne-nasumični upiti primljeni od klijenta šalju zahtjev poslužitelju. Ovaj okvir ne-nasumičnosti ima dvije glavne prednosti: prenosivost koja je primijenjena sa širokim rasponom DBMS-a i sigurnost. Predložena shema ima dobre performanse. Dakle, učinkovito je s obzirom na postignute rezultate i obranu protiv ubrizganih upita. Međutim, još uvijek zahtijeva daljnje testiranje i podršku od strane programera u izgradnji alata koji koriste SQLrand (Boyd, Keromytis, 2004).

U literaturi se najviše raspravlja o prevenciji, pa samim time nema puno govora i prijedloga kako se obraniti od napada ubacivanjem SQL izraza. Prioritet je dan na prevenciji, odnosno želi se na vrijeme zaštiti aplikacije i baze podataka kako do napada ne bi ni došlo, što u konačnici uvelike ima smisla provoditi.

6. PREGLED PRISTUPA

Do sada su korišteni mnogi okviri i pristupi za prevenciju i obranu od napada ubacivanjem SQL izraza u web-aplikacijama. Ovdje će se ukratko spomenuti istaknuta rješenja i njihove metode rada, kako bi čitatelji dobili bolji pregled o temeljnim idejama.

U ovom radu (Panah i sur., 2016) predstavljen je novi pristup za rasijecanje HTTP prometa i pregled složenih napada ubacivanjem SQL izraza. Model koji je predstavljen je hibridni sustav za sprečavanje ubrizgavanja (HIPS) koji koristi klasifikator strojnog učenja i mehanizma za provjeru uzorka, koji se temelji na smanjenim skupovima sigurnosnih pravila. Njihova arhitektura vatrozida web aplikacija ima za cilj optimizirati performanse otkrivanja pomoću modula predviđanja koji isključuje legitimne zahtjeve iz procesa nadzora (Makiou i sur., 2014).

U (Appelt i sur., 2014) autori predstavljaju pristup automatiziranog testiranja pod nazivom µ4SQLi i njegov temeljni skup operatora mutacija. µ4SQLi može proizvesti učinkovite ulaze koji dovode do izvršnih i štetnih SQL izraza. Izvršnost je ključna jer se inače ne može iskoristiti ranjivost ubrizgavanja. Njihova procjena je pokazala da je pristup učinkovit za otkrivanje ranjivosti pri

napadu ubacivanjem SQL izraza i za proizvodnju ulaza koji zaobilaze vatrozide aplikacija, što je uobičajena konfiguracija u stvarnom svijetu.

Alnabusi i suradnici predlažu inovativno rješenje za filtriranje napada ubacivanjem SQL izraza pomoću SNORT IDS-a. Predložena tehnika detekcije koristi SNORT alat povećavajući brojna dodatna SNORT pravila. Predloženo rješenje su usporedili s još nekoliko postojećih tehnika. Eksperimentalni rezultati pokazuju da predložena metoda nadmašuje druge slične tehnike koristeći isti skup podataka (Alnabusi i sur., 2015).

Ovaj rad (Das, Bhattacharrya, 2018) prikazuje metodu provjere web-dokumenata i izvodi se eksperimentiranje na strani klijenta koristeći benignu strukturu skripte. Ova metoda može otkriti sve zlonamjerne skripte koje se ubacuju u web-dokument tijekom transporta od poslužitelja do klijenta ili zbog prethodno pohranjenih sadržaja u operaciji klijenata ili poslužitelja. Zadovoljavajući rezultati pronađeni su vlastitim generiranjem i javno dostupnim skupom podataka.

Ovaj rad (Kar i sur., 2015) predstavlja novi pristup za detekciju napada ubacivanjem SQL izraza u realnom vremenu pomoću transformacije upita i mjere sličnosti dokumenata. Djelujući kao vatrozid baze podataka, predloženi sustav pod nazivom SQLiDDS može zaštititi više web aplikacija pomoću poslužitelja baze podataka. Uz dodatne ulazne podatke od stručnjaka za ljudske resurse, SQLiDDS također može postati robusniji tijekom vremena. Provedeni eksperimenti kao rezultat su dobili potvrdu da ovaj pristup može učinkovito otkriti i spriječiti sve vrste napada ubacivanjem SQL izraza s dobrom točnošću, ali zanemarivim utjecajem na performanse sustava. Pristup je testiran na web aplikacijama izrađenim pomoću PHP-a i MySQL-a, no može se lako primijeniti u drugim platformama s minimalnim promjenama.

U (Ruse i sur., 2010) Ruse i suradnici predlažu tehniku i okvir koji koriste automatsko generiranje testnog slučaja radi otkrivanja napada ubacivanjem SQL izraza. Glavna ideja ovog okvira temelji se na stvaranju specifičnog modela koji se automatski bavi SQL upitima. Pokazalo se kako metodologija može specifično identificirati 85% točnosti. Štoviše, ne proizvodi lažne pozitive ili lažne negative i u stanju je otkriti stvarne slučajeve napada. Usprkos učinkovitosti tehnike, nedostatak je to što nije testirana sa stvarnim upitima u stvarno postojećoj bazi podataka.

U (Shin i sur., 2009) Shin i suradnici su predložili SQLUnitGen, alat koji se temelji na statičkoj analizi i koji automatizira testiranje za identifikaciju ulaznih podataka manipulacijske ranjivosti. Pokazalo se da je predloženi mehanizam učinkovit s obzirom na činjenicu da su lažni pozitivi u potpunosti odsutni u eksperimentima. Međutim, za različite scenarije uočeni su lažni negativi na malom broju. Osim toga, utvrđeno je da se zbog nekih nedostataka može pojaviti značajna stopa lažnih negativnih rezultata za druge aplikacije. Stoga autori govore o usredotočenosti oslobađanje od tih značajnih lažnih negativa i daljnje poboljšanje pristupa za pokrivanje ulaznih manipulacija ranjivosti kao i njihovi budući radovi.

Rochman i Gudes predstavili su novu metodu zaštite internetskih baza podataka na temelju istančanog mehanizma kontrole (engl. *fine-grained acces*) pristupa. Ova metoda koristi ugrađene mehanizme kontrole pristupa bazama podataka s prikazanim parametrima i prilagođava ih radu s web aplikacijama. Predloženi mehanizam kontrole pristupa je primjenjiv na sve postojeće baze

podatka i sposoban je spriječiti mnogo vrsta napada, čime se znatno smanjuje napad na površini baze podatka (Roichman, Gudes, 2007).

Kemalis i Tzouramanis su razvili prototip SQL sustava za detekciju ubrizgavanja (SQLIDS) koji implementira predloženi algoritam. Sustav nadzire Java aplikacije i otkriva SQL ubrizgavanje napada u stvarnom vremenu. Proveli su nekoliko preliminarnih eksperimenata tijekom nekoliko napada ubacivanjem SQL izraza. Rezultati su bili vrlo uspješni. Provođenjem eksperimenata nije došlo do lažnih pozitiva i negativa. Dakle, novi pristup se je pokazao vrlo učinkovitim u praksi. Međutim, pristup zahtijeva više analize kao i dostupne tehnike otkrivanja pod zajedničkim i fleksibilnim okruženjem (Kemalis, Tzouramanis, 2008).

Napadi nastali ubacivanjem SQL izraza javljaju se zbog ranjivosti u strukturi upita gdje zlonamjerni korisnik može iskoristiti mogućnost unosa za umetanje kôda u upite koji mijenjaju uvjete upita, što rezultira neovlaštenim pristupom bazi podataka. Sarkar i suradnici pružaju novu tehniku kako bi identificirali mogućnost takvih napada. Središnja tema njihove tehnike temelji se na automatskom razvoju modela za SQL upit tako da model bilježi zavisnost između različitih komponenti upita. Između ostalog, analiziraju model pomoću generatora CREST test slučaja i identificiraju uvjete pod kojima se upit koji odgovara modelu smatra ranjivim. Također, analiziraju i dobiveni uvjet kako bi identificirali njegov podskup. Taj se podskup naziva uzročnim skupom ranjivosti. Njihova tehnika razmatra semantiku uvjeta upita, to jest odnos između uvjeta i kao takva nadopunjuje postojeće tehnike koje se oslanjaju samo na sintaktičku strukturu SQL upita. Ukratko, ova tehnika može otkriti ranjivost ugniježđenih SQL upita i daje rezultate bez lažnih negativa u usporedbi s postojećim tehnikama (Ruse i sur., 2010).

U radu (Halfond, Orso, 2008) predložen je pristup kombiniranja statističke analize i praćenje izvršavanja upita baze podataka. Tehnika koristi analizu programa za automatsku izradu legitimnih upita koji će biti generirani od strane aplikacije. Tehnika prati dinamički generirane upite i provjerava njihovu prihvatljivost sa statički generiranim modelom. Upit koji se ne podudara s modelom predstavlja potencijalni napad ubacivanja SQL izraza pa je stoga spriječeno njegovo izvršavanje na bazi podataka.

Shema i suradnici (Ali i sur., 2009) su usvojili pristup hash vrijednosti prema daljnjem poboljšanju mehanizma autentifikacije korisnika. Oni koriste korisničko ime i hash vrijednosti lozinke SQLIPA (engl. SQL *Injection Protector for Authentication*). Prototip je razvijen kako bi testirao okvire. Korisničko ime i vrijednosti hash-a za lozinku stvaraju se i izračunavaju tijekom izvođenja kada se prvi put kreira određeni korisnički račun. Hash vrijednosti pohranjuju se u tablici korisničkih računa. Iako je predloženi okvir testiran na nekoliko uzoraka podataka i dalje zahtijeva daljnje poboljšanje kako bi se smanjilo vrijeme opterećenja. Također, zahtijeva da bude testiran od strane veće količine podataka.

Ovaj rad (Som i sur., 2016) daje pregled sprječavanja napada ubacivanjem SQL izraza u pohranjenim procedurama. Aplikacija je osigurana od napada pomoću tehnike koja je podijeljena u dvije faze, a to su: faza sučelja, i faza pozadine. U slučaju kada prva faza nije u stanju zaštiti od SQL napada, tada druga faza sprječava napad. Tijekom skupljanja informacija o upitu ova tehnika koristi AES algoritam, kako bi se izbjegao napad ubacivanjem SQL izraza. Njihovi rezultati istraživanja pokazali

su kako postoji širok raspon stranica koji može zaštiti od napada. Ova metodologija potiče brzo i stručno upoznavanje sustava s bazom podataka te se drži podalje od memorije.

7. REZULTATI I ANALIZA

Više od polovice ranjivosti može se klasificirati kao manipulacija ulazom. Iz toga se razloga sve više koriste alati za automatsku statističku analizu kako bi se identificirala upravo ranjivost tih ulaza. Međutim, ovi alati ne mogu otkriti prisutnost ili učinkovitost crnih ili bijelih ulaznih filtera popisa pa mogu imati visoku razinu lažnih rezultata (Roichman i Gudes, 2007).

Radi bolje obrane i prevencije dolazi se do ideje spajanja više algoritama u jedan, što daje značajnu efikasnost. U radu je kao primjer takve ideje prikazan hibridni algoritam.

Sljedeća tablica prikazuje nekoliko važnijih metoda za prevenciju i obranu od napada ubacivanjem SQL izraza.

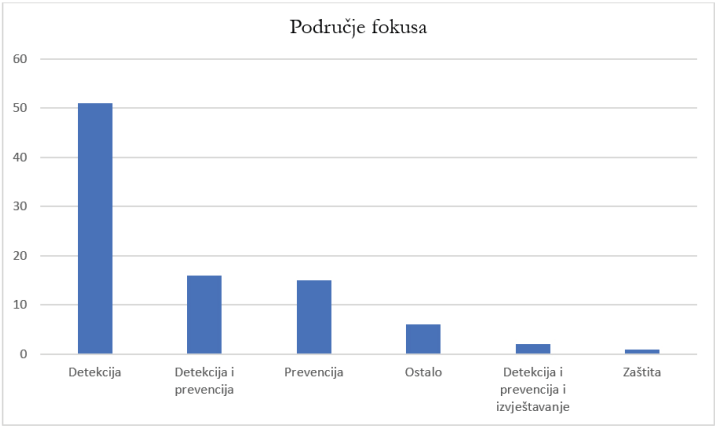
Tablica 3. Prikaz metoda

Metoda	Osvrt
CANDID	Metoda za automatsko sprječavanje napada ubacivanjem SQL izraza *Prednost*: primjenjiv na sve baze podataka *Nedostatak*: smanjena učinkovitost pri suočavanju s vanjskim funkcijama
SAFELI	Okvir statičke analize dizajna koji se temelji na prepoznavanju SQL napada *Prednost*: vrlo učinkovito prepoznavanje napada *Nedostatak*: detektira samo napade ubacivanjem SQL izraza koji se posebno provode na Microsoftovim stranicama
PSR-generator	Automatizira generiranje pripremljenog koda *Prednost*: zamjenjuje SQL izjave sa već pripremljenim izjavama *Nedostatak*: analiza je provedena samo pomoću Jave
SQLUnitGen	Alat koji se temelji na statističkoj analizi *Prednost*: lažan pozitiv je potpuno odsutan u pokusima *Nedostatak*: za neke aplikacije se može pojaviti određena stopa lažnih negativnih rezultata

Proučavanjem literature može se primijetiti kako su dosadašnja istraživanja najviše posvećena detekciji i prevenciji od napada ubacivanjem SQL izraza, dok je u manjoj mjeri zastupljena

literatura koja govori o obrani od napada ubacivanjem SQL izraza. Sljedeći graf prikazuje upravo tu zastupljenost.

Graf 1. Istraženo područje fokusa



Izvor: Panah i sur. (2016)

8. ZAKLJUČAK

Iako su napretkom tehnologije interaktivne web-stranice svakoga dana sve sigurnije njihova nepobitnost je i dalje pod upitnikom. Napad ubacivanja SQL izraza predstavlja najveću prijetnju online tvrtkama koje u pozadini imaju bazu podataka. U ovom je radu dan pregled najpopularnijih metoda i tehnika za prevenciju i obranu od napada ubacivanjem SQL izraza. Postoje metode koje su tek implementirane te im je potrebno dodatno testiranje i uvođenje. S druge strane neke metode se već dugi niz godina unaprjeđuju te već mogu uvelike pomoći u prevenciji i obrani od napada ubacivanjem SQL izraza.

U istraživanjima se opsežno govori o prevenciji te se predlaže niz spomenutih metoda i tehnika, dok kod obrane od napada ubacivanjem SQL izraza to nije slučaj. Iz pregleda literature vidljivo je kako svaka detekcija i tehnika prevencije ne može pružiti potpunu zaštitu protiv napada ubacivanjem SQL izraza jer se mnogi pristupi ne bave tim problemom u cijelosti ili pak stvaraju poteškoće prilikom njihova praktičnog uvođenja. Unatoč tome kombinacija prikazanih mehanizama zasigurno će pokriti široki raspon zaštite baze podataka od napada ubacivanjem SQL izraza. Općenito područje vezano uz obranu i prevenciju od napada ubacivanjem SQL izraza je dobro istraženo pa je stoga i osmišljeno mnogo tehnika i mehanizama u tu svrhu.

Može se reći da postoji još prostora za istraživanje i napredovanje u spomenutim područjima s obzirom da tehnike, pa tako i ljudsko znanje svakim danom sve više širi svoje granice. Posebno treba istaknuti brzo rastući trend računarstva u oblaku, odnosno korištenja „oblaka" za pohranu podataka, kao i aplikacije koje u potpunosti funkcioniraju u „oblaku". Ovaj aspekt problema napada ubacivanjem SQL izraza nije bio toliko u fokusu ovoga rada, međutim svakako treba istaknuti

da ostaje za daljnje istraživanje. Daljnjim razvitkom web tehnologija, kao i postojećih tehnika za obranu i prevenciju napada ubacivanjem SQL izraza povećava se i spektar oblika napada, stoga je od izrazite važnosti voditi računa o konstantnom unaprjeđenju tehnika, metoda i mehanizama potrebnih za prevenciju i obranu od napada ubacivanjem SQL izraza. Kako bi se to postiglo valja ustrajati u osmišljavanju novih rješenja kako bi se u budućnosti osigurala potpuna kontrola i zaštita od napada ubacivanjem SQL izraza. Ono što ostaje za budući rad jest napraviti detaljniji pregled problema i izazova vezanih uz napad ubacivanja SQL izraza nad web aplikacijama te bazama podataka u „oblaku".

ZAHVALE

Ovaj se rad temelji na radu koji je Sveučilište u Rijeci podržalo u okviru projekata „uniri-drustv-18-182".

LITERATURA

Ali S., Rauf A., Javed H. (2009) „SQLIPA: An authentication mechanism against SQL injection", European Journal of Scientific research, vol.38, pp 604-611

Alnabulsi H., Islam M. R., Mamun Q. (2015) „Detecting SQL injection attacks using SNORT IDS"

Amirtahmasebi K., Jalalinia S. R., Khadem S. (2009) „A survey of SQL injection defense mechanisms"

Appelt D., Nguyen C. D., Briand L. C., Alshahwan N. (2014) „Automated testing for SQL injection vulnerabilities: an input mutation approach", international Symposium on Software testing and Analysis, pp 259-269.

Balasundram I., Ramaraj E., (2012) „An efficient techinque for deection and prevention of SQL injection attack using ASCII based string matching".

Bisht P., Madhusasudan P., Venkatakrishnan V. N. (2010) „CANDID: Dynamic Candidate Evaluations for Automatic Prevention of SQL Injection Attacks "

Boyd S. W. i Keromytis A. D. (2004) „SQLrand: Preventing SQL injection attacks", department of Computer Science

Centar informacijske sigurnosti (2012) FER

Clark J. (2012) „SQL injection attacks and defense"

Coles M. (2007) „Pro T-SQL Programmer's Guide"

Cova M., Balzarotti D. Felmetsger V., Vigna G. (2007) „An approach for the anomaly-based detection of state violations in web applications"

Das D., Bhattacharrya D. K., (2018) „Defeating Cyber Attacks Due to Script Injection", International Journal of Network Security, vol.20, pp. 225-234.

Ezumalai R., Aghila G. (2009) „Combinatorial approach for preventing SQL injection attacks"

Fouad Y., Elshazly K. (2013) „Detection and Prevention of SQL Injection Attacks on Web Applications", IJCSNS International Journal of Computer Science and Network Security, vol. 13.

Fu X., Qian K. (2008) „SAFELI: SQL injection scanner using symbolic execution"

Haixia Y., Zhihong N. (2009) „A database security testing scheme of web application"

Halfond W. G. J., Orso A. (2008) „WASP: Protecting W eb Applications Using Positive T ainting and Syntax-Aware Evaluation", IEE Transaction on Softeare engineering

Halfond W. G. J., Orso A., Manolios P. (2006) „Using Positive tainting and Syntax-Aware Evaulation to Counter SQl Injection Attacks ", College of Computing

Indrani B., Ramaraj E. (2011) „X-Log authentication technique to prevent SQL injection attacks"

Johari R., Sharma P. (2012) „A survey on web application vulnerabilities (SQLIA, XSS) exploitation and security engine for SQL injection"

John A., Agarwal A., Bhardwaj M. (2012) "An adaptive algorithm to prevent SQL injection", American Journal of Networks and Communications

Junjin M. (2009) „An approach for SQL injection vulnerability detection"

Kar D., Panigrahi S., Sundararajan S., (2015) „SQLiDDS: SQL injection detection using query transformation and document similarity, pp.377-390.

Kemalis K., Tzouramanis T. (2008) „SQL-IDS: A specification-based approach for SQL-injection detection",

Khan A. (2005) „SQL Injection"

Kindy D. A., K. Pathan K. Al-S. (2012) „A Detailed Survey on Various Aspects of SQL Injection in Web Applications: Vulnerabilities, Innovative Attacks, and Remedies", International Journal

Kumar P. i Pateriya R. K. (2012) „A survey on SQL injection attacks, detection and prevention techinque"

Lawal M.A., Bakar A., Sultan M., Shakiru A. O. (2016) „Systematic Literature Review on SQL Injection Attack", International Journal of Soft Computing, pp 26-35.

Liu A., Yuan Y., Wijesekera D. (2009) „SQLProb: A Proxy-based Architecture toward Preventing SQL Injection Attacks"

Makiou A., Begriche Y., Serhrouchni A. (2014) „Improving web application firewalls to detect advancedd SQL injection "

Mavromoustakos S., Patel A., Chaudhary K., Chokshi P., Patel S. (2016) „Causes and Prevention of SSQL Injection Attacks in Web Applications"

Panah M. V., Bayat N. K., Asami A., Shahmirzadi A. (2016) „SQL injection attacks: A System Review"

Parameswari S. i Kavitha K. (2018) „SQL Injection Attack on Web Application", Asisan Journal of Computer Science and Technology, vol. 7, pp. 11-15.

Roichman A., Gudes E. (2007) „Fine-grained acces control to web database"

Ruse M., Sarkar T., Basu S. (2010) „Analysis & detection of SQL injection vulnerabilities via automatic test case genration of programs"

Sajjadi S. M. S., Pour B. T. (2013) „Study of SQL injection attacks and countermeasures", International Journal of Computer and Communication Engineering, vol. 2.

Senthilkumar S., Reddy K. T. (2017) „Preventing SQL Injection Attack Using Pattern Matching, Parsee Tree Validation and Cryptography Algorithms", Jouurnal of Environmental Science, Computer Science and Engineering and Techonolgy, vol.6, pp 246-253.

Shin Y., Williams L., Xie T. (2009) „SQLUnitGen: Test Case Generation for SQL injection detection"

Som S., Sinha S., Kataria R. (2016) „Study on SQL injection attacks: mode, detection and prevention", International Journal of Engineering Applied Sciences and Technology, vol.1, pp 23-29.

Tajpour A., Ibrahim S., Masrom M. (2011) „SQL injection detection and prevention techniques"

Thomas S., L. Williams, Xie T. (2007) „On automated prepared statement generation to remove SQL injection vulnerabilities", Information and Software technology

Zhang K-X., Lin C-J., Chenn S-J., Hwang Y. (2011) „TransSQL:A Translation and Validation-Based Solution for SQL-injection Attacks"

Review article
https://doi.org/10.31784/zvr.8.1.10
Received: 25. 11. 2019.
Accepted: 8. 1. 2020.

SQL INJECTION – PREVENTION AND DEFENSE

Nikolina Ljubičić
Univ. bacc. inf, Student, 51000 Rijeka, Croatia; e-mail: ljubicic.nikolina@gmail.com

Danijela Jakšić
PhD, Assistant professor, University of Rijeka, Department of Informatics, Radmile Matejčić 2, 51000 Rijeka, Croatia; e-mail: danijela.jaksic@inf.uniri.hr

Patrizia Poščić
PhD, Full Professor, University of Rijeka, Department of Informatics, Radmile Matejčić 2, 51000 Rijeka, Croatia; e-mail: patrizia@inf.uniri.hr

ABSTRACT

SQL injection is one of the most serious security threats to applications that have their own database. In fact, it allows the attacker to gain control of the application database by giving attacker the ability to modify the data. Many researchers have been trying to solve this problem. They have suggested different approaches to detect and prevent this vulnerability, but even these approaches do not fully represent a successful solution. This paper introduces several types of SQL injection attacks, as well as various tools and methods that can provide some prevention and defense against these attacks. Various methods have been proposed to address the problem of SQL injection attacks in the form of extensive reviews of variations of SQL injection attacks. Also, descriptions and examples were offered in order to understand their adverse impact and the consequences better.This paper also provides a brief overview of what some authors suggest when it comes to preventing and defending from SQL injection. Finally, a conclusion is drawn with an objective review and analysis of the overall research. The main contribution of the paper is an overview of the research available so far, limited to relational databases and categorized on: a) prevention of SQL injection and b) defence from SQL injection.

Key words: *SQL injection, prevention, defense, attack, techniques*