Sigurnost web aplikacija

Maja, Rukav

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Maja Rukav

SIGURNOST WEB APLIKACIJA

ZAVRŠNI RAD

Varaždin, 2018.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Maja Rukav

Matični broj: 44003/15–R

Studij: Informacijski sustavi

SIGURNOST WEB APLIKACIJA

ZAVRŠNI RAD

Mentor:

Prof. dr. sc. Dragutin Kermek

Varaždin, rujan 2018.

Maja Rukav

Izjava o izvornosti

Izjavljujem da je moj završni rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autor/Autorica potvrdio/potvrdila prihvaćanjem odredbi u sustavu FOI-radovi

Sažetak

U završnom radu opisane su osnove sigurnosti u Web okruženju. Navedene su osnovni problemi s kojima se korisnici susreću spajanjem na Internet i korištenjem Web aplikacija. U nastavku su također opisane i osnove sigurnosti zbog krađe osobnih podataka. Navedena su i objašnjena moguća mjesta napada na elemente Web aplikacija te opisana najbolja praksa kako programeri mogu i trebaju osigurati da privatni dio aplikacije bude zaštićen. Temeljem OWASP-a navedene su najčešće korištene metode napada na Web aplikacije i preporuke za sigurnosti Web aplikacija. Na temelju Web aplikacije kao praktičnog dijela rada, prikazane su ranjivosti Web aplikacije i objašnjeno je kako zaštititi aplikaciju kako bi bila sigurna za korisnike. Praktični dio završnog rada je Web aplikacija koja koristi sve temeljne preporuke zaštite i sigurnosti.

Ključne riječi: web, PHP, baza podataka, autentikacija, sigurnost, autorizacija, javascript

Sadržaj

1. Uvod

Za razliku od prošlosti kada su se na Internetu nalazile samo statičke Web stranice čiji je sadržaj bio otvoren i dostupan svim korisnicima, u današnje vrijeme većina sadržaja koji se nalaze na Internetu je privatna i sadrži vrlo osjetljive podatke. Svi podaci na poslužitelju za sve korisnike bili su javno dostupni, a protok bitnih informacija bio je jednosmjeran, od poslužitelja do korisnika. Upravo iz toga razloga, zaštita podataka bila je jednostavnija, a sigurnosni propusti nalazili su se jedino na poslužitelju. Kako se sadržaj Interneta bitno promijenio, statičke Web stranice sada su Web aplikacije čiji se sadržaj generira dinamički ovisno o zahtjevu pojedinog korisnika. Protok informacija sada je dvosmjeran, a podatke koji se razmjenjuju između korisnika i Web poslužitelja potrebno je zaštititi kako ne bi došlo do njihovog otkrivanja i zlouporabe.

Web aplikacija je bilo koji računalni program koji obavlja određene funkcije koristeći se pri tome Internet vezom. Web aplikacije se otvaraju u bilo kojemu Web pregledniku kao što su Google Chrome, Mozilla Firefox, Opera i Microsoft Edge. Kako nije potrebna dodatna edukacija za njihovo korištenje, već je dovoljno osnovno poznavanje i razumijevanje korištenja Interneta, Web aplikacije postaju sve popularnije i prisutnije u današnjem svijetu. Mogućnost ažuriranja aplikacije bez instaliranja softvera na računalo ili bilo koji drugi uređaj te pristup aplikaciji s uređaja koji imaju pristup Internetu pridodaju na njihovoj popularnosti.

Web aplikacije imaju različite namjene, od edukacijskih, poslovnih, društvenih pa sve do zabavnih. Primjeri Web aplikacija čija je namjena edukacija korisnika su Udacity (eu.udacity.com), Codecademy (www.codecademy.com), edX (www.edx.org) i drugi. Društvene aplikacije koriste se za povezivanje korisnika te razmjenu korisničkih iskustava i priča, a neke od društvenih aplikacija su Facebook, Instagram, LinkedIn, YouTube i Skype. Ostali primjeri Web aplikacija su Web trgovine (eng. Webshop), bankarske aplikacije, pretraživači i mnoge druge.

Zbog sve veće popularnosti Web aplikacija javlja se problem sigurnosti. Sigurnost se definira kao stupanj zaštite od opasnosti, gubitka ili kriminalne aktivnosti, no u smislu sigurnosti Web aplikacija pod pojmom sigurnost mislimo na zaštitu podataka koje aplikacija koristi, ali i na zaštitu osobnih podataka korisnika koji se koriste Web aplikacijom. Kako bi korisnici bili sigurni da njihovi osobni podaci neće biti zlouporabljeni važno je da aplikacije imaju visoku razinu sigurnosti. Upravo iz toga razloga u ovome radu su opisani i objašnjeni najčešći problemi s kojima se susreću korisnici, ali i najbolja praksa za sprječavanje istih.

2. Metode i tehnike rada

Metode i tehnike koje su korištene pri razradi teme tijekom pisanja teorijskog dijela završnoga rada su proučavanje znanstvenih radova, knjiga o sigurnosti Web aplikacija i literature s web-a.

Prilikom izrade Web aplikacije kao praktičnog dijela završnog rada korišteni su programski jezici HTML, CSS, PHP, AJAX, JavaScript i JQuery.

Korišteni alati:

- NetBeans IDE
- MYSQL Workbench
- XAMPP
- phpMyAdmin
- Mozilla Firefox
- Google Chrome

Vanjski izvori koji su bili pomoći pri izradi Web aplikacije su php.net, w3schools.com i stackocerflow.com.

3. Osnove sigurnosti u web okruženju

World Wide Web je sustav poslužitelja, tj. servera koji podržavaju posebno formatirane dokumente sa vezama prema drugim dokumentima. Web je jedna od najkorištenijih usluga Interneta koja omogućava dohvaćanje hipertekstualnih dokumenata. Kako se Web koristi za razmjenu podataka na Internetu možemo reći da je on zaslužan za umrežavanje korisnika. Web stranice i aplikacije olakšavaju korisnicima da relativno brzo dođu do željenih informacija i koriste ih kada im je to potrebno. Iako nam te informacije mogu biti od velike koristi, nepažnja prilikom korištenja Web stranica i aplikacija te sadržaja koji se na njima nalaze mogu nam stvoriti velike probleme. Ukoliko ne pazimo na osobne podatke koje koristimo ili nismo dovoljno pažljivi prilikom unosa podataka može doći do krađe i zlouporabe istih. Prilikom korištenja aplikacija često dolazi do prijevara pri čemu korisnici ostaju oštećeni, zaraze svoje računalo ili drugi uređaj raznim virusima i time postaju meta hakera. Broj prijevara se sve više povećava, a prevaranti pronalaze i osmišljavaju nove taktike za prijevaru potencijalnih korisnika. Upravo iz tog razloga važno je znati koje su to najčešće prijevare s kojima se korisnici mogu susresti i kako se zaštititi od istih te kako zaštititi svoje osobne podatke.

3.1. Najčešće prijevare prilikom korištenja web aplikacija

Problemi koji se javljaju korištenjem Web sadržaja su krađa i zlouporaba osobnih podataka korisnika. Neke od najčešćih prijevara s kojima se susreću korisnici na internetu tokom prijave i korištenja Web aplikacija su mrežna krađa identiteta, scam prijevare, povrede privatnosti i nigerijsko pismo 419.

3.1.1. Mrežna krađa identiteta

Mrežna krađa identiteta ili pecanje (eng. phising) je kriminalna aktivnost, odnosno vrsta prijevare prilikom koje se šalju lažne poruke koristeći pritom postojeće Internet servise.

Korištenjem manipulacija, prevaranti prikupljaju povjerljive podatke korisnika kao što su korisničko ime i lozinka, podatci s kreditnih kartica i slično kako bi u većini slučajeva ostvarili financijsku korist ili kako bi žrtvi nanijeli neku drugu štetu. [1] Postoje različite metode pecanja kojima prevaranti pokušavaju doći do podataka, a najčešće su one putem elektroničke pošte.

E-mailovi koje primamo na svoje račune često izgledaju vjerodostojno, a poruke su u obliku obavijesti koje primamo od organizacija kao što su Microsoft ili Dropbox. Na taj način se korisnicima čini da poruka dolazi od organizacije kojoj vjeruje.

Jedan od načina je e-mail poruka u kojoj se pošiljatelj lažno predstavlja i zahtjeva od korisnika da u odgovoru poruke pošalje osobne podatke radi verifikacije podataka, računa ili naloga, nadogradnje i održavanja sustava pa čak i unaprjeđenja sigurnosnog sustava. Drugi način je da se u e-mail poruci navede lažnu poveznicu (eng. link) koji korisnika vodi na zloćudnu Web stranicu na kojoj se od korisnika također traži da ostavi svoje korisničko ime i lozinku ili druge povjerljive podatke. Iduća metoda mrežne krađe identiteta su lažna Web sjedišta prilikom kojih se klikom na poveznicu poslužitelja izmijeni stvarnu URL adresu Web sjedišta i na taj način se korisnika obmanjuje da se nalazi na legalnoj stranici. Lažna stranica u velikoj većini slučajeva izgleda skoro identično originalnoj stranici, ali je razlika u adresi koja se nalazi u adresnoj traci Web preglednika. Mogući način prijevare je i iskakanje lažnog prozora (eng. popup) sa poljima za unos podataka na legitimnim Web poslužiteljima. Najnovija vrsta prijevare koja spada u mrežnu krađu identiteta je „tabnabbing“. Pri toj metodi kod korisnika koji imaju otvoreno više prozora istovremeno osvježava se jedan od neaktivnih prozora, ali sa zloćudnim sadržajem koji je imitacija legitimne Web stranice te se na taj način iskorištava nepažnja korisnika koji najčešće ne primjećuje novu adresu.

3.1.2. Scam prijevare

Prijevaru ili plan za izmamljivanje novca ili bilo kojih drugih dobara od osoba koje ne sumnjaju da je riječ o prijevari nazivamo scam prijevarama. Pri toj prijevari osoba ili organizacija pruža lažne podatke prilikom davanja ponude, odnosno sklapanja dogovora. Ova vrsta prijevare

uspješna je kako u virtualnom, tako i u realnom životu, a žrtva scam prijevare može postati bilo tko u bilo kojemu trenutku.

3.1.3. Povrede privatnosti

Znamo kako je temeljno ljudsko pravo zapravo pravo na zaštitu osobnih podataka i kako se osobni podaci korisnika mogu prikupljati, obrađivati i koristiti samo uz pristanak pojedinca čiji se podaci zapisuju i koriste. Povreda privatnosti i nedozvoljena uporaba osobnih podataka kriminalno su djelo pa je pravo na zaštitu osobnih podataka osigurano međunarodnim i nacionalnim propisima. Objavljivanje poruka, e-mailova, fotografija, ali i osobnih podataka bez pristanka druge osobe čiji se podaci javno objavljuju smatra se kriminalnim djelom. U tim slučajevima najčešće se radi o psihičkom maltretiranju osobe, narušavanju njezinog društvenog položaja i digniteta. U povredu privatnosti spada i povreda tajnosti pisama i drugih pošiljaka koja se javlja u slučaju ako je nekoj osobi hakiran račun, mail ili profil na društvenim mrežama te su podaci o toj osobi postali dostupni drugima.

3.1.4. Nigerijsko pismo 419

Nigerijska shema, pismo ili „419“ je jedna od najuspješnijih i najpoznatijih prijevara koja je zaživjela na Internetu. Samo ime je dobila po članku nigerijskog zakona koji prijevaru definira kao kazneno djelo. Riječ je o prijevari u kojoj se korisniku na e-mail šalje pismo bogate nigerijske obitelji u kojemu se navodi smrt bliskog člana i u kojoj osoba želi izvući novac iz zemlje zbog sigurnosnih razloga. Naime, prijevara se krije u dijelu poruke u kojoj od osobe traže da se uplati beznačajni iznos provizije, a nakon toga će novac biti prebačen na korisnikov račun i za taj čin će korisnik biti nagrađen.

3.2. Zaštita osobnih podataka

Osobni podaci, ali i drugi povjerljivi podaci koje ostavljam na Internetu uvijek su na meti kradljivaca i zbog toga je potrebno posvetiti više pažnje kako bismo ih zaštitili. Korisnici mogu spriječiti krađu osobnih podataka i vlastitog identiteta pazeći pritom gdje i kome ostavljaju svoje podatke. Ukoliko se izdvoji malo vremena i detaljnije prouči Web mjesto na kojemu ostavljamo podatke može se spriječiti veća katastrofa.

Kako smo već naveli, jedna od najčešćih prijevara je mrežna krađa identiteta. Korisnici se ne mogu zaštititi od primanja neželjenih poruka, ali bi uvijek trebali dobro provjeriti tko zapravo šalje te e-mailove i jesu li oni lažni. Korisnicima se preporučuje da nikada ne unose svoje osobne podatke na Web stranicu ukoliko nisu sigurni da je stranica verificirana. Prilikom dolaska na Web stranicu potrebno je provjeriti adresu u adresnoj traci Web preglednika. Ako URL adresa nije identična adresi poduzeća, primjerice banke ili druge organizacije koja sadrži osobne podatke korisnika, lako je moguće da se radi o prijevari. Adresa krivotvorene Web stranice često se zna razlikovati od adrese legitimne stranice u samo jednom znaku. Ukoliko je potrebno posjetiti Web mjesto banke, sigurnije je da korisnik sam unese adresu, nego da je otvara klikom na navedenu poveznicu u primljenim porukama. Ako se unose podatci u obrazac Web stranice potrebno je provjeriti digitalni certifikat Web poslužitelja te koristi li stranica kojom se prenose povjerljivi podaci HTTPS protokol.

Kada govorimo o e-pošti, korisnicima se savjetuje da nikada ne otvaraju poveznice ili datoteke koje su primili u spam porukama te da ne odgovaraju na iste. E-pošta je vrlo nesiguran softver i stoga je preporučljivo da se povjerljivi podaci, računi ili lozinke nikada ne šalju u poruci e-pošte. Od znatne pomoći mogu biti softveri za filtriranje spam poruka. Ti programi mogu prepoznati lažne poruke i tako smanjiti broj sumnjivih i neželjenih poruka koje korisnici primaju. Na svojim računalima korisno je imati i antivirusni i antyspyware programe koji mogu prepoznati maliciozne softvere. Sumnjive aktivnosti moguće je uočiti i praćenjem Internet prometa, no za to je potrebno koristiti vatrozid (eng. firewall). Vatrozid štiti korisničke podatke od neautoriziranih korisnika blokiranjem i zabranom prometa prema precizno

određenim nizom pravila koje definira sam korisnik. Jedan krivi klik može rezultirati preuzimanjem malicioznog softvera koji prati sve što korisnik radi na svom računalu pa čak i aktivirati Web kameru te ih snimati. Trik koji pomaže u takvim slučajevima je lijepljenje trake preko kamere.

Osobne podatke korisnici nikada ne bi trebali davati, kako ni u porukama, tako ni putem telefona osim u slučaju kako je korisnik sam nazvao i siguran je da broj nije lažan. Ako korisnik primi poziv i zatraže ga podatke, savjetuje se da se od pozivatelja zatraži ime i broj tako da ih se može naknadno kontaktirati. Tek nakon što se provjeri broj s kojeg je primljen poziv veća je sigurnost da osobni podaci korisnika neće biti zloupotrijebljeni. Ako je broj lažan ili se ne može provjeriti sigurnije je ne otkrivati podatke. Korisnik u svakome trenutku ima pravo znati zašto se od njega traže određeni podaci i to je vrlo dobar argument ukoliko ne želi ostaviti više podataka nego što je potrebno.

Još jedno pravilo koje bi trebali primjenjivati je korištenje jakih i složenih lozinki, ali i različitih lozinki za sve račune i aplikacije. Ako se koriste iste lozinke za sve račune onda se lako može pristupiti velikom broju privatnih informacija. Savjet je da se za lozinke koriste i velika i mala slova, brojevi te simboli jer je tada manja vjerojatnost da će ju netko pogoditi.

Jedan od savjeta je i korištenje najnovijih verzija operacijskog sustava te redovita nadogradnja (eng. update) sustava. U novijim verzijama pokušavaju se popraviti svi sigurnosni propusti, no kako se novi napadi otkrivaju svakoga dana, preporučuje se i korištenje snažnih sigurnosnih programa za dodatnu zaštitu. Najefikasniji oblik obrane od pokušaja prijevare i napada putem Interneta je znanje o istima. Iz tog razloga korisnici bi trebali pratiti koje prijevare kruže Internetom kako bi ih na vrijeme prepoznali i zaštitili se.

Zlouporaba osobnih podataka i krađa identiteta kazneno su djelo i ako dođe do povrede prava nužno je odmah reagirati i slučaj prijaviti policiji. Osoba čiji su osobni podaci zloupotrijebljeni može podnijeti zahtjev za zaštitu prava, a u Republici Hrvatskoj za to je zadužena Agencija za zaštitu osobnih podataka.

4. Moguća mjesta napada na elemente aplikacija

Postoje različiti načini na koje napadač može dobiti pristup sustavu i iskoristiti njegove slabosti. Tipovi napada koji se mogu izvršiti su napadi na operacijski sustav, lošu konfiguraciju, programski kod i napadi na aplikaciju. Kod napada na operacijske sustave, napadači traže ranjivosti u izradi, implementaciji ili konfiguraciji operacijskog sustava. Napadačima olakšava činjenica da suvremeni operacijski sustavi, zbog svoje složenosti, imaju veliki broj ranjivosti implementiranih u izradi. Najčešći napadi na operacijske sustave su prodori u datotečni sigurnosni sustav i u implementacijske mrežne protokole te napad na sustav autentikacije, a kako bi dobili pristup napadači se služe cracking-om zaporki i enkripcijskih mehanizama.

Svaka promjena sustava koji je loše konfiguriran za posljedicu ima nesiguran sustav i prije nego se takav sustav isporuči, od administratora se očekuje da promjeni konfiguraciju. U protivnom se inicijalne postavke mogu iskoristiti za napad, a ranjivosti sustava mogu rezultirati preuzimanjem sustava ili drugim ilegalnim aktivnostima. Napadi na programski kod odnose se na iskorištavanje inicijalnih postavki koda i različitih biblioteka (eng. libraries) koje programeri koriste kako bi reducirali razvoj i cijenu aplikacije. Programeri često nemaju dovoljno vremena za razvoj aplikacije te ona dolazi sa velikim brojem funkcionalnosti koje moraju izvršavati složene zadatke. Zbog toga, sigurnost je samo posebna funkcionalnost aplikacije, a za cjelovito ispitivanje aplikacije prije nego se ona isporuči naručitelju ne preostaje puno vremena.

Elementi Web aplikacija koji su moguća mjesta napada su registracija i prijava te uloge korisnika, košarica, galerija, vođenje dnevnika rada i sl. Uzmemo li u obzir da svaka Web aplikacija mora biti povezana na neku bazu podataka, ta baza također postaje jedno od mogućih mjesta napada. Napadom na bazu, napadači mogu doznati korisničko ime i lozinku računa a potom i sve ostale korisnikove podatke koji su navedeni u aplikaciji. Kako bi bilo teže doznati lozinku korisnikovog računa potrebno ju je dodatno zaštititi, na primjer korištenjem hash algoritama i dodavanjem dodatnog stringa sol-i (eng. salt) koje ćemo detaljnije opisati u nastavku.

4.1. Napadi vezani za autentikaciju

Autentikacija (eng. Authentication) ili kraće „AuthN“ je proces u kojem subjekt dokazuje da je ono što zapravo tvrdi. Taj proces se sastoji od dva koraka: identifikacije i potvrde. Identifikacija omogućuje subjektu da tvrdi da je određeni subjekt, dok potvrda omogućuje da tu tvrdnju potvrdi. [2] Drugim riječima, autentikacija je postupak potvrde identiteta korisnika.

Prilikom korištenja aplikacije svaki korisnik identificira se svojim korisničkim imenom i lozinkom temeljem kojih mu je dozvoljeno korištenje svih ili samo određenih dijelova aplikacije. Web autentikacija može se provesti na dva načina, a to su internom autentikacijom Web poslužitelja i vlastitom autentikacijom s formularom, bazom podataka i pohranom privremenih podataka u sjednicu (eng. session). Kod interne autentikacije Apache poslužitelja, cijeli sustav temelji se na datoteci .htaccess u kojoj se određuje konfiguracija autentikacije. Autentikacija korisnika korištenjem sesije započinje kreiranjem sjednice prije izvršavanja bilo kojeg sadržaja u skripti, nakon čega korisnik ima mogućnost korištenja aplikacije.

4.1.1. Napad grubom silom

Napad grubom silom (eng. Brute force attack) ili napad uzastopnim pokušavanjem je jednostavna, ali uspješna tehnika rješavanja problema koja se sastoji od sustavnog pronalaženja svih mogućih kandidata za rješenje i isprobavanja svakog od njih. [3] Ova vrsta napada koristi se u svrhu otkrivanja korisničkog imena, lozinke ili sigurnosnog broja kreditne kartice. Web aplikacija može biti na meti napadača ukoliko dozvoljava neograničen broj pokušaja prijave korisnika u sustav. Napadi mogu biti uspješni ukoliko aplikacija nema dovoljnu kontrolu nad kvalitetom lozinki. Točnije, kratke lozinke ili lozinke koje su identične korisničkom imenu vrlo su česta pojava, a vrijeme koje je napadaču potrebno da dozna takve podatke najviše ovisi o broju znakova od kojih se ti podaci sastoje.

Ukoliko aplikacija ima veliki broj korisnika, napadači se koriste rezerviranim napadom grubom silom. Ova metoda pretpostavlja da više korisnika ima iste lozinke pa napadači pokušavaju doznati korisnička imena koja im pripadaju.

4.1.2. Napadi na propuste u dizajnu autentikacije

Dizajn autentikacije je prva linija obrane od napada koju programeri mogu implementirati u aplikaciji. Propusti u dizajnu kojima aplikacija privlači napadače su opširne poruke o neuspjehu, nesiguran prijenos podataka HTTP protokolom te implementacija funkcionalnosti za promjenu lozinke i zaboravljene lozinke.

Opširne poruke o neuspjehu mogu pomoći napadačima pri otkrivanju informacija o korisničkim podacima te provjeru točnosti korisničkoga imena. Prijenos podataka HTTP protokolom šalje nešifrirane korisničke podatke što omogućuje napadačima presretanje istih raznim alatima za osluškivanje mreže.

Ukoliko korisnici koriste različite lozinke za različite aplikacije koje koriste postoji mogućnost da neke zaborave. Većina aplikacija omogućuje korisnicima obnovu lozinke, no kako je ova funkcionalnost prednost za korisnike, tako je i potencijalna prijetnja ako se nađu na meti napadača. Mehanizmi za oporavak lozinke često dozvoljavaju neograničen broj pokušaja pogađanja odgovora na tajno pitanje ili mu umjesto odgovora na pitanje omogućava unos savjeta kako da se sjeti lozinke. Korisnici često izabiru ona pitanja s malim skupom odgovora koji napadačima omogućuju lako pogađanje lozinke.

4.1.3. Napadi na propuste u implementaciji autentikacije

Propusti u implementaciji teže se uočavaju nego propusti u dizajnu, no za razliku od njih, ovi propusti dovode napadače do mogućeg zaobilaženja čitave sigurnosti aplikacije. Propusti koji se mogu dogoditi prilikom implementacije autentikacijskog mehanizma su ne kvalitetna implementacija rukovanja iznimkama, višeslojna autentikacija te autentikacija korisnika korištenjem metode odgovora na tajna pitanja. Višeslojna autentikacija koja nema kontrolu redoslijeda izvršavanja koraka stvara mogućnost napadaču da mijenja podatke u prethodnim koracima što omogućava preskakanje nekih od potrebnih koraka autentikacije. Isto tako, ukoliko aplikacija za svakoga korisnika generira nova pitanja, omogućuje napadaču da prolazi kroz sva pitanja te odgovori na ono na koje zna odgovor.

4.2. Napadi vezani uz autorizaciju

Autorizacija (eng. Authorization) ili kraće „AuthZ“ je proces utvrđivanja ima li subjekt dozvolu za izvođenje određenih operacija na zaštićenom resursu. Koristimo pojam subjekta jer osim osoba, subjekti mogu biti i Web servisi, baze ili druga računala. Resurse nam predstavljaju zaštićeni podaci ili funkcionalnosti koje subjekt želi koristiti. [2]

Nedovoljnom autorizacijom, napadaču se omogućava pristup funkcionalnostima aplikacija koje bi mu trebale biti nedostupne što ugrožava sigurnost Web aplikacije. Velika količina informacija koje se nalaze na Web aplikacijama su vrlo osjetljivi korisnički podaci i kako se radi o privatnim i povjerljivim podacima, neautorizirane osobe ne bi smjele imati pristup istima. Autorizacija se iz toga razloga koristi za ograničavanje pristupa i filtriranje podataka u skriptama, a korisnici aplikacije ovisno o razini autorizacije imaju različita ograničenja i različitu razinu pristupa informacijama.

4.2.1. Prisilno pregledavanje

Prisilno pregledavanje (eng. Forced browsing) je napad na autorizaciju kada napadač manualno unosi URL za resurs kojem ne može pristupiti putem aplikacijskog sučelja. [2]

Prema OWASP-u, cilj ovoga napada je pristup resursima koji se ne odnose na aplikaciju, ali su i dalje dostupni. Pri traženju tih podataka, napadači se često koriste tehnikama napada grubom silom, a pretraživanje se vrši nad privremenim direktorijima, konfiguracijskim datotekama te sigurnosnim kopijama. U tim datotekama mogu se nalaziti vrijedni i osjetljivi podaci koji napadačima služe za planiranje jačih napada. Prisilno pregledavanje napad je koji se može izvoditi ručno, već spomenutim upisivanjem URL adrese u adresnoj traci Web preglednika kada su stranice aplikacijskih indeksa temeljene na generiranim brojevima predvidljivih vrijednosti ili pomoću automatiziranih alata za zajedničke datoteke i nazive direktorija.

4.2.2. Falsifikacija parametara

Falsifikacija Web parametara (engl. Web Parameter Tempering) je vrsta napada kada napadač falsificira parametre zahtjeva prije nego li zahtjevi stignu Web aplikaciji. [2] Manipulacijom parametara pokušavaju se izmijeniti podaci aplikacije koji se najčešće pohranjuju u kolačićima, formama, skrivenim poljima ili unutar URL parametara, a služe za povećanje funkcionalnosti i kontrole aplikacije. Napadači izvode ovakvu vrstu napada kada želi iskoristiti aplikaciju za vlastitu koristi ili ako žele napasti treću osobu koristeći napad čovjeka u sredini. Uspjeh falsifikacije parametara može rezultirati drugim posljedicama kao što su XSS napad, napad ubrizgavanjem SQL naredbama (eng. SQL Injection), uključivanje datoteka i napad otkrivanja staze.

4.2.3. Manipulacija HTTP zaglavlja

HTTP protokol (eng. HypertextTransfer Protocol) je komunikacijski protokol koji služi za pristup podacima na Internetu, a danas ga koriste sve Web aplikacije. HTTP komunikacija temelji se na porukama koje razmjenjuju klijent i poslužitelj. Poruka koju klijent šalje poslužitelju dolazi u obliku zahtjeva, a poslužiteljeva poruka klijentu u obliku odgovora. Web aplikacija može biti na meti napada manipulacijom podataka HTTP zaglavlja ukoliko vjeruje podacima koji se nalaze u zaglavlju zahtjeva, a kreiran je od strane korisnika. Prilikom napada, napadač mijenja metapodatke o zahtjevu unutar HTTP zaglavlja. Na slici 1. primjer je snimljenog HTTP zaglavlja, a pomoću podatka Accept-Language može se izvesti napad na bazu podataka, ako aplikacija koristi direktni unos kao upit na bazu.

Slika 1. HTTP zaglavlje snimljeno programom Wireshark [autorski rad]

4.3. Napadi na upravljanje sesijom

Sesija ili sjednica (eng. Session) je spremište podataka koje se nalazi na poslužitelju i vezano je uz rad pojedinačnog korisnika. Podaci koji se spremaju u sesiji dostupni su kroz više različitih skripata aplikacije. Pri tome, podaci se odnose na rad jednog korisnika u jednom, odabranom Web pregledniku. Sesija traje sve dok ne istekne vrijeme trajanja sesije određeno postavkama aplikacije ili do gašenja i ponovnog pokretanja Web poslužitelja. Cilj napada je manipulacija informacijama kako bi se pristupilo zaštićenim resursima. Napad na sesiju uzrokuju propusti u implementaciji, tj. dizajnu mehanizma za upravljanje sesijom. U napad na sesiju spada i propust u implementaciji odjave korisnika kada je prilikom odjave omogućeno brisanje kolačića na strani korisnika, a na strani poslužitelja i dalje postoje identifikatori sesije.

4.3.1. Trovanje kolačića

Kolačići sadržavaju bitne informacije koje omogućuju Web mjestima koje korisnik posjećuje provjeru autentičnosti korisničkog identiteta. Kolačiću mogu pristupiti i neovlaštene osobe, a ako ne postoje sigurnosne mjere, napadači mogu pregledati kolačić i doznati podatke s Web stranice koja ih je poslala. Trovanje kolačića (eng. Cookie poisoning) je metoda izmjene kolačića, tj. osobnih podataka na računalu korisnika, od strane napadača kako bi se dobile informacije o korisniku u svrhu krađe identiteta. Napadač koristi otkrivene podatke za otvaranje novih računa ili za pristup postojećim računima korisnika. [4] Kako bi se zaštitili od napada trovanjem kolačića, Web stranice bi trebale štititi kolačiće enkripcijom prije nego ih pošalju na korisničko računalo.

4.3.2. Krađa i predvidljivost identifikatora sesije

Najčešća vrsta napada na upravljanje sesijom je krađa identifikatora sesije (eng. Stealing Session ID). Prilikom napada, napadač pokušava ukrasti identifikator sesije drugog korisnika u svrhu daljnje impersonalizacije. [2] Krađa identifikatora moguća je prisluškivanjem mreže koja nema zaštićen prijenos podataka, ali i drugim vrstama napada kao što je XSS

napad (eng. Cross-Site Scripting) koji će biti detaljnije opisan u nastavku. Propust prilikom generiranja identifikatora omogućuje napadačima da predvide vrijednost ID-a što rezultira ulaskom u aplikaciju bez autentikacije i autorizacije korisnika. Ovaj propust događa se ukoliko su identifikatori sesije nastali transformiranjem korisničkih podataka kao što su korisničko ime, e-mail ili uloga korisnika ili su vezani uz vrijeme nastanka identifikatora. U oba slučaja podaci se šifriraju, no napadači analiziranjem podataka lako otkrivaju prave vrijednosti identifikatora. Još jedan od propusta je i loš generator brojeva kojim se generiraju predvidljivi identifikatori.

4.3.3. Krađa sesije

Krađa ili otmica sesije (eng. Session Hijacking) često se naziva i otmicama kolačića, a napad se temelji na metodi iskorištavanja valjanje sesije računala pri čemu se koriste ukradenim identifikatorom sesije. Napadači otmicom sesije, na neovlašten način, pokušavaju doći do informacija u računalnom sustavu. Kako se na Web stranicama kolačići koriste za održavanje sesije, napadač pomoću posredničkog računala ili izravnim pristupom spremljenim kolačićima na žrtvinom računalu može doći do podataka koji služe za autentikaciju korisnika. Podvrste ovog napada su prisluškivanje sesije (eng. Session Sniffing) i napad čovjeka u sredini (eng. Man-in-middle).

4.3.4. Fiksacija sesije

Napad fiksacijom sesije (eng. Session Fixation) omogućuje napadaču da otme valjanu sesiju korisnika. On popravlja uspostavljenu sesiju na Web pregledniku žrtve pa napad počinje prije nego se korisnik prijavi u sustav. Prilikom napada istražuju se ograničenja kojima Web aplikacija upravlja identifikatorom sesije. Prijavom korisnika u sustav aplikacija mora provjeriti autentičnost korisnika i dodijeliti mu novi ID, no prilikom napada ne dodjeljuje se novi ID već se omogućava korištenje postojećeg ID-a sesije. [5] Napadač upotrebljava validni identifikator i prosljeđuje ga korisniku putem kolačića, URL parametara ili skrivenom formom. Prijava korisnika s podmetnutim ID-om omogućava napadaču otmicu sesije i samim time krađu korisničkih podataka.

4.4. Napadi na baze podataka

Baza podataka je kolekcija podataka, ograničenja i operacija koja reprezentira neke aspekte realnog svijeta, a percipira se kao skup povezanih relacija, odnosno tablica. [6] Baze podataka se koriste kako bi svi podaci bili na jednom mjestu i kako bi olakšali pristup do potrebnih podataka. Kako su u bazama podataka pohranjene sve informacije o korisnicima tako one postaju čestim mjestima napada.

4.4.1. Napad ugniježđenim SQL naredbama

Napad ubrizgavanjem SQL naredbi (eng. SQL Injection) podrazumijeva umetanje ili ubrizgavanje SQL upita putem ulaznih podataka od klijenta do Web aplikacije. uspješni napad SQL ubrizgavanjem rezultira čitanjem i mijenjanjem osjetljivih podataka iz baze te izvršavanjem administrativnih operacija. [7] Na taj način ugrožavaju se bitna svojstva Web aplikacija, a to su tajnost, integritet i dostupnost. Tajnost se ugrožava krađom osjetljivih podataka, integritet aplikacije postaje narušen ako se unutar baze podataka dodaju novi ili izmjenjuju stari podaci, a ukoliko napadač obriše neku od relacija u bazi podataka ona prestaje biti dostupna za korištenje.

Kao primjer SQL ubrizgavanja možemo navesti autentikaciju korisnika u sustav koristeći se dolje navedenim PHP kodom. Prilikom prijave provjerava se postoji li korisnik u bazi podataka te se korisničko ime i lozinka čitaju iz GET varijabli. Ako upit vrati odgovor tada korisnik postoji i ima pristup aplikaciji.

```
<?php
$korisnik = $_GET['korisnik'];
$lozinka = $_GET['lozinka'];
$upit = ''SELECT * FROM korisnici WHERE korisnik = '$korisnik' AND lozinka = '$lozinka' '';
$rezultat = mysql_query($upit);
?>
```

Prilikom napada, napadač ne mora znati niti korisničko ime niti lozinku kako bi provalio u sustav, već upisivanjem uvijek istinite SQL tvrdnje dobiva pristup sustavu. Takav upit izgledao bi ovako: *SELECT * FROM korisnici WHERE korisnik = '1' AND lozinka = '1' OR '1' = '1' '';*
Ovakav upit će se tretirati valjanim, a podaci s kojima će napadač biti prijavljen u sustav odgovarat će podacima prvoga korisnika koji se nalazi u bazi podataka. Kako bi se zaštitili od SLQ ubrizgavanja dovoljno je koristiti ugrađenu funkciju mysql_real_escape_string(). Funkcija u poljima za unos pronalazi znakove koji se koriste u mysql naredbama i ispred njih postavlja znak „\“ kako bi spriječila njihovo interpretiranje kao dio SQL naredbe. Korištenje funkcije u PHP kodu izgledalo bi ovako:

$korisnik = mysql_real_escape_string($_GET['korisnik']);

$lozinka = mysql_real_escape_string($_GET['lozinka']); [8]

4.4.2. Slijepi napad ugniježđenim SQL naredbama

Slijepi napad SQL naredbama (eng. Blind SQL Injection) je podvrsta napada ugnježđivanjem SQL naredbi, a temelji se na upitima nad bazom podataka pitanjima koja su ili istinita ili lažna. Ova vrsta napada često se upotrebljava kada sustav ne ispisuje dovoljno detalja o pogreškama jer na taj način napadači mogu saznati koji su podaci istiniti, a koji ne i na temelju tih podataka s vremenom može otkriti važne informacije vezane uz bazu podataka.

4.5. Napadi na web preglednike

Web preglednici imaju mnogo ranjivosti zbog kojih se često nalaze na meti napadača. Ranjivosti se iskorištavanju kako bi se nanijela šteta računalu korisnika, ukrali njegove podatke ili napali drugu stranicu koristeći pritom korisnikov Web preglednik. Sigurnosti i zaštiti poslužitelja mogu pomoći redovno nadograđivani Web preglednici uz sigurno oblikovanje Web aplikacije. [8]

4.5.1. HTML ubrizgavanje

Prema OWASPU, HTML ubrizgavanje ili HTML injekcija (eng. HTML Injection) je vrsta napada ubrizgavanjem koja se pojavljuje kada napadač ima mogućnost kontrole nad unosom podataka i može ubrizgati proizvoljni HTML kod u ranjivu Web stranicu. Ova ranjivost može imati mnoge posljedice kao što je otkrivanje korisničkih kolačića sesije koji bi se mogli upotrijebiti za lažno predstavljanje žrtve ili omogućiti napadaču izmjenu sadržaja stranice koju vide korisnici. Također, napadač na Web stranicu ili aplikaciju može dodati formu za prijavu korisnika te podatke koje korisnici unesu na tu lažnu formu prikupljati u svoju bazu podataka čime su korisnici ugroženi jer bi došlo do krađe njihovih podataka.

4.5.2. XSS napad

Izvršavanje skripte s drugog računala ili XSS (eng. Cross-Site Scripting) je vrsta napada u kojoj napadač ima mogućnost umetanja skripata s vlastitim kodom unutar ranjive Web aplikacije. Pristupom stranici od strane korisnika, Web preglednik preuzima i pokreće skriptu koja je umetnuta. [10] postoje tri vrste XSS napada:

- Spremljeni XSS napad (eng. Stored XSS attack)
- Reflektirani XSS napad (eng. Reflected XSS attack)
- DOM baziran XSS napad (eng. DOM-based XSS attack)

Spremljeni XSS najopasnija je vrsta XSS napada u kojoj napadač trajno pohranjuje zloćudnu skriptu unutar aplikacije kojega aplikacija bez ikakve provjere prikazuje korisnicima.

Reflektirani napad je napad pri kojemu napadač pohranjuje skriptu u Web stranicu kao odgovor na korisnikov zahtjev te na taj način omogućuje pristup korisničkim podacima koji se nalaze unutar aplikacije. DOM bazirani XSS napad omogućen je kada skripte unose korisnikove podatke u DOM (eng. Document Object Model). Posljedice napada su iste kao i kod reflektiranog XSS napada.

4.5.3. CSFR napad

Krivotvorenje zahtjeva s drugog računala ili CSRF (eng. Cross-Site Request Forgery) je vrsta napada na Web stranicu koja se izvodi posredstvom klijentskog programa. Ukoliko se zloćudna stranica učita u nesiguran Web preglednik moguće je pokrenuti zahtjev prema drugoj ranjivoj stranici u ime korisnika Web preglednika. Pritom zloćudna stranica mora sadržavati kod za pokretanje takvog zahtjeva koji će se s ostatkom koda te stranice automatski učitati i pokrenuti u Web pregledniku. Ukoliko ranjiva stranica ne izvodi provjere sigurnosti i identiteta korisnika, zahtjev poslan posredno sa zloćudne stranice obradit će se kao da ga je poslao klijent. [8]

4.5.4. Napad na korisničko sučelje

Napad na korisničko sučelje (eng. Clickjacking ili UI Redress attack) je napad u kojemu napadač koristi više transparentnih slojeva s ciljem prijevare korisnika. Na ovaj se način korisnici klikom na prikazani gumb ili poveznicu usmjeravaju na druge stranice od one koju je namjeravao otvoriti ili koristiti.

5. Najbolja praksa za autentikaciju, autorizaciju i dnevnik rada

Kako je navedeno u prethodnom poglavlju, Web autentikacija može se provesti na dva načina, a to su internom autentikacijom Web poslužitelja i vlastitom autentikacijom s formularom, bazom podataka i pohranom privremenih podataka u sjednicu (eng. session) što su ujedno i najbolje prakse za provođenje autentikacije.

Kako bi se mogla koristiti interna autentikacija Apache poslužitelja potrebno je konfigurirati httpd.conf datoteku te opciju AllowOverride postaviti na AuthConfig. Time se omogućuje autentikacija .htaccess metodom na kojoj se temelji cijeli sustav. U datoteci .htaccess nalaze se opcije AuthType Basic koji određuje tip autentikacije, AuthUserFile koji sadrži putanju do datoteke koju je potrebno autenticirati, AuthName koji predstavlja naziv autentikacije i opciju Require valid-user koja je potrebna za autentikaciju sustava. Da bi se autentikacija mogla koristiti potrebno je kreirati i .htpasswd datoteku u kojoj će se nalaziti podaci korisnika za autentikaciju. Autentikacija korisnika provodi se kada korisnik zatraži prikaz dokumenata koji se nalaze na direktoriju. Web poslužitelj tada pokreće postupak autentikacije i prikazuje formu za unos autentikacijskih podataka korisnika. Podaci o korisniku se nakon uspješne autentikacije upisuju u varijablama okoline, a vrijede sve dok se ne zatvori preglednik u kojemu je autentikacija obavljena.

Autentikacija korisnika korištenjem sesije započinje kreiranjem sjednice pozivom funkcije *session_start();* prije izvršavanja bilo kojeg sadržaja u skripti, nakon čega korisnik ima mogućnost korištenja aplikacije. Podaci o korisniku se upisuju i preuzimaju putem varijable *$_SESSION*. Sesija se može obrisati pozivnom funkcije *session_destroy();,* a moguće je i brisanje samo podataka sesije pozivom funkcije *session_unset();*. Poslužitelj pamti vrijeme zadnje aktivnosti korisnika i na temelju neaktivnost korisnika sesija se briše. Vrijeme neaktivnosti nakon kojega se briše sesija određuje se postavkama, a brisanje svih sesija automatski se vrši gašenjem i ponovnim pokretanjem Web poslužitelja.

Kada se govori o autorizaciji, Web aplikaciju dijelimo na javni i privatni dio. Kako privatni dio koriste samo administratori aplikacije (neke dijelove i moderatori) treba osigurati da taj dio bude zaštićen. Pri tome, Web aplikaciju možemo zaštititi različitim metodama kao što su autentikacijska metoda, metoda na temelju uloge ili uloga, metoda na temelju ovlaštenja, metoda na temelju akcija ili mješovitom metodom.

Autentikacijska metoda se zasniva na vrsti korisnika koji mogu biti anonimni ili prijavljeni što se provjerava u zapisima prijavljivanja u sesiji. Metoda na temelju uloge zasniva se, kako i sam naziv metode kaže, temeljem dodijeljene uloge korisniku. Baza podataka sadrži dvije tablice, „Korisnik“ i „Uloga“ koje su povezane tako da tablica „Korisnik“ sadrži stupac „uloga“ koji je vanjski ključ na istoimenu tablicu. Za izvršavanje određene skripte autorizacija se provodi provjeravanjem zahtijevane uloge korisnika. Slično ovoj metodi, metoda autorizacije na temelju uloga sastoji se od pomoćne, treće tablice „Korisnik_Uloga“ koja sadrži vanjske ključeve na tablice „Korisnik“ i „Uloga“, a sadržava zapise uloga pojedinog korisnika te se za izvršavanje skripte provjeravaju uloge korisnika. U ove dvije metode, skripta se može izvršavati za samo jednu ulogu ili za više njih. Metoda autorizacije na temelju ovlaštenja zasniva se na dodjeli dozvola korisniku za korištenje pojedine skripte. U bazi podataka stvara se nova tablica koja sadrži zapise skripata pojedinog korisnika i sadrži vanjski ključ prema tablici „Korisnik“. Skripta prije izvršavanja provjerava ima li korisnik dozvolu za izvršenje te skripte. metoda zaštite na temelju akcija podrazumijeva kreiranje tablice „Korisnik_Akcija“ koja sadrži vanjske ključeve na tablice „Korisnik“ i „Akcija“, te dodjelu akcija korisnicima koje smiju izvršavati. Mješovita metoda se temelji na grupama i korisnicima te dodjeli određenih skripti ili akcija za izvršavanje. Skripta se izvršava korisniku ako je član grupe koja ima dozvolu za izvršavanje.

Autorizacija u odnosu na skriptu mogu biti statičkog i dinamičkog tipa. Kod metode autorizacije statičkog tipa, skripte sadrže podatke o zahtijevanom elementu autorizacije te ih provjeravaju u odnosu na korisnikove atribute dok se kod dinamičkog tipa provjeravaju dozvole na temelju identifikatora ili naziva u odnosu na podatke iz tablice.

Također, vrlo je važno praćenje rada korisnika, odnosno kreiranje dnevnika rada (eng. log) gdje se podaci uobičajeno upisuju u bazu podataka. Kako postoji potreba za praćenjem rada korisnika, Web aplikacije koriste dnevničke zapise koji se mogu provesti na dva načina, a to su Apache log te vlastita evidencija. Apache log se postavlja u datoteci konfiguracija httpd.conf gdje je potrebno konfigurirati format zapisa podataka, a za pretragu podataka zapisanih u Apache log-u koriste se razni alati koji su razvijeni za tu namjenu. Web aplikacije mogu koristiti i vlastiti dnevnik rada gdje se željeni podaci kao što su podaci o korisniku, URL, adresa računala korisnika, vrijeme pristupa i slično upisuju u bazu podataka. Dnevnik rada služi za analizu raznih podataka kao što je ispitivanje funkcionalnosti aplikacije, praćenje rada ne samo korisnika, nego i sustava pa i testiranja istoga.

6. OWASP

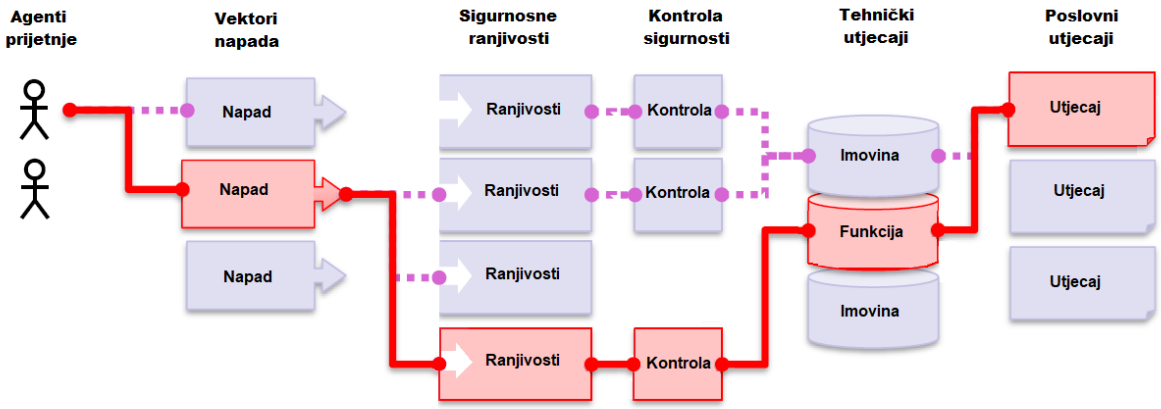
Otvoreni sigurnosni projekt Web aplikacija (eng. Open Web Application Security Project) ili skraćeno OWASP je svjetska, neprofitna i dobrotvorna organizacija čiji je cilj poboljšanje sigurnosti softvera. OWASP pruža nepristrane ideje o sigurnosti aplikacija kako pojedincima tako i korporacijama, sveučilištima, vladinim agencijama i drugim organizacijama širom svijeta. Kako je OWASP otvorena organizacija, svatko je slobodan ne samo koristiti besplatnu i otvorenu licencu softvera, nego i sudjelovati u kreiranju materijala OWASP-a. Neki su članovi jedni od najvećih i najboljih stručnjaka u području sigurnosti softvera.

Kako bi osigurali kvalitetu, OWASP broji tisuće aktivnih korisnika diljem svijeta koji pregledavaju promjene na Web mjestu. Također, zajednica od više od 45 tisuća volontera odgovara na pitanja koje korisnici mogu poslati korištenjem odgovarajuće forme. [11]

6.1. Sigurnosni rizici aplikacija

Kako bi naštetili organizacijama ili njihovom poslovanju, napadači mogu koristiti mnoge različite puteve kroz aplikaciju. Svaki od tih puteva predstavlja rizik koji može, ali ne mora biti ozbiljan kako bi privukli pozornost. Ponekad su putevi trivijalni za pronalaženje i iskorištavanje, no ponekad ih je izuzetno teško pronaći. Slično tome, šteta koju prouzroče napadači može biti ogromna i imati za posljedice probleme s poslovanjem ili može biti bez ikakvih posljedica.

Kako bi se utvrdio rizik neke organizacije mora se procijeniti vjerojatnost vezana uz svakog agenta koji prijeti napadom na aplikaciju, vektorom napada i sigurnosnim ranjivostima. Te procjene potrebno je kombinirati i s procjenom tehničkog i poslovnog utjecaja na organizaciju. Ukupni rizik određen je svim navedenim čimbenicima zajedno. Na slici 2. prikazani su sigurnosni rizici aplikacija kao i mogući putevi kojima se mogu kretati napadači.



Slika 2. Sigurnosni rizici aplikacija [13]

OWASP organizacija usredotočuje se na identificiranje najozbiljnijih rizika za mnogobrojne organizacije različitih namjena. Koristeći se jednostavnom shemom ocjenjivanja, OWASP daje generičke informacije o vjerojatnosti i tehničkom utjecaju za svaki od rizika. Shema ocjenjivanja temelji se na metodologiji ocjenjivanja procjene rizika, a prikazana je tablicom 1.

Agenti prijetnje	Iskoristivost	Prevalencija slabosti	Otkrivanje slabosti	Tehnički utjecaj	Poslovni utjecaj
Specifična aplikacija	Lagano	Rasprostranjen	Lagano	Ozbiljno	Specifična/ Poslovna aplikacija
	Prosječno	Uobičajen	Prosječno	Umjereno	
	Teško	Rijedak	Teško	Nizak	

Tablica 1. OWASP shema ocjenjivanja procjene rizika [13]

Svaka od organizacija jedinstvena je na svoj način, ima jedinstvene ciljeve i utjecaj, pa su tako i prijetnje jedinstvene toj organizaciji. Iako dvije različite organizacije za svoje podatke koriste isti ili vrlo slični sustav to ne znači da će prijetnje i tehnički utjecaj biti isti. Na temelju kritičnosti podataka, obje organizacije imaju prilagođene agente prijetnji i poslovne utjecaje za taj softver.

U idealnom sustavu, sve bi organizacije koristile jedan univerzalni sustav procjene rizika koji bi precizno mogao procijeniti sve rizike. No kako su sve organizacije jedinstvene, ranjivosti ključne za jednu organizaciju možda nisu važne nekoj drugoj organizaciji. Iz tog razloga, OWASP shema procjene rizika predstavlja osnovni okvir koji se mora prilagoditi određenoj organizaciji.

6.2. OWASP metodologija procijene rizika

Otkrivanje ranjivosti vrlo je važno, no jednako važna je i procjena rizika za poslovanje. Pomoću modela prijetnji mogu se identificirati sigurnosna pitanja u arhitekturi i dizajnu aplikacija. Pregledom koda i penetracijskim testiranjem mogu se pojaviti novi sigurnosni problemi, a postoji i mogućnost otkrivanja problema nakon što aplikacija dospije na tržište čime postaje ugrožena. OWASP metodologijom procjene rizika moguće je procijeniti ozbiljnost svih rizika za poslovanje i donijeti odluku o tome što učiniti s tim rizicima. Ovakvim pristupom moguće je uštedjeti vrijeme te osigurati da poslovanje ne gubi vrijeme na manje rizike ignorirajući pri tome ozbiljnije rizike koji su manje razumljivi.

Postoji mnogo različitih pristupa analizi rizika, a u nastavku je opisan standardni model koji se temelji na činjenici da je rizik jednak umnošku vjerojatnosti i utjecaja. Kako bi se utvrdio ukupan rizik, ovaj model provodi se u 6 faza:

- Identificiranje rizika
- Faktori za procjenu vjerojatnosti
- Faktori za procjenu napada
- Utvrđivanje ozbiljnosti rizika
- Odlučivanje što popraviti
- Prilagodba modela procjene rizika.

Identificiranje rizika

Identificiranje sigurnosnog rizika kojega treba ocijeniti, prvi je korak OWASP metodologije. Prilikom identifikacije, tester prikuplja informacije o uključenom agentu prijetnji, napadu koji će se koristiti, uključenoj ranjivosti i utjecaju uspješnog iskorištavanja na poslovanje. Kako postoji više agenata, odnosno više skupina mogućih napadača. korištenje najlošijih mogućih scenarija rezultira najvećim ukupnim rizikom pa se preporučuje takav pristup pri identifikaciji rizika.

Faktori za procjenu vjerojatnosti

Prepoznavanjem potencijalnih rizika potrebno je procijeniti njegovu ozbiljnost. To je mjera kojoj se prikazuje koliko je vjerojatno da će ranjivost biti otkrivena i iskorištena od strane napadača. Procjena vjerojatnosti u ovoj fazi nije precizna već je dovoljno odrediti je li vjerojatnost niska, srednja ili visoka. Faktori koji pomažu testerima odrediti vjerojatnosti imaju skup opcija, a svaka opcija ima ocjene od 0 do 9 gdje je 0 najniža, a 9 najviša ocjena. Te ocjene se kasnije koriste za procjenu ukupne vjerojatnosti.

Prvi skup faktora povezan je s uključenim agentom prijetnji, a cilj je procijeniti vjerojatnost uspješnog napada skupine mogućih napadača.

Faktori agenata prijetnji:

- Razine vještina – Koliko je tehnički kvalificirana skupina agenata prijetnji?
- Motivi - Koliko je motivirana skupina za otkrivanje ranjivosti?
- Prilike - Koji resursi su potrebni za pronalazak i iskorištavanje ranjivosti?
- Veličina – Kolika je veličina, tj. brojnost agenata?

Kod razina vještina najnižom ocjenom ocjenjuju se skupine prijetnji kada ne postoje tehničke vještine dok se najvišom ocjenom ocjenjuju vještine penetracijske sigurnosti. Između ovih ocjena nalaze se skupine agenata nekih tehničkih vještina (3), skupina naprednih korisnika računala (5) i skupina agenata s vještinama mreža i programiranja (6). Druga opcija

kod ocjenjivanja ovog faktora je motiv. Najnižu ocjenu ima motiv niske ili nikakve nagrade, ocjenom 4 je motiviranost s mogućom nagradom, a najvišu ocjenu ima motiv visoke plaće. Kod opcije prilika, ako su resursi skupi dodjeljuje im se najniža ocjena, a ako nije potreban pristup ili resursi dodjeljuje se najviša ocjena. Brojnost agenata ima najviše definiranih ocjena. Ukoliko u skupinu agenata prijetnji spadaju razvojni programeri ili administratori sustava dodjeljuje se ocjena 2, ako su to korisnici Intraneta ocjena 4, partnerima se dodjeljuje ocjena 5, autentičnim korisnicima ocjena 6, dok najvišu ocjenu 9 imaju anonimni korisnici Intraneta.

Drugi faktor je povezan s ranjivosti, a cilj je procijeniti vjerojatnost koliko je neka ranjivost uključena u otkrivanje i iskorištavanje, pod uvjetom kako je agent prijetnji odabran prethodnim faktorom.

Faktor ranjivosti:

- Jednostavnosti otkrića - Koliko je lako skupini agenata otkriti ranjivost?
- Jednostavnosti iskorištavanja - Koliko je lako skupini agenata iskoristiti ranjivost?
- Svijest - Koliko je poznata ranjivost prema odabranoj skupini agenata?
- Otkrivanja upada - Koliko je vjerojatno da će biti otkriven upad u sustav?

Prilikom određivanja prve opcije najniže ocjenjena mogućnost je nemoguće otkriće, teško otkriće ima ocjenu 3, lako otkriće ocjenu 7, dok je ocjenom 9 ocjenjeno otkriće dostupnim automatiziranim alatima. Istu najvišu ocjenu s istim objašnjenjem koristi i opcija jednostavnosti iskorištavanja, no najniža ocjena ove opcije je teorijska (1) što se odnosi na lakoću iskorištavanja ranjivosti. Kod opcije svijesti skupini agenata ranjivost može, ali ne mora biti poznata pa se ocjena 1 dodjeljuje nepoznatom , a ocjena 9 javnom znanju o ranjivosti. Opcija otkrivanja upada ocjenjuje se najnižom ocjenom za aktivnu detekciju u aplikaciji, a najvišom ocjenom za ne prijavljenu ranjivost.

Faktori za procjenu napada

Prilikom razmatranja utjecaja uspješnog napada, važno je shvatiti da postoje dvije vrste utjecaja. Prvi je tehnički utjecaj na aplikaciju, podatke koje koristi i funkcije koje pruža. Drugi i mnogo važniji je poslovni utjecaj na tvrtku i tvrtku koja upravlja aplikacijom. Kao i prethodna faza, svaki faktor ima skup opcija kojima se dodjeljuju ocjene od 0 do 9 koje će se kasnije koristiti za procjenu ukupne težine rizika.

Tehnički se utjecaj može podijeliti na čimbenike usklađene s tradicionalnim sigurnosnim područjima, a to su povjerljivost, integritet, dostupnost i odgovornost. Cilj je procijeniti veličinu utjecaja na sustav ako bi se ranjivost iskoristila.

Tehnički utjecajni faktori:

- Gubitak povjerljivosti – Koliko se podataka može otkriti i koliko je to osjetljivo?
- Gubitak integriteta – Koliko podataka može biti oštećeno?
- Gubitak dostupnosti – Koliko bi usluga mogla biti izgubljena i koliko je vitalna?
- Gubitak odgovornosti – Jesu li akcije agenata prijetnji moguće pratiti pojedincu?

Kod gubitka povjerljivosti najnižom ocjenom 2 prikazani su minimalni podaci koji nisu osjetljivi, minimalni kritični podaci te opsežni podaci koji nisu osjetljivi ocijenjeni su ocjenom 6, opsežni kritični podaci ocjenom 7, a najvišom ocjenom 9 ako su svi podaci otkriveni. Minimalno neznatni oštećeni podaci kod gubitka integriteta imaju najnižu ocjenu, a najvišu ocjenu imaju svi potpuno oštećeni podaci. Treću opciju ocjenjivanja, gubitak dostupnosti, ocjenjuje se sljedećim ocjenama: prekinute minimalne sekundarne usluge ocjenom 1, prekinute minimalne primarne usluge ocjenom te prekinute opsežne sekundarne usluge ocjenom 5, prekinute opsežne primarne usluge ocjenom 7 dok se sve potpuno izgubljene usluge ocjenjuju ocjenom 9. Gubitak odgovornosti može biti potpuno sljediv (ocjena 1), vjerojatno sljediv (ocjena 3) ili potpuno anoniman (ocjena 9).

Utjecaj poslovanja proizlazi iz tehničkog utjecaja, ali zahtijeva duboko razumijevanje onoga što je važno za tvrtku koja pokreće aplikaciju. Poslovni rizik opravdava ulaganja u

rješavanje sigurnosnih problema, a faktori koji se razmatraju zajednička su područja za sve organizacije.

Faktori poslovnog utjecaja:

- Financijska šteta – Kolika će biti financijska šteta nakon iskorištavanja?
- Oštećenje ugleda – Rezultira li iskorištavanje oštećenju ugleda?
- Neispunjenje – Kolika izloženost predstavlja nepoštovanje?
- Kršenje privatnosti – Koliko se osobnih podataka može otkriti?

Opcija financijske štete ocjenjuje se ocjenom 1 ukoliko je financijska šteta manja od troškova rješavanja ranjivosti, ocjenom 3 ako ima manji utjecaj na godišnju dobit, a ocjenom 7 ako je taj utjecaj značajniji. Najvišom ocjenom ocjenjuje se šteta koja uzrokuje stečaj organizacije. Kod oštećenja ugleda minimalna šteta ocjenjuje se ocjenom 1, gubitak glavnih računa ocjenom 4, gubitak imovine koja predstavlja buduće ekonomske koristi (eng. Goodwill) ocjenom 5 te oštećenje marke ocjenom 9. Neispunjenje ima definirane tri ocjene, 2 za manje prekršaje, 5 za jasno kršenje te 7 za prekršaje visokog profila. Ako se prilikom napada otkrivaju podaci jednog pojedinca razina opasnosti ocijenjena je ocjenom 3, ako se radi o stotinama ljudi ocjenom 5, tisuće ljudi ocjenom 7, a najviša opasnost je kršenje privatnosti milijuna ljudi.

Utvrđivanje ozbiljnosti rizika

U ovom koraku računa se ukupna težina za rizik, a temelji se na tome je li vjerojatnost niska, srednja ili visoka, baš kao i utjecaj. Skala od 0 do 9 podijeljena je u tri razine prikazane u sljedećoj tablici:

Ocjena	Razina
0 - <3	Nisko
3 - <6	Srednje
6 - 9	Visoko

Tablica 2. Ljestvica ukupnog rizika [14]

Ozbiljnost rizika može se procijeniti različitim metodama. Neformalna metoda temelji se na pregledu faktora i prikupljanju odgovora. Na taj način utvrđuju se ključni faktori koji utječu na rezultat. Ovom metodom tester može otkriti da je njihov prvi dojam bio pogrešan s obzirom na aspekte rizika koji nisu bili očiti. Formalniji proces ocjenjivanja faktora i utvrđivanja rezultata temelj je ponovljene metode. Cilj ove metode je dolazak do osjetljivih rezultata, a za njihov lakši izračun, proces je podržan automatiziranim alatima.

Procjenom vjerojatnosti i utjecaja dolazi se do konačnog stupnja ozbiljnosti za određeni rizik. Prilikom određivanja konačnog rezultata treba gledati podatke o poslovnom utjecaju, ukoliko su te informacije kvalitetne. Ako ne postoje ili testeri nemaju podatke o organizaciji, konačni rezultat utvrđuje se temeljem informacija o tehničkom utjecaju. Ovisno o razini vjerojatnosti i utjecaja donosi se konačni rezultat ukupne težine rizika prema sljedećoj tablici:

		Vjerojatnost		
		Niska	**Srednja**	**Visoka**
Utjecaj	**Nizak**	Zanemariva	Niska	Srednja
	Srednji	Niska	Srednja	Visoka
	Visok	Srednja	Visoka	Kritična

Tablica 3. Određivanje ukupne težine rizika temeljem vjerojatnosti i utjecaja [14]

Odlučivanje što popraviti

Nakon utvrđivanja ukupne težina rizika potrebno je sastaviti popis prioriteta o tome što treba popraviti. Prvo se utvrđuju najozbiljniji rizici iako ih treba duže ili skuplje popravljati, a zatim se rješavaju manje važni rizici. Također, neki rizici nisu vrijedni popravljanja, a neki gubitci ne samo da su očekivani, već su predviđeni i opravdani u troškovima popravljanja problema.

Prilagodba modela procjene rizika

Kako su sve organizacije jedinstvene potrebno je prilagoditi model procjene koji će vjerojatnije proizvesti rezultate dostatne te organizacije. Prilagodba modela može se provesti na nekoliko načina. Prvi način je dodavanje faktore prilikom kojeg tester odabire različite faktore koji bolje predstavljaju organizaciju. Također, tester može dodati i faktore vjerojatnosti kao što je algoritam šifriranja ili prozor prilika za napadača. Drugi način je prilagodba opcija. Korištenjem ovog načina prilagodbe model može biti mnogo učinkovitiji jer se opcije povezane sa svakim faktorom prilagode poslovanju određene organizacije. Ovaj model pretpostavlja da su svi faktori jednako važni, a pravi rezultati dobivaju se usporedbom ocjena proizvođača s ocjenama tima stručnjaka. Posljednji i najsloženiji način su faktori ponderacije gdje se izdvajaju faktori koji su značajniji za određenu djelatnost. Ova metoda koristi ponderirani prosjek, a model je moguće prilagoditi podudaranjem s ocjenom rizika za koje organizacija smatra da su točne.

7. Najčešće korištene metode napada na web aplikacije

OWASP organizacija prati napade na Web aplikacije i u skladu s rezultatima istraživanja svakih nekoliko godina objavljuje liste top 10 vrsti napada. Zadnji objavljeni popis izašao je 2017. godine. Kako većina Web aplikacija nije na najvišoj razini sigurnosti, napadači iskorištavaju njihove ranjivosti. Iako napadi štete korisnicima aplikacija zbog otkrivanja njihovih ne samo korisničkih, nego i osobnih podataka, napadi mogu pomoći programerima pri izradi sigurne Web aplikacije. Ukoliko se sigurnost pri izradi aplikacije ne promatra samo kao dodatna funkcionalnost, već se provodi na najvišoj razini prilikom pisanja koda lako je moguće da će aplikacija biti sigurna za korištenje. Svaki napad ukazuje na novi problem koje programeri nisu predvidjeli ili koji nisu implementirani, a svakom nadogradnjom sustava sprječavaju se daljnji napadi. Popisi najčešćih napada trebali bi biti putokaz programerima kako stvoriti siguran sustav, tj. Web aplikaciju koja će imati minimalne ranjivosti. Treba naglasiti kako se sigurnost ne odnosi samo na sustav, već i na pažnju korisnika koji svoje podatke unose u obrasce aplikacija.

7.1. OWASP top 10 metoda napada 2013. godine

Prema OWASP-u top 10 vrsta napada 2013. godine bila su [15]:

- A1 SQL, LDAP i OS injektiranje (eng. Injection)
- A2 Prekinuta autentikacija i upravljanje sesijom (eng. Broken Autentication and Session Management)
- A3 Izvršavanje skripte s drugog računala - XSS napad (eng. Cross-Site Scripting)
- A4 Nesigurno direktno referenciranje objekta (eng. Insecure Direkt Object References)
- A5 Neispravna sigurnosna konfiguracija (eng. Security Misconfiguration)
- A6 Otkrivanje osjetljivih podataka (eng. Sensitive Dana Exposure)

- A7 Nedostatak funkcijske razine kontrole pristupa (eng. Missing Function Level Access Control)
- A8 Krivotvorenja zahtjeva drugog računala – CFRS napad (eng. Cross-Site Request Forgery)
- A9 Korištenje komponenata s poznatim ranjivostima (eng. Using Componenents with Known Vulnerabilities)
- A10 Neprovjerena preusmjeravanja i prosljeđivanja (eng. Unvalidated Redirects and Forwards)

Injektiranje ili SQL, OS i LDAP ubrizgavanje nastaju kada se nepouzdani podaci šalju interpretatoru kao dio naredbe ili upita. Napadačevi podaci mogu obmanuti interpretatora i tako izvršiti nenamjerne naredbe ili pristupiti podacima bez odgovarajućeg odobrenja.

Funkcije aplikacije koje se odnose na provjeru autentičnosti i upravljanje sesijama često se ne implementiraju ispravno pa napadačima omogućuju kompromitiranje zaporki, ključeva ili oznaka sesija te iskorištavanje drugih nedostataka u implementaciji kako bi se preuzeo, odnosno ukrao, identitet drugih korisnika.

Izvršavanje skripte s drugog računala ili XSS napad se pojavljuje svaki put kad aplikacija preuzima nepouzdane podatke i šalje ih Web pregledniku bez pravilne provjere valjanosti ili bijega. XSS omogućuje napadačima izvršavanje skripti u Web pregledniku žrtve i na taj način im omogućuje otmicu korisničke sesije, odbacivanje Web stranica ili preusmjeravanje korisnika na zlonamjerne Web stranice.

Nesigurno direktno referenciranje objekta događa se kada programer izlaže referencu na interni objekt implementacije, kao što je datoteka, direktorij ili ključ za bazu podataka. Bez nadzora kontrole pristupa ili druge zaštite, napadači mogu manipulirati referencama kako bi pristupili neovlaštenim podacima.

Sigurnost zahtijeva definiranje i implementaciju sigurne konfiguracije za aplikaciju, okvire, aplikacijski i Web poslužitelj, poslužitelj baze podataka i platformu. Sigurne postavke treba definirati, implementirati i održavati, jer su zadane postavke često nesigurne.

Otkrivanje osjetljivih podataka problem je s kojim se suočava mnogo korisnika. Mnoge Web aplikacije ne štite ispravno osjetljive podatke korisnika. Zbog slabe zaštite, napadači imaju lak pristup tim podacima, a mogu ih ukrasti ili modificirati kako bi ukrali identitet korisnika ili počinili neki drugi zločin. Kako bi korisnici bili sigurni, osjetljivi podaci zaslužuju dodatnu zaštitu vlastitih podataka, kao i posebne mjere zaštite prilikom razmjene podataka s Web preglednikom.

Nedostatak funkcijske razine kontrole pristupa problem je koji nastaj jer Web aplikacije provjeravaju prava pristupa na razini funkcije prije nego što funkcionalnost bude vidljiva u korisničkom sučelju. Aplikacije bi trebale obavljati nadzor kontrole pristupa na poslužitelju pri svakom pristupu funkciji. Napadači mogu krivotvoriti zahtjeve ako oni nisu potvrđeni te na taj način dobivaju pristup funkcionalnostima bez odgovarajućeg odobrenja.

Krivotvorenja zahtjeva drugog računala ili CFRS napad prisiljava Web preglednik prijavljenog korisnika da šalje krivotvoreni HTTP zahtjev uključujući kolačić sesije korisnika i druge informacije kao što su podaci o autentičnosti. Na taj način napadač može prisiliti korisnikov preglednik da generira zahtjeve za koje ranjiva aplikacija smatra da su legitimni zahtjevi korisnika.

Korištenje komponenata s poznatim ranjivostima kao što su biblioteke, okviri i drugi softverski moduli koji gotovo uvijek rade s punim povlasticama mogu stvoriti ranjivu Web aplikaciju. Ako se koristi ranjiva komponenta napad može prouzročiti ozbiljne gubitke podataka ili preuzimanje poslužitelja.

Neprovjerena preusmjeravanja i prosljeđivanja rezultat su korištenja nepouzdanih podataka u Web aplikacijama koje te podatke koriste za određivanje odredišnih stranica. Bez odgovarajuće provjere, napadači mogu preusmjeriti korisnike na Web lokacije za krađu

identiteta ili povjerljivih podataka, koristiti napad za pristup neovlaštenim stranicama ili navesti korisnike da koriste zlonamjerne stranice i programe.

7.2. OWASP top 10 metoda napada 2017. godine

Top 10 vrsta napada 2017. godine nisu se puno promijenila za razliku od 2013. godine. Ipak, na listi se nalaze male promjene, a to su [13]:

- A1 SQL, NoSQL, LDAP i OS injektiranje (eng. Injection)
- A2 Prekinuta autentikacija (eng. Broken Authentication)
- A3 Otkrivanje osjetljivih podataka (eng. Sensitive Data Exposure)
- A4 XML vanjski entiteti – XXE napad (eng. XML External Entities)
- A5 Prekinuta kontrola pristupa (eng. Broken Access Control)
- A6 Neispravna sigurnosna konfiguracija (eng. Security Misconfiguration)
- A7 Izvršavanje skripte s drugog računala – XSS napad (eng. Cross-Site Scripting)
- A8 Nesigurna deserijalizacija (eng. Insecure Deserialization)
- A9 Korištenje komponenata s poznatim ranjivostima (eng. Using Components with known Vulnerabilities)
- A10 Nedovoljni dnevnički zapisi i nadvor (eng. Insufficient Logging&Monitoring)

Injekcija ili ubrizgavanje podataka i 2017. godine bio je najčešći oblik napada na Web aplikacije. Jedina promjena u odnosu na 2013. godinu kod injektiranja je ubrizgavanje ne samo SQL, LDAP i OS podataka, već i NoSQL podataka koji se interpretiraju kao dio upita.

Na drugom mjestu je prekinuta autentikacija, no za razliku od 2013. godine napadi na upravljanje sesijom više nisu na listi deset najčešćih napada.

Otkrivanje osjetljivih podataka se sa šestog mjesta popeo na visoko treće mjesto. Ovakvi podatci na velikoj su meti napadača jer korisnici ostavljaju sve više podataka na Web aplikacijama, a ukoliko aplikacija nije dovoljno dobro zaštićena, napadači dolaze do velikog

broja podataka o korisnicima koji se mogu iskoristiti za krađu identiteta, ali i za krađu financijskih sredstava korisnika.

XXE napad nije bio na listi 2013. godine, a u 2017. godini nalazi se na četvrtom mjestu. Stariji i slabo konfigurirani XML procesori vrednuju vanjske entitetske reference unutar XML dokumenata i upravo zbog loše konfiguracije XXE napad postaje sve popularniji. Vanjski entiteti koriste se za otkrivanje unutarnjih datoteka pomoću URI rukopisa, internog dijeljenja datoteka, internog skeniranja mrežnih vrata, odnosno portova, izvršavanje daljinskog koda i DoS napada (eng. Denial-of-Service Attack).

Na listi top 10 najčešćih napada 2017. godine, među napadima koji nisu bili na listi 2013. godine, nalazi se i prekinuta kontrola pristupa. Funkcionalnosti autentikacije korisnika i ograničenja koja određuju što korisnici mogu raditi u aplikaciji često se ne primjenjuju pravilno pa napadači iskorištavaju ove nedostatke kako bi pristupili neovlaštenim funkcionalnostima i podacima. Prekinutom kontrolom pristupa napadač ima mogućnost pristupa računima korisnika aplikacija, pregled osjetljivih datoteka, izmjenu podataka korisnika, promjenu prava pristupa te slične aktivnosti kojima se ugrožavaju korisnici.

Neispravna sigurnosna konfiguracija i dalje je jedna od najčešćih meta napada, a obično je rezultat nesigurnih, nepotpunih zadanih ili „ad hoc“ konfiguracija, otvorene pohrane podataka u oblaku, pogrešno konfiguriranih HTTP zaglavlja i poruka o pogreškama koje sadrže osjetljive podatke. Kako bi se spriječila ovakva vrsta napada operacijski sustavi i aplikacije trebali bi biti pravilno konfigurirani da budu sigurni za korištenje, ali se moraju pravovremeno popravljati i nadograđivati.

XXS napad je pao s trećeg na sedmo mjesto najčešćih napada, ali je još uvijek problem s kojim se susreću korisnici Web aplikacija. Kako je objašnjeno u prethodnom poglavlju, napadačima XSS napad omogućuje izvršavanje skripti u Web preglednicima korisnika aplikacije koje sadrže skrivene aktivnosti.

Nesigurna deserijalizacija napad je koji 2013. godine nije bio na OWASP popisu napada. Nedostaci deserijalizacije dovode do daljinskog izvršavanja koda ili se mogu

upotrijebiti upotrjebiti za izvođenje napada kao što su ubrizgavanje koda ili napad na eskalaciju povlastica (eng. Privilege Escalation Attack).

Na devetom mjestu liste i dalje se nalazi korištenje komponenata s poznatim ranjivostima, a na desetom mjestu nalazi se nedovoljno bilježenje i praćenje dnevničkih zapisa. Zajedno sa neučinkovitom ili nedovoljnom integracijom omogućuje napadačima daljnje napade, zakretanje na više sustava te manipuliranje, izdvajanje ili uništavanje podataka.

8. Postavke web i aplikacijskog poslužitelja

Poslužitelj (eng. Server) je namjensko računalo ili softver koji šalje i prima podatke od klijenata, a služi za izgradnju statičkih i dinamičkih Web stranica te za dijeljenje sadržaja. Poslužitelji su otvoreni i svaki korisnik ima mogućnost pristupiti nekom poslužitelju putem internetske adrese, tj. domene ili IP adrese. Po svojoj namjeni, poslužitelji mogu biti:

- Web poslužitelj (eng. Web Server)
- Datotečni poslužitelj (eng. File Server)
- Poslužitelj e-pošte
- Poslužitelj za detektiranje štetnog softvera
- Posrednički poslužitelj (eng. Proxy Server)
- Aplikacijski poslužitelj (eng. Application Server)
- Poslužitelj baza podataka.

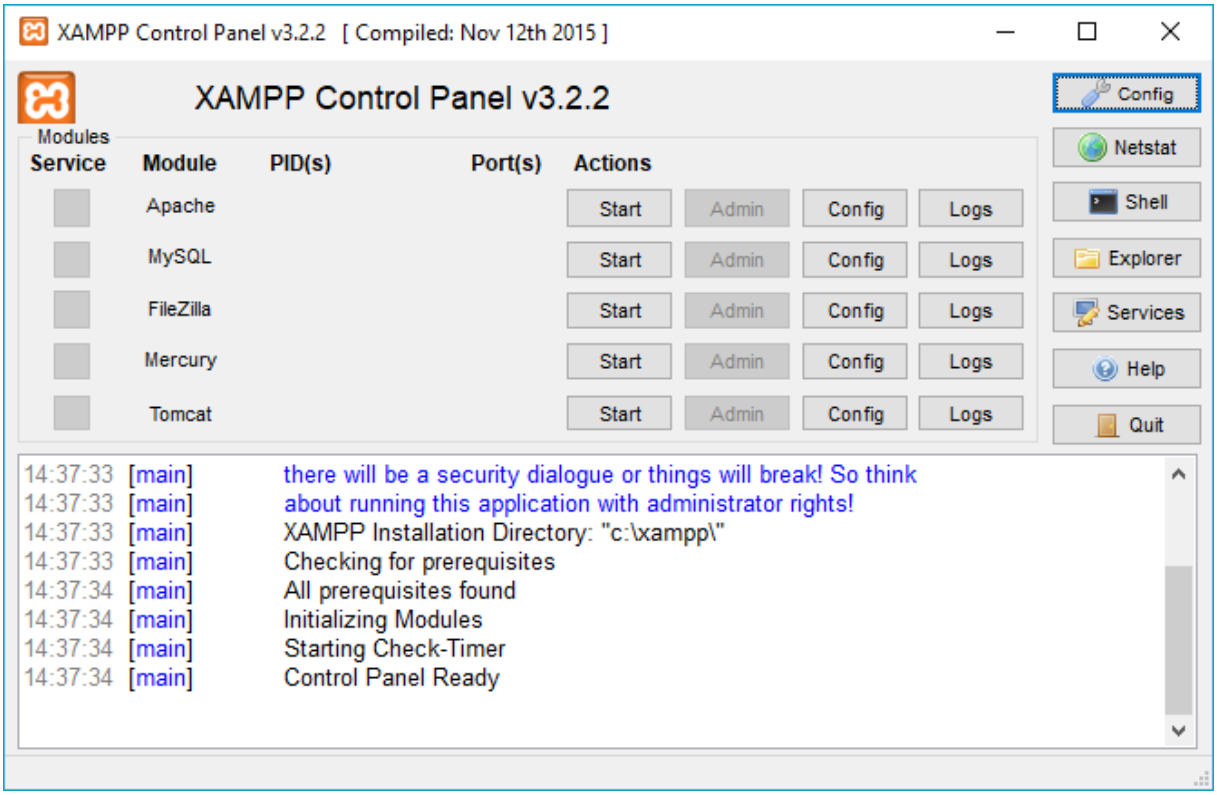
Web poslužitelj je računalo na kojemu se nalaze Web stranice. Pristup Web poslužitelju određen je portom 80, a svaki server ima vlastitu IP adresu. Prilikom posjete nekoj stranici odnosno poslužitelju, zapravo se posjećuje adresa oblika xxx.xxx.xxx.xxx:80 gdje xxx.xxx.xxx.xxx predstavlja IP adresu čije znamenke xxx predstavljaju brojeve od 0 do 255, a broj 80 govori da se radi o Web serveru. Prostor na serveru se može zakupiti kod davatelja usluga posluživanja (eng. Hosting) stranica.

Aplikacijski poslužitelj je poslužitelj unutar mrežnog okruženja, čija je funkcija pokretanje određenih aplikacija, tj. softvera. Kako bi omogućili interakciju aplikacija različitih usluga kvalitete kao što su pouzdanost, sigurnost i neredundantnost, aplikacijski poslužitelji implementiraju posrednički softver (eng. Middleware). Međusobna komunikacija usluga kvalitete i aplikacija odvija se preko HTML-a (eng. HyperText Markup Language) i XML-a (eng EXtenable Markup Language) u obliku linkova prema drugim aplikacijama ili različitim bazama

podataka. Također, aplikacijski poslužitelji pružaju programerima aplikacijsko sučelje (API) što im smanjuje brige o operativnom sustavu te sučeljima koje im pruža web okruženje.

Web i aplikacijski poslužitelj čije će konfiguracije biti opisane u nastavku dio su besplatnog i javno dostupnog (eng. Open Source) multiplatformnog paketa XAMPP razvijenog od strane Apache Friends. Naziv XAMPP označava platformu „Cross-Platform“ (X) koja se sastoji od Apache HTTP poslužitelja (A), baze podataka MariaDB (M), i interpretatora skripata napisanih u programskim jezicima PHP (P) i Perl (P). Sve što je potrebno za postavljanje Web i aplikacijskog poslužitelja uključeno je u XAMPP komprimiranoj datoteci. Kako većina stvarnih implementacija Web poslužitelja koristi iste komponente kao XAMPP, prelazak s lokalnog poslužitelja na stvarni izuzetno je jednostavan proces.

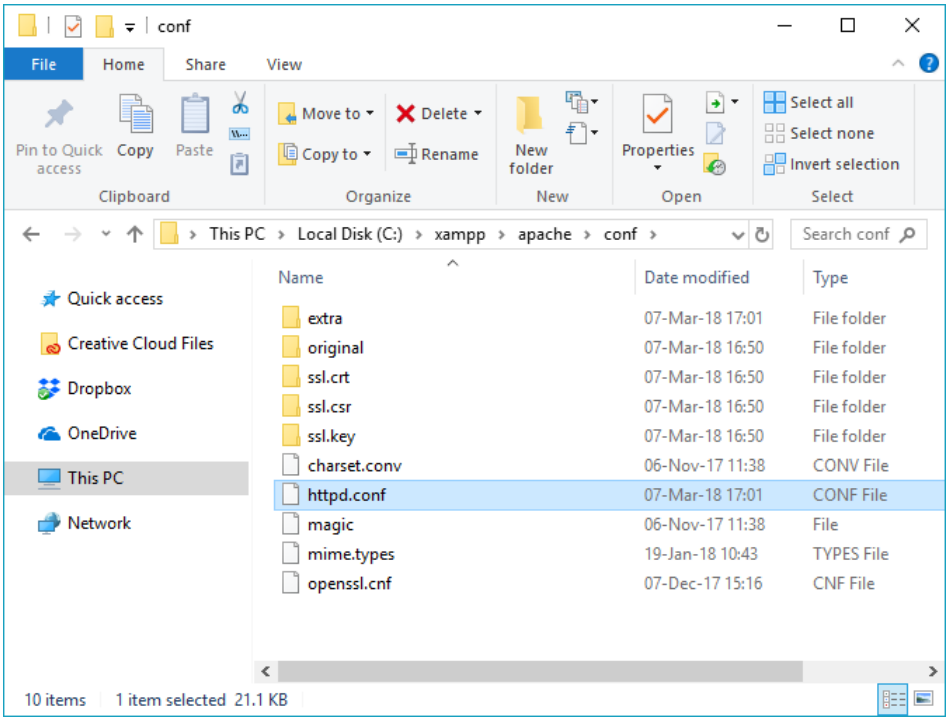
XAMPP se može preuzeti sa stranice apachefriends.org, a dostupan je ne samo za Windows okruženje, već i za Linux te MAC OS. Također, ovaj multiplatformni paket jednostavan je za instalaciju i korištenje.



Slika 3. XAMPP kontrolni prozor

8.1. Postavke konfiguriranja Apache poslužitelja

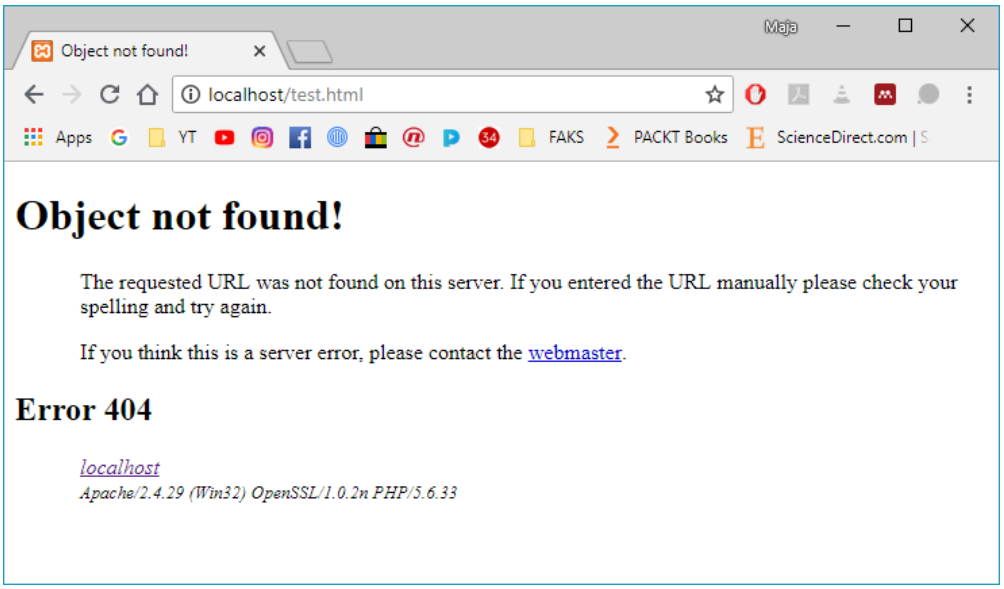
Konfiguraciju Apache poslužitelja moguće je mijenjati u instalacijskom direktoriju do kojega je najlakše doći iz kontrolnog prozora XAMPP programa klikom na „Explorer“. Odabirom „apache“, a zatim „conf“ datoteke dolazi se do konfiguracijske datoteke Apache poslužitelja httpd.conf kao što je prikazano na slici 4.



Slika 4. Prikaz konfiguracijske datoteke htttpd.conf u instalacijskoj datoteci

U httpd.conf datoteci možemo promijeniti HTTP i HTTPS ulaz (eng. Port). Zadani HTTP port je 80, a HTTPS port je 443. Promjena HTTPS porta omogućena je httpd-ssl.conf datotekom koja se nalazi u datoteci „extra“. Oba porta se mogu promijeniti mijenjanjem opcije „Listen“, a nakon svake promjene potrebno je ponovno pokrenuti Apache poslužitelj.

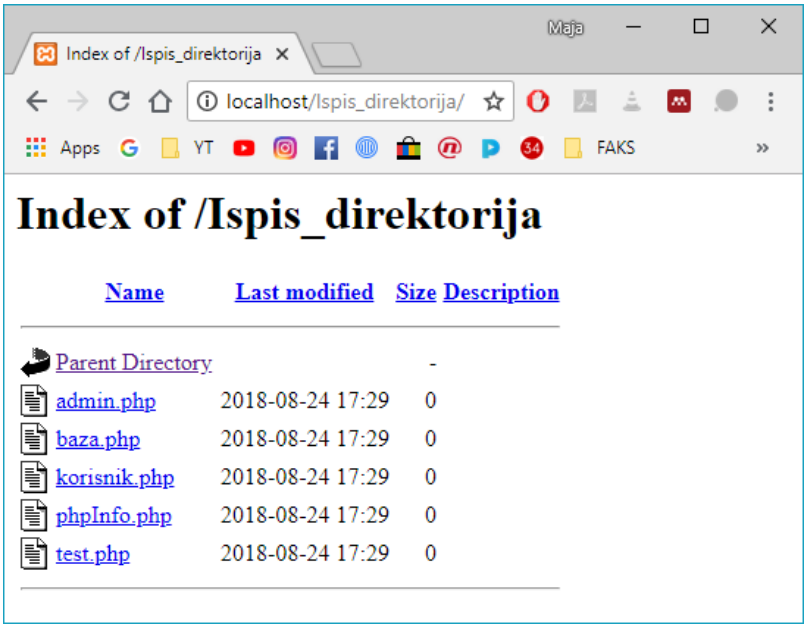
Instaliranjem izvornog paketa Apache poslužitelja u datotekama pogrešaka prikazuje se verzija Apache poslužitelja instalirana na poslužitelju, naziv operacijskog sustava kao i informacije o instaliranim modulima na poslužitelju. Kako bi provjerili ispisuje li poslužitelj ove podatke u adresu traku je potrebno upisati localhost/test.html kao što je prikazano na slici 5. Ako postavke nisu dobro konfigurirane ispis informacija biti će sličan onome na slici.



Slika 5. Apache 404 pogreška s ispisom podataka o poslužitelju

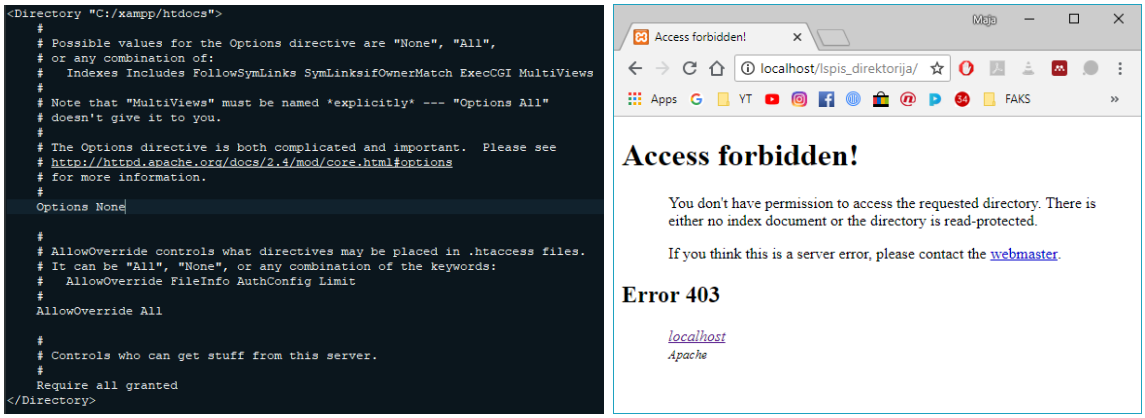
Kako bi spriječili ispis podataka potrebno je u konfiguracijskoj datoteci httpd-default.conf promijeniti opciju „ServerSignature“ iz „On“ u „Off“, a opciju „ServerTokens“ u „Prod“. S ovakvim postavkama podatci se neće ispisivati.

Po zadanim postavkama, Apache poslužitelj ispisuje listu datoteka koje se nalaze u nekom direktoriju ukoliko u njemu nije definirana početna stranica indeks.php, a za potrebe prezentacije ispisa kreiran je novi direktorij prikazan na slici 6.



Slika 6. Ispis datoteka direktorija na poslužitelju

Za sprječavanje izlistanja direktorija potrebno je u datoteci httpd.conf promijeniti opciju „Options“ iz zadane vrijednosti „Indexes Includes FollowSymLinks“ samo u „None“. Promjenom ove postavke i ponovnim pokušajem pristupa kreiranom direktoriju bez početne stranice indeks.php, direktorij neće izlistati datoteke koje se nalaze u direktoriju već će javiti grešku 403 s porukom o nedozvoljenom pristupu stranici.



Slika 7. Opcija „Options“ u httpd.conf konfiguracijskoj datoteci i zabrana pristupa direktoriju

Dva popularna modula Apache poslužitelja koji su vezani za sigurnost su „mod_security“ i „mod_evasive“. Modul „mod_security“ funkcionira kao vatrozid za naše Web aplikacije i omogućuje nam praćenje prometa u stvarnom vremenu. Također nam pomaže da zaštitimo naše Web stranice ili Web poslužitelja od napada grubom silom (eng. Brute Force Attack). Modul „mod_evasive“ djeluje vrlo učinkovito, a sprječava napade grubom silom, DOS i DDOS napade.

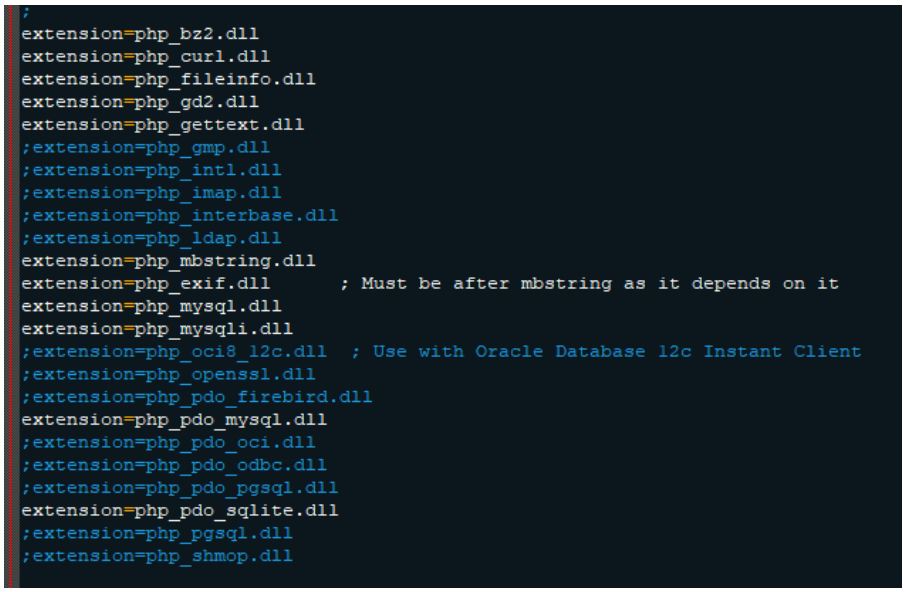
Iako se Web aplikacije ne mogu u potpunosti zaštititi od DDOS napada, postoje neke postavke koji mogu pomoći pri kontroli nad istima. Prva postavka je vrijeme poslužitelja (eng. TimeOut) kojim se određuje koliko će vremena poslužitelj čekati da se izvrše događaju prije neuspjeha, a preporučuje se da se njegova vrijednost postavi na 300 sekundi. Druga mogućnost je maksimalni broj klijenata (eng. MaxClients) koji ograničava broj veza koje se poslužuju istovremeno. Svaka nova veza iznad zadane vrijednosti (256) biti će u redu čekanja. Vrijeme čekanja poslužitelja (eng. KeepAliveTimeout) postavka je kojom se određuje koliko

vremena će poslužitelj čekati na neki zahtjev prije zatvaranja veze. Zadana vrijednost ove opcije je 5 sekundi. Polja ograničenja napada (eng. LimitRequestFields) opcija je koja postavlja ograničenje broja polja zaglavlja HTTP zahtjeva koje mogu biti prihvaćene od klijenata. Zadana vrijednost opcije je 100, a preporučeno je smanjiti tu vrijednost ukoliko se događaju DDOS napadi. Posljednja postavka je granica, odnosno limit veličine polja zahtjeva (eng. LimitRequestFieldSize) koji pomaže pri postavljanju ograničenja veličine zaglavlja HTTP zahtjeva.

Posljednja postavka kojom možemo zaštititi komunikaciju putem Interneta i podatke koji se prenose je korištenje SSL certifikata. Na taj način svi osjetljivi podaci koje Web poslužitelj šalje na Web stranice i aplikacije ne šalju se običnim već šifriranim tekstom. SSL certifikati se mogu kupiti od različitih SSL dobavljača, no Web stranici ili aplikaciji može se dodijeliti vlastito potpisani certifikat. Nakon što se stvori i potpiše SSL certifikat potrebno ga je dodati u Apache konfiguraciju. Za podršku SSL certifikatima Apache poslužitelj koristi „mod_ssl“ modul.

8.2. Postavke konfiguriranja aplikacijskog poslužitelja

Konfiguraciju aplikacijskog poslužitelja (PHP) također je moguće je mijenjati u instalacijskom direktoriju XAMPP-a. Kako je aplikacijski poslužitelj koji se koristi s Apache poslužiteljem PHP, konfiguracijska datoteka u kojoj je moguće prilagoditi postavke poslužitelja naziva se php.ini. U php.ini datoteci moguće je uključivanje raznih ekstenzija za rad s različitim bazama podataka kao što su Oracle ili PostgreSQL. Instaliranjem XAMMP paketa, automatski uključene ekstenzije za rad s bazama podataka su MySQL i MySQLi.

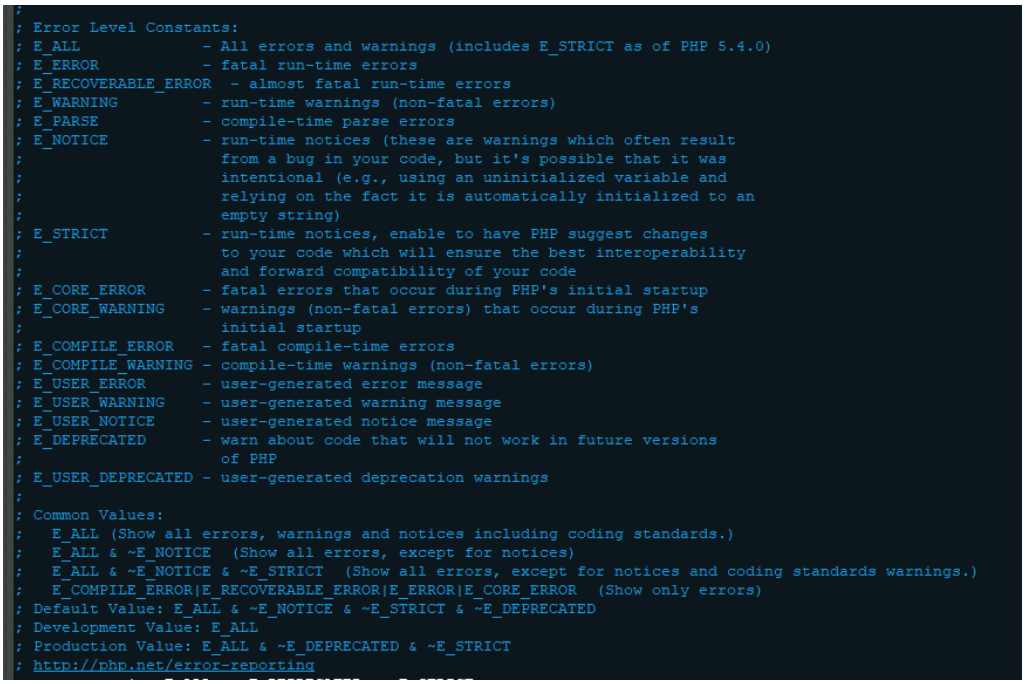


Slika 8. Prikaz ekstenzija koje je moguće uključiti u php.ini datoteci

Jedna od postavki je „short_open_tag“ koja omogućava prepoznavanje PHP koda unutar „<*?*“ i „*?*>“ oznaka umjesto izvornih oznaka početka „<*?php*“ i kraja „*?*>“ PHP koda. Ova postavka je po zadanim vrijednostima isključena.

Iduća postavka koja konfiguraciju poslužitelja čini sigurnijom je „expose_php“ čiju je vrijednost „On“ potrebno promijeniti u „Off“. Ovom postavkom se sprječava ispis instalirane verzije PHP-a na poslužitelju unutar HTTP zaglavlja, a pomaže nam pri zaštiti od napadača jer se na temelju verzije PHP-a mogu doznati sigurnosne „rupe“ koje hakeri iskorištavaju pri napadu na poslužitelja.

U php.ini datoteci nalazi se i konfiguracija razine izvještaja o greškama. U postavkama je moguće uključiti ispis svih grešaka, upozorenja i obavijesti, no moguće je i kombinirati koje razine ispisati, a koje zanemariti.



Slika 9. Konstante pogrešaka koje je moguće koristiti u izvještajima o greškama

U konfiguracijskoj datoteci moguće je uključiti i isključiti postavke za prijenos datoteka putem HTML veze korištenjem opcije „file_uploads“. Ukoliko se radi o održavaju servera, važne postavke su i opcije „Allow_url_fopen“ te opcija „Allow_url_include“. „Allow_url_fopen“ je opcija koja indicira na to da se vanjske datoteke mogu uključivati. Zadana vrijednost ove opcije je „On“, a za veću zaštiti preporučeno ju je isključiti. Opcija „Allow_url_include“ indicira na to da skripte mogu uključivati ili zahtijevati vanjske datoteke. Preporučeno je da i ova opcija bude isključena, što je korišteno i u zadanim vrijednostima.

Aplikacijski poslužitelj može biti konfiguriran da podržava slanje e-pošte i pri tome koristi SMTP protokol (eng. Simple Mail Transfer Protocol). Također, moguće je promijeniti vrijednost port-a koji se koristi za komunikaciju između poslužitelja, a zadana vrijednost je 25.

9. Preporuke za izradu sigurne web aplikacije

Web aplikacija nikada neće biti 100% sigurna, ali kako bi programeri izradili što sigurniju aplikaciju potrebna su znanja i vještine programera kao i dobra oprema koju mnogi programeri još uvijek nemaju. Prilikom izrade Web aplikacije potrebno je shvatiti njezinu svrhu i u skladu s tim pokušati stvoriti što veću sigurnost kako za samu aplikaciju tako i za njezine korisnike. U nastavku su navedene preporuke kojima Web aplikacije postaju sigurnije prilikom njihove izrade. Neke od preporuka za izradu sigurne Web stranice su i testiranje aplikacije te postavljanje vatrozida kao sredstvo za blokiranje napada koji „stvara“ vrijeme za rješavanje problema.

9.1. Kvalitete programera za izradu sigurne web aplikacije

Izrada sigurne Web aplikacije ne zahtjeva samo implementiranje sigurnosnog koda u aplikaciju, već zahtjeva i kvalitete programera koji ju izrađuju. Jedna od kvaliteta programera je mogućnost kontinuirane suradnje sa voditeljima projekta koji su zaduženi za razvoj Web aplikacije. Kontinuirana suradnja među članovima tima, ali i članovima ostalih timova koji nisu zaduženi za sigurnosni aspekt od presudne je važnosti za izradu sigurnog softvera. Međusobna komunikacija članova tima i ostalih timova olakšava donošenje važnih odluka, ali i dobivanje svih potrebnih resursa koji se odnose na razvoj aplikacije.

Prilikom izrade aplikacija važan aspekt sigurnosti je i razvojna disciplina. Sigurnost se ne smije zaboraviti implementirati i ona bi trebala biti provođena tijekom procesa razvoja aplikacije te na svim ključnim prekretnicama. Početak provedbe sigurnosti u aplikacijama trebao bi započeti kada se razmotre i definiraju svi zahtjevi i funkcionalnosti aplikacije. Kako je već spomenuto, sigurnost nije samo dodatna funkcionalnost, već se ona treba provoditi od definiranja zahtjeva, tijekom kodiranja, u fazi analize provjere kvalitete softvera ili kraće QA fazi (eng. Quality Assurance) pa sve do isporuke softvera, uključujući pri tome i održavanje sustava nakon isporuke kupcima.

Kvaliteta programera su i programerskih vještine koje zahtijevaju posebnu obuku. Prilikom razvoja aplikacija vrlo je važno da razvojni programeri budu svjesni kako može doći do sigurnosnih propusta. Prema provedenim istraživanjima Instituta Ponemon otkriveno je kako više od polovice ispitanih programera nije formalno obučeno za implementaciju sigurnosti aplikacija. Kako bi znali zaštititi aplikaciju i podatke, potrebno je znanje o potencijalnim napadima, razumijevanje istih i znanje o načinima njihova preventivnog sprječavanja. Programeri trebaju kritički razmišljati o tome tko može imati motiv i priliku za napasti aplikaciju. Takvim pristupom izradi aplikacije i informacijama do kojih se dođe kritičkim razmišljanjem programeri mogu unaprijed dizajnirati što sigurniju aplikaciju.

9.2. Testiranje web aplikacije

Jedna od preporuka za izradu sigurne Web stranice svakako je i automatsko testiranje koje pronalazi ranjivosti i omogućuje programerima da izbjegnu ne predviđene sigurnosne propuste. Pronalaženje sigurnosnih propusta u današnjim Web aplikacijama zahtjeva mnogo više od samog pregleda kodova. Automatizirani testovi omogućuju programerima i testerima da izrade što sigurnije aplikacije, a testiranje se provodi ne samo na statičkim, nego i na dinamičkim aplikacijama.

Testiranje sigurnosti statičkih aplikacija ili kraće SAST (eng. Static Application Security Testing) odnosi se na pregledavanje koda statičke aplikacije definiranim alatima pri čemu traži određene vrste pogrešaka kao što su logičke bombe, ubrizgani SQL kod i drugi nedostatci. Rezultate testiranja provodi analitičar koji je upoznat s kodom te poslovnim procesima. Ovakvo testiranje funkcionira tijekom razvoja aplikacija u fazi analize, a pomaže pri određivanju specifičnih problema.

Testiranje sigurnosti dinamičkih aplikacija ili kraće DAST (eng. Dynamic Application Security Testing) temelji se na tehnologiji koja analizira rezultat rada u aplikaciji, odnosno interakcije korisnika s aplikacijom na način kojim bi napadač pristupio aplikaciji. DAST omogućava testiranje za one uvijete koji se mogu otkriti tijekom stvarne uporabe sustava.

Ovakav način testiranja moguće je provoditi tijekom cijelog ciklusa izrade aplikacije na bilo kojem jeziku, od samog razvoja, u fazi testiranja i ispitivanja kvalitete pa sve do njezinog puštanja na tržište. Testiranje dinamičkih aplikacija omogućuje ispitivanje cijele aplikacije, a ne samo dijelove za koje je dostupan kod. Novi DAST sustavi temeljeni na oblaku omogućuju testiranje velikog broja aplikacija bez troškova ili tereta postavljanja lokalnog softvera u vrlo kratkom vremenskom periodu.

Osim automatskog testiranja, postoji i penetracijsko testiranje koje je metoda evaluacije sigurnosti sustava ili mreže simuliranjem napada s ciljem pronalaska ranjivosti koje mogu biti iskorištene. Razlikujemo dva tipa penetracijskog testiranja, a to su funkcionalno testiranje (eng. Black-Box Testing) i strukturalno testiranje (eng. White-Box Testing). Funkcionalno testiranje simulira napad od strane počinitelja koji nema nikakvo znanje o sustavu kojeg napada, a strukturalno testiranje je simulacija napada od strane počinitelja koji posjeduje cjelovito znanje o sustavu kojeg napada. Nakon provedenog penetracijskog testiranja potrebno je kreirati izvještaj sa svim uočenim ranjivostima sustava tijekom testiranja kao i s prijedlogom mjera koje se mogu implementirati u sustav s ciljem njegove zaštite.

9.3. Blokiranje napada na web aplikaciju korištenjem vatrozida

Web aplikacije obrađuju podatke koje im šalje korisnik i iz tog razloga su meta napadača za krađu podataka ili širenje malicioznog koda. Preporuka za smanjenje rizika takvih neželjenih događaja je vatrozid (eng. Firewall). Lako podesivi vatrozidi upozoravaju korisnike na lošu konfiguraciju, a koriste se za zaštitu i kontrolu mrežnog prometa. Noviji vatrozidi posjeduju mogućnost kontrole prometa na aplikacijskom sloju (HTTP i HTTPS), no njihove mogućnosti kontrole Web prometa ne mogu se mjeriti sa specijaliziranim uređajima. Kako Web aplikacije nisu dovoljno dobro zaštićene, napadači uz malo uloženog truda i vremena mogu postići isplative rezultate. Vatrozidi za zaštitu Web aplikacija prate radnje koje se događaju na Web aplikaciji i temeljem tih radnji kreiraju pravila za zaštitu, no poznavanjem strukture Web

aplikacija omogućena je zaštita od deset najkritičnijih sigurnosnih rizika aplikacija opisanih u prethodnom poglavlju. Ovakvi uređaji imaju i funkciju zaštite od DOS (eng. Denial of Service) i DDOS (eng. Distributed Denial of Service) napada koji uzrokuju preopterećenost poslužitelja. Bitno je spomenuti kako postoje i uređaji koji omogućavaju autentikaciju pri čemu promet do Web aplikacija ne može doći sve dok se korisnik ne autenticira ispravnim korisničkim podacima. Također, neki uređaji omogućavaju antivirusnu provjeru datoteka što štiti korisnike od preuzimanja malicioznog koda.

9.4. OWASP CLASP

Metoda sigurnog razvoja softvera OWASP CLASP (eng. OWASP Comprehensive Lightweight Application Security Process) specificira aktivnosti koje članovi razvojnog tima moraju provesti kako bi razvili što sigurniji softver. Pri izradi programa, CLASP metoda kategorizira aktivnosti po ulogama, a svaki član tima pripada jednoj od sljedećih uloga [12]:

- Projektni menadžer
- Specifikator zahtjeva
- Arhitekt
- Dizajner
- Implementator
- Tester
- Revizor sigurnosti

Svaka uloga ima definirane odgovornosti i zadatke koje mora provesti tijekom procesa razvoja softvera. Raspored aktivnosti po pojedinim ulogama prikazan je u tablici:

Uloga	Aktivnosti
Projektni menadžer	• Uspostavljanje svijesti o sigurnosti programa • Upravljanje procesom razotkrivanja sigurnosti (u suradnji sa dizajnerom) • Praćenje sigurnosti
Specifikator zahtjeva	• Identificiranje globalne politike sigurnosti • Navođenje operativnog okruženja (u suradnji sa arhitektom) • Opisivanje detalja nepravilnog korištenja slučajeva • Dokumentiranje zahtjeva vezanih za sigurnost (u suradnji s arhitektom)
Arhitekt	• Identifikacija resursa i granica povjerenja (u suradnji sa specifikatorom zahtjeva) • Identifikacija uloga korisnika i mogućnosti resursa (u suradnji sa specifikatorom zahtjeva)
Dizajner	• Istraživanje i pridruživanje sigurnosnih stanja pojedinim tehnološkim rješenjima • Identifikacija mogućih područja napada • Uključivanje sigurnosnih principa u dizajn • Bilježenje klase dizajna sa sigurnosnim svojstvima • Rješavanje prijavljenih sigurnosnih problema • Izrada operativnog sigurnosnog vodiča (u suradnji sa arhitektom i implementatorom)

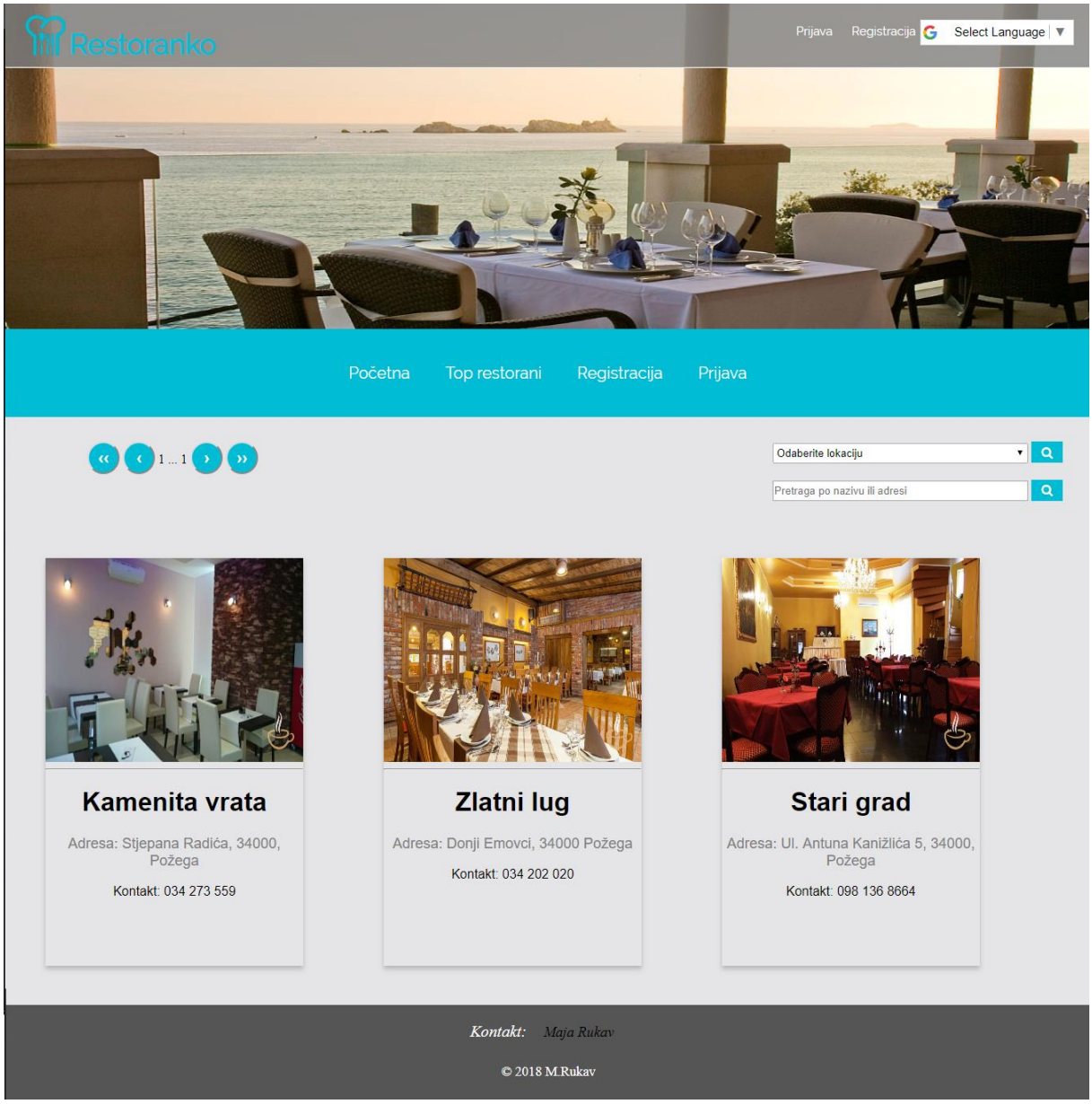
Implementator	• Implementacija i razrada politike resursa i sigurnosne tehnologije • Implementacija sučelje ugovora • Integracija sigurnosne analize u izvor upravljanja procesom • Potpisivanje izvornog koda
Tester	• Identifikacija, implementacija i provedba testova sigurnosti • Verifikacija sigurnosnih atributa resursa
Revizor sigurnosti	• Modeliranje mogućih prijetnji • Kontrola razine sigurnosti izvornog koda (u suradnji s implementatorom i dizajnerom)
Dizajner baze podataka	• Specificiranje sigurnosnih postavki baze podataka

Tablica 4. Podjela aktivnosti po ulogama temeljene na OWASP CLASP metodi [12]

Svaka od navedenih uloga ima definirane aktivnosti koje mora provesti prilikom izrade sigurne web aplikacije. Također, iz tablice možemo vidjeti da neke od aktivnosti provode dvije ili više uloga ovisno o području kojim se bave. Iako dizajner baze podataka nije definirana uloga OWASP CLASP metode, članovi tima koji su zaduženi za izradu i održavanje baze podataka zaduženi su i za njezinu sigurnost.

10. Praktični dio: web aplikacija restoranko.online

Praktični dio rada zamišljen je kao Web aplikacija, odnosno sustav koji korisnicima omogućava pregled restorana, menija te rezervaciju stolova. Također, sustav prati zadovoljstvo korisnika te na temelju ocjena korisnika i nekih drugih obilježja kreira statističke podatke. Aplikacija ima definirane uloge (neregistrirani i registrirani korisnik, moderator i administrator). Kroz završni rad, na primjeru Web aplikacije, prikazane su ranjivosti Web aplikacije te je objašnjeno kako zaštititi aplikaciju da bi ona bila sigurna za korisnike.



Slika 10. Početna stranica web aplikacije www.restoranko.online

10.1. Uvod u rad web aplikacije

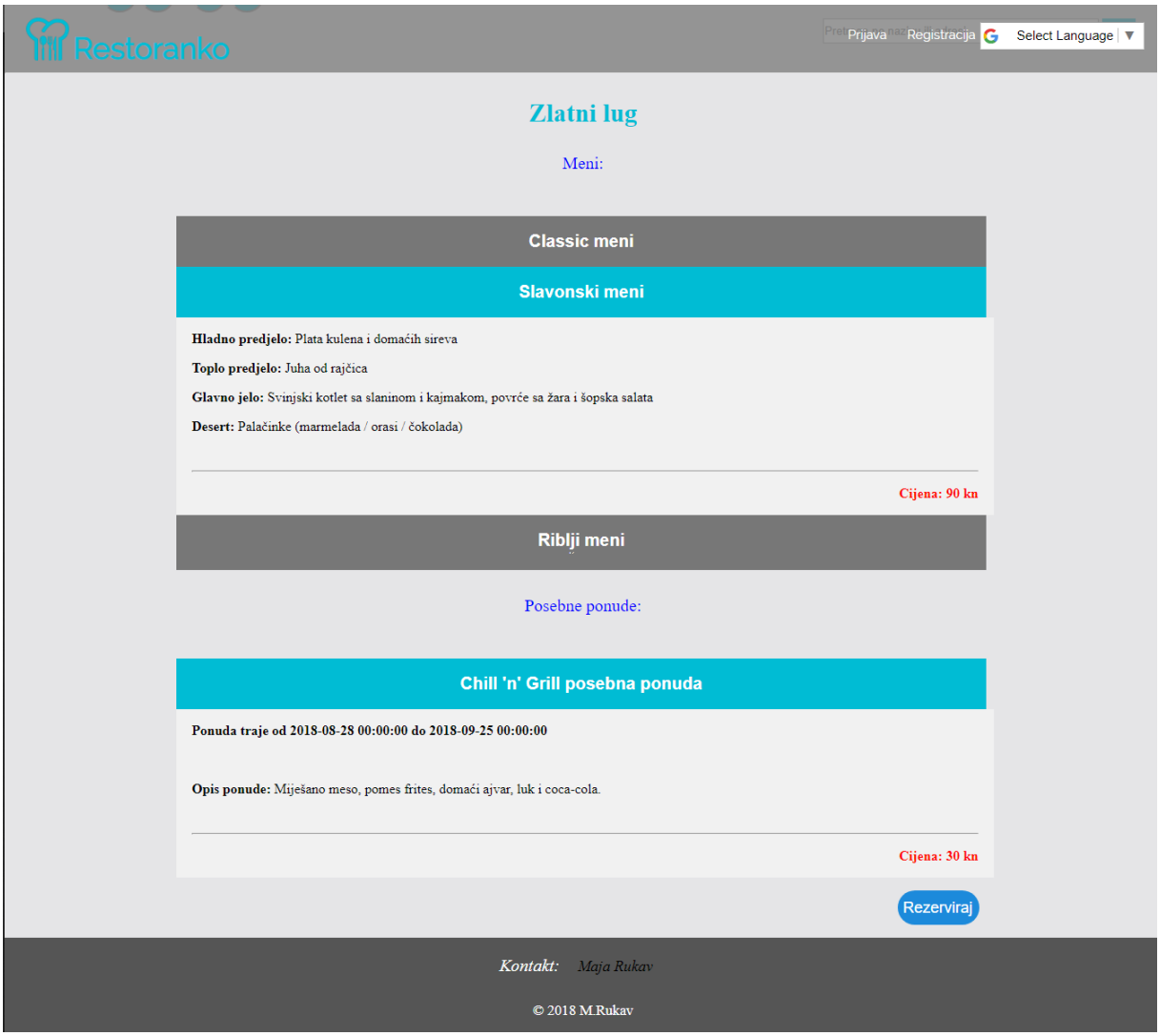
Na početnoj stranici aplikacije prikazani su restorani koji se nalaze u sustavu, te su navedeni njihovi osnovni podaci kao što su adresa i kontakt. Svim korisnicima omogućena je pretraga restorana po dostupnim lokacijama koje definira administrator sustava. Odabirom željene lokacije, na početnoj se stranici prikazuju samo oni restorani koji se nalaze na toj lokaciji. Restorane je moguće pretraživati i po njihovom nazivu ili adresi. Klikom na neki od restorana prikazuju se osnovne informacije te galerija fotografija za isti. Korisnik u otvorenom prozoru prikazanom na slici 11. ima mogućnost dodati odabrani restoran u popis „Omiljeno“, pregledati dostupne menije i posebne ponude restorana.

Slika 11. Prikaz informacija i fotogalerije odabranog restorana

Svakim klikom na restoran bilježi se njegov pregled što omogućuje statističko praćenje popularnosti restorana kod korisnika, ali i kreiranje top liste.

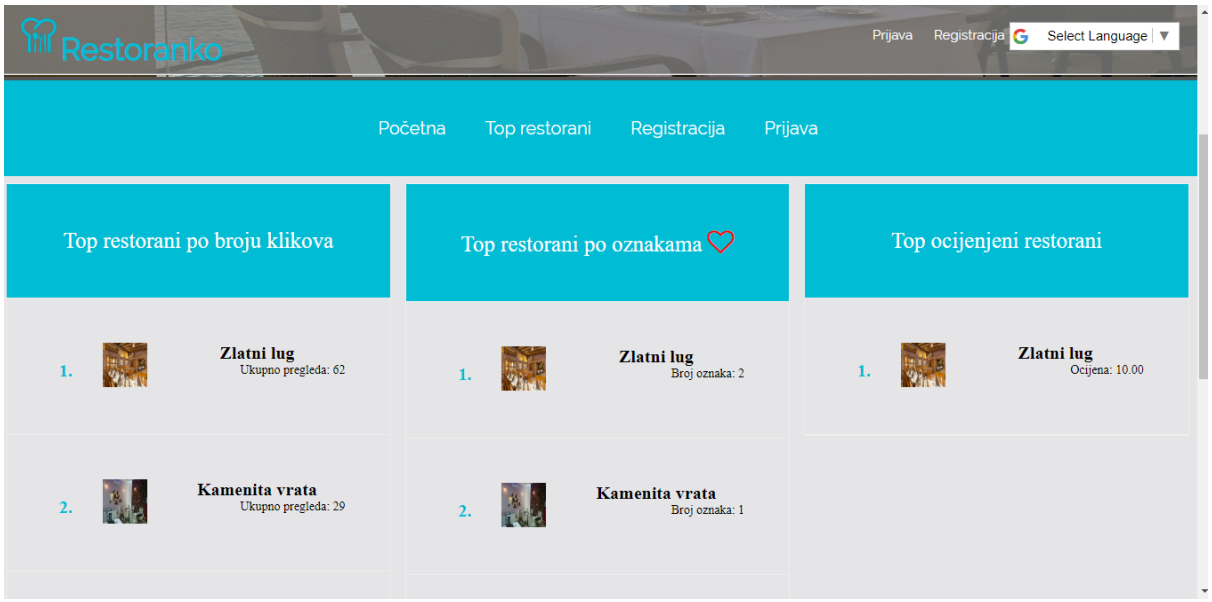
Klikom na gumb „Meni“ prikazuju se svi trenutno dostupni meniji i posebne ponude restorana. Klikom za željeni meni, izlistava se ponuda, te je navedena cijena menija, odnosno posebne ponude. Za posebne ponude navedeni su i datumi početka i završetka promotivne

ponude. Kao što je prikazano na slici 12. iz otvorenog prozora moguće je i automatsko prebacivanje na stranicu rezervacije restorana.



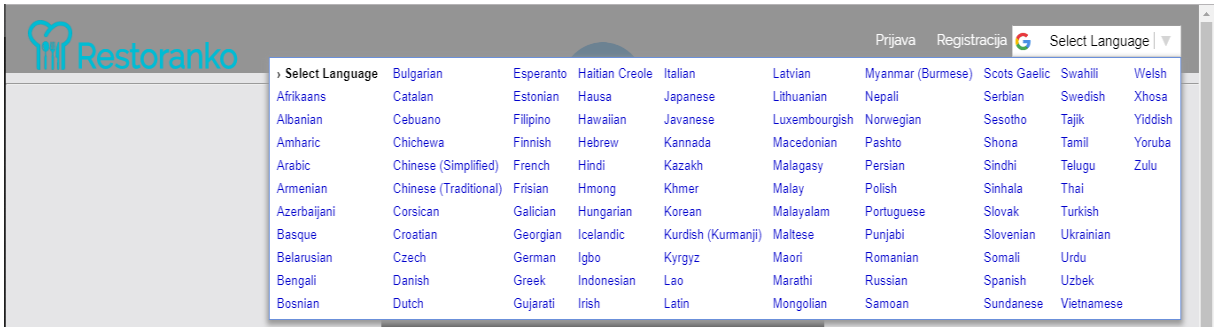
Slika 12. Prikaz menija i posebnih ponuda restorana

Top liste su također vidljive svim korisnicima i može im se pristupiti s početne stranice aplikacije. Korisnicima su dostupne tri top 10 liste. Prva se odnosi na top liste po broju pregleda restorana zabilježenih pregledom informacija i fotogalerije restorana. Druga top lista prikazuje top 10 restorana s najviše oznaka „Omiljeno“. Tom oznakom prijavljeni korisnici mogu označiti restorane koji im se sviđaju kako ne bi zaboravili njihovo ime i kako ih kasnije ne bi morali tražiti. Popis omiljenih restorana nalazi se na profilu korisnika. Treća top lista odnosi se na top ocijenjene restorane. Korisnici imaju mogućnost ocijeniti restoran prilikom unosa recenzije.



Slika 13. Prikaz top listi restorana

Aplikaciju je moguće prevesti na bilo koji od dostupna 104 jezika koji su podržani Google Translate API-jem.

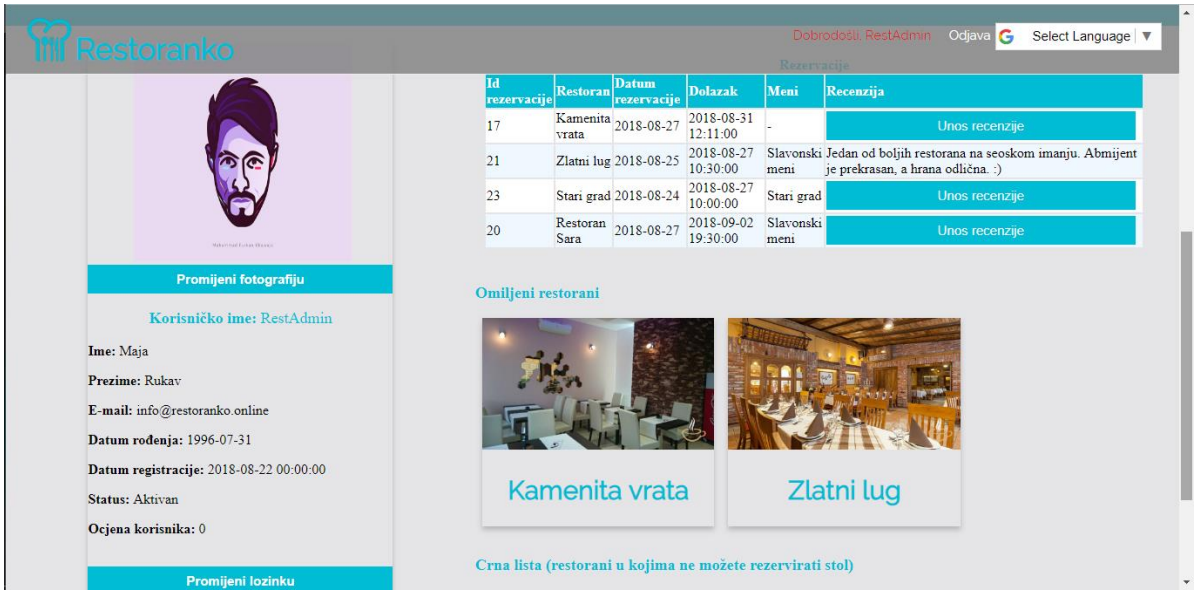


Slika 14. Prikaz integriranog Google Translate API-ja

10.2. Osnovne funkcionalnosti aplikacije

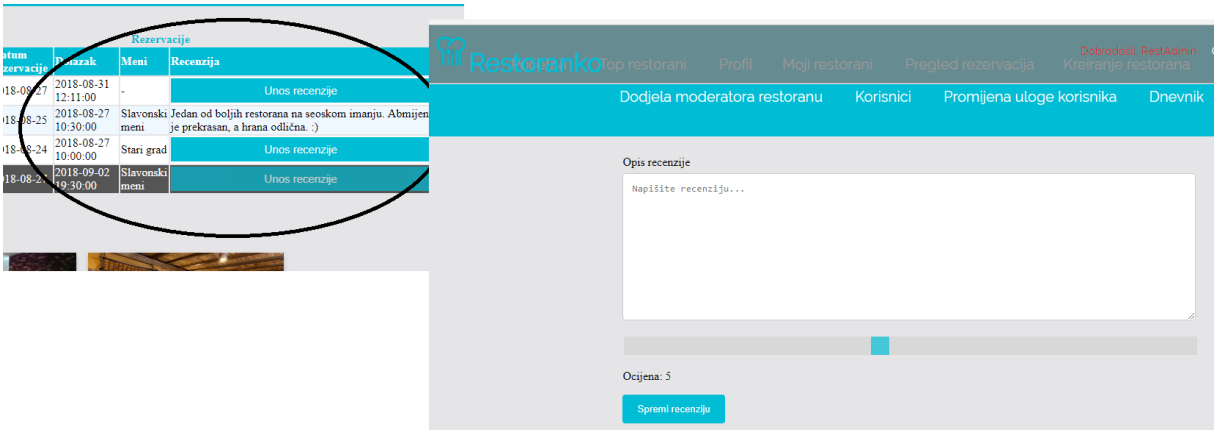
U aplikaciji je omogućena registracija i prijava korisnika te su implementirane potrebne provjere prilikom istih. Kod registracije novog korisnika važno je napomenuti da se na e-mail šalje aktivacijski kod koji traje 7 sati i korisnik se bez aktivacije računa ne može prijaviti u aplikaciju. Registracija i prijava u sustav odvijaju se putem HTTPS protokola. Prilikom prijave u sustav, ukoliko je korisnik netočno upisao lozinku tri puta, njegov korisnički račun se zaključava i može ga jedino otključati samo administrator. Ukoliko je korisnik zaboravio lozinku, može poslati zahtjev za novu pri čemu se generira slučajna lozinka koju korisnik prima na e-mail račun.

Registrirani korisnici u panelu „Profil“ imaju uvid u svoje korisničke podatke, ali i pregled svojih rezervacija, te listi omiljenih restorana, kao i liste restorana gdje su svrstani na crnu listu. Svi korisnici imaju ocjenu koja se računa u dolascima i nedolascima u rezervirane termine. Najveća ocjena korisnika je 5, a ukoliko korisnik ima manju ocjenu od 2,5 njegov korisnički račun se automatski blokira. Njegov status može promijeniti jedino administrator sustava. Obavijest o blokiranju i ponovnom aktiviranju računa šalje se na korisnikov e-mail račun. Korisnici na svojemu profilu mogu promijeniti sliku korisničkog profila i lozinku.



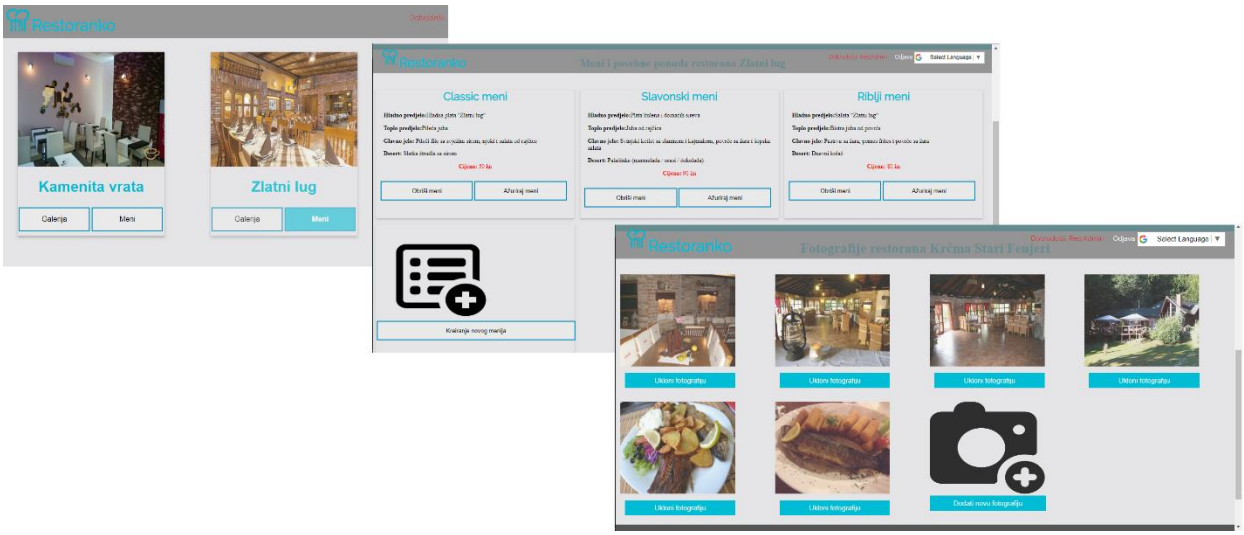
Slika 15. Prikaz profila korisnika

Također, registrirani korisnici imaju mogućnost rezerviranja stola i menija u restoranima. Prilikom rezervacije stola potrebno je navesti datum i vrijeme dolaska, a postoji i mogućnost odabiranja menija kako bi hrana bila spremna za gosta prilikom njegova dolaska. Rezervaciju moraju prihvatiti moderatori restorana, nakon čega se obavijest s potvrdom šalje korisniku na e-mail. Nakon završetka termina rezervacije, korisnici imaju mogućnost unijeti recenziju te ocijeniti restoran.



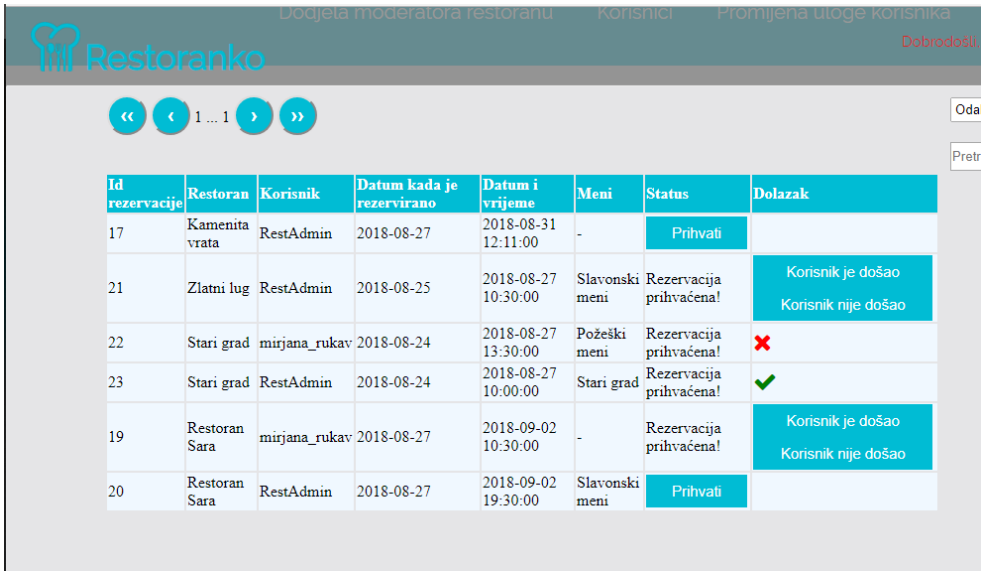
Slika 16. Pregled rezervacija i unos recenzije rezerviranih termina

Moderatori imaju uvid u dodijeljene restorane te imaju mogućnost kreiranja, brisanja i ažuriranja menija te posebnih ponuda. Također, moderatori mogu vidjeti sve fotografije restorana, brisati ih ili dodavati nove kao što je prikazano na slici 17.



Slika 17. Kreiranje, brisanje i ažuriranje menija te brisanje i dodavanje fotografija restorana

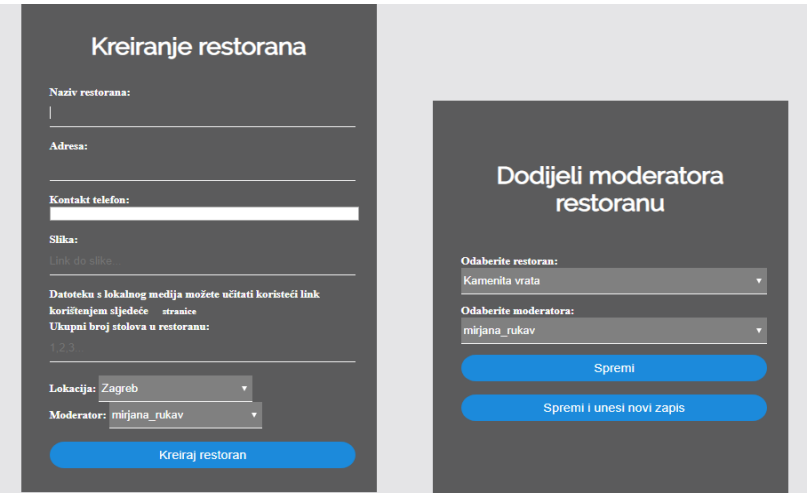
Za iste, moderatori imaju uvid u sve rezervacije te mogu prihvatiti ili odbiti rezervaciju. Nakon završetka termina rezervacije mogu zabilježiti dolazak, odnosno ne dolazak gosta koji utječe na ocjenu korisnika i njegov status. Ukoliko se korisnik nađe na crnoj listi nekog restorana tada mu je zabranjeno rezerviranje termina u istome.



Slika 18. Pregled rezervacija u sustavu

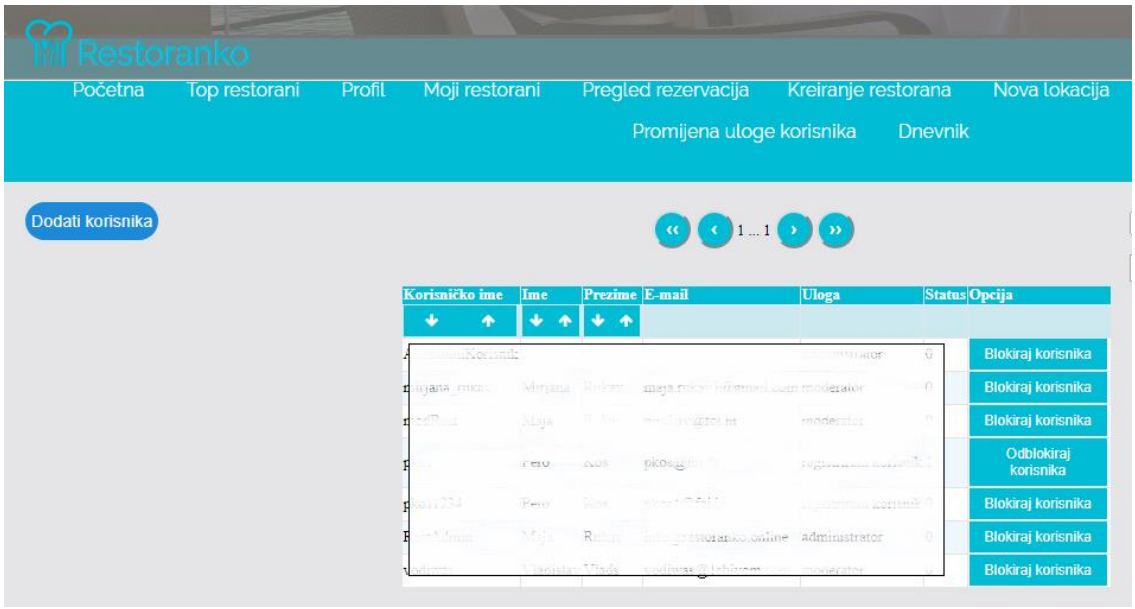
Slika 19. Prikaz popisa korisnika na crnoj listi za restorane kojima je prijavljeni korisnik moderator

Administrator sustava ima mogućnost kreiranja lokacija i kreiranja novih restorana. Prilikom kreiranja novog restorana, administrator mora odabrati lokaciju na kojoj se restoran nalazi te mu mora dodijeliti jednog ili više moderatora. U svakome trenutku, administrator može dodijeliti restoranu novog moderatora.



Slika 20. Kreiranje i dodjela moderatora restoranima

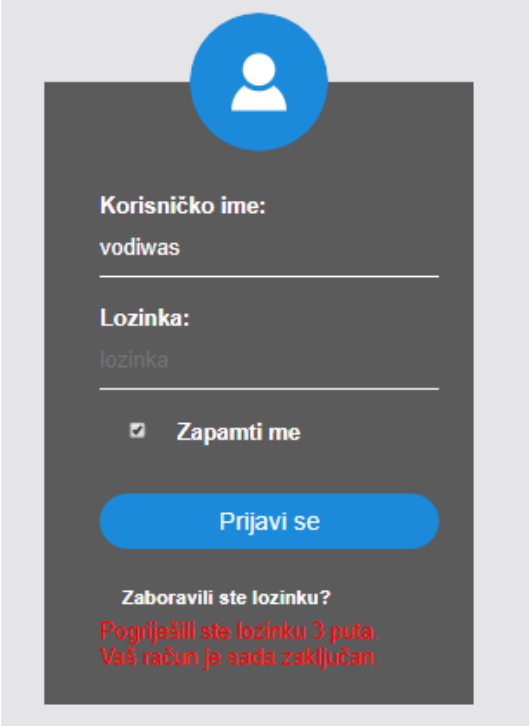
Administrator ima uvid u korisničke podatke svih korisnika aplikacije te ima mogućnost mijenjanja njihovog statusa. Također, on može dodati novog korisnika pri čemu se korisnički podaci s slučajno generiranom lozinkom šalju na e-mail korisnika, ali i mijenjati uloge korisnika, odnosno dodijeliti mu ili oduzeti prava moderatora i/ili administratora.



Slika 21. Prikaz podataka o korisnicima

10.3. Ranjivosti i implementirana zaštita

U teorijskom dijelu opisane su ranjivosti koje može imati svaka Web aplikacija. Kako bi zaštitili aplikaciju vrlo je važno implementirati zaštitu u svakome dijelu aplikacije tijekom pisanja samoga koda. U aplikaciji, autentikacija je provedena vlastitom metodom s bazom podataka, a validacija podataka vrši se i na korisničkoj i na poslužiteljskoj strani. Prilikom prijave u sustav onemogućen je beskonačni broj pokušaja prijave u sustav čime je smanjena ranjivost aplikacije od napada grubom silom. Tako se nakon 3 pokušaja prijave s krivim korisničkim podacima korisnički račun zaključava.



Slika 22. Zaključavanje korisničkog računa nakon 3 neuspješna pokušaja prijave

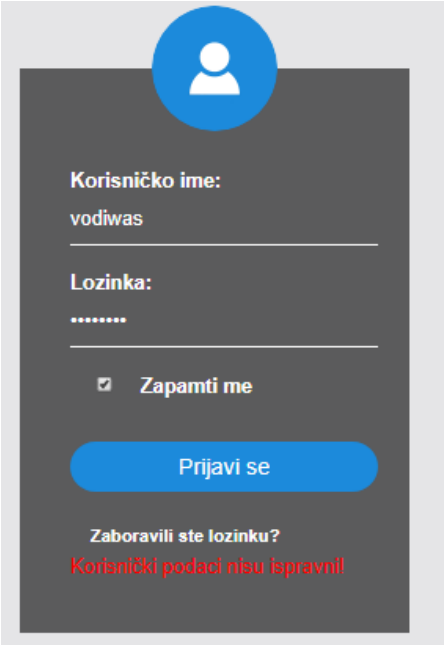
Prilikom prijave u sustav u bazi podataka se provjeravaju upisani korisnički podaci. Ako je korisnik pronađen dohvaćaju se svi podaci o korisniku koji su zapisani u bazi podataka te se provjerava je li korisnikov račun aktiviran ili možda zaključan. Ako je korisnički račun aktiviran i nije zaključan korisnik se prijavljuje u sustav, a ako je račun blokiran ili nije aktiviran ispisuju se poruke o pogreškama. Programski kod je naveden u nastavku.

```
if ($korisnikPronadjen) {
    $podaciKorisnik = 'SELECT * FROM `korisnik` WHERE `korisnickoIme`="
    '.$korime_prijava.' " ';
    while($rezultat= ($bp->selectDB($podaciKorisnik))->fetch_assoc()){
        $korime = $rezultat['korisnickoIme'];
        $status = $rezultat['status'];
        $aktiviran = $rezultat['aktiviran'];
    }
   //provjera je li račun aktiviran: 1-aktiviran, 0-nije aktiviran
    if ($aktiviran) {
        //provjera statusa računa: 0-nije zaključan, 1-zaključan
        if ($status == 0) {
            //ažuriranje podataka o uspješnoj prijavi u sustav i kreiranje kolačića
        } else {
            $prijavaIspisGreska = "Račun s upisanim korisničkim imenom je
            zaključan!";
        }
      } else {
        $prijavaIspisGreska = "Račun s upisanim korisničkim imenom nije aktiviran.";
      }
}
```

Prilikom neuspješne prijave u sustav provjerava se postoji li korisničko ime u bazi podataka i ako postoji ažuriraju se njegovi podaci o neuspjelim prijavama u sustav. Također se provjerava broj neuspjelih pokušaja prijave i kako je već navedeno, nakon 3 neuspjela pokušaja prijave korisnički račun se zaključava, a otključati ga može jedino administrator sustava. Programski kod provjere neuspješne prijave i zaključavanja računa je sljedeći:

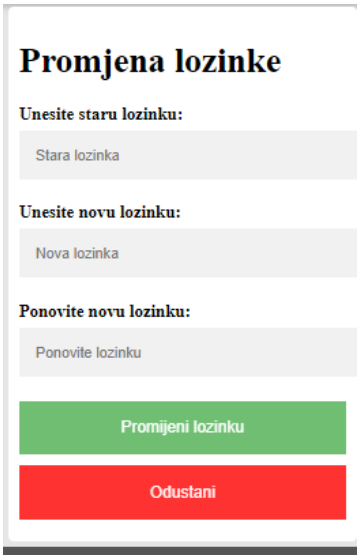
```
elseif (!$korisnikPronadjen) {
        $provjeriKorIme = 'SELECT `korisnickoIme`,`neuspjesna_prijava` FROM `korisnik`
        WHERE `korisnickoIme`=" ' . $korisnicko_ime . ' " ';
        if (mysqli_num_rows($bp->selectDB($provjeriKorIme)) == 1) {
                $prijavaIspisGreske = "Korisnički podaci nisu ispravni!";
                $rezultat = ($bp->selectDB($provjeriKorIme)) ->fetch_assoc();
                $korisnik = $rezultat["korisnickoIme"];
                $neuspjesnaPrijava = $ rezultat ["neuspjesnaPrijava"];
                $neuspjesnaPrijava++;
                $sql = 'UPDATE `korisnik` SET `neuspjesnaPrijava`= " ' . $neuspjesnaPrijava .
                        ' " WHERE `korisnickoIme`= " ' . $korisnik . ' " ';
                $izvrsiUpit = $bp->selectDB($sql);
                if ($neuspjesnaPrijava == 3) {
                        $sqlUpd = 'UPDATE `korisnik` SET `status`=1 WHERE
                            `korisnickoIme` = " ' . $korisnik . ' " ';
                        $IzvrsiUpitUpd = $bp->selectDB($sqlUpd);
                        $prijavaIspisGreske = "Pogriješili ste lozinku 3 puta. Vaš račun je sada
                            zaključan.";
                }
        } else {
                $prijavaIspisGreske = "Korisnički podaci nisu ispravni";
        }
}
```

Također, prilikom upisivanja krivih korisničkih podataka dana je pažnja i propustima u dizajnu autentikacije, pa svaki krivi pokušaj ispisuje pogrešku, no poruka koja se ispisuje nije opširna i ne sadrži informacije koji podaci su krivo upisani kao što je prikazano na slici 23.



Slika 23. Poruka o krivo upisanim korisničkim podacima

Ranjivosti koje programeri mogu implementirati u dizajnu autentikacije su i mogućnost oporavka lozinke ukoliko ju je korisnik zaboravio ili ju želi promijeniti. Prilikom oporavka lozinke, korisnik mora unijeti e-mail adresu korisničkog računa na koji se šalje nova lozinka. Ako e-mail račun kojega korisnik unese ne odgovara korisničkome računu lozinku nije moguće oporaviti. Ako su e-mail računi jednaki korisnik na svoj račun dobiva privremenu, slučajno generiranu lozinku, a promjena lozinke je moguća na panelu „Profil“ kada se korisnik prijavi u sustav.



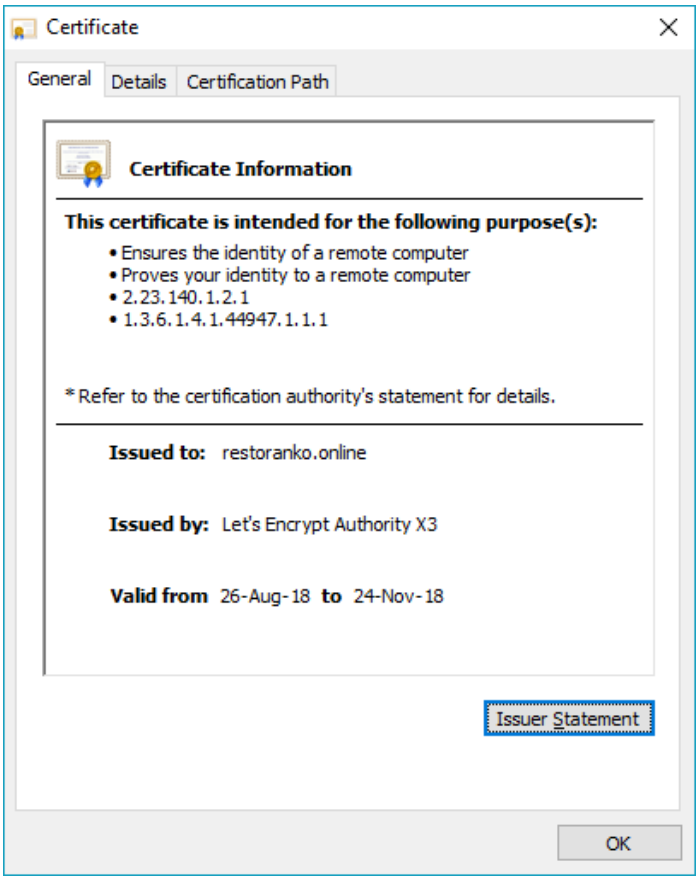
Slika 24. Obrazac za promjenu lozinke na panelu „Profil“

Prijava i registracija korisnika se odvijaju preko sigurne linije, odnosno HTTPS protokola što omogućava slanje kriptiranih podataka od korisnikova računala do poslužitelja. Dodatna zaštita je i SSL certifikat prikazan na slici 25 kojim se kriptira web promet, a koji sprečava pristup Web paketima osobama kojima ti paketi nisu namijenjeni. Realizacija prijave i registracije korištenjem HTTP-s protokola provedena je na sljedeći način:

```
$uri = filter_input(INPUT_SERVER,'REQUEST_URI',FILTER_SANITIZE_STRING);
$pos = strrpos($uri, "/");
$dir = filter_input(INPUT_SERVER,'SERVER_NAME',FILTER_SANITIZE_STRING) . substr($uri, 0, $pos + 1);

if (!isset($_SERVER["HTTPS"]) || strtolower($_SERVER["HTTPS"]) != "on") {
    $adresa = 'https://' . $dir . 'prijava';
    header("Location: $adresa");
    exit();
}
```

Iz navedenog programskog koda možemo vidjeti kako smo pomoću filter_input naredbe, koja se koristi za zaštitu od XSS napada, dohvatili podatke o poslužitelju i URI adresi te se provjerava koristi li skripta HTTPS protokol.



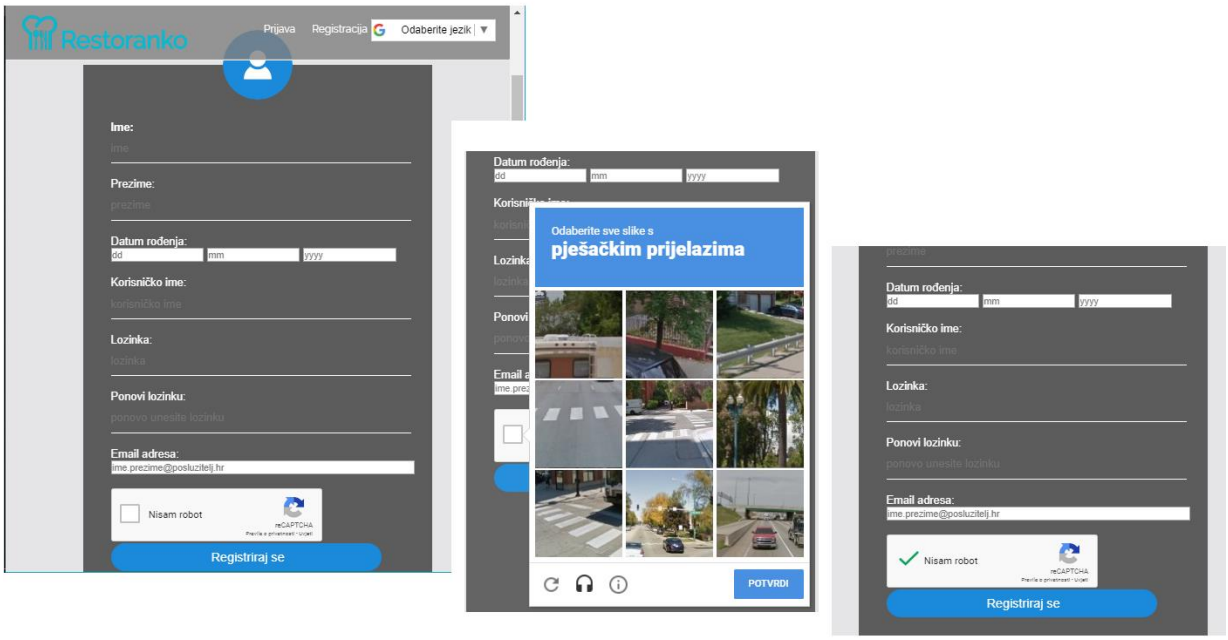
Slika 25. SSL certifikat aplikacije restoranko.online

Kod registracije korisnika implementirana je i reCAPTCHA provjera kontrolnog niza znakova te na taj način osigurava sigurnost i zaštita od automata. Sve lozinke su u bazi spremljene u kriptiranom obliku korištenjem SHA1 algoritma za izračunavanje sažetka (eng. hash) i primjenom soli (eng. salt), a primjer koda koji koristi navedenu metodu za kriptiranje lozinki u sustavu naveden je u nastavku:

```
$sol = sha1(time());
$sifriranaLozinka = sha1($sol . "-" . $unesenaLozinka);
```

Aktivacija korisničkog računa moguća je u trajanju od 7 sati nakon registracije korisnika u sustavu, a prilikom registracije svakom korisniku se dodjeljuje vlastiti token kreiran na sljedeći način:

```
$token = md5(md5(time() . „tajnaRijec“ . rand(0, 1000)));
```



Slika 26. Implementirana reCAPTCHA provjera kontrolnog niza znakova

Provjera kontrolnog niza znakova reCAPTCHA implementirana u programsko rješenje:

<p class="g-recaptcha" data-sitekey=

"6Lf8hmwUAAAAAJiZeRQLmEMMRXudFaVOXi3l8UYq"> </p>

Kod autorizacije implementirana je zaštita od napada prisilnim pregledavanjem. Pokušaji upisivanja URL adrese u adresnu traku rezultirat će porukama o neautoriziranom pristupu. U aplikaciji je primijenjena i osnovna zaštita za XSS koristeći „*filter_input*“ kod unosa i „*htmlspecialchars*“ oznaka kod ispisa podataka, a prikazan je sljedećim kodom:

<?php echo htmlspecialchars($ispis); ?>

Zbog različitih analiza, ispitivanja funkcionalnosti, testiranja sustava, praćenja rada korisnika, ali i sustava implementiran je i dnevnik rada aplikacije. Administrator sustava ima pregled dnevničkih zapisa u kojima su upisani osnovni podaci korisnika i njegova računala kojima se pristupa određenim skriptama aplikacije.

11. Zaključak

U današnje vrijeme sadržaji koji se nalaze na Internetu su većinom privatni i sadrže vrlo osjetljive podatke, a sadržaj se generira dinamički ovisno o zahtjevima korisnika. Zbog sigurnosti korisnika i njihovih podataka sadržaj koji se razmjenjuju između korisnika i Web poslužitelja potrebno je zaštititi. Pristup Web aplikacijama dostupan je s bilo kojega uređaja te su zbog toga Web aplikacije vrlo popularne, a ovisno o svojoj namjeni javlja se problem sigurnosti. Sigurnost se odnosi na zaštitu podataka koje aplikacija koristi, ali i na zaštitu osobnih podataka korisnika koji se koriste Web aplikacijom.

Sigurnosni propusti najčešće se događaju prilikom pisanja koda zbog nedovoljne edukacije programera o sigurnosnim aspektima Web aplikacija i njihovoj implementaciji. Najčešća mjesta napada na Web aplikacije koje hakeri mogu iskoristiti su autentikacija i autorizacija sustava te baza podataka. Napadima na tri navedena mjesta napada hakeri najlakše mogu doći do korisničkih podataka kojima mogu pristupiti Web aplikaciji, a tako i svim ostalim dostupnim podacima o korisniku koji su zapisani u aplikaciji. Kako bi aplikacija bila sigurna za korištenje važno je pratiti koji napadi su hakerima zanimljivi te kako ih spriječiti. Praćenjem najčešćih napada bavi se i organizacija OWASP koja svakih nekoliko godina objavljuje listu top 10 najčešćih napada na Web aplikacije. OWASP je otvorena zajednica koja svim korisnicima pomaže i smjernicama za zaštitu aplikacija.

Osim samog pisanja koda za zaštitu aplikacija, važno je postaviti i kvalitetnu sigurnosnu konfiguraciju poslužitelja kojom se također mogu spriječiti napadi hakera. Kvalitetnoj sigurnosnoj zaštiti u prilog idu i kvalitete programera koji su uvijek spremni usavršavati svoje znanje i biti u stalnoj interakciji sa timom koji je zadužen za razvoj aplikacije. Sigurnost aplikacije se može postići i stalnim testiranjem web aplikacija. Na taj način se mogu pronaći propusti u implementaciji, a pravovremenim testiranjem ti propusti se mogu popraviti prije puštanja aplikacije u korištenje što sprječava da ranjivost aplikacije pronađu napadači i iskoriste ju bez znanja razvojnog tima. Napadi se mogu blokirati i zaustaviti korištenjem

vatrozida koje „stvara“ vrijeme za rješavanje problema. Kvalitetni vatrozidi imaju i mogućnost kontrole prometa na aplikacijskom sloju. Poznavanjem strukture Web aplikacija kreirani su specijalizirani uređaji kojima je omogućena zaštita od deset najkritičnijih sigurnosnih rizika aplikacija, a imaju i funkciju zaštite od DOS i DDOS napada.

Sigurnost korisnika i njihovih osobnih podataka vrlo je važna i zbog toga je važno i znati kako ih zaštititi. Nitko ne bi htio biti na meti hakera, niti se suočavati s problemom krađe identiteta. Stoga se savjetuje da se prilikom izrade aplikacija sigurnost ne promatra samo kao dodatna funkcionalnost, nego da se provodi tijekom cijelog njezinog razvoja. Od programera se traži da kritički razmišljaju o namjeni aplikacije i korisnicima koji bi se njome trebali koristiti, te na temelju tih informacija implementiraju sigurnosne zaštite za sprječavanje napada na korisnike i njihove podatke.

Popis literature

[1] https://www.cert.hr/, dostupno: srpanj 2018.

[2] B. Sullivan i V. Liu, *Web Application Security, A Beginner's Guide*, McGraw Hill Professional, New York, 2011.

[3] *Brute force napadi*, https://www.cis.hr/www.edicija/LinkedDocuments/CCERT-PUBDOC-2007-08-201.pdf, dostupno: srpanj 2018.

[4] *Cookie poisoning*, https://searchsecurity.techtarget.com/definition/cookie-poisoning, dostupno: srpanj 2018.

[5] *Session fixation*, https://www.owasp.org/index.php/Session_fixation, dostupno: srpanj 2018.

[6] Mirko Maleković, Kornelije Rabuzin, *Uvod u Baze podataka*, Fakultet organizacije i informatike Sveučilište u Zagrebu, Varaždin, 2016.

[7] *SQL Injection*, https://www.owasp.org/index.php/SQL_Injection, dostupno: kolovoz 2018.

[8] Cho Ying-Chiang, *Uvođenje i analiza sustava za probijanje sigurnosti web mjesta s primjenom na akademske mreže*, https://hrcak.srce.hr/138078, dostupno: kolovoz 2018.

[9] *CSRF napadi*, https://www.cis.hr/www.edicija/LinkedDocuments/NCERT-PUBDOC-2010-04-297.pdf, dostupno: srpanj 2018.

[10] D. Stuttard i M. Pinto, *The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws*, John Wiley & Sons, Indianapolis, 2011.

[11] *About The Open Web Application Security Project*, https://www.owasp.org/index.php/About_The_Open_Web_Application_Security_Project, dostupno: srpanj 2018.

[12] K. M. Khan, *Developing and evaluating security-aware software systems*, IGI Global, 2012.

[13] *OWASP Top 10 – 2017 The Ten Most Critical Web Application Security Risks,* https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf, dostupno: kolovoz 2018.

[14] *OWASP Risk Rating Methodology*, https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology, dostupno: kolovoz 2018

[15] *OWASP Top 10 – 2013 The Ten Most Critical Web Application Security Risks,* https://www.owasp.org/images/f/f8/OWASP_Top_10_-_2013.pdf , dostupno: kolovoz 2018.

Popis slika

Popis tablica