Primjeri kibernetičkih napada i moguće protumjere

Šimić, Mihovil

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni diplomski studij

Primjeri kibernetičkih napada i moguće protumjere

Diplomski rad

Mihovil Šimić

Osijek, 2023.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

Obrazac D1: Obrazac za imenovanje Povjerenstva za diplomski ispit

Osijek, 05.07.2023.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za diplomski ispit

Ime i prezime Pristupnika:	Mihovil Šimić
Studij, smjer:	Diplomski sveučilišni studij Računarstvo
Mat. br. Pristupnika, godina upisa:	D-1248R, 08.10.2021.
OIB studenta:	48525482169
Mentor:	izv. prof. dr. sc. Krešimir Grgić
Sumentor:	,
Sumentor iz tvrtke:	
Predsjednik Povjerenstva:	doc. dr. sc. Višnja Križanović
Član Povjerenstva 1:	izv. prof. dr. sc. Krešimir Grgić
Član Povjerenstva 2:	mr. sc. Anđelko Lišnjić
Naslov diplomskog rada:	Primjeri kibernetičkih napada i moguće protumjere
Znanstvena grana diplomskog rada:	**Telekomunikacije i informatika (zn. polje elektrotehnika)**
Zadatak diplomskog rada:	Kibernetičke prijetnje u umreženom digitalnom svijetu postale su svakodnevica, te je postalo nužno voditi računa o problematici sigurnosti i mogućim preventivnim protumjerama. U radu je potrebno provesti i analizirati neke primjere mogućih kibernetičkih napada (u kontroliranom laboratorijskom okruženju), te dati smjernice i upute o mogućim preventivnim protumjerama. Također, potrebno je analizirati učinak mogućih protumjera. Tema rezervirana za: Mihovil Šimić
Prijedlog ocjene pismenog dijela ispita (diplomskog rada):	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene od strane mentora:	05.07.2023.

Potvrda mentora o predaji konačne verzije rada:	*Mentor elektronički potpisao predaju konačne verzije.*
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 14.07.2023.

Ime i prezime studenta:	Mihovil Šimić
Studij:	Diplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	D-1248R, 08.10.2021.
Turnitin podudaranje [%]:	7

Ovom izjavom izjavljujem da je rad pod nazivom**: Primjeri kibernetičkih napada i moguće protumjere**

izrađen pod vodstvom mentora izv. prof. dr. sc. Krešimir Grgić

i sumentora ,

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SADRŽAJ

1. UVOD

Kibernetički napadi svakim danom ostavljaju sve veći i ozbiljniji utisak na današnje društvo. Žrtve kibernetičkog kriminala mogu biti individualne osobe, korporacije, organizacije ili državne institucije. Razlozi napada mogu varirati od trivijalnog djelomičnog zaustavljanja rada do krađe financijskih sredstava ili identiteta. Provođenje kibernetičkih napada podrazumijeva korištenje sofisticiranih tehnika i alata što znači da se za uspješno sprečavanje istih treba koristiti interdisciplinarni pristup. Polja računalnih i informacijskih tehnologija neprestano se razvijaju i šire, te se analogno tomu i kibernetički napadi unaprjeđuju, što povećava njihov utjecaj na društvo i gospodarstvo. Iz navedenih razloga može se zaključiti da polje kibernetičke sigurnosti igra relevantnu ulogu u zaštiti od istih. Kibernetička sigurnost obuhvaća široki raspon strategija, tehnika i praksi koje su usmjerene na otkrivanje, sprječavanje i ublažavanje posljedica kibernetičkih napada. Ista se ne ograničava samo na računalnu zaštitu, već uključuje i upravljanje rizikom, edukaciju zaposlenika, razvoj politika i slično. Iz svih navedenih razloga jasno je da kibernetički napadi mogu imati iznimno štetne posljedice stoga je diplomski rad usredotočen na provedbu i analizu posljedica kibernetičkih napada u kontroliranom laboratorijskom okruženju.

U prvom dijelu objašnjen je i detaljno analiziran utjecaj kibernetičkih napada, od povijesno popularnih napada koji su potaknuli razvoj kibernetičke sigurnosti do detaljne statistike aktualnih kibernetičkih napada. Također su objašnjeni ekonomski i društveni utjecaji kibernetičkih napada. Provedba svih napada odrađena je unutar sigurnog laboratorijskog okruženja, a postavljanje istoga je objašnjeno pri početku drugog dijela rada. Provedba i analiza posljedica kibernetičkih napada odrađena je u drugom dijelu te su za svaki napad detaljno objašnjene protumjere i strategije za uspješnu obranu. U trećem dijelu rada su detaljno opisane generalne protumjere koje predstavljaju učinkovita rješenja za obranu od većine kibernetičkih napada.

2. ANALIZA UTJECAJA KIBERNETIČKIH NAPADA

Kibernetički napadi već dugi niz godina imaju značajan ekonomski i društveni učinak. Krajnji ciljevi napada mogu varirati od krađe podataka korisnika do pukog uništavanja sustava. Prije prelaska na istraživanje kibernetičkih napada potrebno je objasniti pojam kibernetičkog napada. Kibernetički napad je svaka aktivnost koja može prouzročiti ozbiljnu štetu računalnom sustavu, mreži ili bilo kojoj drugoj digitalnoj imovini organizacije ili pojedinca koja igra bitnu ulogu u svakodnevnom životu istih. Kibernetičke napade se također može i opisati kao napade osmišljene za iskorištavanje stvarnih ranjivosti u sustavima i mrežama, kako bi im se pristupilo. Budući da jedan kibernetički napad može uključivati iskorištavanje više ranjivosti, obično je prikladnije nazvati ga kombiniranim kibernetičkim napadom. Kibernetičke napade mogu provesti zaposlenici unutar organizacije ili neidentificirane vanjske strane na udaljenim lokacijama. Svaka nova tehnologija uz svoje predviđene koristi donosi i nove prijetnje. Ovo je posebno važno kod implementacije IT sustava unutar organizacije kao dijela poslovnog procesa. Iako nove tehnologije donose značajna poboljšanja u radu organizacije, u isto vrijeme otvaraju mogućnost za kibernetički napad [1].

2.1. Povijesni pregled kibernetičkih napada

Kibernetički napadi datiraju još od samih početaka korištenja računalnih sustava te svakim danom postaju sve učestaliji i sofisticiraniji. S godinama se sve više razvijaju postojeći i pronalaze novi načini za iskorištavanje ranjivosti sustava. Proučavanje prethodnih kibernetičkih napada i detaljno istraživanje kako su izvedeni uvelike može pomoći u obrani od budućih.

2.1.1. Morris računalni crv

Računalni crvi predstavljaju značajan rizik za sigurnost mreže. Sadrže isti destruktivni potencijal kao i drugi objekti zlonamjernog koda ali s dodatnom funkcionalnošću. Prema [2] računalni crv je definiran kao samo replicirajući zlonamjerni kod koji se može izvršiti i širiti neovisno o žrtvinim aplikacijama ili datotekama. Za razliku od virusa, crvi se razmnožavaju sami, bez ljudske intervencije.

Godine 1988. jedan je student izveo prvi veliki napad na Internetu i postao prva osoba osuđena za novu vrstu zločina. Ovaj kibernetički crv počeo se širiti nevjerojatnom brzinom i zaustavio brojna računala. Unutar 24 sata, procjenjuje se da je pogođeno od 6000 do 60000 računala koja su tada bila spojena na Internet. Zlonamjerni program zarazio je sustave na nizu prestižnih fakulteta, javnih i privatnih istraživačkih centara koji su činili ranu nacionalnu elektroničku mrežu. Među

brojnim žrtvama bili su Harvard, Princeton, Stanford, Johns Hopkins, NASA i Nacionalni laboratorij Lawrence Livermore. Znatno je usporio rad vitalnih vojnih i sveučilišnih funkcija. Elektroničke poruke su kasnile danima. Mrežna zajednica trudila se otkriti kako crv radi i kako ga ukloniti. Točnu štetu bilo je teško kvantificirati, ali procjene su počele od 100000 USD i skočile su do nekoliko milijuna. Nakon što je incident postao javan, FBI je pokrenuo istragu. Agenti su brzo utvrdili da Robert Tappan Morris, 23-godišnji student Sveučilišta Cornell, stoji iza napada te su počeli ispitivati njega i njegove suradnike te dešifrirati njegove računalne datoteke, što je dovelo do mnoštva inkriminirajućih dokaza. Godine 1986. Kongres je odobrio Zakon o računalnim prijevarama i zlouporabi, čime su zabranili neovlašteni pristup zaštićenim računalima. Tužitelji su optužili Morrisa 1989. godine. Sljedeće godine porota ga je proglasila krivim, čime je postao prva osoba osuđena prema zakonu iz 1986. godine. Morris je, međutim, bio pošteđen zatvora, umjesto toga dobio je novčanu kaznu, uvjetnu kaznu i nalog da završi 400 sati društveno korisnog rada. Ovaj događaj imao je značajan utjecaj na naciju koja je tada shvatila koliko su računala postala važna i ranjiva. Ideja kibernetičke sigurnosti postala je nešto što su korisnici računala počeli shvaćati ozbiljnije. Samo nekoliko dana nakon napada, u Pittsburghu je prema uputama Ministarstva obrane osnovan prvi tim za hitne slučajeve u zemlji. Programeri su također počeli stvarati neophodne programe za otkrivanje neovlaštenih upada u računalo [3].

Slika 1. Izvorni kod Morris crva na disketi čuvan u muzeju [4]

Slika 1. prikazuje disketu s izvornim kodom Morris crva koja se nalazi u Muzeju Povijesti Računala u gradu Mountain View, Kalifornija. Detaljna analiza načina rada crva je krucijalna kako bi se točno objasnio način na koji se crv širio. Mete crva bila su Sun 3 i VAX računala koja su koristila operacijski sustav Berkeley 4.3 UNIX te su sadržavala TCP/IP Internet protokole. Njegova jedina svrha bila je prelazak na nova računala zaobilaženjem postupaka provjere autentičnosti i širenje novih kopija samog sebe. Svaki novi crv je prilikom nastanka izgradio popis udaljenih računala koje može napasti. Popis je napravio uz pomoć tablice koja je sadržavala računala koja su smatrala trenutnog domaćina crva povjerljivim, te uz popis kontakata korisnika domaćina, tablica po kojima se korisnicima zadaju dopuštenja za pristup udaljenim računima te na posljetku iz programa koji javlja status mrežnih veza. Na svaki od potencijalnih novih domaćina, crv je pokušao ući na razne načine:

- lažno se predstavljajući kao korisnik prijavom u njegov račun nakon pronalaženja lozinke istoga
- iskorištavanjem greške u *finger* protokolu, koja javlja gdje se nalazi udaljeni korisnik
- iskorištavanjem ranjivosti putem opcije otklanjanja pogrešaka na udaljenom procesu, koji prima i šalje poštu.

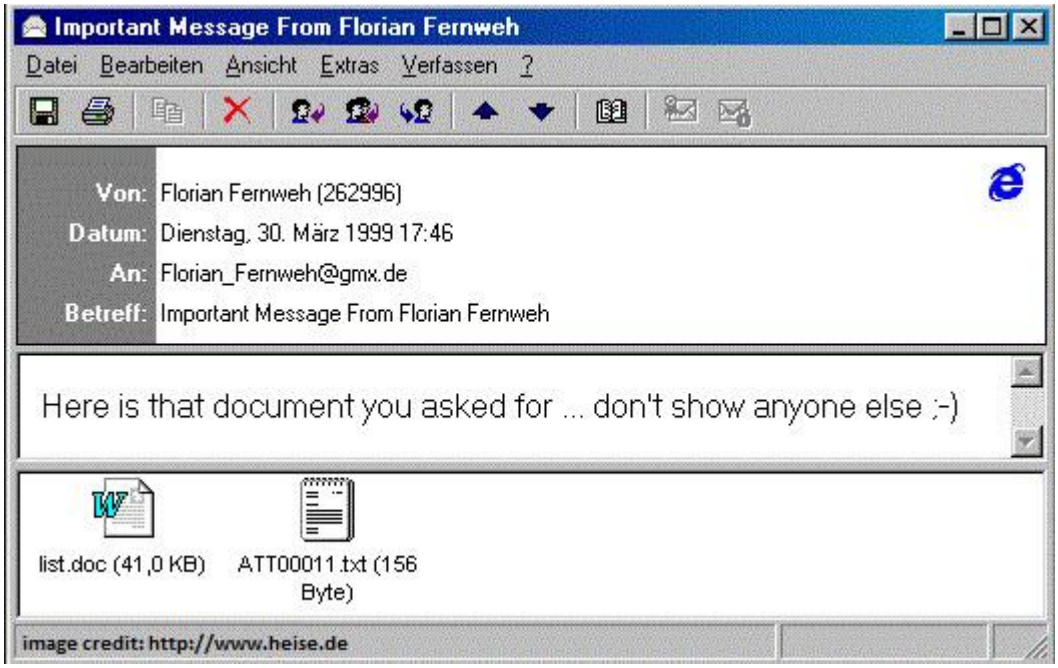
Paralelno s napadima na nova računala, crv je pokušavao pogoditi lozinke korisničkih računa na trenutnom domaćinu. Prvobitno je isprobao naziv računa i njegove jednostavne permutacije, zatim popis od 432 ugrađene lozinke, te na kraju sve riječi iz lokalnog rječnika. Neotkriveni crv mogao je provesti dane pokušavajući probiti lozinku. Ako je bilo koji od njegovih napada na nove domaćine uspio, crv bi imao mogućnost komunikacije s naredbenim retkom na udaljenom stroju. U naredbeni redak bi unio prethodno pripremljeni program od 99 linija, s naredbama za prevođenje i izvršavanje, a zatim bi prekinuo vezu. Ako bi se program uspješno pokrenuo, pozvao bi nadređenog crva u roku od 120 sekundi. Roditeljski crv kopirao je šifrirane datoteke koje su sadržavale cijeli kod crva. Roditeljski crv je zatim pokrenuo naredbe za konstruiranje novog crva iz šifriranih datoteka i pokretanje istoga. Crv je također pokušao kontrolirati populaciju, tražeći druge crve u istom domaćinu i odlučivanje koji će od njih biti terminiran. Međutim, crv koji je trebao biti terminiran bi prvo napao mnogo drugih računala što je dovelo do znatno brojnijeg stvaranja crva u odnosu na terminiranje. Štoviše, jedan od svakih sedam crva proglasio se besmrtnim i potpuno je zaobišao svako sudjelovanje u kontroli populacije. Izvorni kod crva je šifriran i poslan na udaljenog domaćina jedino kada bi se prethodno pripremljeni program uspješno pokrenuo, što je uvelike poboljšalo skrivenost crva. Novi crv nije ostavio nikakve tragove u datotečnom sustavu, kopirao je sve svoje datoteke u memoriju i izbrisao ih iz direktorija sustava.

Crv je onemogućio funkciju sustava koja stvara datoteku s izvatkom (engl. *memory dump*) u slučaju pogreške, i kriptirao je sve nizove znakova, tako da bi u slučaju dolaska do datoteke s izvatkom, ista bila beskorisna. Datoteka s izvatkom bilježi sav sadržaj memorije kada se računalo neočekivano zaustavi, što bi znatno olakšalo otkrivanje i zaustavljanje crva. Crv si je dodijelio ime koje je izgledalo bezazleno programu koji ispisuje procese sustava i često je mijenjao svoj identifikator procesa [5].

2.1.2. Melissa virus

Virus je zlonamjerni kod koji se replicira nakon što se priloži datotekama na žrtvinom uređaju. Kada se žrtvine datoteke pokrenu, virus može izvršiti svoj kod. Drugim riječima, virusi se ne mogu sami razmnožavati [2]. Virusi mogu biti iznimno štetni i mogu trajno uništiti podatke, usporiti sustav i bilježiti pritiske tipki.

Krajem ožujka 1999. godine, programer po imenu David Lee Smith oteo je jedan America Online (AOL) račun i upotrijebio ga za postavljanje datoteke na internetsku grupu pod nazivom *alt.sex*. Objava je obećavala desetke besplatnih lozinki za Internet stranice sa sadržajem za odrasle. Kada su korisnici zagrizli mamac, preuzeli dokument i zatim ga otvorili u Microsoft Wordu, virus je pušten na njihova računala. 26. ožujka virus se počeo rapidno širiti. Virus Melissa, kojeg je Smith navodno nazvao po striptizeti na Floridi, započeo je preuzimanjem kontrole nad Microsoft Word programom na žrtvinom računalu. Zatim je pomoću makronaredbe preoteo njihov Microsoft Outlook sustav elektroničke pošte i poslao poruke na prvih 50 adresa na njihovim listama kontakata. U programu Word često korišteni zadatci se mogu automatizirati stvaranjem i pokretanjem makronaredbi. Makronaredba je niz naredbi i uputa koje se grupiraju kao jedna naredba u svrhu automatizacije zadataka. Te su poruke dovodile primatelje u iskušenje da otvore privitak zaražen virusom dajući mu imena kao što su *sexxxy.jpg* ili *naked women* ili lažno tvrdeći da je u privitku dokument koji je primatelj tražio. Uz pomoć lukavog društvenog inženjeringa, virus je djelovao poput automatiziranog lančanog pisma [7].



Slika 2. Poruka s privitkom koji sadrži Melissa virus [6]

Slika 2. prikazuje kako je izgledala poruka koja sadrži Melissa virus u privitku. Primatelji nisu imali nikakve sumnje jer na prvi pogled poruka izgleda bezopasno i intrigantno. Virus nije bio namijenjen krađi novca ili informacija, ali je svejedno uzrokovao dosta problema. Poslužitelji elektroničke pošte u više od 300 korporacija i vladinih agencija diljem svijeta postali su preopterećeni, a neki su morali biti potpuno ugašeni, uključujući Microsoft. U roku od nekoliko dana, stručnjaci za kibernetičku sigurnost su velikim dijelom zaustavili širenje virusa i vratili funkcionalnost svojih mreža, iako je trebalo određeno vrijeme da se virus u potpunosti ukloni. Uz svoju istražnu ulogu, FBI je slao upozorenja o virusu i njegovim učincima, pomažući u smanjenju destruktivnih učinaka napada. Ipak, kumulativna šteta bila je ogromna, otprilike 80 milijuna USD za čišćenje i popravak pogođenih računalnih sustava. Pronalaženje krivca nije dugo trajalo, zahvaljujući dojavi predstavnika AOL-a i gotovo besprijekornoj suradnji između FBI-a, policije New Jersey-a i drugih partnera. Vlasti su pratile elektronske tragove virusa do Smith-a, koji je uhićen u sjeveroistočnom New Jersey-u 1. travnja 1999. godine. Smith je priznao krivnju u prosincu 1999., a u svibnju 2002. osuđen je na 20 mjeseci saveznog zatvora i novčanu kaznu od 5000 USD. Također je dogovorio suradnju s federalnim i državnim vlastima. Virus Melissa, koji se u to vrijeme smatrao zarazom koja se najbrže širila, za mnoge je Amerikance bio grubo upoznavanje s tamnom stranom Interneta. Svijest o opasnosti otvaranja neželjenih privitaka elektroničke pošte počela je rasti sa stvarnošću virusa i štete koju mogu učiniti. Kao i crv Morris nešto više od desetljeća ranije, virus Melissa bio je dvosjekli mač, koji je doveo do poboljšanja Internet sigurnosti, a istodobno je poslužio kao inspiracija za val još skupljih i moćnijih

kibernetičkih napada. Za FBI i njegove kolege, virus je bio znak upozorenja na veliku rastuću prijetnju i potrebu da se brzo pojačaju kibernetičke sposobnosti i partnerstva. Prikladno, nekoliko mjeseci nakon što je Smith-u izrečena kazna, FBI je uspostavio svoj novi nacionalni kibernetički odjel usredotočen isključivo na Internet zločine, s resursima i programima posvećenim zaštiti američkih elektroničkih mreža od štete [7].

Nakon što je korisnik otvorio Word dokument u kojemu se nalazio Melissa virus, isti je napravio sljedeće:

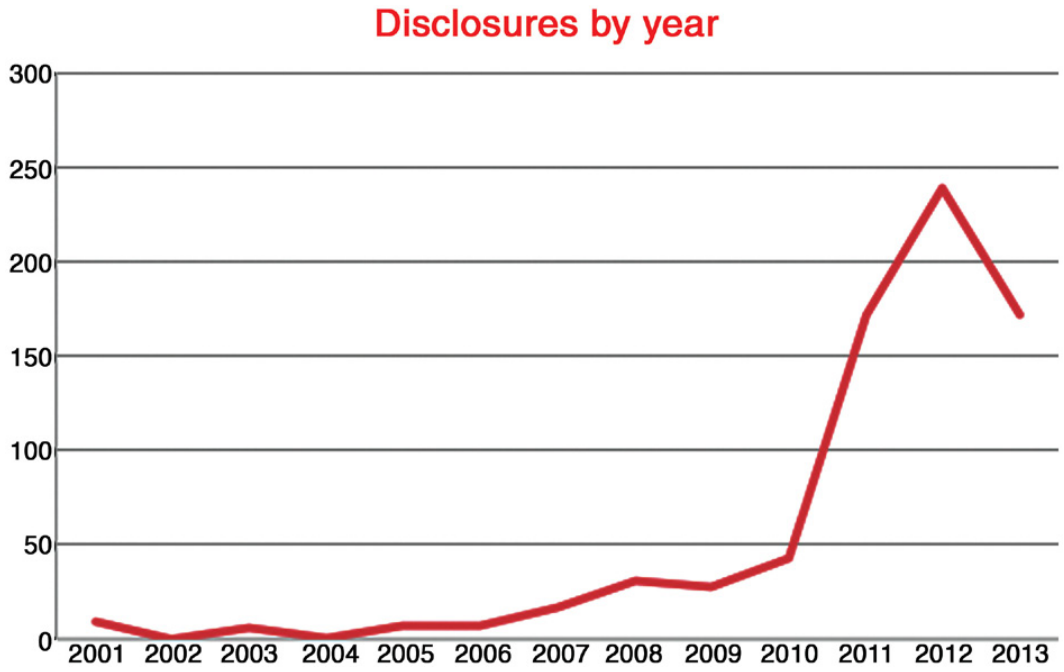
- provjerio je li instaliran Word 97 ili Word 2000
- onemogućio određene značajke softvera, što je otežalo otkrivanje virusa
- poslao kopije zaraženog dokumenta na 50 drugih adresa koristeći kompatibilne verzije Microsoft Outlook-a
- modificirao Word program tako da virus zarazi svaki dokument kojim je korisnik upravljao. Ako se ti dokumenti dijele, virus se širi.

Ako je lista kontakata korisnika sadržavala dosta adresa elektroničke pošte iste organizacije, virus je mogao dosta brzo preopteretiti poslužitelje elektroničke pošte što je rezultiralo uskraćivanjem usluge. Prema CERT Koordinacijskom Centru na Sveučilištu Carnegie Mellon, jedno je mjesto prijavilo primanje 32000 kopija elektroničke pošte koje su sadržavale Melissa virus unutar 45 minuta. Zapravo, ono po čemu se virus uistinu razlikovao od ostalih makro virusa bila je njegova sposobnost da iskoristi prednosti Microsoft-ove aplikacije za elektroničku poštu i brzina kojom se širio. Prema CERT Koordinacijskom Centru, prvi potvrđeni izvještaji o virusu primljeni su u petak, 26. ožujka 1999. godine. Do ponedjeljka, 29. ožujka, zaraženo je više od 100000 računala u više od 300 organizacija. Naime, osim što je virus preopteretio sustave elektroničke pošte, nije trajno oštetio informacijske sustave državnog i privatnog sektora i nije ugrozio osjetljive državne podatke. Analogno širenju Melissa virusa, na površinu su izašle različite varijacije istoga, uključujući virus Papa, Microsoft Excel 97 ili Excel 2000 makro virus koji se također širio putem elektroničke pošte. Prema Microsoft-u, navedeni virus je mogao generirati naredbe koje bi rezultirale značajnim zagušenjem mrežnog prometa [8].

2.1.3. Stuxnet

Stuxnet je odigrao relevantnu ulogu kada je u pitanju zlonamjerni softver u industrijskim kontrolnim sustavima (engl. *industrial control system*) ili skraćeno ICS jer je bio prvi ciljani

naoružani kibernetički napad na industrijske kontrolne sustave. Prema [9] industrijski kontrolni sustav se definira kao informacijski sustav koji se koristi za kontrolu industrijskih procesa kao što su proizvodnja, rukovanje proizvodima i distribucija. Industrijski kontrolni sustavi uključuju sustave nadzorne kontrole i prikupljanja podataka koji se koriste za kontrolu geografski raspršene imovine, kao i distribuirane upravljačke sustave i manje upravljačke sustave koji koriste programabilne logičke kontrolere za kontrolu lokaliziranih procesa. Prije Stuxnet-a, još je uvijek bilo uvriježeno mišljenje da su industrijski sustavi imuni na kibernetičke napade zbog opskurnosti i izolacije sustava te da nikada neće biti meta kibernetičkih kriminalaca. Prije Stuxnet-a uglavnom se smatralo da su kibernetičke prijetnje ograničene na slučajne zaraze sustava ili unutarnje prijetnje. Iz navedenih razloga se jasno može zaključiti zašto je Stuxnet bio toliko razglašen i zašto se o njemu i danas govori [10].



Slika 3. Otkrivanje ranjivosti ICS sustava po godinama (2001. – 2013.) [10]

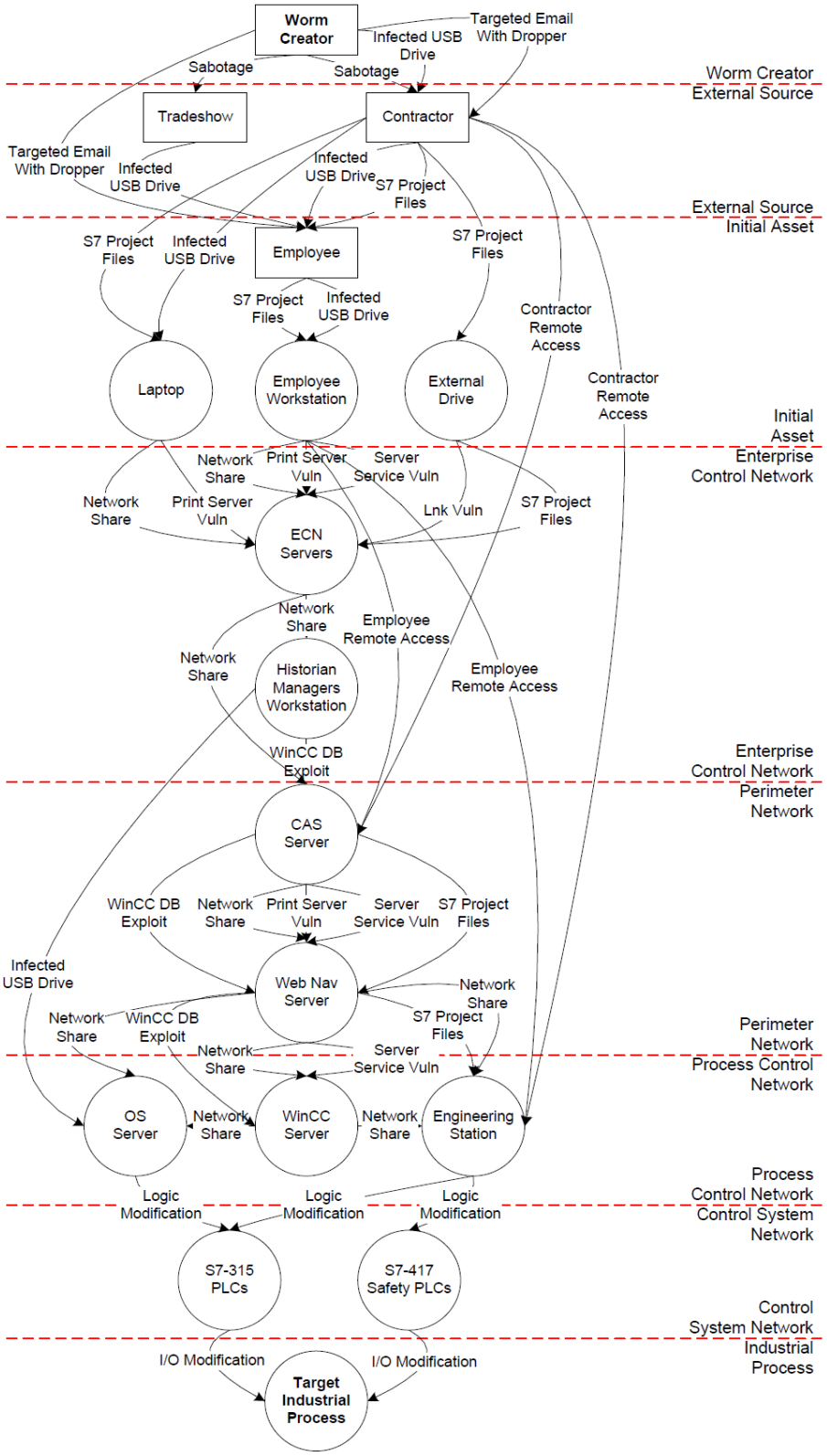
Iz slike 3. se jasno može vidjeti utjecaj Stuxnet-a na istraživanja sigurnosti ICS sustava. Nakon što je opasnost naoružanih kibernetičkih napada na industrijske kontrolne sustave pravilno shvaćena broj otkriven ranjivosti se naglo povećao. Cijela industrija je počela pružati više pažnje sigurnosti vlastitih sustava. Iako su se prve verzije Stuxnet-a počele pojavljivati početkom studenog 2007. godine, široko rasprostranjene rasprave o istome nisu počele sve do ljeta 2010. godine, nakon što je kibernetički tim pod imenom Industrial Control Systems Cyber Emergency Response Team ili skraćeno ICS-CERT izdao upozorenje [10]. To je bio prvi poznati zlonamjerni softver prilagođen ugrožavanju PLC softvera i imao je sljedeće karakteristike:

- četiri iskorištene ranjivosti nultog dana (greške koje su prethodno bile nepoznate programerima)
- Windows *rootkit* (omogućuje zlonamjernom softveru privilegirana prava i skriva njegovo postojanje od softvera za otkrivanje upada)
- uključuje različite datume zaustavljanja izvršenja za onemogućavanje širenja zlonamjernog softvera i rad u unaprijed određenim budućim vremenima.

Stuxnet je mogao ugroziti računala iskorištavanjem ranjivosti nultog dana putem zaraženog USB memorijskog štapića. Uz navedeni način širenja, postojala su još dva, putem lokalne mreže i putem Siemens projektnih datoteka. Kada je uređaj bio zaražen, Stuxnet je pokušao ažurirati svoj kod s Interneta. Osim ako je zaraženi uređaj imao specifičnu platformu koju je Stuxnet ciljao, isti je ostao neaktivan i nastavio je širiti zarazu. Koristeći ugrožene digitalne certifikate, Stuxnet je imao mogućnost zaobilaska vatrozida i nastavak širenja lokalnom komunikacijskom mrežom SCADA sustava. Prema [11] sustavi za nadzor kontrole i prikupljanje podataka (engl. *Supervisory Control and Data Acquisition*) ili SCADA koriste se za upravljanje, nadzor i analizu industrijskih uređaja i procesa. Sustav se sastoji od softverskih i hardverskih komponenti i omogućuje daljinsko i prikupljanje podataka na licu mjesta iz industrijske opreme. Tako tvrtkama omogućuje daljinsko upravljanje industrijskim lokacijama kao što su vjetroelektrane, jer tvrtka može pristupiti podacima o turbinama i kontrolirati ih bez boravka na licu mjesta. Putem vršnjačke komunikacije (engl. *peer to peer*) Stuxnet je imao mogućnost samostalnog ažuriranja, čak i kada zaraženi uređaj nije imao izravan pristup Internetu. Nakon što je ciljani PLC bio zaražen, Stuxnet je promijenio svoj način rada. Koristeći PLC *rootkit*, zlonamjerni softver je modificirao PLC kod kako bi izvršio napad u svrhu razotkrivanja i dokumentiranja primljenih podatka. Nakon određenog vremena dokumentiranja podataka, Stuxnet je započeo sa sabotiranjem sklopovlja sustava. U slučaju pronalaska određenih postavki regulatora frekvencije, Stuxnet je izmijenio postavke frekvencije sabotirajući sustav centrifuge usporavanjem, a zatim ubrzavanjem motora na različite brzine u različitim vremenima. Prilikom promjene upravljačkog signala koji se šalje aktuatorima, Stuxnet je sakrio štetu na postrojenju tako što je poslao prethodno dokumentirane podatke u SCADA sustav [12].

Ako kritične infrastrukture u svijetu žele biti sigurne, onda vlasnici i radnici moraju prepoznati da su njihovi kontrolni sustavi sada meta sofisticiranih napada i trebaju prilagoditi svoje sigurnosne programe u skladu s tim. Konkretno, prema [13] sigurnosni programi moraju:

- razmotriti sve moguće načine zaraze i imati strategije za ublažavanje istih, umjesto fokusiranja na jedan način kao što su USB memorijski štapići
- instalirati tehnologije za otkrivanje upada i podizanje alarma kada je oprema ugrožena ili u opasnosti od ugrožavanja
- implementirati, upravljati i održavati na maksimalnoj učinkovitosti sigurnosne tehnologije i prakse prikladne za industrijske kontrolne sustave, uključujući vatrozid, antivirusnu tehnologiju, sustave zakrpa i popis dopuštenja dizajniran za SCADA/ICS sustav, kako bi se napadi sofisticiranog zlonamjernog softvera uvelike otežali
- proučiti vatrozide sposobne za dublje proučavanje paketa od ključnih SCADA i ICS protokola
- uključiti sigurnosne procjene i testiranja kao dio razvoja sustava i povremene procese održavanja
- prepoznati i ispraviti potencijalne ranjivosti, čime se smanjuje vjerojatnost uspješnog napada
- raditi na poboljšanju kulture industrijske sigurnosti među menadžmentom i tehničkim osobljem



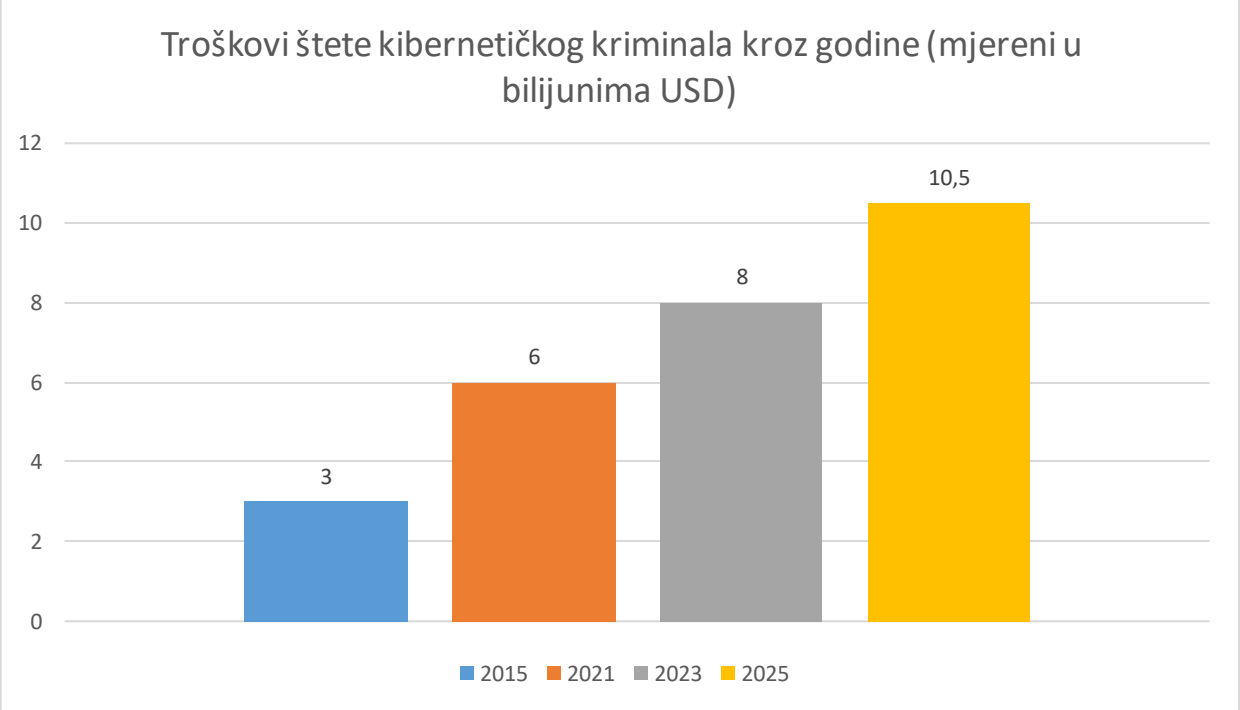
Slika 4. Graf napada Stuxnet-a [13]

Slika 4. prikazuje graf napada Stuxnet-a te se jasno mogu vidjeti različiti vektori i posljedice napada. Stuxnet je imao izričito razvijen način širenja, u svakom koraku se mogao nastaviti širiti na više načina te je pritom održavao visoku razinu prikrivenosti.

2.2. Ekonomski i društveni učinak kibernetičkih napada

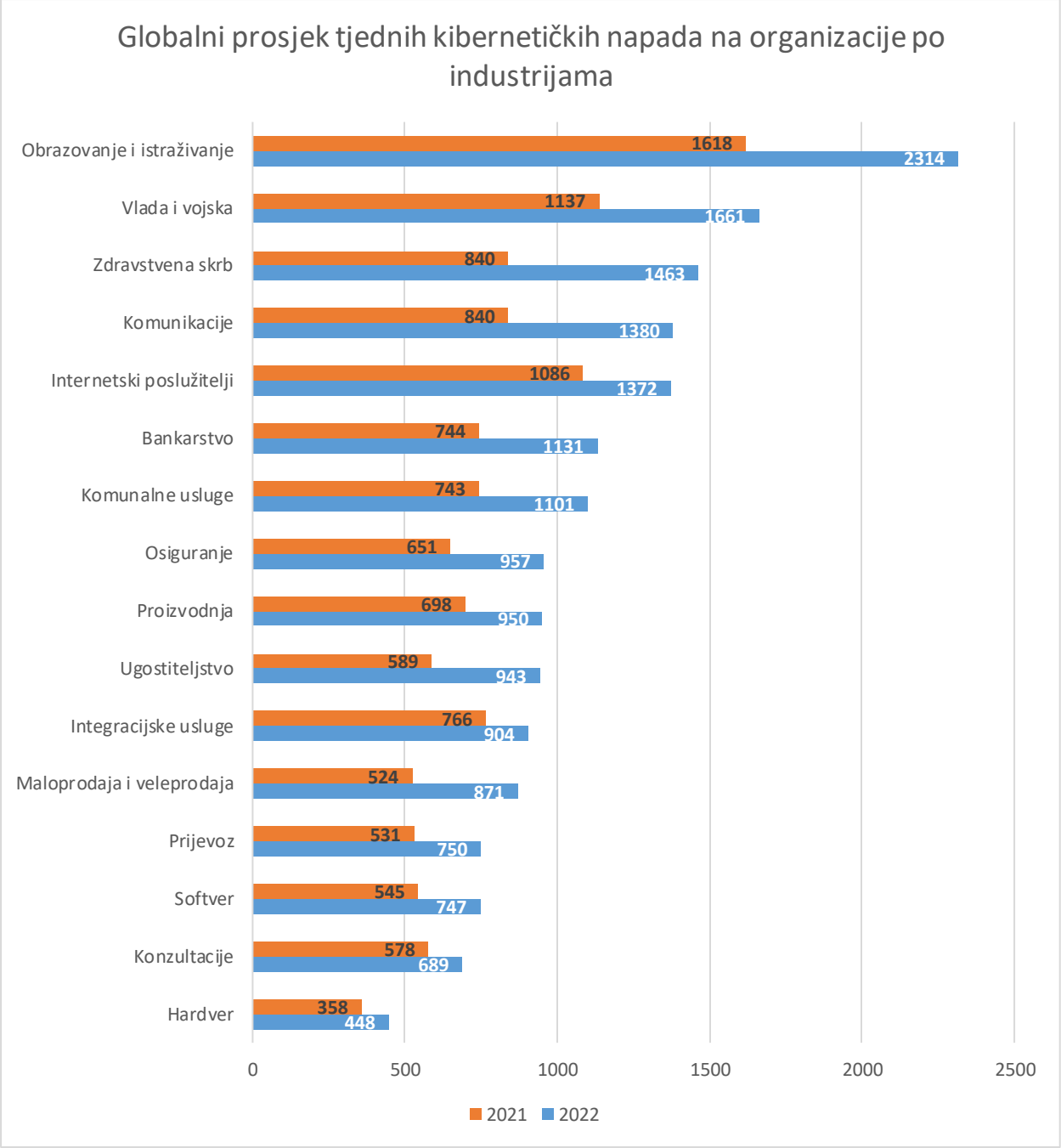
Sve snažniji i utjecajniji učinak kibernetičkih napada na ekonomiju i društvo je neporeciv. Posljedice povrede podataka postaju sve ozbiljnije, što kao rezultat ima značajan financijski i reputacijski udar na organizacije i institucije. Kibernetički napadi predstavljaju prijetnju na globalnoj razini, te zavisno od dosta faktora neke zemlje i područja su sklonija napadima. Jedan od glavnih faktora je spremnost zemalja za sprečavanje kibernetičkih napada te mogućnost pravilnog upravljanja kibernetičkim incidentima te minimiziranje posljedica.

Predviđa se da će globalni godišnji trošak kibernetičkog kriminala dosegnuti 8 bilijuna USD u 2023. godini. Troškovi kibernetičkog kriminala uključuju oštećenje i uništavanje podataka, ukradeni novac, izgubljenu produktivnost, krađu intelektualnog vlasništva, krađu osobnih i financijskih podataka, prijevare, poremećaje normalnog tijeka poslovanja nakon napada, forenzičku istragu, vraćanje i brisanje ugroženih podatka i sustava te narušavanje ugleda i kredibiliteta [14].



Grafikon 1. Troškovi štete kibernetičkog kriminala kroz godine [14]

Grafikon 1. prikazuje kretanje troškova kibernetičkog kriminala kroz prijašnje godine te predviđanja za naredne godine. Troškovi štete rastu iz godine u godinu te postaju sve ozbiljniji.



Grafikon 2. Globalni prosjek tjednih kibernetičkih napada na organizacije po industrijama [15]

Grafikon 2. jasno prikazuje rast globalnog prosjeka kibernetičkih napada u 2022. u odnosu na 2021. godinu. Obrazovna i istraživačka industrija bila je najčešća meta, s prosjekom od 2314 napada tjedno, što iznosi povećanje od 43% u odnosu na 2021. godinu. Zdravstveni sektor zabilježio je najveći porast u napadima, čak 75 % više napada nego prijašnje godine. Kibernetički kriminalci su se usredotočili na napadanje institucija u zdravstvenom sektoru još od početka epidemije bolesti COVID-19, kako bi ostvarili financijsku dobit. Što se tiče količine napada s obzirom na geografsko područje, Afrika je imala najveću količinu napada s 1875 tjednih napada po organizaciji, a iza nje slijedi područje Azije i Pacifika s 1691 tjednim napadom po organizaciji.

Napadi u Europi narasli su za 26 % u odnosu na 2021. godinu, što iznosi najveće povećanje u odnosu na ostala područja [15].

2.2.1. Nacionalni indeks kibernetičke sigurnosti

Prema [16] nacionalni indeks kibernetičke sigurnosti (engl. *national cyber security index*) ili NCSI se definira kao indeks koji mjeri spremnost zemalja da spriječe kibernetičke prijetnje i upravljaju kibernetičkim incidentima. Pokazatelji nacionalnog indeksa kibernetičke sigurnosti razvijeni su u skladu s nacionalnim okvirom kibernetičke sigurnosti. Temeljne kibernetičke prijetnje su uskraćivanje usluga, povreda integriteta podataka te kršenje povjerljivosti podataka. Ove prijetnje izravno utječu na normalno funkcioniranje nacionalnih informacijskih i komunikacijskih sustava, te na elektroničke usluge. Kako bi upravljala tim kibernetičkim prijetnjama, država mora imati razvijenu kibernetičku sigurnost i sustav upravljanja incidentima. NCSI se fokusira na mjerljive aspekte kibernetičke sigurnosti koje implementira vlada:

- važeće zakonodavstvo – pravni akti, propisi, naredbe
- osnovane jedinice – postojeće organizacije, odjeli
- formati suradnje – odbori, radne skupine
- ishodi – politike, vježbe, tehnologije, web stranice, programi.

NCSI pokazuje postotak koji je zemlja dobila od maksimalne vrijednosti pokazatelja. Uz NCSI, također se mjeri i razina digitalnog razvoja (engl. *digital development level*) ili DDL. DDL se izračunava prema razini razvijenosti informacijskih i komunikacijskih tehnologija (engl. *ICT development index*) ili IDI te razini umrežene spremnosti (engl. *networked readiness index*) ili NRI. DDL je prosječni postotak koji je zemlja dobila od maksimalne vrijednosti oba indeksa. Uz sve navedeno računa se i razlika, a ista pokazuje odnos između NCSI-a i DDL-a. Pozitivan rezultat pokazuje da je razvoj kibernetičke sigurnosti zemlje u skladu s digitalnim razvojem društva ili ispred njega. Negativan rezultat pokazuje da je digitalno društvo zemlje naprednije od nacionalnog područja kibernetičke sigurnosti [16].

Rang	Država	NCSI	DDL	Razlika
1.	Belgija	94,81	74,07	20,74
2.	Litva	93,51	67,34	26,17
3.	Estonija	93,51	75,59	17,92
4.	Češka	92,21	69,21	23,00
5.	Njemačka	90,91	80,01	10,90
6.	Rumunjska	89,61	59,84	29,77
7.	Grčka	89,61	64,02	25,59
8.	Portugal	89,61	68,46	21,15
9.	Ujedinjeno Kraljevstvo	89,61	79,96	9,65
10.	Španjolska	88,31	72,21	16,10
11.	Poljska	87,01	65,03	21,98
12.	Austrija	85,71	75,76	9,95
13.	Finska	85,71	78,35	7,36
14.	Saudijska Arabija	84,42	63,89	20,53
15.	Francuska	84,42	77,29	7,13
16.	Švedska	84,42	81,51	2,91
17.	Danska	84,42	82,68	1,74
18.	Hrvatska	83,12	64,63	18,49
19.	Slovačka	83,12	65,44	17,68
20.	Nizozemska	83,12	81,86	1,26
21.	Srbija	80,52	59,81	20,71
22.	Malezija	79,22	62,19	17,03
23.	Italija	79,22	67,26	11,96
24.	Švicarska	76,62	82,93	-6,31

Tablica 1. Države rangirane prema NSCI-u [17]

Iz tablice 1. se može vidjeti da je Belgija najbolje rangirana država s nacionalnim indeksom kibernetičke sigurnosti od 94,81 %. Švicarska ima negativnu razliku između NCSI-a i DDL-a koja iznosi -6,31, a razlog tomu je naprednije digitalno društvo od nacionalnog područja kibernetičke sigurnosti u zemlji. Švicarska ima najveću razinu digitalnog razvoja na svijetu koja iznosi 82,93 %. Iako nije prikazano u tablici, prema podacima najgoru razliku ima Sveti Kristofor i Nevis koja iznosi -60,71. Razlika je tolika jer je NSCI nizak, a DDL dosta visok. Najbolji omjer je postigao

Bangladeš, gdje razlika iznosi 34,42. Iako je digitalno društvo slabo razvijeno, područje kibernetičke sigurnosti u državi je vrlo napredno, što je dovelo do ovakvog rezultata. Najniži NCSI imaju Južni Sudan, Palau, Mikronezija i Maršalovi Otoci, a isti iznosi svega 1,30 %. Hrvatska se plasirala na 18. mjesto s NCSI-om od 83,12 %. Prema [18] sljedeće kategorije opisuju na koji način se računa NCSI. Svaka kategorija ima nekoliko zahtjeva koji trebaju biti ispunjeni za dobivanje bodova. U svrhu primjera uzeti su podaci za Hrvatsku.

Opći pokazatelji kibernetičke sigurnosti:

- razvoj politike kibernetičke sigurnosti: 100 %
- analiza kibernetičkih prijetnji i informacija: 100 %
- obrazovanje i profesionalni razvoj: 33 %
- doprinos globalnoj kibernetičkoj sigurnosti: 33 %

Osnovni pokazatelji kibernetičke sigurnosti:

- zaštita digitalnih usluga: 100 %
- zaštita neophodnih usluga: 83 %
- usluga za digitalnu identifikaciju i povjerenja: 89 %
- zaštita osobnih podataka: 100 %

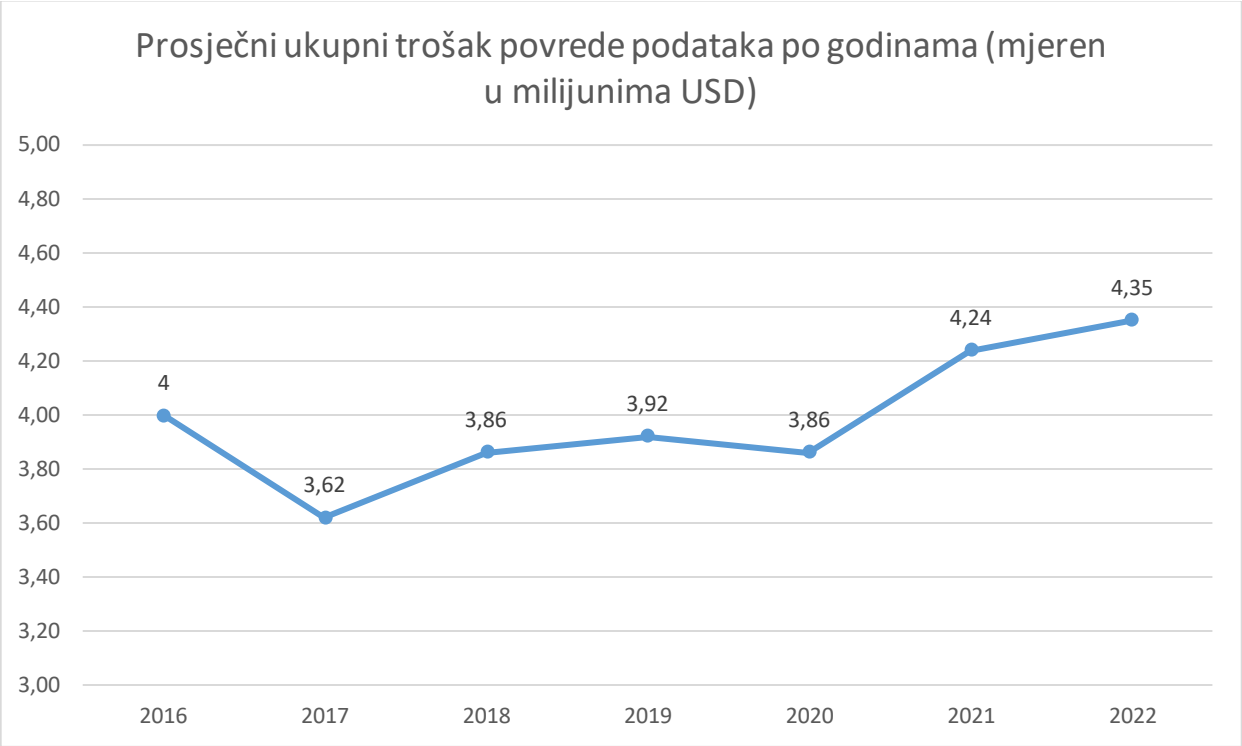
Pokazatelji upravljanja incidentima i krizom:

- odgovor na kibernetičke incidente: 100 %
- upravljanje kibernetičkom krizom: 80 %
- borba protiv kibernetičkog kriminala: 100 %
- vojne kibernetičke operacije: 100 %

ICT Hrvatske je 72 % čime je 36. zemlja na svijetu, a NRI je 57 % što ju plasira na 45. mjesto. Na temelju ova dva broja se može vidjeti kako se došlo do DDL-a Hrvatske, a isti se računa kao prosječna vrijednost oba indeksa [18].

2.2.2. Posljedice povrede podataka tijekom kibernetičkih napada

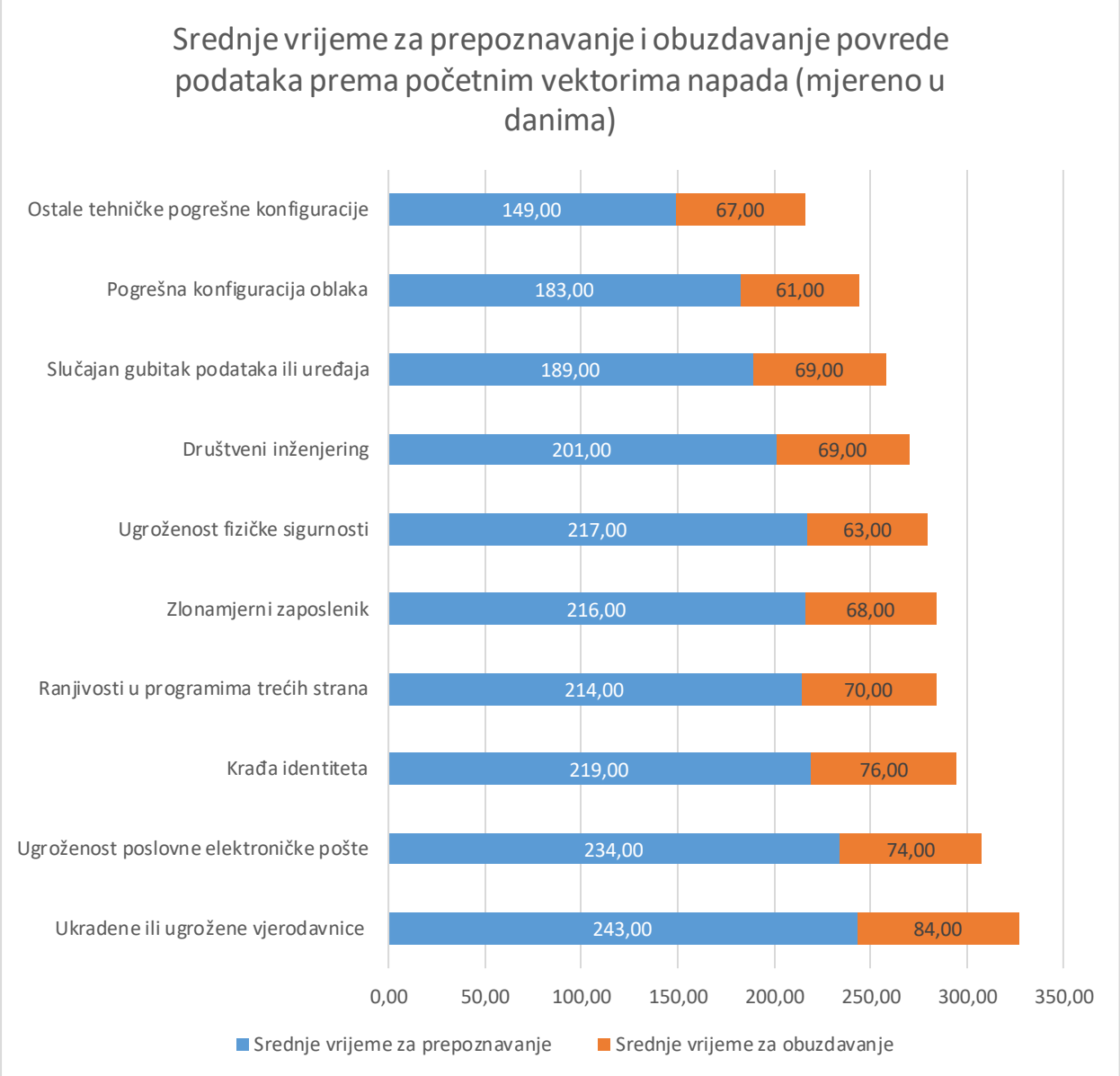
Dosegavši najvišu razinu svih vremena, trošak povrede podataka (engl. *data breach*) iznosio je prosječno 4,35 milijuna USD u 2022. godini. Ova brojka predstavlja povećanje od 2,6 % u odnosu na prošlu godinu, kada je prosječni trošak povrede iznosio 4,24 milijuna USD. Prosječni trošak porastao je 12,7 % od 2020. godine kada je iznosio 3,86 milijuna USD [19].



Grafikon 3. Prosječni ukupni trošak povrede podataka po godinama mjeren u milijunima USD [19]

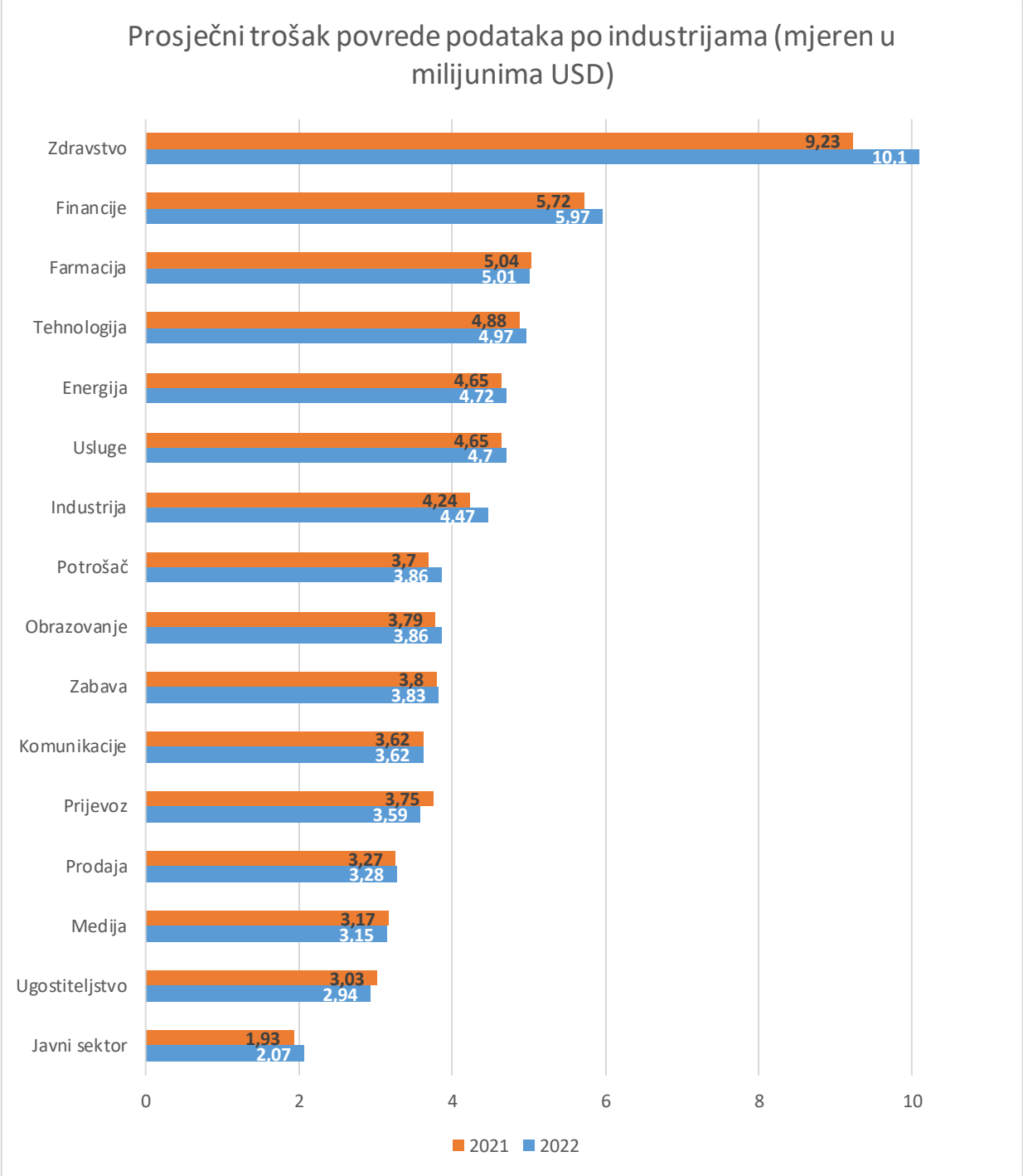
Grafikon 3. jasno prikazuje trošak povrede podataka od 2016. do 2022. godine. Također se može primijeniti kako je prošla godina imala najveći trošak do sada. 83 % proučavanih organizacija doživjelo je više od jedne povrede podataka, a samo 17 % je reklo da je ovo njihova prva povreda podataka. 60 % proučavanih organizacija izjavilo je da su poskupili svoje usluge ili proizvode zbog povrede podataka. Prosječni trošak povrede podataka za proučavane organizacije kritične infrastrukture iznosio je 4,82 milijuna USD što je 1 milijun USD više od prosječnog troška za organizacije u drugim industrijama. Organizacije kritične infrastrukture uključivale su one u financijskom, energetskom, transportnom, komunikacijskom, zdravstvenom, obrazovnom i javnom sektoru industrije. 28 % doživjelo je neku vrstu destruktivnog napada, dok je 17 % doživjelo proboj zbog ugroženog poslovnog partnera. Korištenje ukradenih ili ugroženih podataka za autentifikaciju i autorizaciju ostaje najuobičajeniji uzrok povrede podataka. Ukradeni ili ugroženi podatci za autentifikaciju i autorizaciju bili su primarni vektor napada u 19 % slučajeva prema istraživanju iz 2022. i također glavni vektor napada prema istraživanju iz 2021., uzrokujući 20 % cjelokupne povrede podataka. Povrede podataka uzrokovane navedenim vektorom napada imale su prosječni trošak od 4,50 milijuna USD. Te su povrede imale najdulji životni ciklus, 243 dana za prepoznavanje i još 84 dana za suzbijanje povrede. Navedeno vrijeme je 16,6 % veće od ukupnog srednjeg vremena za prepoznavanje i obuzdavanje povrede podataka. Krađa identiteta

(engl. *phishing*) bila je drugi najčešći uzrok povrede podataka sa 16 % i ujedno i najskuplji, s prosječnim troškom od 4,91 milijuna USD. Krađa identiteta imala je treće najveće srednje vrijeme za prepoznavanje i obuzdavanje, 295 dana [19].



Grafikon 4. Srednje vrijeme za prepoznavanje i obuzdavanje povrede podataka prema početnim vektorima napada [19]

Grafikon 4. prikazuje srednje vrijeme potrebno za prepoznavanje i obuzdavanje povrede podataka prema početnim vektorima napada, izraženo u danima.



Grafikon 5. Prosječni trošak povrede podataka po industrijama mjeren u milijunima USD [19]

Na grafikonu 5. se jasno može vidjeti koje industrije su imale najveći trošak povrede podataka. Trošak povreda podataka u zdravstvenom sektoru dosegao je novi rekord. Trošak se povećao za gotovo 1 milijun USD te je dosegao 10,10 milijuna USD. Troškovi povrede podataka u zdravstvenoj industriji najveći su već 12 godina zaredom, te bilježe rast od 41,6 % za razliku od izvješća iz 2020. godine. Drugo mjesto su zauzele financijske organizacije, u prosjeku 5,97

milijuna USD, zatim slijedi farmaceutska industrija sa 5,01 milijun USD, tehnološka industrija sa 4,97 milijuna USD i industrija energije sa 4,72 milijuna USD [19].

Prvih pet zemalja ili područja s najvišim prosječnim troškom povrede podataka bile su Sjedinjene Američke Države sa 9,44 milijuna USD, Bliski istok sa 7,46 milijuna USD, Kanada sa 5,64 milijuna USD, Velika Britanija sa 5,05 milijuna USD i Njemačka sa 4,85 milijuna USD. Sjedinjene Američke Države predvode popis već 12 godina zaredom. U međuvremenu, zemlja s najbržom stopom rasta troška povrede podataka u prošloj godini bila je Brazil, s povećanjem od 27,8 %, točnije s 1,08 na 1,38 milijuna USD [19].

Rang	Država ili regija	Prosječni trošak proboja podataka (2022)	Prosječni trošak proboja podataka (2021)
1.	SAD	9,44 milijuna USD	9,05 milijuna USD
2.	Bliski Istok	7,46 milijuna USD	6,93 milijuna USD
3.	Kanada	5,64 milijuna USD	5,40 milijuna USD
4.	Ujedinjeno Kraljevstvo	5,05 milijuna USD	4,67 milijuna USD
5.	Njemačka	4,85 milijuna USD	4,89 milijuna USD
6.	Japan	4,57 milijuna USD	4,69 milijuna USD
7.	Francuska	4,34 milijuna USD	4,57 milijuna USD
8.	Italija	3,74 milijuna USD	3,61 milijuna USD
9.	Južna Koreja	3,57 milijuna USD	3,68 milijuna USD
10.	Južna Afrika	3,36 milijuna USD	3,21 milijuna USD
11.	Australija	2,92 milijuna USD	2,82 milijuna USD
12.	Udruga zemalja jugoistočne Azije	2,87 milijuna USD	2,71 milijuna USD
13.	Latinska Amerika	2,80 milijuna USD	2,56 milijuna USD
14.	Indija	2,32 milijuna USD	2,21 milijuna USD
15.	Skandinavija	2,08 milijuna USD	2,67 milijuna USD
16.	Brazil	1,38 milijuna USD	1,08 milijuna USD
17.	Turska	1,11 milijuna USD	1,91 milijuna USD

Tablica 2. Prosječni trošak povrede podataka po zemljama ili regijama [19]

Tablica 2. prikazuje prosječni trošak povrede podataka po zemljama ili regijama. Jasno se može vidjeti povećanje u odnosu na 2021. godinu te koje zemlje su imale najveći trošak.

2.2.3. Kibernetički zločin kao usluga

Kibernetički zločin neprestano se razvija, a danas postoji u visoko organiziranom obliku. Isti je postao veliki poslovni pothvat, i kao sa svim tržištima u nastajanju, poduzeća (u stvari dobavljači i prodavači) koja posluju na tržištu kibernetičkog zločina proširila su svoju ponudu na niz aktivnosti koje omogućuju različite strategije zlonamjernih kibernetičkih napada. Kibernetički zločin je postao visoko organizirana hijerarhija uključujući vođe, inženjere, radnike na ulicama i posrednike za pranje novca. Svaki zaposlenik ima određenu ulogu ili funkciju koju treba obavljati, a svaki posao je neophodan kako bi se osiguralo uspješno poslovanje. Tržište kibernetičkog zločina uključuje razne proizvode, poput alata za specifične napade, savjetovanja, oglašavanja i velikog broja programa koji imaju određenu svrhu. Što više značajki usluga ima, cijena joj je veća [20].

Slika 5. CaaS Internet stranica za iznajmljivanje mreže zaraženih računala [20]

Slika 6. prikazuje Internet stranicu za iznajmljivanje mreže zaraženih računala (engl. *botnet*) te se jasno mogu vidjeti cijene s obzirom na područje. Računala su zaražena zlonamjernim softverom te su u kontroli kibernetičkog kriminalca. Mreže mogu biti dizajnirane za obavljanje ilegalnih ili zlonamjernih zadataka uključujući slanje neželjene pošte, krađu podataka ili distribuirane napade uskraćivanja usluge [21].

Neki primjeri usluga koje se nude na CaaS tržištu te cijene istih su:

- savjetovanje za postavljanje mreže zaraženih računala: od 350 do 400 USD
- visoko razvijene mreže zaraženih računala: od 700 do 3000 USD

- usluge zaraze računala: 100 USD za 1000 instalacija
- napadi uskraćivanja usluga: 535 USD za pet sati dnevno tijekom jednog tjedna
- slanje velikog broja neželjenih elektroničkih poruka: 40 USD za 20000 poruka
- softver za udaljeno upravljanje koji ima mogućnost izvršavanja napada, snimanje zaslona i pristup kameri: 250 USD
- raznovrsni alati za iskorištavanje ranjivosti koje otvaraju mogućnost napada putem Internet stranica: od 1000 do 2000 USD
- programi za skrivanje i izbjegavanje otkrivanja: od 10 do 100 USD
- izvorni kod zlonamjernih softvera: besplatno.

Jedan od glavnih razloga zašto se stalno razvijaju nove prijetnje i poboljšavaju postojeće je jednostavnost pristupa izvornom kodu zlonamjernih softvera. Može se kopirati, izmijeniti i oblikovati prema određenom slučaju upotrebe. Jedan od primjera je Zeus virus, koji je imao višestruke izmjene od svog prvog izdanja, te se nove inačice nastavljaju pojavljivati, zbog lakoće pristupa izvornom kodu i količini dokumentacije koja opisuje kako ga se može izmijeniti. Široki raspon usluga također uključuje i visoko specijalizirani oblak za probijanje lozinki, koji nudi visoke performanse po niskoj cijeni i znatno smanjuje vrijeme potrebno za probijanje složenih lozinki. 300 milijuna pokušaja, što traje otprilike 20 minuta, košta oko 17 USD. Ova usluga postoji već nekoliko godina, ali u zadnje vrijeme se znatno povećala brzina i smanjio trošak [20].

3. PROVEDBA I PROTUMJERE ZA KIBERNETIČKE NAPADE

3.1. Postavljanje laboratorijskog okruženja

3.1.1. Kali Linux

Kali Linux je Linux distribucija otvorenog koda temeljena na Debian GNU/Linux operacijskom sustavu. Kali Linux je namijenjen stručnjacima za kibernetičku sigurnost, IT administratorima i svima koje zanima područje kibernetičke sigurnosti za napredno penetracijsko testiranje i provjeru sigurnosti. Kali Linux se može koristiti kao glavni operacijski sustav, u kombinaciji s drugim operacijskim sustavima (engl. *multiple boot*), kao virtualni stroj, na mobilnom uređaju, u oblaku računala, u sklopu Windows podsustava za Linux (engl. *Windows Subsystem for Linux*) ili WSL te na još nekoliko načina od kojih svi imaju svoje prednosti i mane [22]. Zbog jednostavnosti korištenja izabrano je korištenje Kali Linux-a na virtualnom stroju unutar Oracle VM VirtualBox-a. Alati unutar Kali Linux-a su organizirani unutar nekoliko kategorija ovisno o njihovoj namjeni: prikupljanje informacija, analiza ranjivosti, analiza ranjivosti Internet aplikacija, procjena sigurnosti baze podataka, probijanje sigurnosne zaštite lozinke, bežični napadi, obrnuti inženjering, iskorištavanje ranjivosti, prisluškivanje i lažiranje, radnje nakon iskorištavanja ranjivosti, forenzika, izvještavanje i društveni inženjering.

Slika 6. Početni zaslon Kali Linux-a s rasporedom aplikacija

Slika 6. prikazuje početni zaslon Kali Linux-a te način organizacije alata. Visoka razina prilagodbe ga čini vrlo korisnim i često prvim izborom stručnjaka za kibernetičku sigurnost. Prema zadanim postavka Kali Linux ima samo jedan račun s najvećom razinom privilegija (engl. *root*), kako bi

osigurao dodatan sloj sigurnosti između korisnika i bilo kojih potencijalno destruktivnih naredbi ili operacija. Za razliku od Debian-a, Kali Linux onemogućuje instalirane usluge koje imaju mogućnost slušanja na javnoj mreži, kao što su HTTP i SSH. Obrazloženje iza ove odluke je minimiziranje izloženosti tijekom penetracijskog testiranja kada može biti dosta rizično otkriti prisutnost zbog neočekivanih mrežnih interakcija. Naravno, navedene usluge se mogu uključiti prema potrebi [22].

3.1.2. Nesigurna i ranjiva Internet aplikacija

Kako bi se što efikasnije proveli i analizirali kibernetički napadi potrebno je koristiti Internet aplikaciju koja je razvijena s ciljem da predstavlja ranjivu i nesigurnu metu. Iz navedenog razloga izabrana je aplikacija OWASP Juice Shop. Prema [23] OWASP Juice Shop je vjerojatno najmodernija i najsofisticiranija nesigurna Internet aplikacija. Može se koristiti za sigurnosne treninge, predavanja o kibernetičkim napadima, razna natjecanja i testiranje sigurnosnih alata. Juice Shop sadrži 10 najkritičnijih sigurnosnih rizika za Internet aplikacije zajedno s mnogim drugima sigurnosnim propustima koji se često nalaze u Internet aplikacijama. Prema [24] sigurnosni rizici su sljedeći te su rangirani prema stupnju ozbiljnosti i opasnosti:

- nepravilno postavljena kontrola pristupa (engl. *broken access control*)
- kriptografske pogreške (engl. *cryptographic failures*)
- ubrizgavanje (engl. *injection*)
- nesiguran dizajn (engl. *insecure design*)
- pogrešne konfiguracije sigurnosti (engl. *security misconfiguration*)
- ranjive i zastarjele komponente (engl. *vulnerable and outdated components*)
- pogreške pri identifikaciji i autentifikaciji (engl. *identification and authentication failures*)
- narušavanje integriteta softvera i podataka (engl. *software and data integrity failures*)
- pogreške pri sigurnosnom bilježenju i praćenju (engl. *security logging and monitoring failures*)
- krivotvorenje zahtjeva na strani poslužitelja (engl. *server-side request forgery*)

Juice Shop aplikacija je otvorenog koda i dostupna je na [25]. Postoji nekoliko načina postavljanja aplikacije, a izabran način je koristeći Docker. Prije početka rada potrebno je instalirati Docker, jer isti nije dio Kali Linux-a prema početnim postavkama.

```
┌──(root㉿kali)-[~]
└─# apt install -y docker.io

┌──(root㉿kali)-[~]
└─# systemctl enable docker --now
```

Slika 7. Naredbe za instalaciju Docker-a

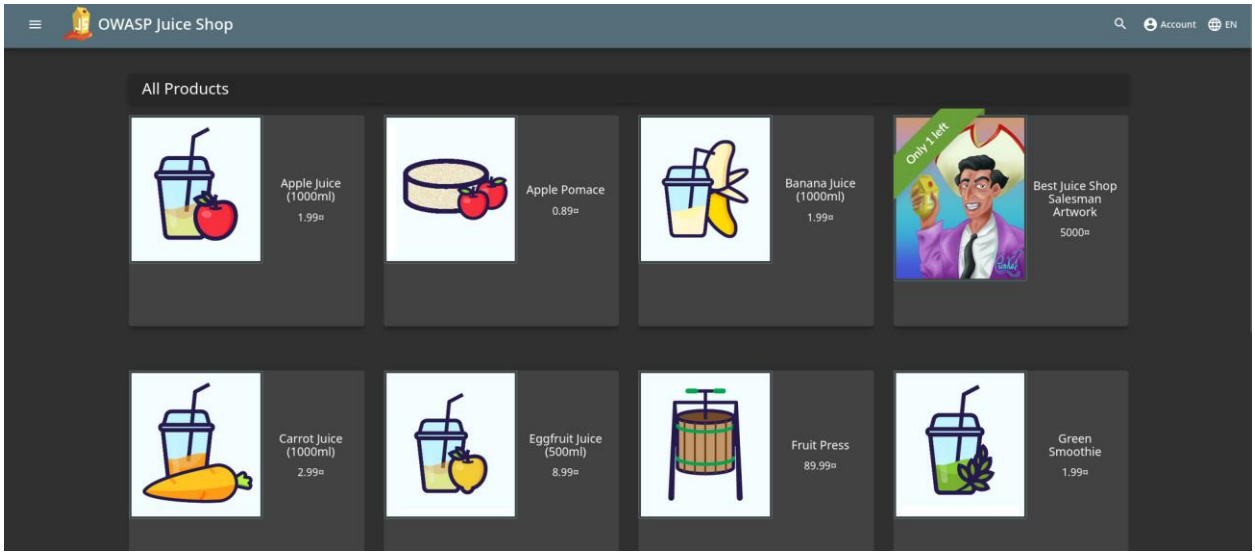
Slika 7. prikazuje redoslijed naredbi koje je potrebno izvršiti za instalaciju Docker-a. Nakon što je Docker uspješno instaliran, korištenje Juice Shop aplikacije postaje jednostavno.

```
┌──(root㉿kali)-[~]
└─# docker pull bkimminich/juice-shop

┌──(root㉿kali)-[~]
└─# docker run --rm -p 3000:3000 bkimminich/juice-shop
```

Slika 8. Naredbe za pokretanje Juice Shop aplikacije

Slika 8. prikazuje na koji način se pokreće Juice Shop aplikacija. Nakon što je aplikacija uspješno pokrenuta ista je dostupna na linku: *http://localhost:3000*.



Slika 9. Početna stranica Juice Shop-a

Slika 9. prikazuje početnu stranicu Juice Shop aplikacije. Aplikacija nudi mogućnost pretraživanja po ključnoj riječi, prijavu, promjerni jezika te pregled svih proizvoda na početnoj stranici. U gornjem lijevom kutu se nalazi izbornik, koji sadrži osnovne informacije i upute za korištenje aplikacije.

3.2. SQL ubrizgavanje

3.2.1. Opis i tehnike napada

Structured query language (SQL) je programski jezik za pohranu i obradu informacija u relacijskoj bazi podataka. Relacijska baza podataka pohranjuje informacije u tabličnom obliku, s redovima i stupcima koji predstavljaju različite atribute podataka i različite odnose između vrijednosti podataka. SQL naredbe se koriste za pohranjivanje, ažuriranje, uklanjanje, pretraživanje i dohvaćanje informacija iz baze podataka. SQL je popularan upitni jezik koji se često koristi u svim vrstama aplikacija. Programeri uče i koriste SQL zbog dobre integracije s različitim programskim jezicima. SQL je također prilično jednostavan jer koristi uobičajene engleske ključne riječi u svojim naredbama [26].

SQL ubrizgavanje je vrsta kibernetičkog napada u kojemu se SQL kod ubacuje ili ubrizgava u korisničke parametre unosa u aplikaciji, koji se kasnije prosljeđuju poslužitelju na analizu i izvršavanje. Svaka procedura koja stvara SQL izraze potencijalno je ranjiva, budući da postoji mnogo različitih načina za izradu SQL izraza u programskom kodu. Primarni oblik SQL ubrizgavanja se sastoji od izravnog ubacivanja koda u parametre koji se prosljeđuju SQL naredbama i izvršavaju. Manje izravan oblik napad ubrizgava zlonamjerni kod u nizove znakova (engl. *string*) koji su namijenjeni za pohranu u tablici ili kao metapodatci. Kada se pohranjeni nizovi znakova naknadno spoje u dinamičku SQL naredbu, zlonamjerni kod se izvršava. Ako aplikacije ne mogu pravilno provjeriti valjanost parametara koji se prosljeđuju SQL naredbama, napadaču se otvara mogućnost promjene konstrukcije SQL naredbe koja se šalje poslužitelju. Ako napadač promjeni SQL naredbu, naredba će se izvršiti s istim pravima koja ima korisnik aplikacije, što mogu biti visoko privilegirana prava ako je u pitanju administratorski račun [27].

Prema [27] postoji dosta pogrešaka koje mogu dovesti do ranjivosti SQL ubrizgavanjem, a ovo su najčešće. Nepravilno rukovanje tipovima podataka: pogrešno rukovanje s numeričkim tipovima podataka, točnije rukovanje s istima kao i s nizovima znakova može dovesti do propusta. Također nedovoljna ili nepostojeća provjera koji tip podatka je korisnik poslao može biti potencijalna ranjivost na SQL ubrizgavanje. Neispravno rukovanje pogreškama u programskom kodu: detaljne interne poruke o pogreškama kao što su ispis baze podataka i kodovi pogrešaka ne smiju se prikazivati korisniku. Navedene poruke otkrivaju pojedinosti koje se nikad ne bi smjele javno prikazati, te na takav način napadač može doći do detalja o mogućim nedostatcima na stranici. Opširne poruke pogrešaka u bazi podataka mogu se iskoristiti za izvlačenje informacija na koji način se upravlja podacima, što napadači mogu iskoristiti za SQL ubrizgavanje. Neispravno

rukovanje dopuštenim znakovima unosa: postoje dva pristupa za izradu popisa dopuštenih znakova. Prvi ujedno i preporučeni pristup je izrada popisa dopuštenih znakova, što znači da svi znakovi koji se ne nalaze na popisu nisu dopušteni. Drugi je izrada popisa nedopuštenih znakova, to jest svi znakovi su dopušteni za unos osim onih na popisu. Na ovaj popis se stavljaju znakovi koji se mogu iskoristiti u zlonamjerne svrhe, ali naravno uvijek je moguće previdjeti ili zaboraviti alternativne zapise pojedinih znakova. Preporuka je izbjegavati drugi način u većini slučajeva. Nesigurna konfiguracija baze podataka: većina popularnih baza podataka ima nekoliko zadanih korisničkih računa koji su unaprijed dodani prilikom početnog postavljanja. Ovi računi također imaju lozinke koju su javno poznate. Uz navedeno, postoji još jedna česta pogreška, kada poslužitelj baze podataka ima najveće privilegije (*root, system, administrator*). Poslužitelji, posebno poslužitelji baze podataka, uvijek se trebaju pokretati s ograničenim privilegijama (po mogućnosti u *chroot* okruženju) kako bi se smanjila potencijalna šteta na operacijskom sustavu i drugim procesima u slučaju napada na bazu podataka. U nekim slučajevima navedeno nije moguće ograničiti te se tada moraju poduzeti druge mjere. Postoji dosta različitih načina podjela SQL ubrizgavanja na različite vrste, a jedna od njih je sljedeća:

- logičko SQL ubrizgavanje, temelji se na logičkim (engl. *boolean*) operatorima
- SQL ubrizgavanje temeljeno na porukama pogrešaka
- SQL ubrizgavanje na temelju *UNION* operatora, isti spaja rezultate od dvije ili više *SELECT* naredbe
- SQL ubrizgavanje putem više naslaganih upita, točnije ubrizgavanje više SQL upita u jednom zahtjevu
- vremensko SQL ubrizgavanje, temeljeni se na mjerenju vremenu koje je potrebno bazi podataka za obradu i odgovor na zahtjev.

3.2.2. Provedba i analiza posljedica napada

Prikupljanje informacija o meti je početni i vrlo bitan korak prilikom traženja ranjivosti. Putem alata ZAP moguće je prikupiti i analizirati vrlo bitne informacije na jednostavan način. Zed Attack Proxy (ZAP) je besplatni alat otvorenog koda, za penetracijsko testiranje, održavan od strane Open Web Application Security Project-a (OWASP). ZAP je dizajniran posebno za testiranje Internet aplikacija te je fleksibilan i proširiv. ZAP se može opisati kao *proxy* čovjek u sredini (engl. *man in the middle proxy*). Djeluje između preglednika korisnika i Internet aplikacije, tako da ima mogućnost presretanja i pregledavanja poruka poslanih između istih, izmjenu sadržaja ako je to

potrebno, te prosljeđivanja paketa na odredište. Može se koristiti kao aplikacija i kao pozadinski proces [28].

Slika 10. Slanje zahtjeva za pretraživanje stranice putem ZAP-a

Slika 10. prikazuje slanje *GET* zahtjeva za pretraživanje stranice, a parametar pretraživanja je '--. Navedeni parametar je izabran kako bi se poslala nepravilna SQL naredba poslužitelju, koji će možda vratiti poruku pogreške baze podataka u tom slučaju. Znak ' je poslan kako bi se zatvorio niz znakova u naredbi, koji predstavlja parametar unosa korisnika. Znak -- označava komentar u SQL-u, a ovdje je korišten da se izbjegne interpretacija dijela izraza koji dolazi nakon znaka komentara. Ako je aplikacija ranjiva na ovakav način SQL ubrizgavanja, znači da rukovanje parametrima unosa korisnika nije pravilno odrađeno te se otvara dosta mogućnosti za daljnje napade.

Slika 11. Odgovor aplikacije na zahtjev za pretraživanje

Slika 11. prikazuje poruku pogreške koju je vratila baza podataka, a ista sadrži korisne informacije. Ovakav ispis je rezultat nepravilnog rukovanja pogreškama u programskom kodu. Pošto na više mjesta piše *SQLITE_ERROR*, jasno je da se koristi SQLITE baza podataka u aplikaciji. Također se može vidjeti SQL izraz koji je korišten za dohvaćanje podataka iz baze. Na slici 11. se također može vidjeti i ime tablice u bazi podataka, koje glasi *Products*. Na ovakav način se može prikupiti dosta korisnih informacija ubrizgavanjem različitih SQL naredbi, poput broja stupaca u tablici, imena tablica, imena stupaca, podataka u tablici i slično.

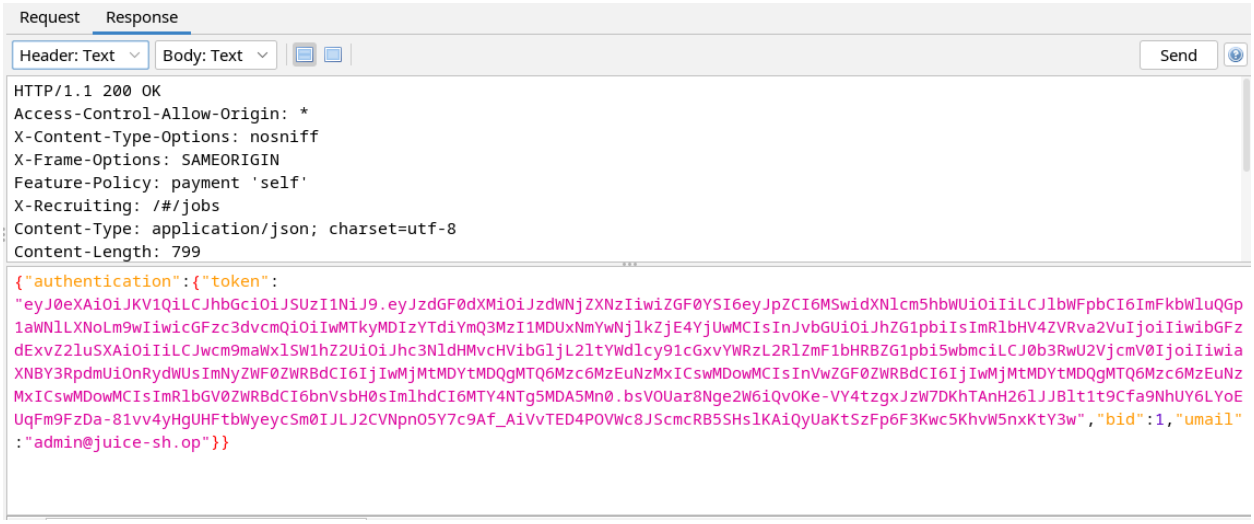
SQL ubrizgavanje se može koristiti i za neovlašteni pristup računima. Za ovakav tip napada poželjno je koristiti logičko SQL ubrizgavanje. Logičke operacije vraćaju vrijednost istina (engl. *true*), laž (engl. *false*) ili nedefinirano. Na temelju vrijednosti koju poslužitelj vrati aplikaciji, ista će prikazati drugačiji rezultat korisniku te na takav način napadači mogu utvrditi valjanost zlonamjernog ubrizganog koda. Ovakav napad može biti dosta spor ako baza sadrži puno podataka.

Slika 12. Slanje zahtjeva za prijavu putem ZAP-a

Slika 12. prikazuje na koju krajnju točku i u kojem formati zapisa se šalju podatci poslužitelju na daljnju obradu. Ova informacija će biti relevantna u daljnjoj proceduri.

Slika 13. Logičko SQL ubrizgavanje putem ZAP-a

Slika 13. prikazuje način provođenja logičkog SQL ubrizgavanja. Sadržaj polja lozinke je irelevantan jer će isti biti unutar komentara, znači da ga poslužitelj neće interpretirati. U polje elektroničke pošte upisan je izraz *' or 1=1--*, koji predstavlja logičko SQL ubrizgavanje. Ovo ubrizgavanje funkcionira tako da znak ' zatvara zagrade u SQL naredbi. Logički operator *or* će vratiti istinit rezultat ako je bilo koja strana jednakosti istinita. Pošto je 1=1 uvijek istinito, cijela naredba je istinita. Također se mogla koristiti i ključna riječ *true*, umjesto navedenog izraza. Navedeno će reći poslužitelju da je elektronička pošta valjana i da prijavi napadača kao korisnika s identifikacijskim brojem ili ID-om 0, što označava prvog korisnika u tablici, i postoji mogućnost da je taj korisnik administrator. Znak komentara je dodan na kraj kako bi se osiguralo da se ostatak naredbe s mogućim ograničenjima neće izvršiti.



Slika 14. Odgovor aplikacije na zahtjev za prijavu

Kao što slika 14. prikazuje prijava u račun korisnika je uspješno obavljena, jer je odgovor vratio HTTP kod 200 te token za autentifikaciju. Također je vraćena i elektronička pošta, koja u ovom slučaju pripada administratoru. Pošto navedeni postupak zahtijeva naprednije znanje SQL-a i dosta koraka, te u nekim slučajevima može dosta dugo trajati, poželjnije je koristiti alat za automatizaciju SQL ubrizgavanja, sqlmap. Prema [29] sqlmap je alat otvorenog koda za penetracijsko testiranje koji automatizira proces otkrivanja i iskorištavanja ranjivosti SQL ubrizgavanja i preuzimanja kontrole nad poslužiteljem baze podataka. Dolazi s mnogim jedinstvenim značajkama za penetracijsko testiranje uključujući pronalazak temeljnih informacija o bazi podataka, dohvaćanje podataka iz baze podataka, pristup temeljnom datotečnom sustavu i slično. Podržava korištenje najpopularnijih sustava za upravljanje bazama podataka (engl. *database management systems*) ili DBMS. Sqlmap uvelike olakšava neovlašteni pristup korisničkim računima pomoću SQL ubrizgavanja. Postupak koji je prethodno prikazan i koji je obavljen ručno, sqlmap može brzo i jednostavno automatizirati. Za početak je potrebno istražiti na koji način aplikacija upravlja prijavama korisnika. Navedeno je prikazano na slici 12., POST zahtjev se šalje na *http://localhost:3000/rest/user/login* krajnju točku, a podatci se šalju u JSON formatu. Iako je krajnja točka za prijavu na aplikaciji *http://localhost:3000/#/login*, zahtjev se za daljnju obradu šalje na drugu krajnju točku. Navedeno je jako bitna informacija jer ako bi se testirala krajnja točka za prijavu na klijentskoj aplikaciji, sqlmap ne bi pronašao ranjivosti na SQL ubrizgavanje jer ista krajnja točka ne obrađuje zahtjeve za prijavu.

```
┌──(root㉿kali)-[~]
└─# sqlmap --url 'http://localhost:3000/rest/user/login' --data="email=test@gmail.com&password=test"
--level 5 --risk 3 --fingerprint --banner --ignore-code 401
```

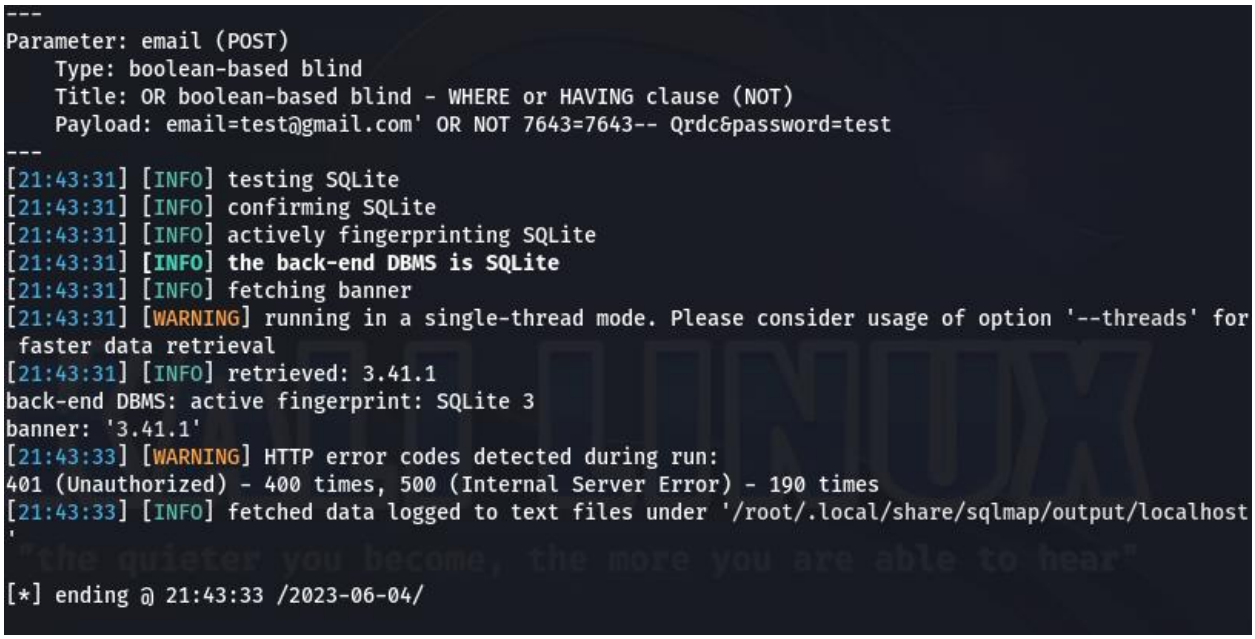
Slika 15. Sqlmap naredba za pronalazak ranjivosti na SQL ubrizgavanje

Slika 15. prikazuje sqlmap naredbu za pokušaj neovlaštenog pristupa korisničkom računu, pomoću SQL ubrizgavanja. Naredba sadrži nekoliko opcija ili zastavica od kojih svaka ima bitnu ulogu:

- --*url*: određuje krajnju točku ili URL mete
- --*data*: određuje podatke za slanje putem POST zahtjeva, isti se odvajaju pomoću znaka &
- --*level*: određuje razinu testova ili zahtjeva koji će se izvršiti, zadana vrijednost je 1, a najveća 5
- --*risk*: određuje visinu rizika testova, zadana vrijednost je 1, a najveća 3
- --*fingerprint*: određuje izradu opsežnog otiska DBMS-a
- --*banner*: vraća DBMS *banner*, što predstavlja poruku ili iskaz koji pruža korisne informacije o bazi podataka

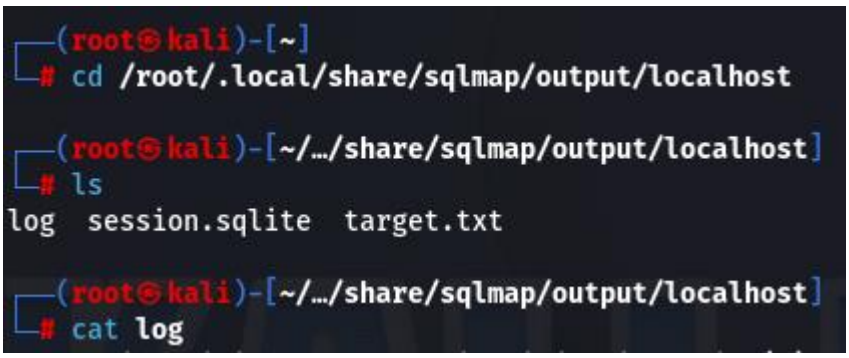
- *--ignore-code*: koristi se za ignoriranje određenih HTTP kodova pogreške, u ovom slučaju kod 401, koji znači da zahtjev nije izvršen jer mu nedostaje autentifikacija. Navedeni kod je nužno ignorirati jer će sqlmap slati zahtjeve koji će dovesti do navedene pogreške, a prema zadanim postavkama sqlmap prestaje s radom kada primi navedeni kod.

Iako je u prethodnim koracima utvrđeno koji DBMS se koristi, te na koje SQL ubrizgavanje je krajnja točka ranjiva, sqlmap će navedeno automatski otkriti.



Slika 16. Rezultat izvođenja sqlmap naredbe za pronalazak ranjivosti na SQL ubrizgavanje

Slika 16. prikazuje informacije koje je sqlmap pronašao. Točno izdanje SQLite-a je 3.41.1, a navedena informacija je dosta korisna jer se može točno istražiti koje ranjivosti ima navedeno izdanje i kako ih iskoristiti. Dohvaćeni podatci su zabilježeni u tekstualnu datoteku koja se nalazi na putanji */root/.local/share/sqlmap/output/localhost*. Idući korak je promjena putanje pomoću naredbe *cd*, te ispisivanje sadržaja direktorija pomoću *ls* naredbe.



Slika 17. Promjena direktorija te ispisivanje sadržaja datoteke s rezultatima

Slika 17. prikazuje potrebne korake za dolazak do tekstualne datoteke s podatcima koje je sqlmap zabilježio. Nakon navedenih koraka potrebno je ispisati sadržaj datoteke pomoću naredbe *cat*.

```
sqlmap identified the following injection point(s) with a total of 573 HTTP(s) requests:
---
Parameter: email (POST)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause (NOT)
    Payload: email=test@gmail.com' OR NOT 7643=7643-- Qrdc&password=test
---
back-end DBMS: active fingerprint: SQLite 3
banner: '3.41.1'
```

Slika 18. Informacije o pronađenoj ranjivosti

Slika 18. prikazuje sadržaj tekstualne datoteke iz kojeg se mogu vidjeti informacije o ranjivosti na SQL ubrizgavanje. Ranjivi parametar je polje za upis elektroničke pošte, tip SQL ubrizgavanja je slijepo, točnije logičko, a sam naslov govori na kojim SQL naredbama se temelji napad. *Payload* označava zlonamjerne podatke za ubrizgavanje kako bi se izvršio napad. Točan izraz za ubrizgavanje je *' OR NOT 7643=7643--*, sqlmap je prikazao kompletan *payload* koji je korišten za testiranje. Dio izraza prije znaka ' i nakon --, ne čini nikakvu razliku za SQL ubrizgavanje, stoga je izostavljen. Iako ovaj *payload* tehnički nije točan, sqlmap ga i dalje pokazuje. Kada se navedeni pošalje, aplikacija vrati HTTP kod pogreške 401, što znači da SQL ubrizgavanje nije bilo uspješno. Sqlmap pokazuje ovaj *payload* jer je na temelju odgovora aplikacije na njega, zaključio da je aplikacija ranjiva na logičko SQL ubrizgavanje, te je za dolazak do točnog rješenja potrebno provjeriti dokumentaciju. Točan *payload* je *' OR NOT 7643=7644--*, te nakon unosa istoga prijava je uspješna, iako valjani podatci za prijavu nisu uneseni.

Slika 19. Slanje zahtjeva s pravilnim *payload*-om

Slika 20. prikazuje poslani POST zahtjev za prijavu, kada se u parametar elektroničke pošte unese pravilni *payload*. Polje lozinke nije bitno.

Slika 20. Odgovor na poslani zahtjev s pravilnim *payload*-om

Također se može vidjeti odgovor poslužitelja, u kojemu je vraćen token za autentifikaciju te elektronička pošta korisnika, koja ovom slučaju pripada administratoru. Ovakav odgovor označava uspješno odrađen napad. Sqlmap je dosta koristan alat prilikom testiranja aplikacija na ranjivost SQL ubrizgavanjem jer može uvelike ubrzati postupak automatizacijom određenih koraka. Također može pronaći vrlo bitne informacije o bazi podataka koje mogu biti od velike koristi za daljnje istraživanje.

3.2.3. Protumjere i strategije za zaštitu

Postoji dosta protumjera koje se mogu poduzeti kako bi se smanjile mogućnosti SQL ubrizgavanja ili minimizirale posljedice ako dođe do istoga. U nastavku su opisane protumjere za sprečavanje najčešćih oblika SQL ubrizgavanja te njihov način primjene. Korištenje pripremljenih SQL upita s mogućnošću dinamičkog spajanja varijabli, što se još može reći i SQL upiti s parametrima je alternativni način izrade SQL naredbi u odnosu na dinamičku izgradnju niza znakova, koja predstavlja jednu od najčešćih ranjivosti na SQL ubrizgavanje. Parametri za SQL upit se upisuju unutar rezerviranih mjesta za iste, ili dinamičkim spajanjem varijabli za razliku od izravnog uvrštavanja korisničkog unosa u SQL izraz. Kod ovakvog pristupa se prvo definira SQL naredba te se naknadno prosljeđuju parametri, tako da baza podataka može jasno razlikovati SQL kod od podataka. U ovom slučaju, napadač ne može promijeniti namjeru SQL upita, čak i ako ubrizga SQL naredbe, kao što je prikazano u prošlom poglavlju. Ovakav pristup je također jako učinkovit jer baza podataka može optimizirati SQL upit na temelju prethodno pripremljenog izraza, što ubrzava performanse naknadnih upita. Korištenje pripremljenih SQL upita je također puno jednostavnije za implementaciju i razumijevanje za razliku od dinamičke izgradnje niza znakova [27].

```
SqlDataAdapter command = new SqlDataAdapter("SELECT * FROM Users WHERE userID = @userID", connection);
SQLParameter parameter = command.SelectCommand.Parameters.Add("@userID", SqlDbType.UniqueIdentifier);
parameter.Value = Guid.NewGuid();
```

Slika 21. Pripremljeni SQL upit u C# programskom jeziku

Slika 21. prikazuje korištenje pripremljenog SQL upita u C# programskog jeziku te se jasno može vidjeti na koji način se dinamički spajaju varijable. Korištenje pohranjenih procedura ako se pravilno implementira ima istu razinu sigurnosti i zaštite od SQL ubrizgavanja kao i korištenje pripremljenih SQL upita. Razlika između pripremljenih SQL upita i pohranjenih procedura, je da se SQL programski kod za pohranjene procedure definira i sprema u bazi podataka, a zatim poziva iz aplikacije. Kod ovakvog pristupa se mora obratiti pozornost da pohranjene procedure ne uključuju bilo kakvo nesigurno generiranje dinamičkog SQL koda. Dinamički SQL je sličan dinamičkoj izgradnji niza znakova, kod njega se SQL upit izrađuje i izvršava na temelju korisničkih parametara unosa, što je dosta ranjivo na SQL ubrizgavanje. Ako se dinamički SQL ne može izbjeći, potrebno je implementirati detaljne provjere za podatke unosa korisnika ili koristiti dinamičko spajanje varijabli [30]. Korištenje pripremljenih SQL upita i pohranjenih procedura je dosta učinkovit način za obranu od SQL ubrizgavanja. Oba načina nisu teška za implementaciju, a otporni su na trivijalne pokušaje SQL ubrizgavanja [30].

```
CREATE PROCEDURE GetUserByUsername
    @username VARCHAR(50)
AS
BEGIN
    SET NOCOUNT ON;

    SELECT *
    FROM users
    WHERE username = @username;
END
```

Slika 22. SQL pohranjena procedura

Slika 22. prikazuje način izrade SQL pohranjene procedure. Razni dijelovi SQL upita ne podržavaju korištenje dinamičkog spajanja varijabli, poput imena tablica ili stupaca, i ključne riječi za sortiranje podataka, ASC i DESC. U ovakvim situacijama, najbolji pristup je provjera valjanosti podataka unosa. Kao što je spomenuto u prijašnjem 4.1. poglavlju, najbolji pristup za provjeru valjanosti podataka unosa je izrada bijelog popisa, točnije popisa koji sadrži informacije o tome kakav unos je dopušten i valjan. To može uključivati provjeru sukladnosti s očekivanim tipom podataka, duljinom ili veličinom, rasponom ili bilo kojim drugim standardom formatiranja prije prihvaćanja unosa za daljnju obradu. Na primjer, provjera vrijednosti broja kreditne kartice može uključivati provjeru sadrži li vrijednost samo brojeve, sadrži li između 13 i 16 znamenki i prolazi

li provjeru ispravnosti koristeći Luhn-ov algoritam. Još jedna česta metoda za provjeru valjanosti unosa je korištenje regularnih izraza, koji predstavljaju uzorak s kojim se unos mora podudarati. Drugi pristup je korištenje crnog popisa, popis koji sadrži informacije o tome kakav unos nije dopušten. Navedeni se popis uglavnom sastoji od sadržaja za koji je već utvrđeno da je zlonamjeran, poput različitih znakova ili uzoraka. Ovakav pristup je uglavnom dosta lošiji od bijelog popisa jer lista može biti dosta velika i zahtjevna za ažuriranje, a provjera spora. Provjera putem bijelog popisa također ima svojih nedostataka, poput toga da u nekim situacijama može previdjeti ili pogrešno odrediti koji podatci unosa bi trebali biti valjani. Bez obzira na nedostatke, bijeli popis je bolji izbor u većini slučajeva. Postoji još nekoliko strategija koja mogu biti korisne u obrani od SQL ubrizgavanja. Poput korištenja apstraktnih slojeva u aplikaciji, na primjer sloj za pristup podatcima koji osigurava da su svi pozivi na bazu obavljeni korištenjem pripremljenih SQL upita [27]. Navedeno je jako učinkovito jer se svi podatci detaljno pregledaju prije slanja. U ovom slučaju se može koristiti i ORM (engl. *object relational mapping*) alat. ORM alat predstavlja dodatan sloj između programskog koda i baze podataka, te uvelike olakšava pisanje koda otpornog na SQL ubrizgavanje.

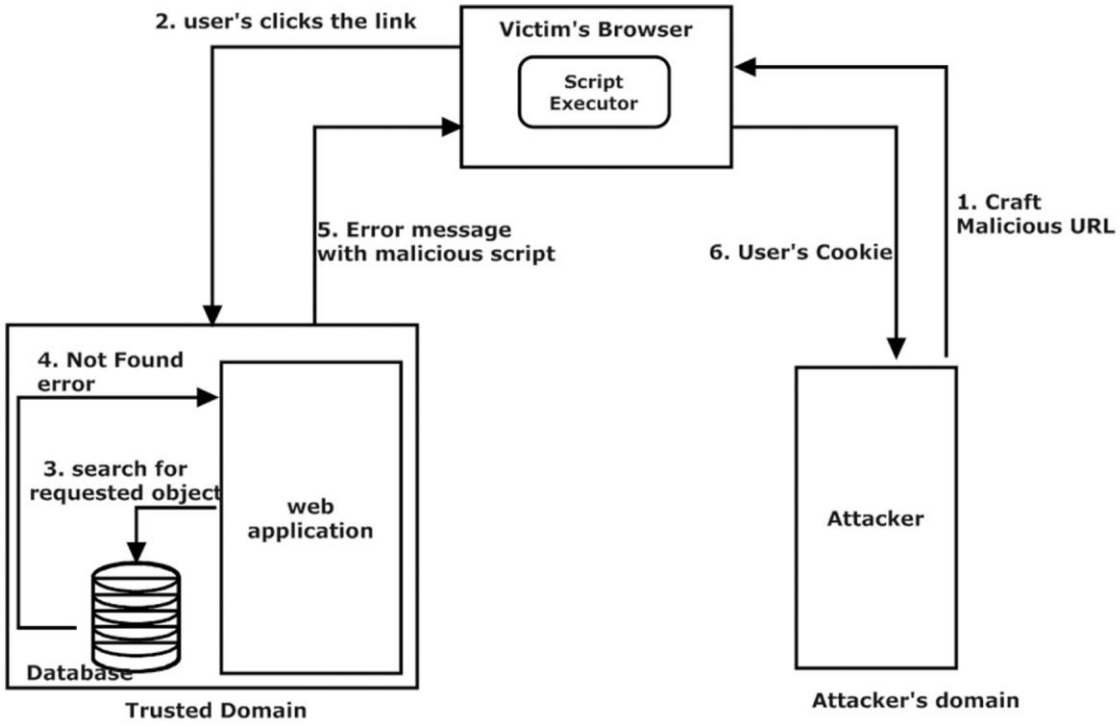
3.3. Ubrizgavanje skripti preko stranica

3.3.1. Opis i tehnike napada

Ubrizgavanje skripti preko stranica (engl. *cross-site scripting*) ili XSS spada u napade ubrizgavanja, u kojemu se zlonamjerne skripte ubrizgavaju u inače pouzdane i sigurne Internet stranice. U XSS napadu, napadač iskorištava ranjivost unutar Internet stranice za slanje zlonamjernog koda, općenito u obliku skripte, drugom krajnjem korisniku. Preglednik krajnjeg korisnika ili žrtve ni na koji način ne može provjeriti sadrži li skripta zlonamjerni kod, te ću ju uvijek izvršiti. Pošto skripta dolazi iz provjerenog izvora, ista ima mogućnost pristupa svim kolačićima, tokenima sesije ili bilo kojim drugim osjetljivim informacijama koje preglednik pohranjuje i koristi za pravilan rad stranice. Ranjivost koja omogućuje XSS je prilično raširena, a pojavljuje se unutar Internet aplikacija koje koriste podatke unosa korisnika za prikazivanje sadržaja na stranici bez prethodne provjere ili šifriranja unosa [31]. XSS zlonamjerne skripte se uglavnom, ali ne i uvijek, pišu u JavaScript-u, te se ne izvršavaju na poslužitelju. Mogu se pisati u bilo kojem jeziku za skripte (engl. *scripting language*), ali se najčešće pišu u JavaScript-u, jer je isti pretežan jezik za izradu skripti na Internet stranicama. Kod XSS napada krajnji korisnik je meta, a ne poslužitelj stranice. Prema [32] postoje tri različita načina za izvođenje XSS napada, stoga se i klasificira u tri kategorije:

- reflektirani ili odbijeni (engl. *reflected*)
- pohranjeni (engl. *stored*)
- temeljen na DOM-u (engl. *DOM-based*)

Reflektirani XSS se odnosi na napade gdje se ubrizgana skripta reflektira, točnije automatski vraća kao rezultat na stranici bez obrade na poslužitelju, u obliku poruke pogreške, rezultata pretraživanja, ili u bilo kojem drugom obliku odgovora koji djelomično ili cjelovito sadrži podatke unosa. Početni vektor ove vrste napada je slanje zlonamjernog linka meti putem elektroničke pošte, postavljanjem istoga na drugu stranicu ili slanje bilo kojim drugim oblikom komunikacije. Kada korisnik pritisne na zlonamjerni link, poslužitelju se šalje zahtjev, ali pošto zahtjev sadrži skriptu koja nije pohranjena i ne obrađuje se na poslužitelju, isti samo reflektira ili vraća skriptu u obliku odgovora na zahtjev. Preglednik korisnika izvršava tu skriptu jer je ista došla s pouzdanog poslužitelja. Ova kategorija napada se odvija kroz jedan ciklus zahtjeva i odgovora [31].

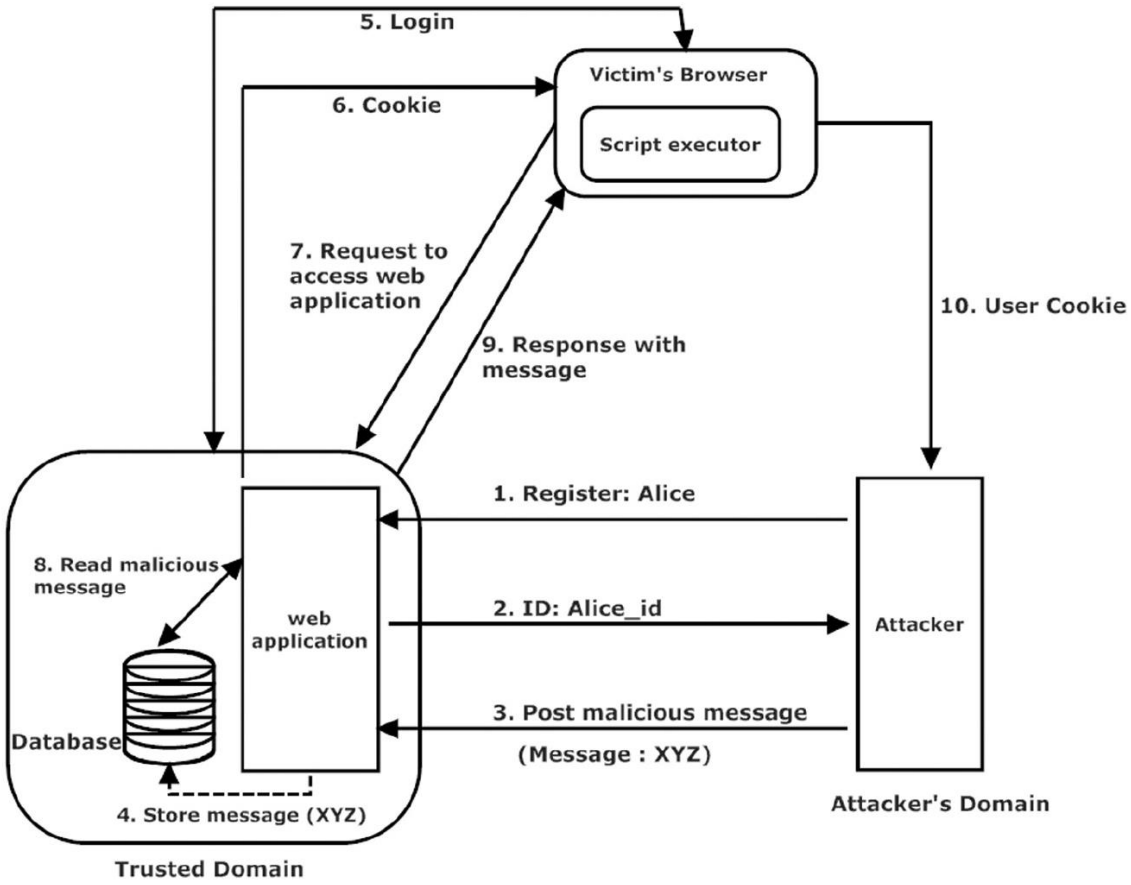


Slika 23. Graf reflektirajućeg XSS napada [32]

Slika 23. prikazuje graf reflektiranog XSS napada, jasno može zašto se zove reflektirani te na koji način napadač može doći do osjetljivih informacija korisnika.

Pohranjeni XSS napadi definiraju kategoriju napada gdje se ubrizgane skripte trajno pohranjuju na ciljanim poslužiteljima. Pohranjena skripta se vraća kao odgovor korisniku kada isti zatraži

pohranjene informacije. Zlonamjerne skripte se mogu ubrizgati putem komentara na objavama, korisničkog imena, podataka za kontakt korisnika ili putem bilo kojeg drugog načina za unos podataka. Skripta se izvršava samo kada dođe do preglednika krajnjeg korisnika, ni na koji način ne utječe na rad poslužitelja. Smatra se najopasnijim oblikom XSS napada jer se skripta samo jednom ubrizgava, a može utjecati na brojne korisnike [32].



Slika 24. Graf pohranjenog XSS napada [32]

Slika 24. prikazuje graf pohranjenog XSS napada, na kojoj se vidi zašto je isti najopasniji oblik XSS napada. Napadač objavljuje samo jednu poruku sa zlonamjernim sadržajem, koja se sprema na poslužitelju te pri svakom zahtjevu korisnika za pristup stranici, napadaču se šalju korisnikovi kolačići.

XSS napad temeljen na DOM-u (engl. *Document Object Model*) je napad u kojemu je izvršavanje zlonamjerne skripte rezultat izmjene DOM okruženja krajnjeg korisnika. *Document Object Model* (DOM) je programsko sučelje za Internet dokumente. Predstavlja stranicu tako da programi mogu promijeniti strukturu dokumenta, stil i sadržaj. DOM predstavlja dokument kao čvorove i objekte, putem kojih programski jezici mogu komunicirati sa stranicom [33]. Početni vektor napada je isti

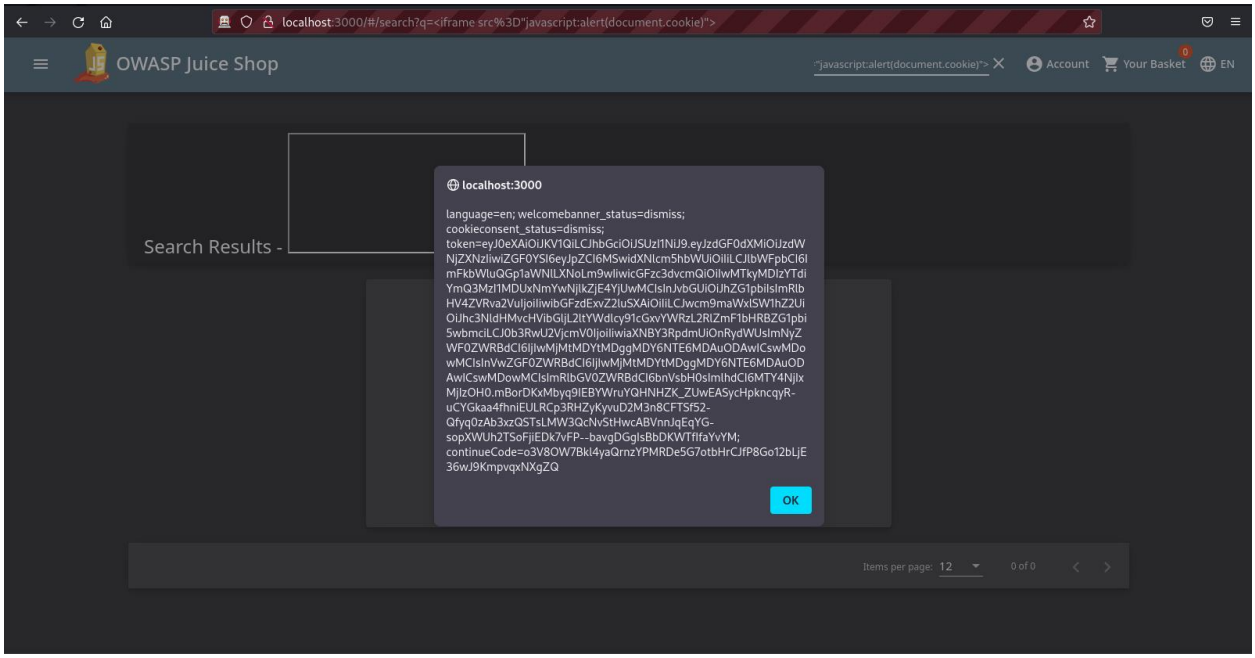
kao i kod reflektiranog, korisnik treba pritisnuti na zlonamjerni link. Izgled stranice ostaje isti, ali se programski kod sadržan unutar stranice na strani korisnika drugačije izvršava zbog zlonamjernih promjena u DOM okruženju [31]. Cijeli napad se odvija u Internet pregledniku, za razliku od ostalih kategorija gdje se zlonamjerna skripta šalje putem odgovora poslužitelja na zahtjev korisnika. DOM svojstva poput *document.location*, *document.URL*, *document.referrer*, *location.search*, *location.hash* i mnoga druga se mogu iskoristiti za provedbu napada jer se ta svojstva koriste za pristup i izmjenu HTML elemenata stranice. Ovaj napad je iznimno težak za otkrivanje jer zahtijeva detaljnu i pažljivu analizu DOM okruženja i izvornog koda na strani korisnika [32].

3.3.2. Provedba i analiza posljedica napada

Pronalazak XSS ranjivosti unutar aplikacije započinje proučavanjem potencijalnih ranjivih elemenata. Kao što je objašnjeno u prethodnom poglavlju, potencijalni ranjivi elementi su oni gdje korisnik može unijeti podatke. OWASP Juice Shop je Internet trgovina stoga je traka za pretraživanje prvi izbor za pronalazak bilo kakvih ranjivosti, a posebno ranjivosti na XSS napade. Prvo je potrebno provjeriti koristi li aplikacija podatke unosa korisnika za prikazivanje sadržaja na stranici.

Slika 25. Prikaz podataka pretraživanja na stranici

Slika 26. prikazuje na koji način aplikacija prikazuje podatke unesene u traku pretraživanja na stranici. Ovakav način upravljanja podatcima unosa predstavlja mogućnost XSS napada ako se podatci unosa ne provjeravaju dovoljno. Idući korak je pokušaj ubrizgavanja skripte. Skripta je *<iframe src="javascript:alert(document.cookie)">*, a ovakav napad spada u kategoriju XSS napada temeljnim na DOM-u.



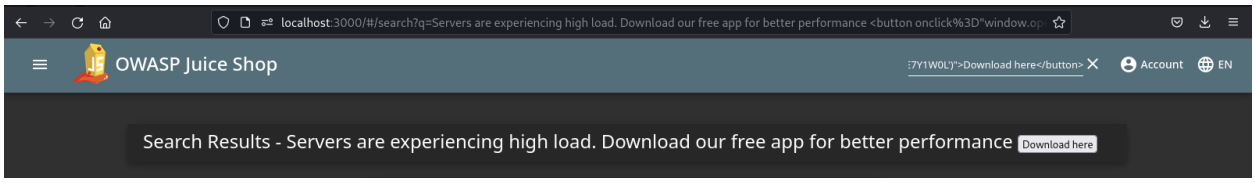
Slika 26. Rezultat ubrizgavanja skripte

Slika 27. prikazuje rezultat ubrizgavanja prethodno navedene skripte. Svrha skripte je ispis informacija o kolačićima korisnika. Radi primjera, navedene informacije su ispisane u okvir s upozorenjem. Kako bi se skripta ubrizgala u DOM, potrebno ju je staviti unutar HTML elementa *iframe*. Iako ova skripta samo ispisuje informacije o kolačićima, ovakav pristup se može iskoristit i za puno opasnije stvari, poput prijevare žrtve na instalaciju zlonamjernog softvera. Prvo korak za takav napad je izrada vjerodostojnog odgovora aplikacije za pretraživanje određenog pojma. Kod nekih Internet stranice postoji mogućnost preopterećenja poslužitelja zbog velikog broja zahtjeva. Također, dosta Internet trgovina ima i svoje aplikacije. Napadač može prevariti korisnika na preuzimanje zlonamjernog softvera, za koji je korisnik na početku dobio dojam da je vjerodostojna aplikacija Internet trgovine. Prvi korak je postavljanje zlonamjernog softvera na bilo koju uslugu koja omogućuje jednostavno preuzimanje kada se posjeti određeni URL. U svrhu primjera izabran je Google Drive. Jednostavna implementacija za usmjeravanje na link za preuzimanje zlonamjernog softvera ostvarena je putem HTML *button* elementa. Za ovakav tip napada nije dovoljno samo ubrizgati skriptu koja sadrži link za preuzimanje zlonamjernog softvera. Potrebno je uključiti i društveni inženjering kako bi rezultat izgledao što vjerodostojnije.

```
Servers are experiencing high load. Download our free app for better performance
<button onclick="window.open('https://drive.google.com/uc?export=download&id=1-65_PNqsRr8-KZ5_ZFLbqK')" >
    Download here</button>
```

Slika 27. Zlonamjerna skripta za preuzimanje zlonamjernog softvera

Slika 28. prikazuje zlonamjernu skriptu za ubrizgavanje. Početna rečenica skripte se prevodi na sljedeći način. Poslužitelji su pod velikim opterećenjem. Preuzmite našu besplatnu aplikaciju za bolje performanse. *Button* element sadrži *onclick* događaj, koji nakon pritiska na gumb usmjerava korisnika na link sa zlonamjernim softverom.



Slika 28. Rezultat ubrizgavanja zlonamjerne skripte

Rezultat ubrizgavanje skripte sa slike 28. je prikazan na slici 29. Nakon ubrizgavanja stranica sadržava gumb koji vodi do URL-a za preuzimanje zlonamjernog softvera. Iako se u ovom slučaju nakon pritiska na gumb preuzima obična tekstualna datoteka, na ovaj način se može prevariti korisnika na preuzimanje puno opasnijih datoteka. Uspješnost ovakvih XSS napada se uglavnom svodi na kreativnost napadača za izradu vjerodostojnog dijela stranice koji sadrži zlonamjerne elemente te na snalažljivo korištenje društvenog inženjeringa kako bi naveo žrtvu da stisne na link.

3.3.3. Protumjere i strategije za zaštitu

Svaki element unutar Internet aplikacije bi trebao biti zaštićen, svi podatci unosa korisnika i sve varijable bi trebale proći kroz detaljne provjere te bi se sa svim podatcima trebalo pažljivo i pravilno upravljati. Ovakva razina sigurnosti predstavlja savršenu otpornost na napade ubrizgavanja. Svaka varijabla koja ne prođe kroz provjeru je potencijalna prijetnja. Korištenje razvojnih okvira, kao što su Angular, Vue.js i ostali, olakšava pravilno upravljanje i provjeru varijabli. Razvojni okviri usmjeravaju razvoj Internet aplikacija prema dobrim sigurnosnim praksama i pomažu u obrani od XSS napada omogućavajući razne predloške, automatsku provjeru unosa i slično. Naravno, i tijekom korištenja razvojnih okvira mogu se javiti problemi zbog načina na koji razvojni okvir upravlja podatcima. Iz navedenog razloga je nužno razumjeti kako točno razvojni okvir upravlja podatcima i gdje postoje potencijalne pogreške. Kada takve pogreške dođu do izražaja potrebno je poduzeti dodatne mjere zaštite. Šifriranje izlaznih rezultata se koristi za sigurno prikazivanje rezultata u obliku u kojemu ih je korisnik unio. Varijable i ključne riječi se trebaju interpretirati kao običan tekst, a ne kao programski kod. Većina razvojnih okvira ima ugrađene opcije za šifriranje izlaznih podataka, koje je poželjno koristiti u većini situacija. U situacijama kada te opcije nisu dovoljne ili kada se ne koristi razvojni okvir za izradu Internet aplikacije, potrebno je uključiti biblioteku za šifriranje izlaznih podataka. Svi podatci unosa bi trebali proći kroz funkciju za šifriranje izlaza. Postoji mnogo različitih metoda za šifriranje izlaza

podataka jer preglednici različito interpretiraju URL, HTML, CSS i JavaScript. Korištenje pogrešne biblioteke ili metode šifriranja izlaznih podataka može dovesti do sigurnosnih propusta [34]. U slučaju dodavanja podataka unutar HTML oznaka poput * *, *<b>*, *<p>* i sličnih potrebno je koristiti šifriranje pomoću HTML entiteta. HTML entitet je dio teksta, to jest niz znakova koji počinje znakom adresnog operatora (&) i završava točkom i zarezom (;). Entiteti se često koriste za prikaz rezerviranih znakova, koji bi se inače interpretirali kao HTML kod, i nevidljivih znakova, poput neprekinutih razmaka. Šifriranje pomoću HTML entiteta je nužno kako bi se pravilno moglo odrediti što su HTML oznake, a što podatci unutar istih.

Znak	**HTML entitet**
&	&amp;
<	&lt;
>	&gt;
"	&quot;
–	&ndash;
—	&mdash;
€	&euro;
#	&num;
≈	&asymp;
@	&commat;
==	&Equal;
ß	&szlig;

Tablica 3. Primjer znakova i njihovih odgovarajućih HTML entiteta [35]

Svaki znak koji se koristi u HTML jeziku, poput znaka manje od „<“ i znaka veće od „>“ koji označuju početak i kraj HTML oznake, ima odgovarajući entitet za sigurno šifriranje. Preglednici interpretiraju entitete kao znakove izlaza (engl. *escape characters*), i pravilno ih prikazuju kao odgovarajući znak. Najvažnije je da ih neće interpretirati kao HTML oznake [36].

```
<div>
    &lt;script&gt;alert(&quot;Pokušaj XSS napada!&quot;)&lt;/script&gt;
</div>
```

Slika 29. Primjer šifriranja unosa pomoću HTML entiteta

Slika 28. prikazuje kako se na siguran način rukuje s unosom korisnika. Unos je *<script> alert("Pokušaj XSS napada!")</script>*, što je jasan pokušaj testiranja ranjivosti na XSS napad. Šifriranje znakova se odrađuje nakon što preglednik izradi DOM za stranicu, stoga neće izvršiti skriptu [36]. Također je bitno stavljati varijable unutar dvostrukih ili jednostrukih navodnika jer navedeno uvelike otežava promjenu sadržaja kojim varijabla upravlja. Stavljanje varijabli unutar navodnika umanjuje broj znakova koje je potrebno šifrirati, što čini aplikaciju pouzdanijom i olakšava šifriranje potrebnih podataka [34]. Vrlo bitna i efektivna protumjera je dodavanje zaglavlja *Content-Security-Policy* u HTTP odgovor. *Content-Security-Policy* ili CSP naziv je zaglavlja HTTP odgovora koje današnji preglednici koriste za poboljšanje sigurnosti Internet stranica. CSP zaglavlje omogućuje ograničavanje koji izvori, kao što su JavaScript, CSS, razne slike i slično, se mogu učitavati te URL-ove s kojih se mogu učitavati. Iako se prvenstveno koristi kao zaglavlje HTTP odgovora, može se primijeniti i putem HTML *meta* oznake. CSP je u početku dizajniran kao protumjera za XSS napade, no današnje verzije također imaju mogućnost obrane i od drugih oblika napada, poput krađe klikova i bilo kakvih oblika napada ubrizgavanja [37].

```
<meta http-equiv="Content-Security-Policy" content="default-src 'self'">
```

Slika 30. Primjer postavljanja CSP-a unutar *meta* oznake

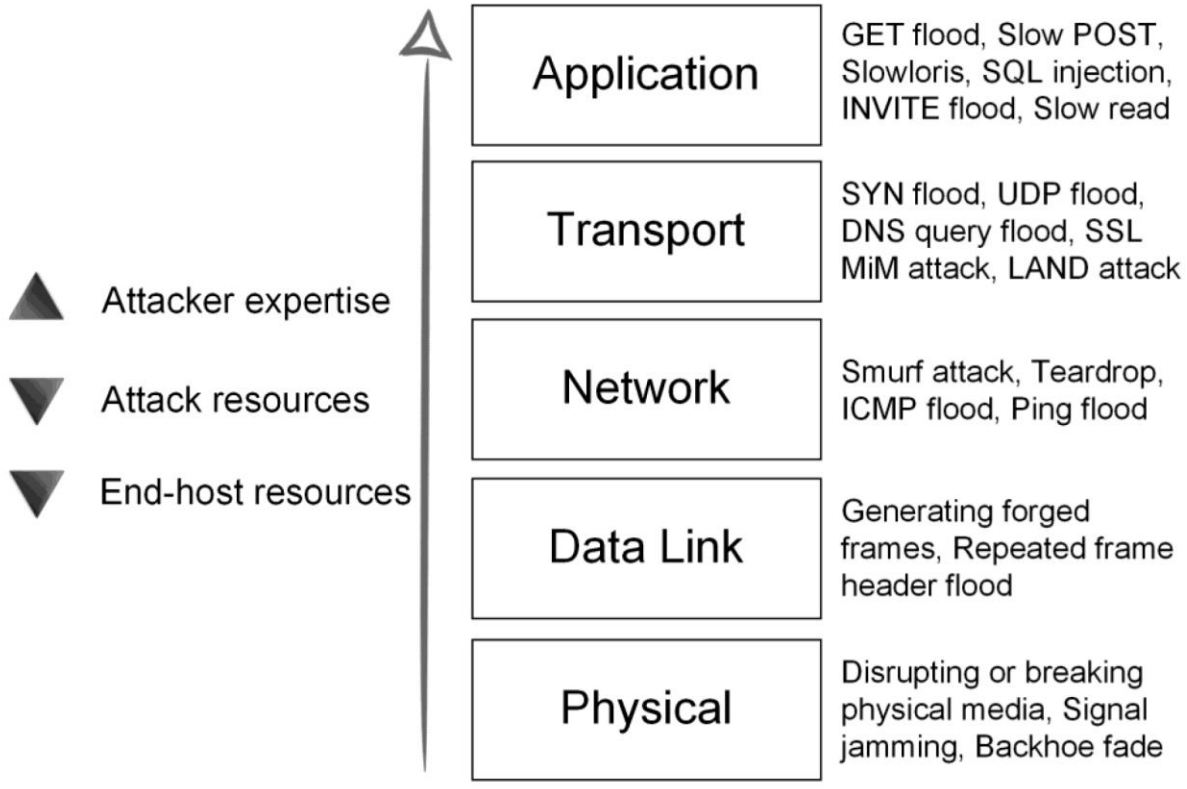
Meta oznaka se postavlja unutar *head* oznake. Pravila CSP-a primjenjuju se samo na sadržaj pronađen nakon obrade *meta* oznake, stoga bi *meta* oznaka uvijek trebala biti pri vrhu dokumenta ili barem prije dinamički generiranog sadržaja. *Default-src* direktiva označuje zadani izvor za sadržaj ako nije definirana nikakva određena direktiva. U ovom slučaju, postavljena je na *self*, što znači da se sadržaj poput JavaScript-a, CSS-a, fontova, slika i slično, može jedino učitati s istog izvora, što je sama Internet stranica, a ne iz bilo kojih drugih vanjskih izvora [37]. CSP može uvelike pomoći u obrani od XSS napada, jer će izvršavati samo skripte poslane od strane odobrenih domena, a sve ostale će se ignorirati.

3.4. Raspodijeljeno uskraćivanje usluge

3.4.1. Opis i tehnike napada

Napad raspodijeljenog uskraćivanja usluge (engl. *distributed denial of service*) ili DDoS je promišljen i koordiniran napad od strane više ugroženih računalnih sustava u svrhu preopterećenja Internet usluge ili poslužitelja. DDoS je pokušaj napada na dostupnost usluge slanjem iznimno velike količine lažnih podataka kako bi ciljana usluga ostala bez resursa. DDoS napad je vrsta napada uskraćivanja usluge (engl. *denial of service*) ili DoS, gdje je razlika u raspodijeljenosti

izvora napada. U DDoS napadu, zlonamjerni promet se šalje iz više distribuiranih izvora, dok u DoS napadu, postoji samo jedan izvor. U DDoS napadu, promet koji dolazi s jednog izvora nije dovoljno velik kako bi utjecao na dostupnost usluge, nego je ugrožavanje dostupnosti usluge rezultat kumulativnog slanja prometa s više različitih računalnih sustava. Napadači uglavnom stvaraju mrežu ugroženih sustava, tako što ih potajno zaraze zlonamjernim softverom. Nakon preuzimanja kontrole nad računalnim sustavima, napadači šalju neželjenu poštu, šire zlonamjerni softver i napadaju druge sustave iskorištavanjem resursa ugroženog sustava. Bilo kakav napad uskraćivanja usluge predstavlja značajan rizik za poduzeće koje primarno posluje na Internetu, jer samo nekoliko minuta prekida rada usluge može imati ozbiljne financijske posljedice i značajan udar na reputaciju poduzeća [38].



Slika 31. DDoS napadi na različitim slojevima mreže [39]

Slika 31. prikazuje koji DDoS napadi se mogu pojaviti na pojedinim slojevima mreže. Postoji mnogo načina i tehnika za izvođenje DDoS napada, stoga postoji i dosta klasifikacija istih. Prema [40] DDoS napadi se klasificiraju na sljedeći način:

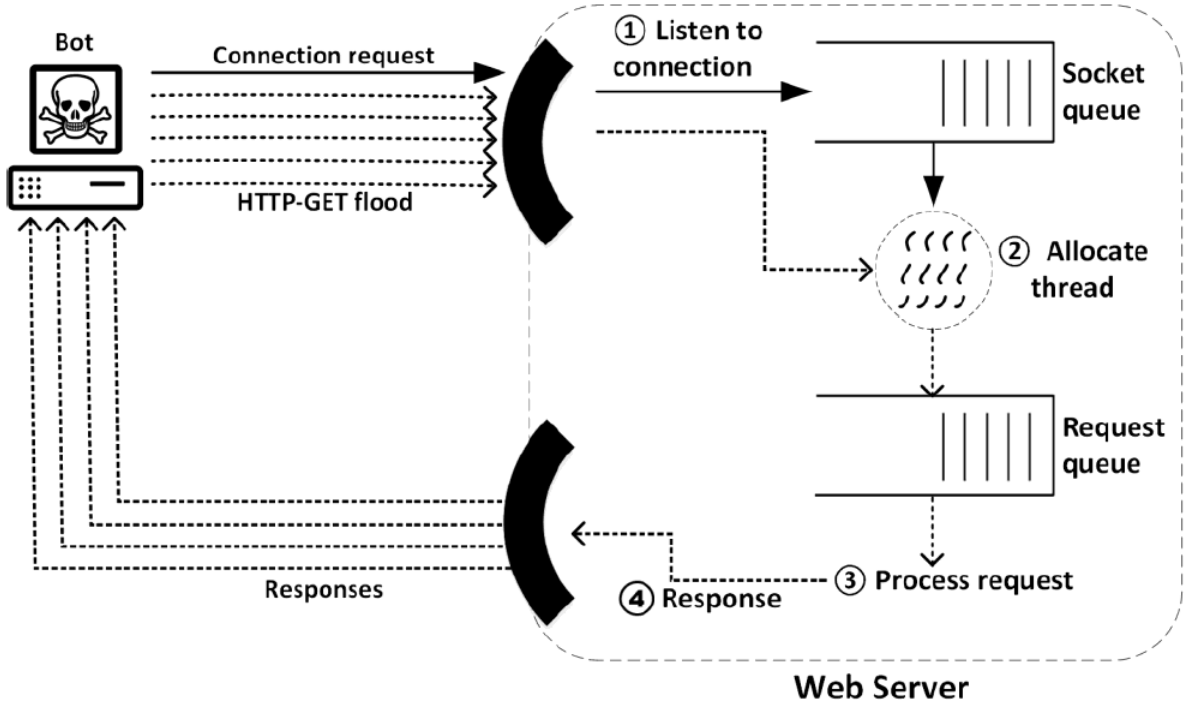
- napadi usmjereni na mrežne resurse ili napadi poplavljivanja: svrha ovih napada je iskorištavanje cijelo ukupne propusnosti (engl. *bandwidth*) mreže žrtve slanjem velikog broja zahtjeva. Ovi napadi su jednostavni, ali učinkoviti. Ova vrsta napada ima nekoliko

podvrsta, ovisno o tome koji protokol se koristi u napadu. Neki primjeri protokola koji se mogu koristiti su UDP (engl. *user datagram protocol*), ICMP (engl. *internet control message protocol*) te IGMP (engl. *internet group management protocol*)

- napadi usmjereni na resurse poslužitelja: ovakvi napadi pokušavaju maksimalno iskoristiti poslužiteljeve resurse za obradu ili memoriju. Napadač može iskoristiti postojeće ranjivosti na poslužitelju ili slabosti u komunikacijskom protokolu za preopterećenje poslužitelja. Jedna od verzija ovog napada se temelji na ranjivosti u TCP/IP protokol. Postoji još nekoliko verzija napada poput TCP SYN poplave, TCP RST napada te TCP PSH+ASK poplava
- niski i spori napadi: za razliku od napada poplavljivanja, niski i spori napadi ne zahtijevaju veliku količinu prometa. Ciljaju točno određene ranjivosti na poslužitelju s relativno malom količinom zlonamjernog prometa. Ova vrsta napada je dosta teška za otkrivanje jer se povezivanje i prijenos podataka odvija normalnom brzinom. Jedan primjer ovakvog napada je *Sockstress*, napad koji iskorištava ranjivosti u TCP stogu
- napadi temeljeni na SSL-u: *Secure Socket Layer* je metoda šifriranja koju koriste razni mrežni komunikacijski protokoli. DDoS napadi temeljeni na SSL-u se mogu izvesti na mnogo načina, na primjer putem SSL mehanizma rukovanja, slanjem lažnih podataka na SSL poslužitelj ili zlouporabom određenih funkcija povezanih sa SSL šifriranjem. Napad temeljen na SSL-u se također može odnositi i na napad koji se provodi preko SSL šifriranog prometa, što ga čini zahtjevnim za prepoznavanje
- napadi na mrežne procese u aplikacijskom sloju: za ovu vrstu napada se može reći da je kombinacija svih prethodnih. Ne ciljaju samo HTTP protokol, nego i HTTPS, DNS, SMTP, FTP, VOIP i ostale aplikacijske protokole koji su ranjivi na DDoS napade. HTTP protokol je najčešća meta za napadače jer je široko korišten protokol na Internetu.

Napadi poplavljivanja su dosta česti jer nisu zahtjevni, a jako su učinkoviti. Cilj poplavljivanja zahtjevima je preopterećenje što većeg resursa sustava slanjem prekomjerne količine zahtjeva za uslugu. Generalno, sustav ne može upravljati tom količinom zahtjeva, te ih stavlja u red čekanja kada stignu. Kada se red zahtjeva prepuni, svi naredni zahtjevi se odbijaju. Jedna od najčešćih vrsta ovog napada je poplavljivanje HTTP zahtjevima. Poplavljivanje HTTP zahtjevima utječe i na propusnost mreže i na resurse sustava žrtve. Međutim, primarna meta ovog napada su resursi sustava. Najčešće se provodi koristeći GET ili POST zahtjeve. Prekomjerna upotreba GET zahtjeva može preopteretiti i sustavne i mrežne resurse pružatelja usluga. POST zahtjevi se uglavnom koriste za kompleksnije zadatke na poslužitelju, poput spremanja podataka u bazu

podataka. Iz navedenog razloga, POST zahtjevi su puno učinkovitiji u preopterećivanju poslužitelja za razliku od GET zahtjeva [41].



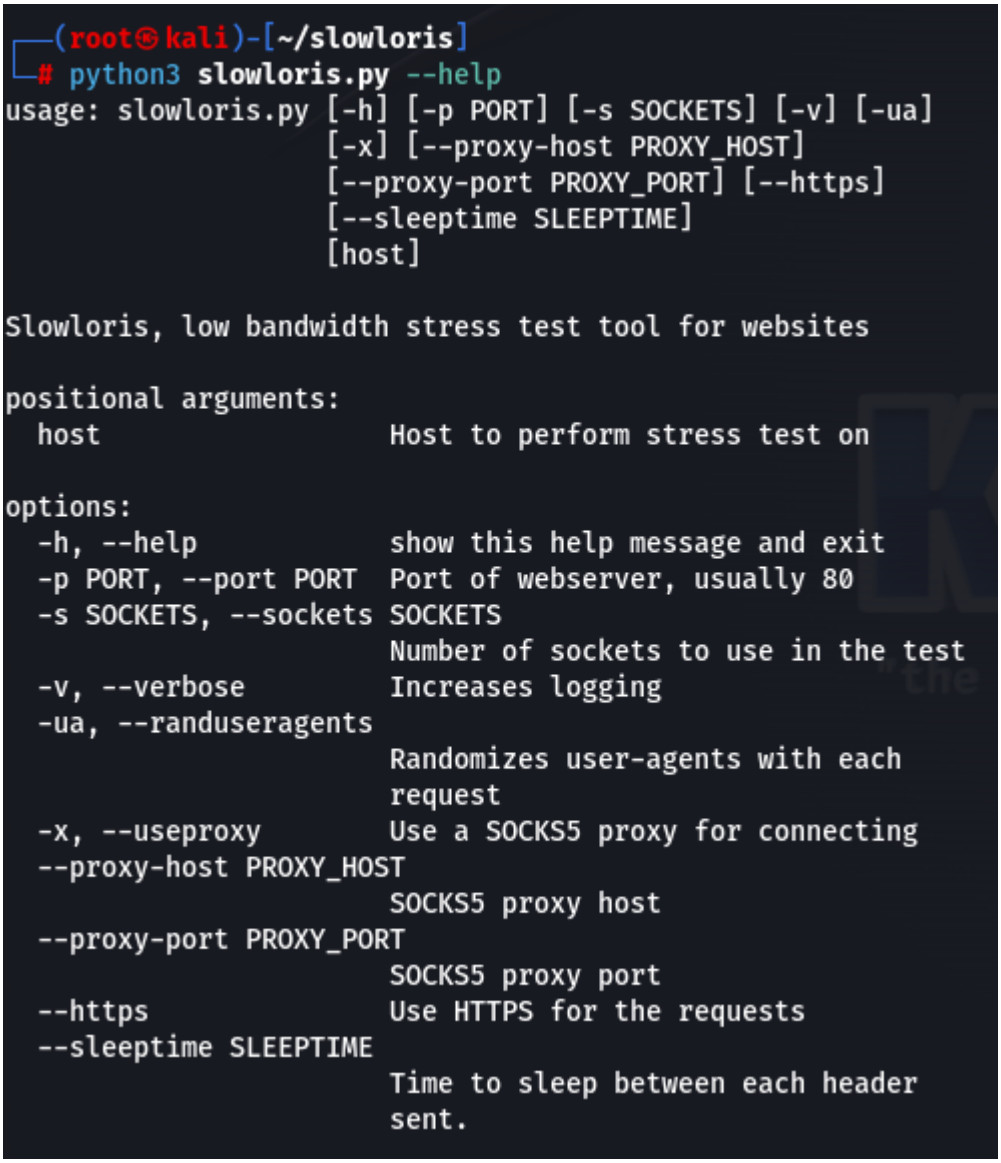
Slika 32. Graf napada poplavljivanja s HTTP GET zahtjevima [39]

Slika 32. prikazuje graf napada poplavljivanja s HTTP GET zahtjevima. Iako je prikazan samo jedan sustav s kojeg se šalju zahtjevi, u napadu uglavnom sudjeluje mnogo više.

3.4.2. Provedba i analiza posljedica napada

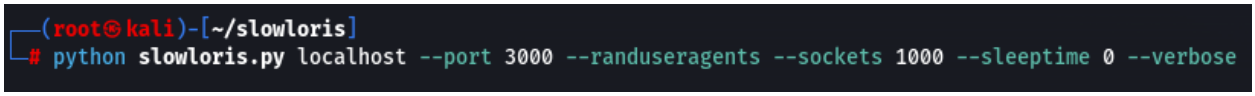
Pošto raspodijeljeni napad uskraćivanja usluge zahtijeva veći broj računalnih sustava, što je dosta zahtjevno za prikazivanje, u svrhu primjera provesti će se napad uskraćivanja usluge s jednog računala, što je zapravo DoS. Kako bi ovakav napad postao DDoS, potrebno ga je provesti s više različitih računala u isto vrijeme. Ranjiva aplikacija je ponovno OWASP Juice Shop. Za napad će se koristiti alat Slowloris. Slowloris je ujedno i vrsta DDoS napada koju koristi navedeni alat. Slowloris je napad na aplikacijski sloj mreže koji preplavljuje poslužitelj korištenjem djelomičnih HTTP zahtjeva. Spada u kategoriju niskih i sporih napada i funkcionira tako da otvara veze s ciljanim poslužiteljem i drži ih otvorenim koliko god je to moguće. Slowloris koristi malu količinu propusnosti i pokušava iskoristiti resurse poslužitelja koristeći zahtjeve koji su sporiji od uobičajenih, ali inače se ponašaju kao normalan promet. Ciljani poslužitelj ima ograničeni broj niti za rukovanje istovremenim vezama. Svaka nit će čekati sve dok se zahtjev ne izvrši, što se naravno

nikada neće dogoditi. Kada poslužitelj dođe do maksimalnog broja otvorenih veza, svaku naknadnu vezu će odbiti, što rezultira uskraćivanjem usluge [42].



Slika 33. Ispis mogućnosti Slowloris alata

Slika 33. prikazuje mogućnosti i način korištenja Slowloris alata.



Slika 34. Naredba za izvršavanje Slowloris napada

Slika 34. prikazuje naredbu i potrebne zastavice za izvršavanje Slowloris napada. Na početku naredbe je potrebno definirati URL mete, što je u ovom slučaju *localhost*. Kada se koristi *localhost*, potrebno je definirati i port, koji je uglavnom i u ovom slučaju 3000. Zastavica *--randuseragents* označuje korištenje nasumičnih *User-Agent* zaglavlja u zahtjevu. *User-Agent* zaglavlje je

karakterističan niz znakova koji sadrži informacije o sustavu korisniku koji šalje zahtjev. Broj *socket*-a se definira zastavicom *--sockets*, te je isti postavljen na 1000 radi slanja većeg broja zahtjeva. Zastavica *--sleeptime* definira vrijeme između svakog zahtjeva, u ovom slučaju je postavljena na 0, ali ju je poželjno postaviti na veći broj kako bi se smanjila mogućnost otkrivanja. Zadnja zastavica je *--verbose*, koja povećava ispis informacija tijekom izvođenja napada. Za što bolju analizu posljedica napada, potrebno je provesti analizu mrežnih paketa pomoći alata Wireshark. Pošto ovaj napad spada u kategoriju niskih i sporih napada, poželjno je koristiti manje *socket*-a i dulje vrijeme između slanja zahtjeva. Radi primjera, navedene vrijednosti su prilagođene kako bi se što efikasnije prikazale mogućnosti napada.

Statistics

Measurement	Captured	Displayed	Marked
Packets	8670167	8670167 (100.0%)	—
Time span, s	119.521	119.521	—
Average pps	72541.1	72541.1	—
Average packet size, B	74	74	—
Bytes	641406526	641406526 (100.0%)	0
Average bytes/s	5,366 k	5,366 k	—
Average bits/s	42 M	42 M	—

Slika 35. Statistika poslanih paketa tijekom DoS napada

Slika 35. prikazuje statističke podatke o poslanim mrežnim paketima tijekom DoS napada. Poslano je preko 8,5 milijuna paketa u oba smjera unutar dvije minute, koliko je i trajao napad. Ovaj broj predstavlja razmijenjeni broj paketa između stranice i jednog korisnika, što je iznimno puno. Tijekom normalnog korištenja stranice unutar dvije minute, broj razmijenjenih paketa između korisnika i stranice je otprilike 1600, ovisno o radnjama korisnika.

Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
Total HTTP Packets	2376005				25.7959	100%	70.3200	62.085
Other HTTP Packets	2373007				25.7634	99.87%	70.3200	62.085
HTTP Response Packets	2998				0.0325	0.13%	5.8200	119.365
???: broken	0				0.0000	0.00%	-	-
5xx: Server Error	0				0.0000	0.00%	-	-
4xx: Client Error	2998				0.0325	100.00%	5.8200	119.365
431 Request Header Fields Too Large	1998				0.0217	66.64%	4.8900	43.622
400 Bad Request	1000				0.0109	33.36%	5.8200	119.365
3xx: Redirection	0				0.0000	0.00%	-	-
2xx: Success	0				0.0000	0.00%	-	-
1xx: Informational	0				0.0000	0.00%	-	-
HTTP Request Packets	0				0.0000	0.00%	-	-

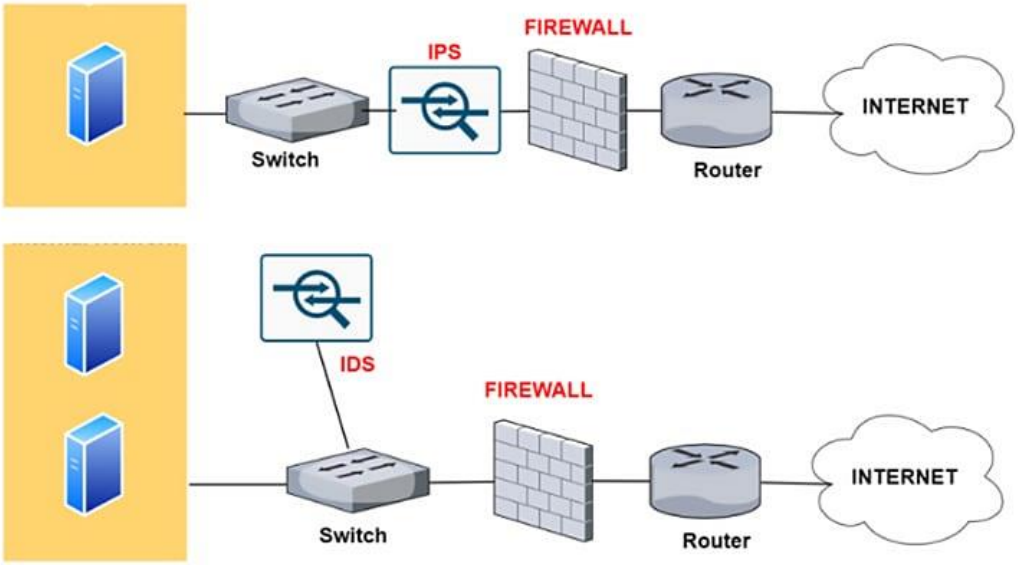
Slika 36. Statistički podaci o HTTP paketima

Slika 36. prikazuje statističke podatke o poslanim HTTP paketima tijekom DoS napada. Poslužitelj aplikacije je poslao 2998 HTTP paketa odgovora, od čega su svi klasificirani kao pogreška na strani klijenta, što je očekivano jer na takav način funkcionira ova vrsta napada. Tijekom normalnog korištenja stranice unutar dvije minute, poslužitelj pošalje oko 100 HTTP paketa odgovora jednom korisniku, ovisno o radnjama korisnika. Kao što je i prethodno navedeno ovakav način izvođenja napada se klasificira kao DoS, jer je uključen samo jedan računalni sustav. Čak i tijekom korištenja jednog računalnog sustava, napad je bio dosta učinkovit, što se može zaključiti prema analizi poslanih mrežnih paketa. Tijekom trajanja napada stranica je bila znatno usporena, i vrijeme odgovora na trivijalne zahtjeve se iznimno odužilo, čak do razine da se stranica nije mogla koristiti. Kada bi se u ovakav napad uključilo više različitih računalnih sustava, posljedice bi bile puno ozbiljnije.

3.4.3. Protumjere i strategije za zaštitu

Izgradnja adekvatne obrane od DDoS napada je zahtjevan i kompleksan postupak za bilo koje poduzeće. Ako napadači posjeduju visoku razinu znanja, postoji mogućnost da i visoko razvijeni obrambeni sustav neće biti dovoljan. Nove vrste DDoS napada se neprestano razvijaju, što znači da se i obrambeni sustavi trebaju prilagođavati istom brzinom. Rano prepoznavanje napada je dosta bitno kako bi se maksimalno mogle ublažiti posljedice napada. Isto tako, sustav za prepoznavanje napada ne bi trebao uzrokovati veliku kolateralnu štetu, dakle trebao bi prepoznati pokušaj DDoS napada bez utjecaja na ostali mrežni promet. Sustav obrane bi se trebao sastojati od modula za nadzor, prepoznavanje i djelovanje. Nadzor mreže se odnosi na prikupljanje informacija o mrežnog prometu i o svim mrežnim čvorovima koji mogu biti meta napada. Također, uz nadzor mreže, bitna je i analiza, kojom se može identificirati neautorizirane usluge koje se koriste na mreži. Primarni cilj modula za prepoznavanje je identifikacija potencijalnih prijetnji, bilježenje informacija te prijava sumnjivih radnji. Uz navedenu svrhu, modul za prepoznavanje se može koristiti i za prepoznavanje problema unutar sigurnosnih pravila i politika, dokumentiranje postojećih prijetnji i upozoravanje pojedinaca koji krše sigurnosna pravila. Modul djelovanja se sastoji od definiranih postupaka za zaustavljanje i ublažavanje posljedica poznatih DDoS napada. Sustav za otkrivanje upada (engl. *intrusion detection system*) ili IDS koristi se za upravljanje sumnjivim radnjama, u obliku prikazivanja upozorenja, bilježenja informacija o radnji ili obavještavanju odgovorne osobe za kibernetičku sigurnost. Bolja opcija od IDS-a je sustav za sprečavanje upada (engl. *intrusion prevention system*) ili IPS. Oba sustava nadziru mrežni promet radi zlonamjernih aktivnosti, međutim za razliku od IDS-a, IPS ima i mogućnost obrane od napada.

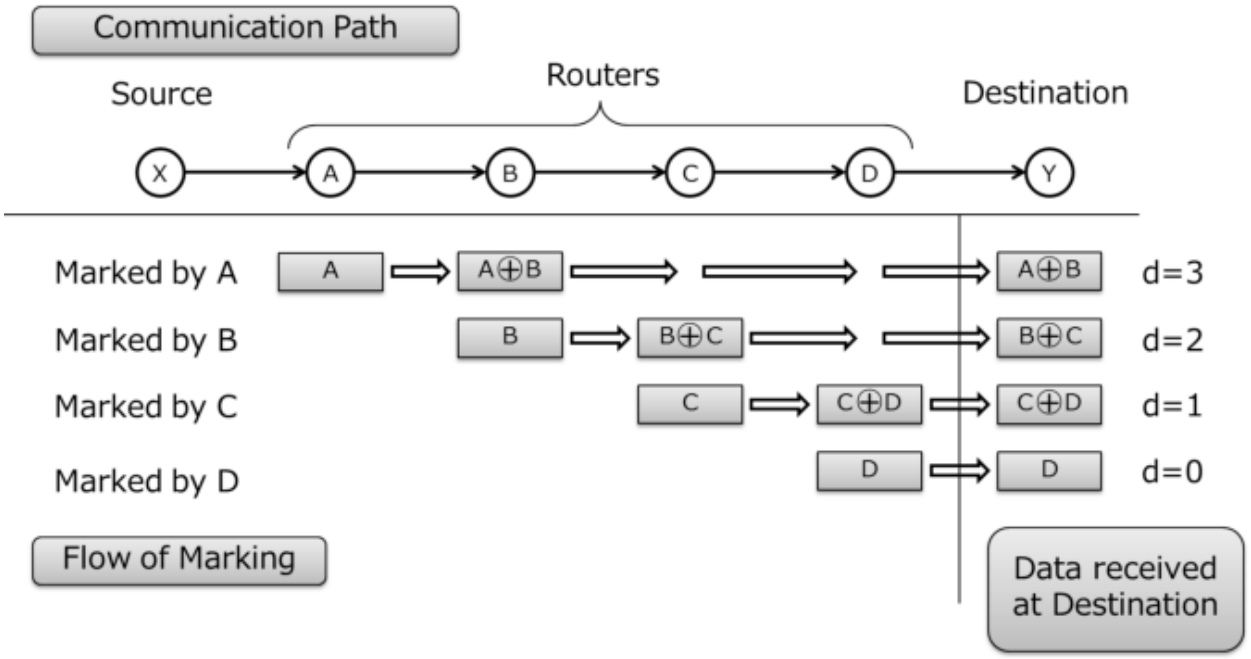
Uglavnom, IPS to čini izdavanjem upozorenja, odbacivanjem zlonamjernih paketa, ponovnim postavljanjem veze te blokiranjem prometa s određenih IP adresa [43].



Slika 37. Dijagram koji prikazuje razliku između IPS-a i IDS-a [44]

Slika 37. jasno prikazuje razliku između IPS-a i IDS-a te je jasno zašto je poželjnije koristiti IPS. Za što učinkovitiju obranu, potrebno je otkriti izvore napada. Budući da je DDoS koordinirani napad, nije jednostavno identificirati izvore napade u stvarnom vremenu. Nadalje, lažiranje IP adresa u zlonamjernim paketima uvelike otežava obranu od DDoS napada. IP praćenje do izvora (engl. IP *traceback*) je jedna od iznimno korisnih strategija u obrani od DDoS napada. Otkrivanje izvora napada može jako dugo potrajati ako postoji dosta izvora i posrednika u napadu. Iz navedenog razloga uvedena je tehnika označavanja paketa (engl. *packet marking*) gdje usmjerivač označava pakete za prosljeđivanje s jedinstvenom adresom. Stoga, kada dođe do napada, putem informacija povezanih s označenim paketima može se doći do izvora napada. Postoje dva pristupa kod označavanja paketa, deterministički (engl. *deterministic*) ili DPM i vjerojatnosni (engl. *probabilistic*) ili PPM. PPM pristup ne zahtijeva prethodno poznavanje cijele mreže za izgradnju stabla napada, to jest mape usmjerivača korištenih u napadu. Može se koristiti tijekom napada i nakon napada. Kod PPM-a IP zaglavlje ima samo jedno polje za pohranu informacija za označavanje. Svaki usmjerivač između izvora i odredišta upisuje svoju jedinstvenu identifikacijsku vrijednost u zaglavlje paketa s određenom vjerojatnošću. Tako prepisuje sve prethodno pohranjene informacije. Na temelju pohranjenih informacija moguće je rekonstruirati put od izvora do odredišta. Tijekom DDoS napada, rekonstrukcija poruka se može obaviti putem

hash funkcije u izvornoj poruci koju je poslao usmjerivač. Značajno ograničenje PPM pristupa su veliki troškovi potrebni za pravilnu rekonstrukciju puta [43].



Slika 38. Dijagram načina označavanja paketa PPM pristupom [45]

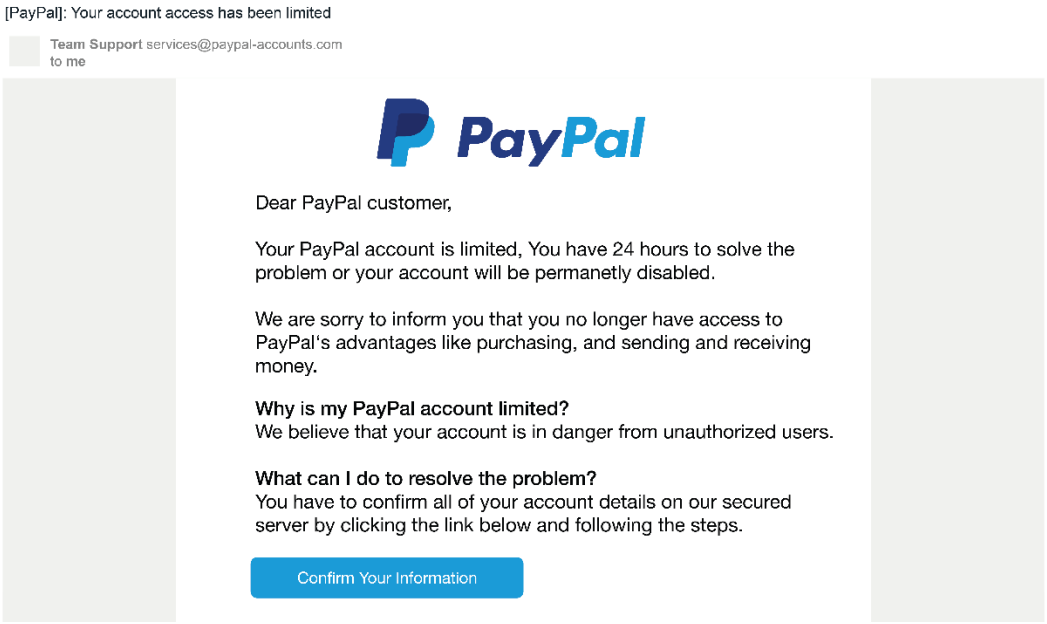
Slika 38. prikazuje na koji način se označuju paketi korištenjem PPM tehnike. Rekonstrukcija puta na temelju pohranjenih informacija se odvija u suprotnom smjeru, od posljednjeg primljenog paketa do prvog primljenog. PPM zahtijeva značajnu količinu paketa koja se treba prikupiti prije procesa rekonstrukcije. Kako se povećava duljina puta, potreban broj paketa za rekonstrukciju također raste. Iz navedenog razloga, poželjnije je koristiti deterministički pristup jer je isti puno efikasniji. Kod determinističkog pristupa, svaki paket je označen potpisom prvog uređaja prilikom ulaska u mrežu. Ova oznaka će se mijenjati prolaskom kroz svaki usmjerivač, sve dok paket ne stigne do mete. Kada paketi stignu do mete, svaki od njih sadrži potpune informacije o putu kroz mrežu. To se zove determinističko označavanje budući da se pohranjuju sve informacije o putu u svaki paket. Kod DPM pristupa se može koristiti bilo koji paket za rekonstrukciju puta bez čekanja za prikupljanje svih paketa, i bez obzira na duljinu puta [46]. DPM pristup rješava problem prevelikih troškova i vremena prilikom korištenja PPM pristupa. Iako PPM nije nužno loša opcija, iz svih navedenih razloga je jasno zašto je DPM ipak efikasnija i bolja opcija u većini slučajeva.

3.5. Krađa identiteta

3.5.1. Opis i tehnike napada

Krađa identiteta (engl. *phishing*) se definira kao oblik društvenog inženjeringa u kojemu napadač (engl. *phisher*) pokušava prijevarom doći do povjerljivih ili osjetljivih podataka korisnika

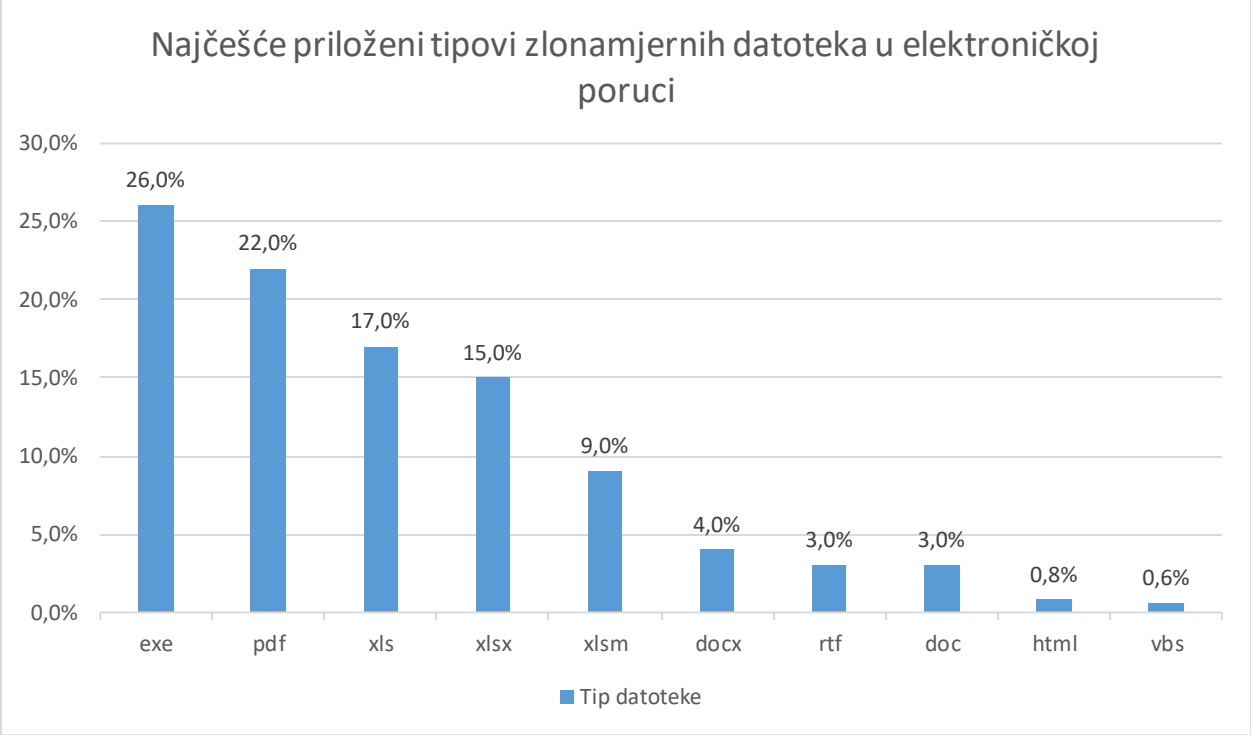
oponašanjem komunikacije od strane pouzdane ili javne organizacije. Prikupljene informacije se koriste za broja kriminalna djela uključujući lažno predstavljanje, prijevare, prodaju podataka, širenje zlonamjernog softvera i slično. Utvrđivanje točnog troška štete uzrokovanog napadima krađe identiteta nije trivijalan zadatak, budući da korporacije i pojedinci često ne žele priznati kada su postali meta napada zbog straha od poniženja ili gubitka kredibiliteta, financijskih troškova ili pravne odgovornosti [47].



Slika 39. Phishing elektronička poruka [48]

Slika 39. prikazuje primjer *phishing* elektroničke poruke, za koju izgleda da je poslana od strane PayPal-a. Nadalje, postoje tri vrste troškova koje treba razmotriti, a to su izravni, neizravni te oportunitetni troškovi. Izravni troškovi su oni koji su direktno uzrokovani napadom, to jest ukupna vrijednost novca i dobara ukradena tijekom *phishing* napada. Neizravni troškovi su troškovi potrebni za popravljanje štete i ublažavanje posljedica nakon uspješnog *phishing* napada, ali ovi troškovi ne uključuju novac i dobra ukradena tijekom napada. Primjer neizravnih troškova su troškovi koje moraju snositi pružatelji određenih usluga kada dođe do ugrožavanja podataka njihovih korisnika, povezanih s korisničkom podrškom i pozivima koji su nužni za obavljanje u svrhu ponovnog postavljanja lozinki korisnika, zamrzavanja korisničkih računa i slično. Neizravni troškovi također mogu uključivati i vrijeme i novac utrošen na povrat podataka individualca ili poduzeća, praćenje neautoriziranih uplata i slično. Oportunitetni troškovi su povezani s propuštenim prilikama jer ljudi obijaju koristiti Internet usluge zbog straha od krađe identiteta ili bilo kojih drugih razloga. Na primjer, ako korisnik ne želi koristiti usluge Internet bankarstva zbog straha od mogućeg gubitka financijskih sredstava, tada su oportunitetni troškovi za banku svi

dodatni troškovi poslovanja s klijentom u banci. Slično tome, sam klijent nema mogućnost obavljanja mnogih bankarskih usluga preko Interneta, stoga mora trošiti i vrijeme i novac za odlazak u banku. Oportunitetni troškovi povezani s gubitkom povjerenja korisnika u Internet trgovine su posebno naglašeni kod trgovaca koji posluju isključivo preko Interneta. Također, oportunitetni troškovi se odnose na potencijalni gubitak budućih korisnika ako poduzeće ima lošu reputaciju zbog određenih *phishing* napada [47]. Česti cilj napadača prilikom slanja *phishing* elektroničke poruke je instalacija zlonamjernog softvera na računalni sustav žrtve. Zlonamjerni softver može imati različite svrhe, ovisno o ciljevima napadača.

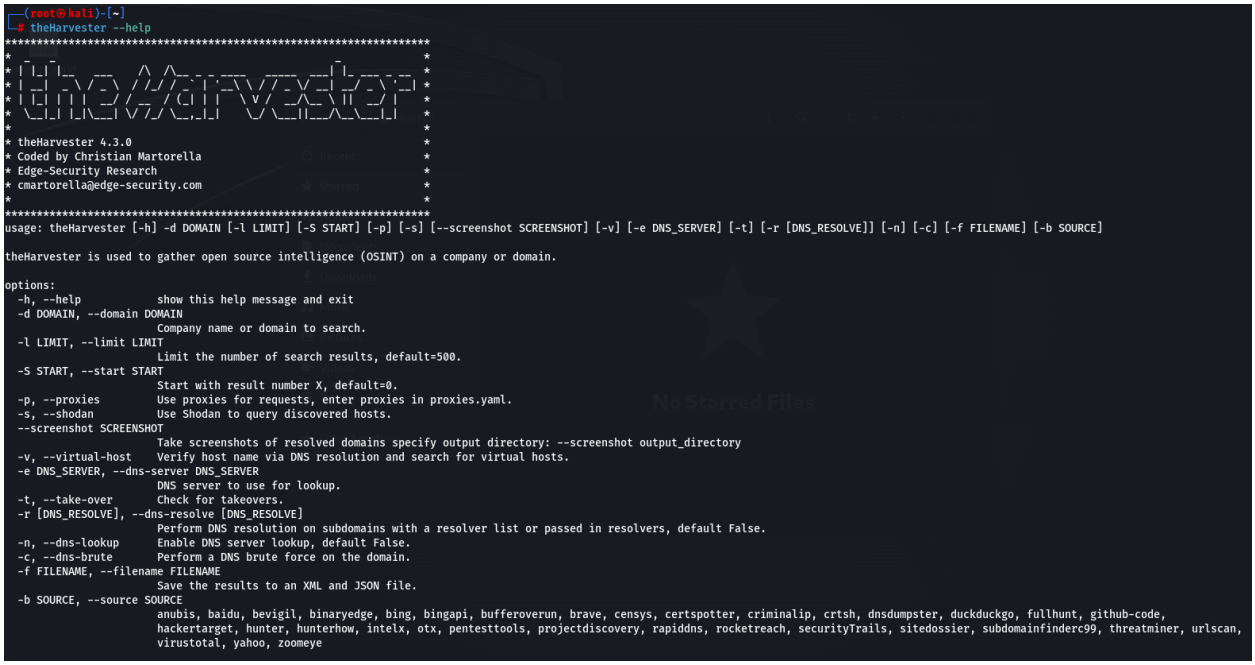


Grafikon 6. Najčešće priloženi tipovi zlonamjernih datoteka u elektroničkoj poruci [15]

Grafikon 6. prikazuje najčešće priložene tipove zlonamjernih datoteka u elektroničkoj poruci kod *phishing* napada. Najčešće priloženi tip zlonamjerne datoteke je očekivano *exe* datoteka. *Exe* označava izvršnu datoteku, što je datoteka koja izvršava određeni programski kod ili slijed instrukcija kada se pokrene. Ako je navedena datoteka poslana sa sumnjive elektroničke adrese, postoji velika mogućnost da sadrži zlonamjerni softver. Zlonamjerni softver se može širiti i putem drugih tipova datoteka, kako je i prikazano na grafikonu 6., PDF datoteke mogu imati skriveni JavaScript programski kod, koji ima mogućnost iskorištavanja ranjivosti u PDF-u. Generalno, svaki tip datoteke može imati nekakav oblik zlonamjernog programskog koda, koji se uglavnom pokreće kada korisnik otvori datoteku.

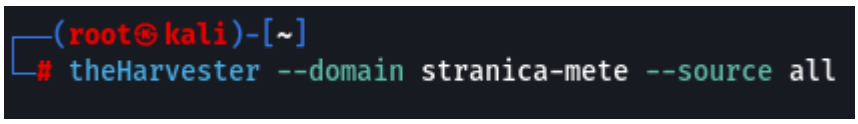
3.5.2. Provedba i analiza posljedica napada

Ovisno o meti i ciljevima, *phishing* napad se može provesti bez uporabe alata za automatizaciju procesa. Međutim, postoje razni alati koji uvelike olakšavaju provedbu *phishing* napada. Bitan prvi korak prije svakog napada, a pogotovo *phishing*-a je prikupljanje informacija o meti. Navedeni korak se može obaviti ručno, istraživanjem društvenih stranica mete, što ponekad može dosta potrajati. Iz toga razloga u svrhu primjera koristit će se alat theHarvester. TheHarvester se koristi za prikupljanje relevantnih informacija o meti, koje se mogu iskoristiti za početne vektore napada. TheHarvester prikuplja imena, adrese elektroničke pošte, IP adrese i slično.



Slika 40. Ispis mogućnosti theHarvester alata

Slika 40. prikazuje ispis mogućnosti alata theHarvester nakon upisivanja naredbe *theHarvester --help*.



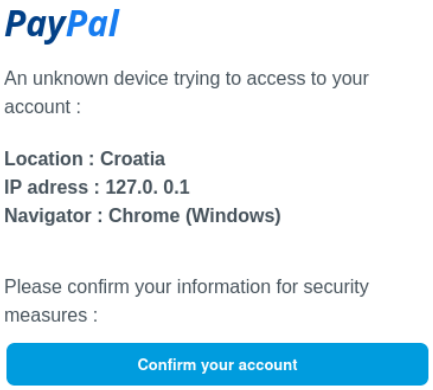
Slika 41. TheHarvester naredba za prikupljanje informacija o meti

Slika 41. prikazuje theHarvester naredbu za prikupljanje relevantnih informacija o meti. Zastavica *--domain* definira domenu Internet stranice mete, a vrijednost je u svrhu primjera postavljena na *stranica-mete*. Zastavica *--source* definira na kojim javnim uslugama će se prikupljati informacije, u ovom slučaju vrijednost je postavljena na *all* kako bi se uključile sve javne usluge koje theHarvester može pretražiti. Nakon pretrage theHarvester ispisuje sve pronađene informacije, a postoji i mogućnost ispisa u datoteku za lakšu daljnju obradu. Nakon potencijalnog pronalaska

adresa elektroničke pošte, idući korak je slanje *phishing* elektroničke poruke. Kao što je već navedeno, slanje poruke se može obaviti ručno, no postoje i alati koji automatiziraju proces. Jedan od njih je *PhishMailer*, alat za izradu predloška elektroničke poruke na temelju popularnih društvenih mreža. Podržan je velik broj društvenih mreža, uključujući Facebook, Instagram, Twitter i slično.

Slika 42. Ispis mogućnosti PhishMailer-a

Slika 42. prikazuje mogućnosti PhishMailer-a, koje se ispisuju nakon pokretanja alata. Nakon odabira na temelju koje stranice će se izraditi predložak, potrebno je postaviti razne opcije za izradu ovisno o kojoj stranici je riječ. Nakon uspješne izrade, predložak se sprema na željeno mjesto na računalu, te se može koristiti za daljnje korake u napadu.

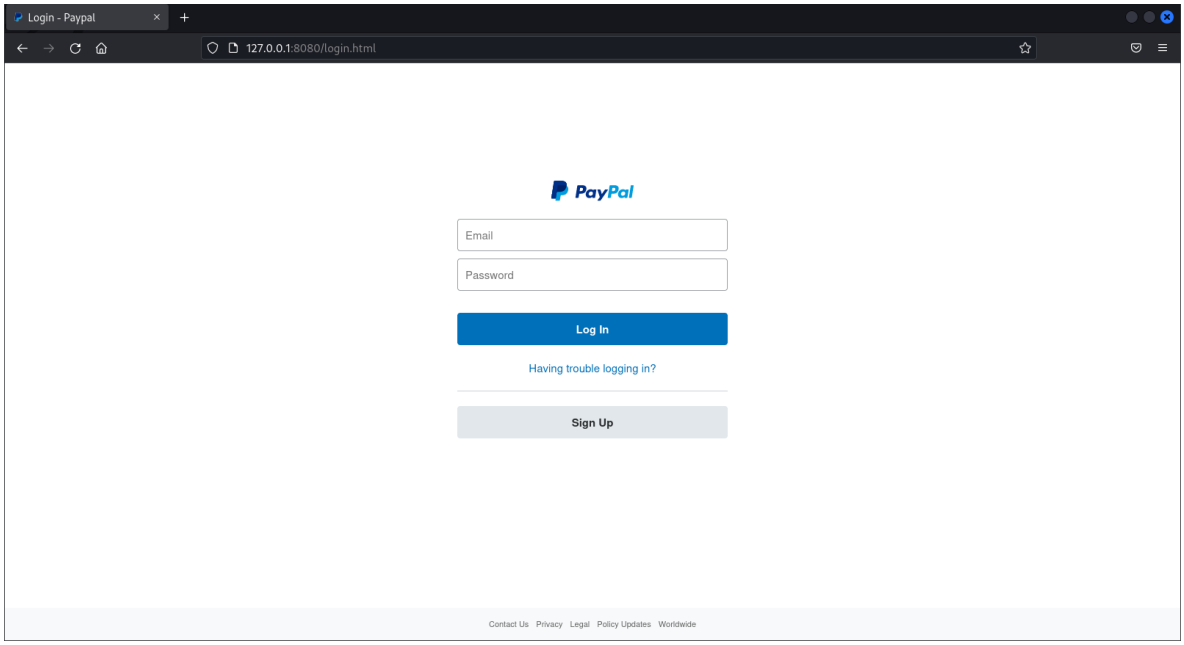


Slika 43. Predložak *phishing* elektroničke poruke za PayPal

Slika 43. prikazuje napravljeni predložak elektroničke poruke za koju izgleda da je došla od strane PayPal-a. Opcije koje su vidljive na slici, poput lokacije i IP adrese se mogu postaviti na željene vrijednosti prema potrebi. Radi primjera, IP adresa je postavljena na adresu lokalnog domaćina. Idući korak ovisi o cilju napada, najčešće je preusmjeravanje mete na određeni URL, na kojemu se nalazi lažna stranica identična određenoj pravoj stranici. Alat Zphisher nudi mogućnosti poput izrade lažne stranice za prijavu, bilježenja podataka za prijavu žrtve i slično.

Slika 44. Mogućnosti Zphisher alata

Slika 44. prikazuje ispis mogućnosti nakon pokretanja Zphisher alata. Korištenjem navedenog alata se mogu izraditi lažne stranice za prijavu koje su identične popularnim stranicama. U svrhu primjera izradit će se PayPal stranica.



Slika 45. Lažna PayPal stranica za prijavu

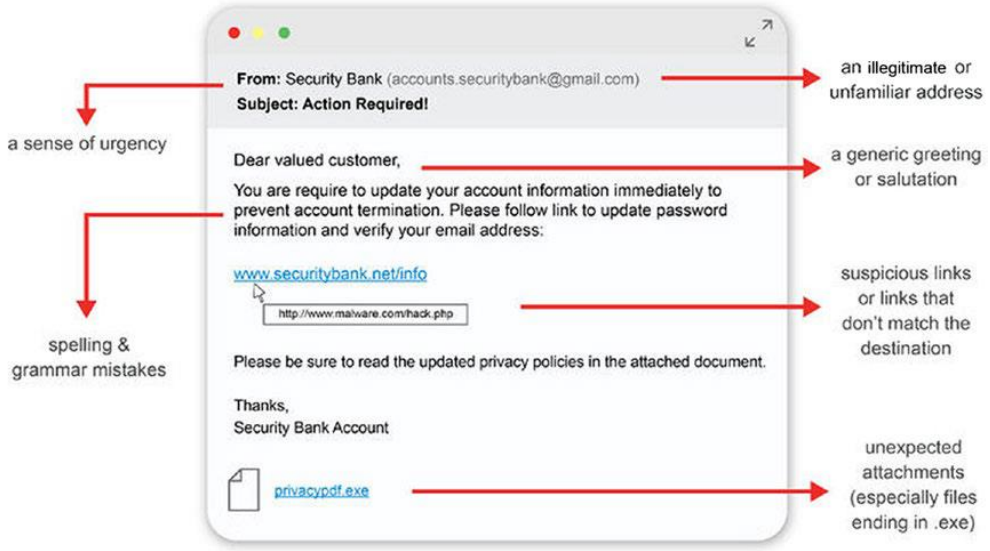
Slika 45. prikazuje lažnu PayPal stranicu za prijavu napravljenu pomoću Zphisher-a. Stranica je postavljena lokalno, no alat nudi razne mogućnosti za postavljanje stranice.

Slika 46. Zabilježeni podatci žrtve

Slika 46. prikazuje zabilježene podatke za prijavu i IP adresu žrtve. Kada žrtva stisne na određeni zlonamjerni link, napadaču se šalju podatci o IP adresi žrtve. Isto tako, nakon što žrtva upiše svoje podatke za prijavu na lažnu stranicu, podatci se šalju napadaču, ispisuju i spremaju na željeno mjesto. Nakon upisa podataka, žrtvu se preusmjerava na pravu stranicu društvene mreže.

3.5.3. Protumjere i strategije za zaštitu

Glavna i često jedina potrebna protumjera za *phishing* napade je kritičko razmišljanje prilikom korištenja Internet usluga. Iako se navedena protumjera može klasificirati kao protumjera protiv bilo koje vrste kibernetičkih napada, kod *phishing* napada je najizraženija. Za uspješnu obranu od *phishing* napada potrebno je razumjeti kako su najčešće strukturirane *phishing* elektroničke poruke te koji znakovi upućuju na potencijalni *phishing* napad.



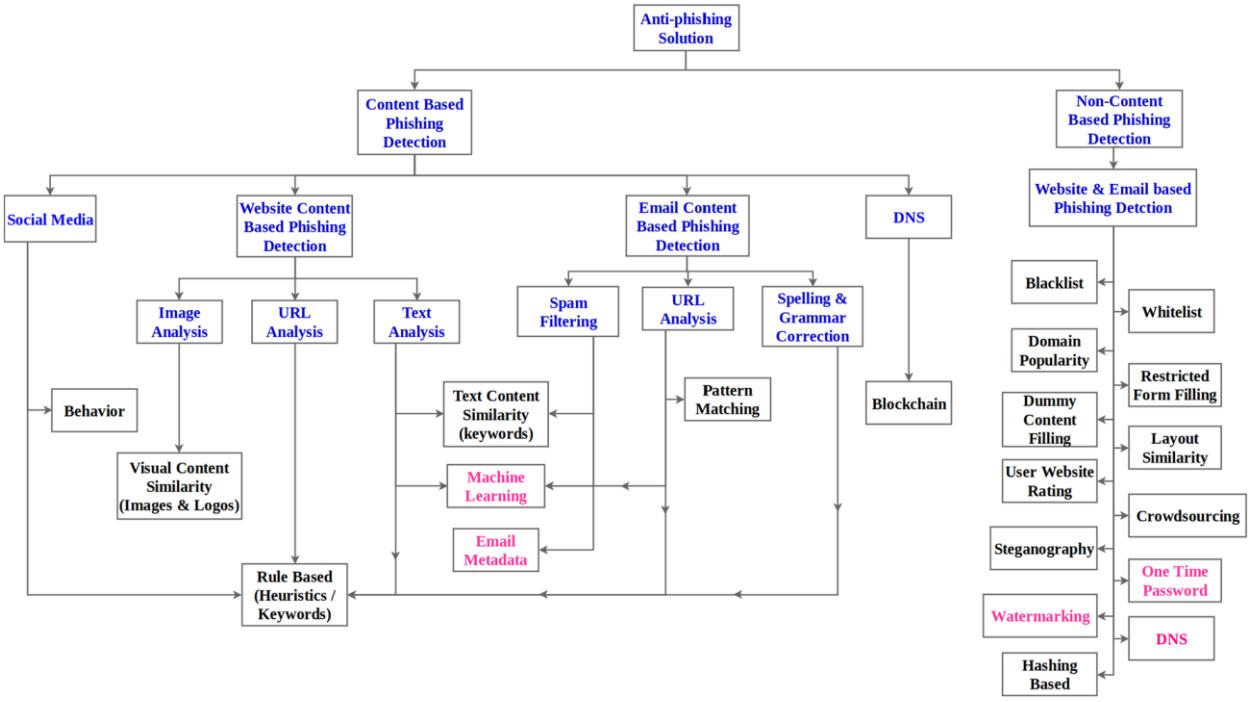
Slika 47. Znakovi koju upućuju na potencijalnu *phishing* elektroničku poruku [49]

Slika 47. prikazuje elektroničku poruku s jasno istaknutim znakovima koji ukazuju na mogući pokušaj *phishing* napada. Postoji dosta znakova koji upućuju na pokušaj *phishing* napada, uključujući sljedeće:

- nepoznata adresa elektroničke pošte
- neusklađena domena elektroničke pošte sa sadržajem poruke
- neutralan pozdrav
- osjećaj užurbanosti
- pravopisne i gramatičke pogreške
- neuobičajeni ili neočekivani zahtjevi
- sumnjivi linkovi ili prilozi.

U slučaju da elektronička poruka sadrži neke ili većinu navedenih znakova, to ne mora nužno značiti da je u pitanju *phishing* napad. Međutim, i dalje je potrebno pažljivo rukovati s linkovima i prilozima u poruci. Potpuno oslanjanje na prepoznavanje znakova *phishing* poruka, ponekad nije dovoljno jer kibernetički kriminalci neprestano pronalaze nove tehnike za izradu *phishing* poruka.

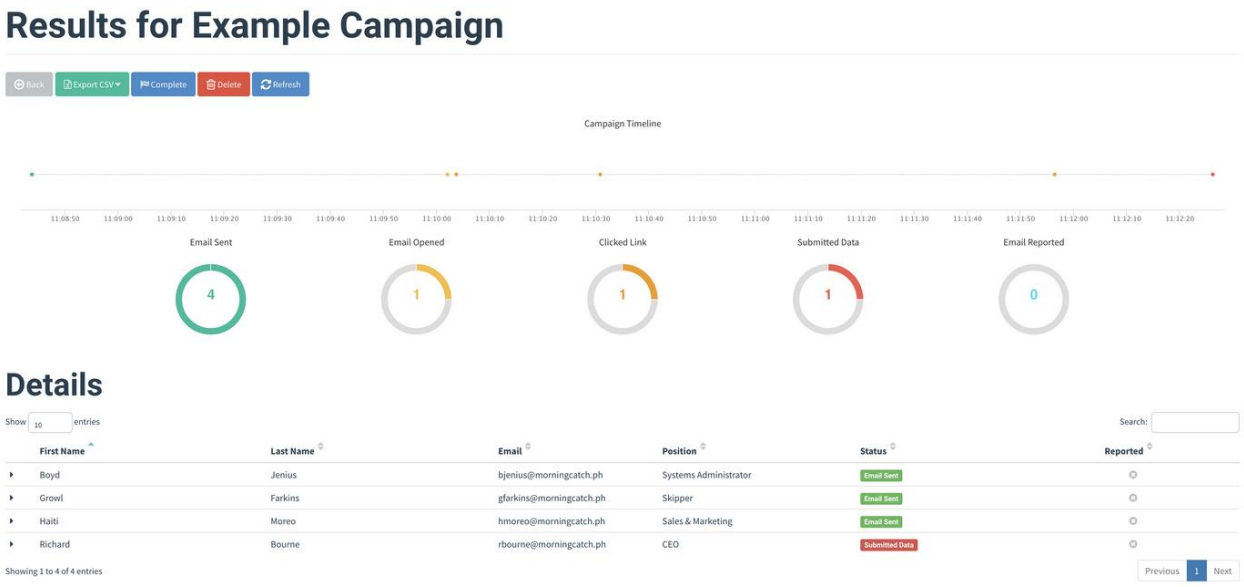
U slučajevima gdje je potrebna viša razina zaštite, potrebno je koristiti i naprednije tehnike otkrivanja *phishing* poruka.



Slika 48. Rješenja za otkrivanje *phishing* poruka [50]

Slika 48. prikazuje podjelu rješenja za otkrivanje *phishing* poruka. Svako rješenje ima svoje prednosti i nedostatke te je prije implementacije bilo kojeg potrebno točno razumjeti na koji način funkcionira. Automatsko otkrivanje *phishing* poruka postoji na razini pružatelja usluga elektroničke pošte, pružatelja Internet usluga te kao dio preglednika u obliku raznih alata ili proširenja. Alati za automatsko otkrivanje *phishing* elektroničkih poruka se temeljne na tehnikama strojnog učenja, statističke klasifikacije i filtriranja. Svi navedeni pristupi imaju različite učinkovitosti jer prijetnje sve više napreduju. Nadalje, kriva klasifikacija elektroničke poruke može utjecati na pouzdanost usluge. Protumjere za *phishing* napade u Internet preglednicima su implementirane u obliku izrade crnog popisa stranica, analiziranje URL-a i imena domene, usporedba izgleda s potvrđenim *phishing* stranicama te analiza sadržaja stranice. Pružatelji Internet usluga koriste slične tehnike za otkrivanje *phishing* stranica te mogu upozoriti korisnike prije pristupa potencijalnim *phishing* stranicama [51]. Također, postoji i mogućnost razvijanja vlastitih metoda otkrivanja *phishing* napada prema potrebi, koje se mogu temeljiti na tehnikama strojnog učenja i povratnim informacijama zaposlenika o potencijalnim pokušajima *phishing*-a. Poduzeća mogu testirati razinu otpora na *phishing* napade korištenjem različitih razvojnih okvira za simulaciju *phishing* napada. Putem istih dobiva se detaljan uvid u razinu znanja zaposlenika o

kibernetičkim prijetnjama te na kojim područjima je potrebna dodatna edukacija. Jedan od takvih razvojnih okvira je Gophish.



Slika 49. Rezultati Gophish phishing simulacije [52]

Slika 49. prikazuje primjer rezultata Gophish simulacije. Razvojni okviri za provedbu *phishing* simulacija poput Gophish-a, pružaju detaljna izvješća o kibernetičkom znanju zaposlenika. Izvješća prikazuju podatke o tome koliko je elektroničkih poruka poslano, koliko je zaposlenika otvorilo poruku, koliko je kliknulo na link sadržan unutar poruke, koliko je unijelo svoje osjetljive podatke, koliko je zaposlenika prijavilo pokušaj *phishing*-a i slično.

4. GENERALNE PROTUMJERE ZA OBRANU OD KIBERNETIČKIH NAPADA

Generalne protumjere se odnose na određene tehnike i strategije koje su učinkovite u obrani od svih vrsta kibernetičkih napada. Naravno, pošto se kibernetički napadi razviju neprestano, nije moguće imati generalno rješenje koje može zaustaviti svaku vrstu napada s jednakom učinkovitošću, no trebalo bi moći otkriti i zaustaviti trivijalne pokušaje napada. Generalno rješenje za obranu od kibernetičkih napada obuhvaća razne metode i alate koji zajedno stvaraju sveobuhvatnu obranu od niza različitih vektora napada.

4.1. Vatrozid

Vatrozid Internet aplikacije (engl. *web application firewall*) ili WAF pomaže u zaštiti Internet aplikacija filtriranjem i nadzorom HTTP prometa između Internet aplikacije i korisnika. Ima mogućnost zaštite stranica od napada poput SQL ubrizgavanja, ubrizgavanja skripti preko stranica, krivotvorenja zahtjeva preko stranica, uključivanja zlonamjernih datoteka i slično. WAF je vrsta obrnutog *proxy*-a, što znači da štiti poslužitelj od zlonamjernih radnji korisnika, tako što detaljno provjerava promet korisnika prije nego što isti dođe do poslužitelja. WAF se temelji na skupu pravila, koja se često nazivaju i politikama. Cilj ovih pravila je filtriranje zlonamjernog prometa za zaštitu od kibernetičkih napada. Jedna od bitnih vrijednosti WAF-a je brzina i lakoća implementiranja novih promjena i ažuriranja pravila, što omogućuje brži odgovor na različite vektore napada. Na primjer, tijekom DDoS napada, ograničenje brzine (engl. *rate limiting*) zahtjeva koju putuju mrežom se može dosta brzo implementirati ažuriranjem pravila [53]. Prema [53] WAF se može implementirati na tri različita načina, gdje svaki ime svoje prednosti i nedostatke:

- mrežni WAF se općenito temelji na hardveru. Budući da se instalira lokalno, minimizira kašnjenje, ali mrežni WAF je najskuplja opcija i također zahtijeva pohranu i održavanje opreme
- WAF temeljen na domaćinu može se u potpunosti integrirati u softver aplikacije. Ovo rješenje je jeftinije od mrežnog WAF-a i nudi veću prilagodljivost. Loša strana WAF-a temeljenog na domaćinu je potrošnja resursa lokalnog poslužitelja, složenost implementacije te troškovi i vrijeme održavanja
- WAF temeljen na oblaku nudi pristupačnu opciju koju je vrlo jednostavno implementirati. Također ima minimalne početne troškove, jer korisnici plaćaju mjesečno ili godišnje za

sigurnost kao uslugu. WAF temeljen na oblaku također može ponuditi rješenje koje se stalno ažurira za zaštitu od najnovijih prijetnji bez ikakvog dodatnog rada ili troškova na strani korisnika. Nedostatak WAF-a temeljenog na oblaku je što korisnici predaju odgovornost trećoj strani, stoga im neke značajke WAF-a mogu biti nepoznate.

Ako je potrebna snažnija zaštita, onda je vatrozid sljedeće generacije (engl. *next generation firewall*) ili NGFW bolja opcija od vatrozida Internet aplikacije. NGFW je sigurnosni uređaj koji obrađuje mrežni promet i primjenjuje pravila za blokiranje potencijalno opasnog prometa. NGFW proširuje i poboljšava mogućnosti tradicionalnog vatrozida. Također, dodaje i mogućnosti koje tradicionalni vatrozidi nemaju poput:

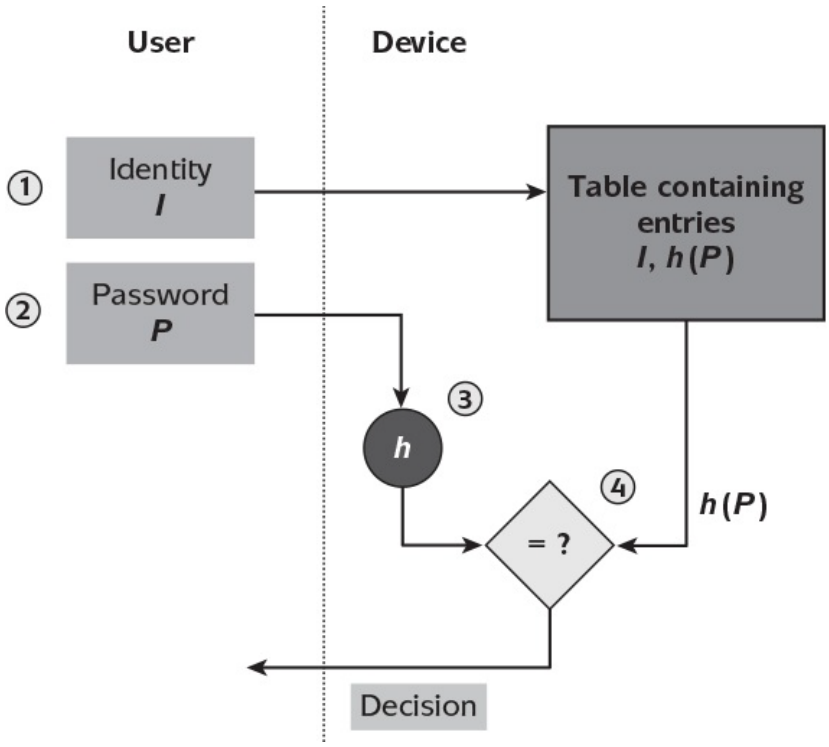
- duboke provjere paketa (engl. *deep packet inspection*) ili DPI uz filtriranje paketa
- sprečavanja upada
- korištenja obavještajnih podataka o prijetnjama (engl. *threat intelligence*)
- mogućnosti implementiranja narednih izvora informacija
- tehnika i strategija za sprečavanje kibernetičkih prijetnji.

Većina ovih značajki je moguća jer, za razliku od uobičajenog vatrozida, NGFW može analizirati promet na nekoliko slojeva u OSI modelu, a ne samo na slojevima 3 (mrežni sloj) i 4 (transportni sloj). NGFW može pregledati HTTP promet sloja 7 i identificirati koje su aplikacije u upotrebi. Ovo je važna mogućnost jer se sloj 7 (aplikacijski sloj) sve više koristi za napade kako bi se zaobišle sigurnosne politike koje na slojevima 3 i 4 primjenjuje uobičajeni vatrozid. NGFW poboljšava filtriranje paketa izvođenjem dubinske provjere paketa. Poput uobičajenog filtriranja paketa, DPI uključuje provjeru svakog pojedinačnog paketa za određivanje izvorne i odredišne IP adrese, izvornog i odredišnog porta i slično. Sve ove informacije sadržane su u zaglavljima paketa sloja 3 i sloja 4. Ali DPI također provjerava tijelo svakog paketa, ne samo zaglavlje. Točnije, DPI provjerava tijelo paketa za uzorke poznatih zlonamjernih softvera i druge potencijalne prijetnje. Uspoređuje sadržaj svakog paketa sa sadržajem poznatih zlonamjernih napada [54].

4.2. Sigurna pohrana lozinki

Sigurna pohrana lozinki korištenjem *hash* funkcije je ključna, kako bi se lozinke zaštitile i u slučaju povrede podataka. Većina modernih programskih jezika i razvojnih okvira nudi ugrađene funkcionalnosti za sigurno pohranjivanje lozinki. *Hash* funkcija je matematička funkcija, odnosno postupak za pretvorbu numeričke ulazne vrijednosti u numeričku izlaznu vrijednost. *Hash* funkcija komprimira ili sažima proizvoljno dugu ulaznu vrijednost u izlaznu vrijednost fiksne duljine. Za

istu ulaznu vrijednost, izračunata izlazna vrijednost će uvijek biti jednaka. Bitna značajka *hash* funkcije je brzina i učinkovitost prilikom računanja. *Hash* funkcija je jednosmjerna, što znači da je gotovo nemoguće, u smislu učinkovitosti i brzine, doći do ulazne vrijednosti na temelju izlazne. Također, vrlo bitna značajka *hash* funkcije je otpornost na podudaranje (engl. *collision resistance*), što znači da bi trebalo biti gotovo nemoguće pronaći dvije različite ulazne vrijednosti za koje će *hash* funkcija izračunati istu izlaznu vrijednost. Za hash funkcije MD4, MD5, SHA1 su pronađene ranjivosti koje uvelike ugrožavaju sigurnost prilikom korištenja istih, stoga ih je potrebno izbjegavati. Prije odabira *hash* funkcije, uvijek je potrebno istražiti sadržava li ista ranjivosti koje mogu utjecati na integritet podataka [55].



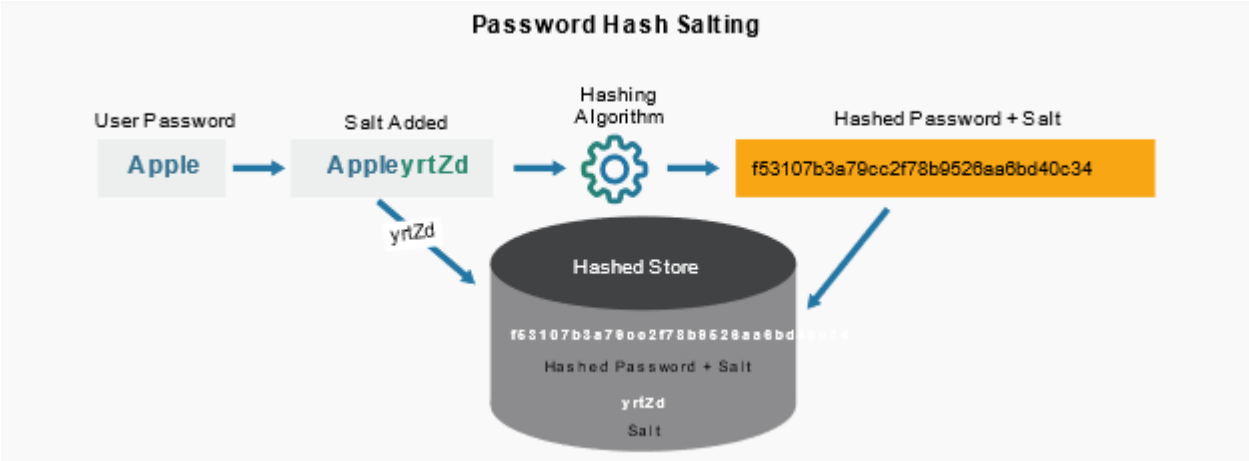
Slika 50. Korištenje *hash* funkcije u svrhu sigurne pohrane lozinki [55]

Slika 50. prikazuje graf korištenja *hash* funkcije u svrhu sigurne pohrane lozinki. Prema [55] proces je sljedeći:

- korisnik unosi svoje podatke ili identitet označen sa slovom *I*
- korisnik unosi lozinku označenu sa slovom *P*
- proces ili dio aplikacije odgovoran za autentifikaciju korisnika unosi lozinku *P* u hash funkciju i izračunava izlaznu vrijednosti, označenu s *h(P)*
- proces ili dio aplikacije odgovoran za autentifikaciju korisnika pretražuje bazu podataka za identitet *I* te uspoređuje pohranjenu *hash* vrijednost lozinke s ranije izračunatom *hash*

vrijednošću *h(P)*. Ako su vrijednosti iste, proces autentifikacije je uspješan, a u suprotnom se korisniku odbija pristup.

Ukoliko neautorizirane osobe dođu do *hash* vrijednosti lozinki, putem raznih metoda, poput napada rječnikom mogu probiti sigurnosnu zaštitu istih. Kako bi se lozinke dodatno zaštitile i u slučaju kibernetičkog napada, potrebno je uključiti dodatne mjere, koje se nazivaju *salting* i *peppering*. *Salt* je jedinstveni, nasumično stvoreni niz znakova koji se dodaje svakoj lozinki prilikom izrade *hash* vrijednosti lozinke. *Salt* je jedinstven za svakog korisnika, što uvelike otežava spomenute napade rječnikom, jer vrijeme potrebno za probijanje sigurnosti zaštite svake lozinke raste s brojem *hash* vrijednosti. Također, ako se koristi *salting* nemoguće je utvrditi koristi li više korisnika jednaku lozinku, jer će zbog različite *salt* vrijednosti, *hash* vrijednosti biti također različite. Moderne *hash* funkcije poput Argon2id-a, bcrypt-a i PBKDF2-a automatski uključuju *salting* tehniku, tako da nisu potrebni nikakvi dodatni koraci prilikom njihove upotrebe [56].

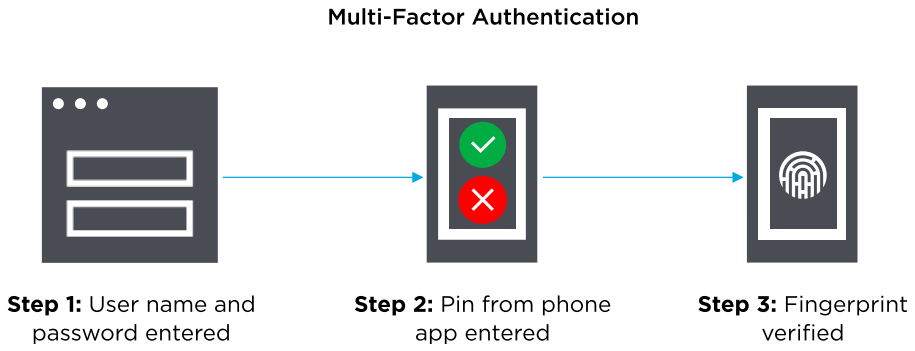


Slika 51. Primjer korištenja *salting* tehnike [57]

Slika 51. prikazuje primjer korištenja *salting* tehnike prilikom sigurnog pohranjivanja lozinke putem *hash* funkcije. Uz *salt* može se koristiti i *pepper* za dodatni sloj zaštite. Svrha *pepper*-a je spriječiti napadača u probijanju sigurnosne zaštite lozinke, ako je isti jedino ostvario pristup bazi podataka. Jedna od strategija *peppering*-a je šifriranje *hash* vrijednosti lozinki koristeći simetrični ključ za šifriranje prije spremanja *hash* vrijednosti lozinki u bazu podataka, gdje je ključ za šifriranje zapravo *pepper*. *Peppering* strategije ni na koji način ne utječu na *hash* funkciju. *Pepper* se razlikuje od *salt*-a po tome što je *salt* jedinstven za svaku pohranjenu lozinku, dok je *pepper* svugdje isti. *Salt* se sprema u bazu podataka, a spremanje *pepper*-a u bazu podataka bi se trebalo izbjegavati. Vrijednost *pepper*-a bi se trebala pohraniti na sigurno mjesto s ostalim osjetljivim podatcima, te bi bilo poželjno uključiti strategije za rotiranje ili periodično mijenjanje *pepper* vrijednosti [56].

4.3. Autentifikacija s višestrukim faktorima

U određenim situacijama snažne lozinke ne predstavljaju dovoljnu razinu sigurnosti, stoga je potrebno uključiti autentifikaciju s višestrukim faktorima (engl. *multi factor authentication*) ili MFA.



Slika 52. Primjer autentifikacije s višestrukim faktorima [58]

Slika 52. prikazuje primjer autentifikacije s višestrukim faktorima, nakon unosa korisničkog imena i lozinke, zahtijeva se dodatna provjera identiteta korisnika putem PIN-a koji se šalje na mobitel te otiska prsta. Autentifikacija s višestrukim faktorima djeluje kao dodatni sigurnosni sloj koji neovlaštenim korisnicima sprječava pristup računalnim sustavima, čak i u slučaju kada je sigurnost lozinke ugrožena. Autentifikacija s višestrukim faktorima zahtijeva više oblika provjere identiteta tijekom registracije ili prijave korisnika [59]. Prema [59] postupak autentifikacije s višestrukim faktorima je sljedeći:

- registracija: korisnik registrira račun sa željenim korisničkim imenom i lozinkom. Zatim povezuje druge stavke, poput mobitela ili fizičkog sigurnosnog ključa sa svojim računom. Postoji i mogućnost povezivanja virtualnih stavki, poput adrese elektroničke pošte ili aplikacije za autentifikaciju. Uz sve navedeno moguće je uključiti i biometrijsku provjeru, što predstavlja jednu od najtočnijih i najsigurnijih metoda
- prijava: korisnik s omogućenim MFA-om se prijavljuje u aplikaciju ili stranicu, nakon uspješnog unosa korisničkog imena i lozinke
- provjera: korisnik dovršava postupak provjere autentičnosti ovjerom ostalih stavki. Ovaj korak se može implementirati na bilo koji od navedenih načina, ili može biti kombinacija više načina. Korisnik dobiva pristup sustavu tek kada su svi ostali podatci ovjereni.

MFA se može implementirati u sustav tako da se višestruka autentifikacija zahtijeva samo pri prvom pristupu na novom uređaju. Nakon toga, sustav će zapamtiti uređaj i tražiti samo lozinku pri svakoj sljedećoj prijavi. MFA je moguće podići na dodatnu razinu sigurnosti. Adaptivna autentifikacija s višestrukim faktorima, koristi poslovna pravila i informacije o korisniku za određivanje faktora za provjeru koje je potrebno primijeniti u određenoj situaciji. Adaptivna autentifikacija s višestrukim faktorima je korisna za postizanje bolje ravnoteže između sigurnosnih zahtjeva i korisničkog iskustva. Na primjer, može se povećati ili smanjiti potreban broj koraka za autentifikaciju korištenjem korisničkih informacija poput:

- broja neuspjelih pokušaja prijave
- zemljopisnog položaja korisnika
- uređaja koji se koristi za prijavu
- vremena pokušaja prijave
- operacijskog sustava
- izvorne IP adrese
- uloge korisnika.

Adaptivna rješenja koriste umjetnu inteligenciju i strojno učenje za analizu i prepoznavanje sumnjivih aktivnosti za pristup sustavu. Ova rješenja mogu nadzirati aktivnosti korisnika tijekom vremena kako bi prepoznala uzorke u ponašanju korisnika te otkrila neobična ponašanja, poput pokušaja prijave u neuobičajeno vrijeme, s neuobičajene lokacije ili s nepoznatih uređaja. Ovisno o procijenjenom riziku mijenja se broj potrebnih faktora za autentifikaciju [59].

4.4. Ostale protumjere

Načelo najmanje privilegije (engl. *the principle of least privilege*) koncept je kibernetičke sigurnosti koji govori da korisnik ili entitet treba imati pristup samo određenim podatcima, resursima, funkcijama ili aplikacijama potrebnim za dovršenje traženog zadatka. Organizacije koje slijede načelo najmanje privilegije uvelike smanjuju mogućnost kibernetičkog napada i rizik od širenja zlonamjernog softvera. Ovo načelo poboljšava radne performanse jer smanjuje vrijeme potrebno za zaustavljanje i ublažavanje posljedica kibernetičkog napada [60].

Penetracijsko testiranje je sigurnosna vježba u kojoj stručnjak za kibernetičku sigurnost pokušava pronaći i iskoristiti ranjivosti u računalnom sustavu. Svrha ovog simuliranog napada je identificirati sve slabe točke u obrani sustava koje bi napadači mogli iskoristiti. Penetracijsko testiranje treba izvoditi osoba s malo ili nimalo prethodnog znanja o tome kako je sustav osiguran

jer se tako mogu otkriti mrtve točke koje su razvojni programeri previdjeli. Iz tog razloga obično se angažiraju vanjski izvođači za provođenje penetracijskih testova. Nakon dovršetka penetracijskog testa, odgovorna osoba ili osobe će objasniti koje ranjivosti su pronađene, zašto predstavljaju rizik te kako se iste trebaju ispraviti [61].

Za što efikasniju obranu od kibernetičkih napada, uz sve navedene protumjere, potrebno je uključiti i redovne izrade sigurnosnih kopija podataka, kako bi se integritet podataka što bolje sačuvao u slučaju kibernetičkog napada. Potrebno je redovno ažurirati računalne sustave zbog implementiranja sigurnosnih zakrpa koje popravljaju pronađene ranjivosti u sustavu ili aplikaciji. Uz redovno ažuriranje, preporuča se i upotreba antivirusnih programa za što učinkovitiju zaštitu od zlonamjernih softvera. Antivirusni programi također nude i mogućnosti periodičnog skeniranja sustava ili određenih datoteka za potencijalne prijetnje. Čak i kada se koristi autentifikacija s višestrukim faktorima potrebno je uvesti stroga pravila za stvaranje lozinki s kombinacijom velikih i malih slova, posebnih znakova i brojeva te određene najmanje duljine. Uz stroga pravila za stvaranje lozinki, poželjno je da zaposlenici redovito mijenjaju lozinke, ovaj proces se može i automatizirati na razne načine. Uz redovito analiziranje kibernetičke sigurnosti u poduzeću, pravilnu edukaciju zaposlenika, te praćenje najboljih sigurnosnih praksi mogućnost za kibernetički napad se svodi na minimum.

5. ZAKLJUČAK

Ovim diplomskim radom obrađena je tema kibernetičkih napada i protumjera za obranu od istih. U radu je detaljno objašnjen pojam kibernetičkog napada, te su objašnjeni povijesni primjeri kibernetičkih napada koji su značajno utjecali na razvoj polja kibernetičke sigurnosti. Za svaki primjer povijesnog napada su detaljno opisane ranjivosti koje su dovele do istoga, te su objašnjene posljedice koje je određeni kibernetički napad ostavio na društvo. Prikazani su statistički podatci o kibernetičkim napadima, koji jasno prikazuju ekonomski utjecaj istih. Također je definiran pojam nacionalnog indeksa kibernetičke sigurnosti, objašnjeno je što navedeni prikazuje te su prikazani primjeri za određene države. Putem definicije kibernetičkog zločina kao usluge, objašnjeno je zašto učestalost kibernetičkih napada neprestano raste te zašto se isti razvijaju značajnom brzinom. Za izabrane primjere kibernetičkih napada je detaljno objašnjeno koje ranjivosti su podloga za iste, što je prikazano provedbom i analizom posljedica napada u sigurnom laboratorijskom okruženju. Izabrani su neki od najučestalijih primjera kibernetičkih napada, te su demonstrirani na sličan način kako bi ih proveli kibernetički kriminalci. Za provedbu napada korištene su razne strategije i tehnike te alati za automatizaciju u svrhu što točnije i kvalitetnije demonstracije koja odražava način na koji kibernetički kriminalci pokušavaju iskoristiti ranjivosti u računalnim sustavima. Za svaki napad su detaljno objašnjene protumjere koje je potrebno poduzeti za uspješnu obranu od istoga. Također su opisana i objašnjena generalna rješenja i strategije koje uvelike doprinose uspješnoj obrani. Primjenom protumjera navedenih u ovom diplomskom radu moguće je značajno podignuti razinu sigurnosti računalnog sustava. Sve objašnjene protumjere predstavljaju učinkovita rješenja koja se mogu jednostavno implementirati u svakodnevne zadatke.

Na temelju prikazanih primjera kibernetičkih napada, jasno je zašto isti mogu imati značajne posljedice ukoliko se određene ranjivosti ne otkriju na vrijeme. Relevantnost razvoja kibernetičke sigurnosti je neporeciva, kako na razini poduzeća, tako i na globalnoj razini. Ključni korak u obrani od kibernetičkih napada je kvalitetna edukacija o važnosti primjene sigurnosnih strategija u svakodnevnim zadatcima. Kibernetički kriminalci neprestano razvijaju nove tehnike i strategije te pronalaze nove ranjivosti za potencijalni kibernetički napad. Zbog toga je nužno održavati korak s najnovijim tehnologijama i rješenjima za zaštitu računalnih sustava.

LITERATURA

[1] K. Sehgal, N. Thzmianis, Cybersecurity Blue Team Strategies: Uncover the secrets of blue teams to combat cyber threats in your organization, Packt Publishing, 2023.

[2] N. Lane, W. A. Conklin, G. White, D. Williams, CASP+ CompTIA Advanced Security Practitioner Certification All-in-One Exam Guide, Second Edition (Exam CAS-003), McGraw Hill, 2019.

[3] Morris Worm, dostupno na: https://www.fbi.gov/history/famous-cases/morris-worm [15.4.2023.]

[4] Intel Free Press, Disk with Morris Worm source code, dostupno na: https://www.flickr.com/photos/intelfreepress/10483246033/in/photostream [15.4.2023.]

[5] P. J. Denning, The Science of Computing: The Internet Worm, American Scientist, No. 2, Vol. 77, pp. 126 – 28, 1989., dostupno na: https://www.jstor.org/stable/27855650 [15.4.2023.]

[6] The Most Famous Virus History: Melissa, dostupno na: https://www.pandasecurity.com/en/mediacenter/src/uploads/2013/10/virus_melisa.jpg [20.4.2023.]

[7] Melissa Virus, dostupno na: https://www.fbi.gov/history/famous-cases/melissa-virus [20.4.2023.]

[8] K. A. Rhodes, Information Security: The Melissa Computer Virus Demonstrates Urgent Need for Stronger Protection Over Systems and Sensitive Data, United States General Accounting Office, 1999., dostupno na: https://www.gao.gov/assets/t-aimd-99-146.pdf [22.4.2023.]

[9] Joint Task Force Transformation Initiative, Guide for Conducting Risk Assessments, National Institute of Standards and Technology, 2012., dostupno na: https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-30r1.pdf [22.4.2023.]

[10] E. D. Knapp, J. Langill, Industrial Network Security: Securing Critical Infrastructure Networks for Smart Grid, SCADA, and Other Industrial Control Systems, Syngress Publishing, Inc., 2014.

[11] SCADA systems explained , dostupno na: https://scada-international.com/what-is-scada [22.4.2023.]

[12] F. Skopik, P. Smith, Smart Grid Security: Innovative Solutions for a Modernized Grid, Syngress Publishing, Inc., 2015.

[13] E. Byres, A. Ginter, J. Langill, How Stuxnet Spreads – A Study of Infection Paths in Best Practice Systems, Tofino Security, Abterra Technologies, ScadaHacker, 2011., dostupno

na: https://scadahacker.com/library/Documents/Cyber_Events/Tofino%20%20How%20Stuxnet%20Spreads%20v1.0.pdf [23.4.2023.]

[14] Cybersecurity Ventures, 2022 Official Cybercrime Report, dostupno na: https://s3.ca-central-1.amazonaws.com/esentire-dot-com-assets/assets/resourcefiles/2022-Official-Cybercrime-Report.pdf [20.5.2023.]

[15] Check Point Research, 2023 Cyber Security Report, dostupno na: https://pages.checkpoint.com/cyber-security-report-2023.html [21.5.2023.]

[16] NCSI Methodology, dostupno na: https://ncsi.ega.ee/methodology/ [22.5.2023.]

[17] NCSI Index, dostupno na: https://ncsi.ega.ee/ncsi-index/ [22.5.2023.]

[18] NCSI Croatia, dostupno na: https://ncsi.ega.ee/country/hr/ [22.5.2023.]

[19] IBM Corporation, Cost of a Data Breach Report 2022, dostupno na: https://www.ibm.com/downloads/cas/3R8N1DZJ [23.5.2023.]

[20] D. Manky, Cybercrime as a service: a very modern business, Computer Fraud & Security, No. 6, Vol. 2013, pp. 9-13, 2013., dostupno na: https://www.sciencedirect.com/science/article/pii/S1361372313700538 [24.5.2023.]

[21] What is a DDoS botnet?, dostupno na: https://www.cloudflare.com/learning/ddos/what-is-a-ddos-botnet/ [24.5.2023.]

[22] R. Hertzog, J. O'Gorman, Kali Linux Revealed: Mastering the Penetration Testing Distribution, Offsec Press, 2017.

[23] OWASP Juice Shop, dostupno na: https://owasp.org/www-project-juice-shop/ [1.6.2023.]

[24] OWASP Top 10, dostupno na: https://owasp.org/Top10/ [1.6.2023.]

[25] Juice-shop, dostupno na: https://github.com/juice-shop/juice-shop [1.6.2023.]

[26] What Is SQL (Structured Query Language)?, dostupno na: https://aws.amazon.com/what-is/sql/ [2.6.2023.]

[27] J. Clarke-Salt, SQL Injection Attacks and Defense, Syngress Publishing, Inc., 2012.

[28] OWASP ZAP Getting Started, dostupno na: https://www.zaproxy.org/getting-started/ [3.6.2023]

[29] Sqlmap Introduction, dostupno na: https://sqlmap.org/ [4.6.2023.]

[30] SQL Injection Prevention Cheat Sheet, dostupno na: https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet [5.6.2023.]

[31] Cross Site Scripting (XSS), dostupno na: https://owasp.org/www-community/attacks/xss/ [7.6.2023.]

[32] B. B. Gupta, P. Chaudhary, Cross-Site Scripting Attacks: Classification, Attack, and Countermeasures (Security, Privacy, and Trust in Mobile Communications), CRC Press, 2020.

[33] Introduction to the DOM, dostupno na: https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction [7.6.2023.]

[34] Cross Site Scripting Prevention Cheat Sheet, dostupno na: https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet [11.6.2023.]

[35] HTML Named character references, dostupno na: https://html.spec.whatwg.org/multipage/named-characters [11.6.2023.]

[36] M. McDonald, Web Security for Developers: Real Threats, Practical Defense, No Starch Press, 2020.

[37] Content Security Policy Reference, dostupno na: https://content-security-policy.com [11.6.2023.]

[38] B. B. Gupta, A. Dahiya, Distributed Denial of Service (DDoS) Attacks, CRC Press, 2021.

[39] K. Singh, P. Singh, K. Kumar, Application layer HTTP-GET flood DDoS attacks: Research landscape and challenges, Computers & Security, Vol. 65, pp. 344 - 372, 2017., dostupno na: https://www.sciencedirect.com/science/article/abs/pii/S0167404816301365 [13.6.2023.]

[40] Radware, DDoS Handbook, 2015., dostupno na: https://www.radware.com/getattachment/Security/Research/702/Radware_DDoS_Handbook_2015.pdf.aspx/?lang=en-US [13.6.2023.]

[41] İ. Özçelik, R. Brooks, Distributed Denial of Service Attacks: Real-world Detection and Mitigation, CRC Press, 2020.

[42] Slowloris DDoS attack, dostupno na: https://www.cloudflare.com/learning/ddos/ddos-attack-tools/slowloris/ [13.6.2023.]

[43] D. K. Bhattacharyya, J. K. Kalita, DDoS Attacks: Evolution, Detection, Prevention, Reaction, and Tolerance, Chapman and Hall/CRC, 2016.

[44] What is an Intrusion Detection System?, dostupno na: https://www.paloaltonetworks.com/content/dam/pan/en_US/images/cyberpedia/ids-ips.png?imwidth=1920 [14.6.2023.]

[45] M. Okada, N. Goto, A. Kanaoka and E. Okamoto, A Device for Transparent Probabilistic Packet Marking, Journal of Information Processing, Vol. 22, pp. 242 - 247, 2013.

[46] S. Yi, Y. Xinyu, L. Ning, Q. Yong, Deterministic Packet Marking with Link Signatures for IP Traceback, Information Security and Cryptology, pp. 144 - 152, 2006.

[47] M. Jakobsson, S. Myers, Phishing and Countermeasures: Understanding the Increasing Problem of Electronic Identity Theft, Wiley-Interscience, 2006.

[48] Paypal Phishing Email Example, dostupno na: https://global-uploads.webflow.com/5e5ff4f0165cd367cc7ca88f/6009e9edba9c2ab78a6245d7_PayPal-01.png [15.6.2023.]

[49] G. Sonowal, Phishing and Communication Channels: A Guide to Identifying and Mitigating Phishing Attacks, Apress Media, LLC, 2021.

[50] S. Chanti, T. Chithralekha, Classification of Anti-phishing Solutions, SN Computer Science, No. 1, Vol. 1, 2019., dostupno na: https://doi.org/10.1007/s42979-019-0011-2 [19.6.2023.]

[51] M. Alsharnouby, F. Alaca, S. Chiasson, Why phishing still works: User strategies for combating phishing attacks, International Journal of Human-Computer Studies, Vol. 82, pp. 69 - 82, 2015.

[52] Gophish User Guide Generating Reports, dostupno na: https://docs.getgophish.com/user-guide/documentation/generating-reports [19.6.2023.]

[53] What is a WAF? | Web Application Firewall explained, dostupno na: https://www.cloudflare.com/learning/ddos/glossary/web-application-firewall-waf/ [21.6.2023.]

[54] What is a next-generation firewall (NGFW)?, dostupno na: https://www.cloudflare.com/learning/security/what-is-next-generation-firewall-ngfw/ [22.6.2023.]

[55] K. Martin, Everyday Cryptography: Fundamental Principles and Applications, Oxford University Press, 2021.

[56] Password Storage Cheat Sheet, dostupno na: https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html [23.6.2023.]

[57] What Is Password Salting & How It Improves Password Security?, dostupno na: https://websitesecuritystore.com/blog/what-is-password-salting/ [23.6.2023.]

[58] What is Multi-Factor Authentication (MFA) and How Does it Work?, dostupno na: https://www.onelogin.com/learn/what-is-mfa [24.6.2023.]

[59] What Is Multi-Factor Authentication (MFA)?, dostupno na: https://aws.amazon.com/what-is/mfa/ [24.6.2023.]

[60] What Is the Principle of Least Privilege?, dostupno na: https://www.paloaltonetworks.com/cyberpedia/what-is-the-principle-of-least-privilege [24.6.2023.]

[61] What is penetration testing? | What is pen testing?, dostupno na: https://www.cloudflare.com/learning/security/glossary/what-is-penetration-testing/ [24.6.2023.]

POPIS SLIKA

POPIS TABLICA

POPIS GRAFIKONA

SAŽETAK

Cilj je ovog rada bio provesti i analizirati neke primjere kibernetičkih napada te dati smjernice o mogućim protumjerama. Povijesni pregled kibernetičkih napada objašnjava utjecaj istih na društvo s ciljem poticanja razvoja kibernetičke sigurnosti. Statistički su podatci kibernetičkih napada vizualno prikazani i objašnjeni, a isti jasno prikazuju ekonomski učinak kibernetičkih napada. Simulacijom kibernetičkih napada u sigurnom laboratorijskom okruženju prikazan je način na koji kibernetički kriminalci mogu doći do osjetljivih informacija pojedinca ili organizacije. Svaki je napad detaljno opisan te su objašnjene ranjivosti koje otvaraju mogućnost za izvođenje istoga. Za svaki su napad detaljno opisane protumjere i strategije koje je potrebno poduzeti za uspješnu obranu od istoga. Također je objašnjen i učinak obrađenih protumjera. Na kraju su objašnjene opće protumjere koje predstavljaju učinkovitu obranu od većine kibernetičkih napada.

Ključne riječi: kibernetički napadi, protumjere, ranjivosti, sigurnost, strategije

ABSTRACT

This paper aimed to conduct and analyze some examples of cyber-attacks as well as to provide guidelines on possible countermeasures. A historical overview of cyber-attacks was used to explain the impact they have on the society and eventually foster cyber security. Statistical data on cyber-attacks were visually represented and elaborated on clearly demonstrating cyber-attacks' economic impact. Simulating cyber-attacks in a secure laboratory environment aimed to demonstrate how cyber criminals can access sensitive information of individuals or organizations. Each attack and underlying vulnerabilities were elaborated on. Countermeasures and strategies necessary for the successful defence against each attack were thoroughly described. The effectiveness of the implemented countermeasures was also explained. Finally, general countermeasures, which provide effective defence against most cyber-attacks, were outlined.

Keywords: countermeasures, cyber-attacks, security, strategies, vulnerabilities

ŽIVOTOPIS

Mihovil Šimić rođen je 27. prosinca 1999. godine u Pakracu. Nakon završetka opće gimnazije u Srednjoj školi Pakrac, 2018. godine upisuje preddiplomski sveučilišni studij Računarstvo na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija Osijek. Isti završava u srpnju 2021. godine, te iste godine nastavlja obrazovanje na diplomskom sveučilišnom studiju Računarstvo, smjer Informacijske i podatkovne znanosti.