Parametrizacija napadačkog prometa i primjena u zadanim topologijama

Pagon, Vilim

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 657

PARAMETRIZACIJA NAPADAČKOG PROMETA I PRIMJENA U ZADANIM TOPOLOGIJAMA

Vilim Pagon

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 657

PARAMETRIZACIJA NAPADAČKOG PROMETA I PRIMJENA U ZADANIM TOPOLOGIJAMA

Vilim Pagon

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 4. ožujka 2024.

DIPLOMSKI ZADATAK br. 657

Pristupnik:	**Vilim Pagon (0036523693)**
Studij:	Računarstvo
Profil:	Računarska znanost
Mentor:	izv. prof. dr. sc. Stjepan Groš

Zadatak: **Parametrizacija napadačkog prometa i primjena u zadanim topologijama**

Opis zadatka:

Značajan izazov u treniranju algoritama strojnog učenja su podaci. Većina prometa na mrežama nije maliciozna, a i kada je, teško je znati koji dio je maliciozan zbog velike količine podataka. Problem nedostatka malicioznog prometa prisutan je i u slučaju kada se želi obučavati ljude da detektiraju napade u mreži. Način na koji se to rješava je da se snima promet tijekom provođenja penetracijskog ispitivanja ili se umjetno generira napadački promet. Nedostatak takvih pristupa je u tome što su specifični za točno određenu topologiju mreže te što brzo zastarjevaju. U diplomskom radu potrebno je pronaći javno raspoložive skupove podataka koji su snimani tijekom napada na računalnu mrežu. Iz tih skupova potrebno je izdvojiti promet koji je posljedica provođenja napada te ga parametrizirati na način da se može upravljati vremenom kada je promet generiran, njegovim intenzitetom, izvorištem i odredištem te svim ostalim parametrima koji su ovisni o topologiji nad kojom se provodi napad. Potom je s tako parametriziranim prometom potrebno generirati napadački promet za zadane topologije. Topologije su zadane u formi koju koristi simulator CCS.

Rok za predaju rada: 28. lipnja 2024.

Zahvaljujem se mentoru doc. dr. sc. Stjepanu Grošu, mag. ing. na stručnom vodstvu i pomoći pri izradi ovoga rada.

Posebno bih želio zahvaliti svim profesorima, asistentima i ostalom osoblju fakulteta koji su podijelili svoje znanje i trud te me uspješno pripremili za daljnju karijeru.

Također, želim se zahvaliti svojim prijateljima za predivna druženja, veselje te zajednička učenja i pripreme za ispite. Vaše društvo i podrška bili su mi od velike važnosti.

Na kraju, najveće zahvale upućujem svojoj obitelji: djevojci Carlli Brunni, mami Nadi, tati Sunčanu, sestri Kristini, baki Ljubici, baki Dragici i didi Anti. Hvala vam što ste uvijek bili uz mene kada sam trebao pomoć, na svakoj riječi hvale koju ste mi uputili te na vašoj neizmjernoj ljubavi.

Sadržaj

1. Uvod

Jedan od značajnih izazova u treniranju algoritama strojnog učenja su podaci. Većina mrežnog prometa nije maliciozna, a čak i kada jest, teško je identificirati koji dio prometa je maliciozan zbog velike količine podataka. Problem nedostatka malicioznog prometa prisutan je i kod obučavanja ljudi za detekciju mrežnih napada. Ovo se često rješava snimanjem prometa tijekom penetracijskih ispitivanja ili umjetnim generiranjem napadačkog prometa. Međutim, takvi pristupi imaju ograničenja jer su specifični za određenu topologiju mreže i brzo postaju zastarjeli.

Cilj ovog diplomskog rada je pronaći javno dostupne skupove podataka koji su snimani tijekom napada na računalnu mrežu. Iz tih skupova potrebno je izdvojiti promet koji je posljedica napada te ga parametrizirati tako da se može upravljati s vremenom generiranja prometa, njegovim intenzitetom, izvorištem i odredištem, kao i svim ostalim parametrima koji ovise o topologiji mreže. Na temelju tako parametriziranog prometa, potrebno je generirati napadački promet za zadane topologije. Topologije će biti zadane u formatu koji koristi simulator Cyber Conflict Simulator, CCS.

Ovaj rad će pridonijeti boljem razumijevanju i sposobnosti detekcije mrežnih napada kroz analizu i manipulaciju stvarnih podataka, omogućujući realističnije treninge i evaluaciju algoritama strojnog učenja.

Nakon kratkog uvoda, u drugom poglavlju prikazani su neki od javno dostupnih skupova podataka pronađenih na internetu. Skupovi podataka detaljno su opisani, uključujući načine na koje su generirani te karakteristike svakog skupa. U trećem poglavlju opisano je programsko rješenje koje je razvijeno te odabrana tehnika generiranja scenarija napada nad zadanom topologijom mreže, implementirano kao praktični dio rada. U četvrtom poglavlju detaljno je opisana tehnička strana razvijenog programa u sklopu praktičnog dijela diplomskog rada. Poglavlje započinje opisom tehnologija i alata korištenih za razvoj, a zatim nastavlja s opisom postupka generiranja scenarija napada. Prikazani su i objašnjeni dijelovi koda potrebni za detaljnije razumijevanje rada programa. Kako bi se cjelina zaokružila, prikazani su primjeri korištenja programa te njegovi izlazi, odnosno rezultati generiranja. Na kraju su prodiskutirana određena ograničenja i moguća potencijalna poboljšanja, kao i smjerovi za nastavak

istraživanja novih tehnika. Rad završava zaključkom kao zadnjim poglavljem, nakon čega slijedi pregled literature korištene u sklopu istraživanja i razvoja programa.

2. Javno dostupni skupovi podataka mrežnog prometa

Broj napada na računalne mreže povećava se godinama, a tehnike napada postaju sve sofisticiranije i teže za otkrivanje [1]. SOC (Security Operations Center) analitičari [2] su stručnjaci odgovorni za praćenje, analizu i odgovaranje na sigurnosne incidente unutar organizacije. Njihove ključne odgovornosti uključuju kontinuirano nadgledanje mrežnih aktivnosti, upravljanje sigurnosnim upozorenjima i suradnju s drugim timovima kako bi se osigurala sigurnost IT infrastrukture.

Trenutno ne postoji alat ili platforma namijenjena edukaciji SOC analitičara koja bi omogućila generiranje napada na temelju proizvoljnog scenarija i mrežne topologije za potrebe njihove obuke. Ovaj rad predstavlja prvi korak prema razvoju takvog alata. Ideja alata je da, na temelju zadane topologije Cyber Conflict Simulatora i detaljno opisanog scenarija napada po fazama i koracima, ubaci maliciozni mrežni promet među normalan promet te mreže. Tako SOC analitičari mogu učiti otkrivati razne napade i pripremati se za detekciju stvarnih napada.

Cyber Conflict Simulator (CCS) je interaktivni simulator dizajniran za timove za odgovor na kibernetičke incidente u civilnom i vojnom sektoru, kako bi im pomogao da se pripreme za incidente. CCS je prepoznat od strane Europske obrambene agencije (EDA) kao inovativan projekt za dualnu (vojnu i civilnu) upotrebu i odabran za tehničku podršku.

U kontekstu računalnih mreža i ovog rada, normalan promet odnosi se na mrežni promet koji predstavlja standardne, svakodnevne aktivnosti i komunikacije unutar mreže. To uključuje sve legitimne i očekivane podatkovne razmjene između uređaja, servisa i korisnika koji koriste mrežu za uobičajene svrhe.

Većina prometa snimljenog na računalnim mrežama nije maliciozna, međutim, kada se maliciozni promet pojavi, često je teško identificirati koji dio prometa predstavlja prijetnju zbog velike količine podataka koja se prenosi. U domeni mrežne sigurnosti, problem nedostatka malicioznog prometa, odnosno manjka jasno označenog malicioznog mrežnog prometa, predstavlja izazov i u scenarijima gdje se provodi obučavanje stručnjaka za detekciju napada na računalnim mrežama. Jedan od načina za rješavanje ovog problema je snimanje prometa tijekom provođenja penetracijskih ispitivanja ili umjetno generiranje napadačkog

prometa. Nedostaci ovih pristupa su što su specifični za točno određenu topologiju mreže te brzo postaju zastarjeli.

2.1 Korišteni skupovi podataka

Jedan od ciljeva ovog diplomskog rada je pronaći javno raspoložive skupove podataka koji su snimani tijekom napada na računalne mreže. Iz pronađenih skupova potrebno je izdvojiti promet koji je posljedica provođenja napada te ga parametrizirati tako da se može upravljati s vremenom kada je promet generiran, njegovim intenzitetom, izvorištem i odredištem te svim ostalim parametrima koji ovise o topologiji nad kojom se provodi napad.

U sklopu rada analizirani su javno dostupni skupovi podataka kako bi se utvrdilo jesu li prikladni za korištenje. U kontekstu analize skupova podataka povezanih s mrežnim napadima, ključne značajke koje su jasno označeni paketi koji se mogu pripisati određenom tipu napada, kategorizacija pojedinih napada radi precizne klasifikacije, dostupnost topologije mreže na kojoj su napadi izvršeni te popis ekstrahiranih značajki iz tih skupova podataka koje omogućuju daljnju analizu i otkrivanje uzoraka. Relevantni pronađeni skupovi podataka te njihovi detaljni opisi prikazani su u nastavku rada.

2.1.1 The UNSW-NB15

UNSW-NB15 [3] predstavlja sintetički skup podataka mrežnog prometa koji uključuje kombinaciju stvarnih normalnih aktivnosti i suvremenih napadačkih aktivnosti generiranih s pomoću IXIA PerfectStorm alata [4] u okvirima Cyber Range Laba na Sveučilištu Novog Južnog Walesa u Canberri [5]. Navedeni skup podataka osmišljen je u svrhu evaluacije sustava za otkrivanje napada te drugih alata namijenjenih analizi mrežnog prometa.

Stvaranje skupa podataka započelo je konfiguracijom IXIA alata s tri virtualna poslužitelja - dva za generiranje normalnog prometa, a jedan za generiranje zlonamjernih aktivnosti. Mreža je također uključivala dvije virtualne mrežne priključne točke (engl. *network taps*), dva usmjerivača i vatrozid kojim se propušta sav promet. Tcpdump alat instaliran je na jednom od usmjerivača kako bi se snimilo ukupno 100 GB neobrađenih podataka (engl. *raw*) `PCAP`

datoteka mrežnog prometa tijekom različitih simulacija. Napadačke aktivnosti generirane su IXIA alatom na temelju realnih prijetnji prikupljenih s CVE (Common Vulnerabilities and Exposures) web stranice [6] kako bi se postigla realna reprezentacija modernih prijetnji.

Za ekstrakciju značajki iz sirovih `PCAP` datoteka korišteni su Argus [7] i Bro-IDS [8] mrežni analizatori te algoritmi u C# programskom jeziku. Ukupno je ekstrahirano 49 različitih značajki podijeljenih u tri skupine: osnovne, sadržajne i vremenske značajke. Osnovne značajke uključuju IP adrese, portove i protokole. Sadržajne značajke opisuju sadržaj mrežnih paketa poput veličine, zastavica i drugih podataka iz zaglavlja. Vremenske značajke prate vremensku domenu prometa poput trajanja veze, međudolaznih vremena paketa itd.

Osim navedenih značajki, dodano je 12 izvedenih značajki koje opisuju opće i vezne karakteristike mrežnog prometa. Ove značajke dizajnirane su za otkrivanje napada koji skeniraju mreže nasumično ili povremeno.

Skup podataka je podijeljen u četiri CSV datoteke koje sadržavaju ukupno 2,540,044 zapisa. Svaki zapis ima sve značajke i oznaku koja označava je li riječ o normalnom ili napadačkom prometu. Zasebno su dostupne dvije datoteke - jedna sa stvarnom istinom (engl. *ground truth*) podacima koja mapira zapise na stvarne kategorije napada, te druga s popisom svih događaja.

Za praktične svrhe poput treniranja modela strojnog učenja, dio zapisa iz skupova podataka je odvojen u posebne datoteke za trening (175,341 zapis) i testiranje (82,332 zapisa).

UNSW-NB15 smatra se jednim od najkvalitetnijih i najaktualnijih javno dostupnih skupova podataka mrežnog prometa. Glavni razlog je njegova veličina, raznolikost napadačkih aktivnosti, bogatstva značajki i načina na koji je stvoren kako bi realno opisivao modernu prijetnju krajnje kompleksnih napada.

Jedan od ključnih čimbenika koji UNSW-NB15 skup podataka čini iznimno vrijednim resursom jest činjenica da obuhvaća širok raspon zlonamjernih aktivnosti i napadačkih tehnika. Skup podataka sadrži uzorke mrežnog prometa za čak devet kategorija napada, uključujući Fuzzers, Analysis, Backdoors, DoS (Distributed Denial of Service), Exploits, Generic, Reconnaissance, Shellcode i Worms. Ovako raznovrsna pokrivenost različitih vrsta prijetnji omogućava sveobuhvatno testiranje i evaluaciju sustava za detekciju napada u realističnim scenarijima koji odražavaju kompleksnost i raznoliku prirodu modernih

kibernetičkih prijetnji. Bogatstvo različitih uzoraka napada u skupu podataka pruža vrijednu osnovu za generiranje širokog spektra mogućih scenarija napada.

2.1.2 CSE-CIC-IDS2018

CSE-CIC-IDS2018 [9] je skup podataka mrežnog prometa stvoren kroz suradnju Communications Security Establishment (CSE) [10] i Canadian Institute for Cybersecurity (CIC) [11]. Glavna svrha skupa podataka je omogućiti testiranje i evaluaciju sustava za detekciju napada usmjerenih na anomalije u mrežnom prometu.

Skup podataka sadrži snimke mrežnog prometa i logove sustava za sedam različitih scenarija napada - Brute-force, Heartbleed, Botnet, DoS, DDoS, Web napadi i infiltracija mreže iznutra. Napadačka infrastruktura sastojala se od 50 računala, dok je žrtva organizacija imala 5 odjela s 420 računala i 30 poslužitelja.

Stvaranje skupa podataka započelo je definiranjem „profila“ - B-profili opisuju normalno ponašanje različitih mrežnih protokola koristeći tehnike strojnog učenja, dok M-profili opisuju scenarije napada koje izvode ljudi ili automatizirani agenti.

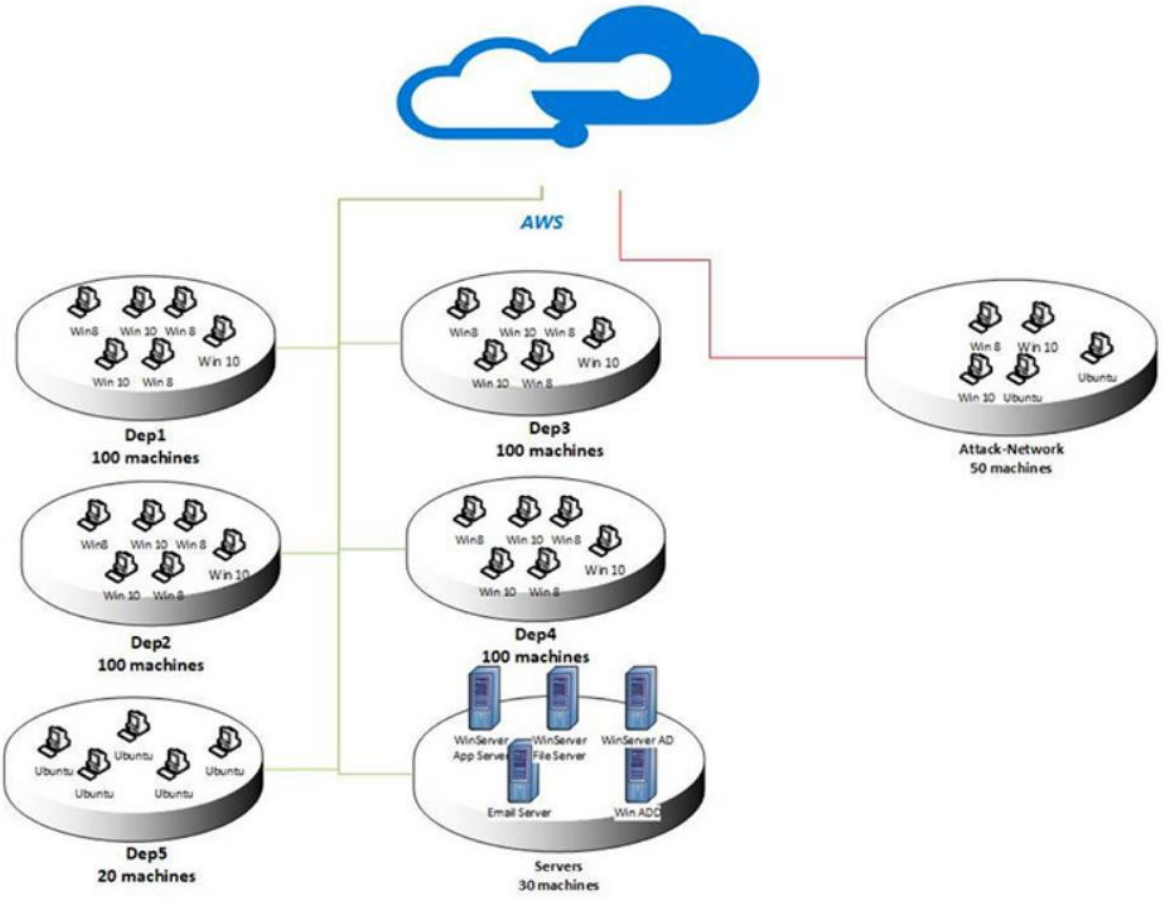
Implementirana je testna mreža s različitim Windows i Linux strojevima raspoređenima u odgovarajuće odjele i poslužiteljske sobe kao što je prikazano na **Slika 1**. Zatim su se izvršavali razni napadi generirani prema definiranim M-profilima, koristeći alate poput Patator, Heartleech, Ares botneta, Slowloris, LOIC, Selenium okvira itd.

Sniman je neobrađeni (engl. *raw*) mrežni promet u obliku `PCAP` datoteka i logovi sustava svakog stroja tijekom napada. Iz sirovih podataka izvučeno je preko 80 značajki prometa s pomoću CICFlowMeter alata [12], uključujući statističke značajke poput trajanja toka, veličina paketa, vremena između paketa, TCP/IP zastavica itd.

Konačni skup podataka organiziran je po danima, sirovima podacima i izvučenim CSV datotekama po danu/stroju. Zapisi su označeni na temelju IP adresa, portova, protokola i rasporeda provedenih napada.

Skup podataka je objavljen za širu upotrebu u analizi mrežnog prometa i evaluaciji IDS/IPS sustava, uz uvjet citiranja izvora pri redistribuciji ili korištenju. Karakterizira ga raznolikost

napada, realistična topologija mreže te ažurirani trendovi u vrstama napada u usporedbi s postojećim javnim skupom.



***Slika 1**: Mrežna topologija*

2.1.3 Kitsune Network Attack

Kitsune Network Attack skup podataka predstavlja iznimno vrijedan resurs za istraživanje i evaluaciju sustava za detekciju i prevenciju mrežnih napada. Ovaj skup podataka sadrži snimke stvarnog mrežnog prometa iz komercijalnih sustava videonadzora i mreža IoT (Internet of Things) uređaja. Prikupljeno je ukupno devet različitih skupova podataka, pri čemu svaki sadrži milijune mrežnih paketa i različite vrste kibernetičkih napada.

Za svaki napad, skup podataka pruža tri različita oblika podataka:

1. Pretprocesiranu CSV datoteku spremnu za strojno učenje, s označenim vektorima za svaki mrežni paket.
2. Odgovarajuću vektorsku datoteku oznaka napada u CSV formatu.
3. Izvornu snimku mrežnog prometa u PCAP formatu, za one koji žele samostalno izvući značajke.

Skup podataka obuhvaća širok raspon kibernetičkih napada koji se mogu pojaviti u stvarnim mrežnim intruzijama, uključujući skeniranje operacijskih sustava, napadne prijetnje, napade na video streamove, pasivne i aktivne mrežne prisluškivače, napade uskraćivanjem usluge, napade ponovnim pregovaranjem SSL-a i napade botnet zlonamjernim softverom.

Podaci su prikupljeni u kontroliranom okruženju koje odražava stvarne mrežne topologije s komercijalnim IP kamerama, usmjernicima (engl. *routers*), preklopnicima (engl. *switches*), DVR-ovima i različitim vrstama mrežnih uređaja. Napadači su simulirani iz različitih točaka u mreži, s različitim vektorima napada poput Etherneta, VPN kanala i bežičnih pristupnih točaka. Snimke mrežnog prometa prikupljene su na kritičnim točkama gdje bi sustav za detekciju i prevenciju upada mogao biti implementiran.

Predobrađene CSV datoteke sadrže 115 značajki za svaki mrežni paket, ekstrahiranih korištenjem AfterImage ekstraktora značajki. Ove značajke pružaju statističku snimku mreže, uključujući informacije o hostovima i ponašanjima u kontekstu trenutnog paketa. AfterImage ekstraktor jedinstveno je dizajniran za učinkovitu obradu milijuna mrežnih tokova u stvarnom vremenu, što ga čini pogodnim za analizu velikih količina mrežnog prometa.

Ovaj skup podataka predstavlja vrijedan resurs za razvoj i evaluaciju naprednih algoritama strojnog učenja i duboke analize za otkrivanje kibernetičkih napada u stvarnom vremenu. Prisutnost različitih vrsta napada, stvarnih mrežnih topologija i velikih količina podataka omogućava sveobuhvatno testiranje i usporedbu različitih pristupa detekciji napada. Uz to, pristup izvornim PCAP datotekama omogućava istraživačima da eksperimentiraju s vlastitim tehnikama ekstrakcije značajki i predobrade podataka.

Unatoč brojnim prednostima koje Kitsune Network Attack skup podataka pruža, ovaj skup podataka ima određena ograničenja koja ograničavaju njegovu primjenu u nekim scenarijima. Jedno od ključnih ograničenja jest nedostatak preciznog označavanja (engl. *labeling*) mrežnih

paketa. Iako skup podataka sadrži različite vrste napada, postoji mogućnost da neki paketi nisu ispravno klasificirani ili označeni, što može dovesti do pogrešnih zaključaka i smanjene učinkovitosti generiranja ispravnog napada. Nadalje, skup podataka obuhvaća relativno uzak raspon vrsta napada u usporedbi s raznolikošću prijetnji koje se mogu pojaviti u stvarnim mrežnim okruženjima, što dodatno ograničava generalizaciju i primjenjivost rezultata dobivenih korištenjem ovog skupa podataka.

2.1.3 Ručno snimljeni skupovi podataka

U sklopu ovog rada korištena su tri različita izvora mrežnog prometa za potrebe generiranja napadačkog prometa temeljenog na scenarijima i topologijama računalnih mreža. Prvi izvor podataka je javno dostupan UNSW-NB15 skup podataka koji sadrži snimljeni mrežni promet kombiniran s normalnim aktivnostima i sintetiziranim napadima generiranim IXIA PerfectStorm alatom. Drugi izvor je CSE-CIC-IDS2018 skup podataka stvoren suradnjom Communications Security Establishmenta (CSE) i Canadian Institute for Cybersecurity (CIC). Ovaj skup podataka sniman je u testnoj računalnoj mreži s različitim Windows i Linux strojevima na kojima su se provodili napadi definirani kroz M-profile (scenarije napada).

Međutim, sva tri skupa podataka, iako recentna i kvalitetna, ograničena su specifičnom mrežnom topologijom te vrstama i redoslijedom napadačkih aktivnosti koje opisuju. Stoga je kao treći izvor podataka prikupljan mrežni promet sniman u kontroliranom okruženju s dvije virtualne mašine - napadačkom Kali Linux mašinom i metom Metasploitable 2 mašinom [13]. Na ovaj način bilo je moguće snimiti realni mrežni promet različitih faza tipičnog napadačkog scenarija kao što su izviđanje (engl. *reconnaissance*), inicijalni napad (engl. *initial compromise*), uspostava trajnog pristupa (engl. *establishing persistence*), lateralno kretanje (engl.*lateral movement*), izvlačenje podataka (engl. *data exfiltration*) te održavanje pristupa (engl. *maintaining access*).

Za svaku od ovih faza snimani su odgovarajući napadi izvedeni s pomoću alata poput Nmap, Metasploit frameworka, Hydra, Nessus, Nikto, MSFvenom, i slično. Mrežni promet tijekom napada sniman je korištenjem Wireshark alata [14], jednog od najpoznatijih mrežnih

analizatora. Snimljeni podaci spremani su u `PCAP` ili `PCAPNG` formatima koji su standardni za pohranu sirovih mrežnih paketa.

Sakupljeni skupovi podataka iz sva tri izvora poslužili su kao ulazni podaci za razvijeni sustav generiranja napadačkog mrežnog prometa. Kombinacijom ovih raznolikih izvora mrežnog prometa bilo je moguće obuhvatiti velik raspon vrsta napada i napadačkih procedura. Generirani napadački promet zatim se mogao koristiti u svrhu generiranje mrežnih napada razvijenog programa.

3. Scenariji napada

Scenariji napada su strukturirani planovi koji definiraju korake i tehnike koje napadač koristi kako bi kompromitirao, upravljao i eksfiltrirao podatke iz ciljanih sustava. U kontekstu simulacija kibernetičke sigurnosti, scenariji napada omogućuju stvaranje realističnih uvjeta za analiziranje sigurnosnih mehanizama i sposobnosti obrane. Scenariji se sastoje od više faza, a svaka faza ima specifične ciljeve, alate i taktike.

Datoteka scenarija strukturirana je tako da omogućuje definiranje globalnih parametara koji se primjenjuju na sve faze napada i pojedinačne napade unutar svake faze. Također je moguće definirati i lokalne parametre koji vrijede na razini pojedine faze ili na razini pojedinog napada. Pri definiranju parametara koji se odnose na specifičnu fazu ili konkretan napad, lokalni parametri imaju prednost nad globalnim parametrima te ih nadjačavaju i zanemaruju ako postoje.

3.1 Format scenarija

Definicija scenarija napada uključuje hijerarhijsku strukturu koja sadrži parametre napada, faze napada i taktike, odnosno napade korištene unutar svake faze. Evo detaljnog opisa formata definicije scenarija. Napad se sastoji od 6 faza:

1. Faza prikupljanja obavještajnih podataka (*reconnaissance*): Ova faza uključuje prikupljanje informacija o ciljanoj mreži i sustavima kako bi se identificirale potencijalne ranjivosti i točke upada. Napadač može koristiti različite taktike poput skeniranja mreže, enumeracije usluga i skeniranja ranjivosti.
2. Faza početnog kompromitiranja (*initial compromise*): Ova faza je usmjerena na iskorištavanje otkrivenih ranjivosti na sustavima mete kako bi se uspostavio početni pristup i stvorio oslonac unutar mreže. Napadač može iskoristiti različite taktike poput eksploatacije poznatih ranjivosti ili *backdoor-a*.
3. Faza uspostavljanja perzistencije (*establishing persistence*): Nakon što je uspostavljen početni pristup, ova faza uključuje taktike za održavanje trajnog pristupa kompromitiranim sustavima tijekom dužeg perioda. Primjeri taktika uključuju

stvaranje novog korisničkog računa s administratorskim ovlastima ili zamjenu legitimnih servisa kompromitiranim verzijama.

4. Faza lateralnog kretanja (*lateral movement*): Ova faza se fokusira na širenje napada na dodatne sustave unutar mreže iskorištavanjem stečenih ovlasti i informacija. Napadač može koristiti taktike poput napada nasilnim metodama (engl. *brute-force*), prolaženja kroz kriptografski sažetak vrijednosti (engl. *pass-the-hash*) ili daljnjeg skeniranja mreže.
5. Faza izvlačenja podataka (*data exfiltration*): Nakon što je uspostavljen pristup na više sustava, ova faza uključuje izvlačenje osjetljivih podataka s kompromitiranih sustava do napadačeve kontrolne točke. Napadač može koristiti različite metode poput prijenosa datoteka ili uspostavljanja povratne ljuske (engl. *reverse shell*).
6. Faza održavanja pristupa (*maintaining access*): Konačna faza osigurava da napadač zadrži pristup kompromitiranim sustavima za buduću eksploataciju. Ovo može uključivati postavljanje zadataka za periodično otvaranje povratne ljuske ili učitavanje zlonamjernog kernel modula za održavanje *rootkita*.

Svaka faza sadrži opće informacije i specifične taktike koje napadač može poduzeti, a pojedine taktike mogu uključivati i alate koji se koriste za njihovo izvršavanje. Ovaj hijerarhijski scenarij predstavlja strukturirani pristup napadaču, omogućujući mu sistematično napredovanje kroz različite faze napada.

Datoteka scenarija strukturirana je tako da omogućuje definiranje globalnih parametara koji se primjenjuju na sve faze napada i pojedinačne napade unutar svake faze. Primjer globalnih parametara prikazan je u **Izvorni kod 1**. Također je moguće definirati i lokalne parametre koji vrijede na razini pojedine faze ili na razini pojedinog napada. Pri definiranju parametara koji se odnose na specifičnu fazu ili konkretan napad, lokalni parametri imaju prednost nad globalnim parametrima te ih nadjačavaju i zanemaruju ako postoje.

Globalni parametri definiraju se na početku datoteke scenarija i vrijede za sve faze napada i pojedinačne napade unutar tih faza. Primjeri globalnih parametara uključuju IP adrese napadača i žrtve, MAC adrese te vrijeme početka napada.

```
attack_model:
   parameters:
      Attacker IP: 192.168.1.100
      Victim IP: 192.168.1.200
      Attacker MAC: 00:0a:95:9d:68:16
      Victim MAC: 00:0a:95:9d:68:17
```

***Izvorni kod 1**: Primjer globalnih parametara*

U ovom primjeru, globalni parametri uključuju IP adrese i MAC adrese napadača i žrtve, kao i vrijeme početka napada. Ovi parametri vrijede za sve faze i napade definirane u scenariju.

Lokalni parametri definiraju se unutar specifičnih faza ili unutar pojedinačnih napada unutar tih faza kao što se može vidjeti u **Izvorni kod 2**. Ovi parametri imaju prednost nad globalnim parametrima i koriste se za prilagodbu specifičnih faza ili napada. Lokalni parametri mogu uključivati specifične alate, taktike, trajanje napada i intenzitet napada.

```
reconnaissance:
   general_info:
      description: This phase involves gathering
       information about the target network and systems to
       identify potential vulnerabilities and entry points
       for exploitation.
      Attacker IP: 192.168.1.100
      Victim IP: 192.168.1.200
      Attacker MAC: 00:0a:95:9d:68:16
      Victim MAC: 00:0a:95:9d:68:17
```

***Izvorni kod 2**: Primjer lokalnih parametara na razini faze*

U fazi `reconnaissance`, lokalni parametri poput IP adresa i MAC adresa mogu biti isti kao globalni parametri, ali u slučaju da su definirani različiti, lokalni parametri će imati prednost. Primjer takvih parametara je dan u **Izvorni kod 3**.

```
tactics:
   - name: Network Scanning
     static: False
     duration: 2
     intensity: 1
     tool: Nmap
     tactic: Use tool like Nmap to discover open ports and
      services running on the network.
```

***Izvorni kod 3**: Primjer lokalnih parametara na razini pojedinačnog napada*

U ovom primjeru, parametri kao što su trajanje, intenzitet i alat (Nmap) definirani su za specifičnu taktiku unutar faze `reconnaissance`. Ovi parametri nadjačavaju globalne parametre ako se odnose na isti aspekt napada.

Obavezni parametri koji moraju biti definirani u datoteci scenarija uključuju:

- `Attacker IP`: IP adresa napadača,
- `Victim IP`: IP adresa žrtve,
- `Attacker MAC`: MAC adresa napadača,
- `Victim MAC`: MAC adresa žrtve,
- `start_attack`: Vrijeme početka napada (u UNIX timestamp formatu).

Ovi parametri su ključni za postavljanje osnovnih elemenata scenarija napada i omogućuju pravilno funkcioniranje simulacije. Lokalni parametri poput specifičnih alata, trajanja i intenziteta napada nisu obavezni, ali su korisni za prilagodbu i precizno definiranje faza i taktika unutar scenarija.

Kroz pravilno definiranje globalnih i lokalnih parametara, scenariji napada mogu biti detaljno prilagođeni kako bi se stvorili što realističniji uvjeti za testiranje i evaluaciju sigurnosnih mjera.

3.2 Primjer scenarija

U ovom scenariju, napadač slijedi strukturirani pristup napadu podijeljenom u više faza, koristeći niz tehnika i alata za postizanje svojih ciljeva. Scenarij započinje fazom prikupljanja obavještajnih podataka (engl. *reconnaissance*) u kojoj napadač koristi alat Nmap za skeniranje mreže i identifikaciju otvorenih portova te verzija usluga koje se izvode na tim portovima. Zatim se koriste alati poput Nessusa ili OpenVAS-a za skeniranje poznatih ranjivosti na identificiranim uslugama.

Sljedeća faza je faza početnog kompromitiranja (engl. *initial compromise*) gdje napadač iskorištava otkrivene ranjivosti poput *backdoor-a* u VSFTPD 2.3.4 i UnrealIRCd-u koristeći Metasploit Framework. Ovo mu omogućava uspostavljanje početnog pristupa i oslonca unutar mreže.

Nakon što je uspostavljen pristup, napadač prelazi u fazu uspostavljanja perzistencije (engl. *establishing persistence*) gdje koristi naredbe poput `useradd` ili `usermod` za stvaranje novog korisničkog računa s administratorskim ovlastima. Također koristi alat msfvenom za zamjenu SSH daemona kompromitiranom *backdoor* verzijom, osiguravajući trajni pristup sustavu.

U fazi lateralnog kretanja (engl. *lateral movement*), napadač koristi alat Hydra za nasilno pogađanje SSH lozinki na drugim sustavima u mreži. Također ekstrahira hash vrijednosti s kompromitiranih sustava koristeći alat samdump2 i koristi ih za autentikaciju na drugim strojevima (engl. *pass-the-hash* napad). Napadač dalje skenira mrežu alatima poput Nmap-a i Metasploit Frameworka kako bi identificirao dodatne ranjive sustave i usluge.

U fazi izvlačenja podataka (engl. *data exfiltration*), napadač koristi SCP za kopiranje osjetljivih datoteka s kompromitiranih sustava na svoj sustav. Također uspostavlja povratnu ljusku (engl. reverse shell) koristeći alat nc za kontinuirani prijenos podataka.

Konačno, u fazi održavanja pristupa (engl. maintaining access), napadač postavlja cron zadatke koji periodično otvaraju povratnu ljusku. Također učitava zlonamjerni kernel modul koristeći naredbu insmod za uspostavljanje rootkit-a i kontinuirano održavanje pristupa.

Kroz ove faze, napadač iskorištava različite ranjivosti i koristi niz alata za skeniranje, eksploataciju, lateralno kretanje, izvlačenje podataka i održavanje pristupa unutar kompromitirane mreže. Svaka faza slijedi logičan slijed aktivnosti koji napadaču omogućuje sustavno napredovanje i proširenje opsega napada.

3.2.1 Objašnjenja korištenih napada

„Eksploatacija backdoor-a u VSFTPD 2.3.4“ sadrži kritičnu ranjivost koja omogućava udaljenom napadaču izvršavanje proizvoljnog koda na sustavu. Ova ranjivost je *backdoor* dizajniran za udaljeni pristup. Napadač koristi Metasploit Framework, popularnu platformu za eksploataciju ranjivosti, kako bi iskoristio ovaj *backdoor*. Metasploit sadrži modul koji cilja ovu specifičnu ranjivost i nakon uspješne eksploatacije, napadaču dodjeljuje sesiju udaljenog pristupa s ljuskom na kompromitiranom sustavu.

„Eksploatacija backdoor-a u UnrealIRCd“ odnosi se na kritičnu ranjivost u popularnom IRC poslužitelju koja je dozvoljavala udaljenom napadaču preuzimanje kontrole nad sustavom. Ova ranjivost je također implementirana kao *backdoor*. Slično kao i kod VSFTPD-a, napadač koristi Metasploit Framework sa specifičnim modulom za ovu ranjivost kako bi je eksploatirao i dobio udaljenu ljusku na kompromitiranom sustavu.

Nakon što je uspostavio pristup sustavu, napadač koristi naredbe `useradd` ili `usermod` na Unix sustavima kako bi stvorio novog korisnika s administratorskim (engl. *root*) ovlastima. Ovo mu omogućava trajni pristup sustavu čak i ako se inicijalna ranjivost zakrpa.

Napadač koristi alat msfvenom iz Metasploit Frameworka za generiranje zlonamjernog *payload-a* - „backdoor verzije SSH daemona“. Zatim zamjenjuje legitimni SSH *daemon* sa svojim kompromitiranim *backdoorom*, čime si osigurava trajni pristup sustavu kroz ovaj kompromitiran SSH servis.

Napadač koristi alat Hydra, popularni alat za nasilno pogađanje lozinki, kako bi pokušao pogoditi SSH lozinke na drugim sustavima u mreži. Hydra podržava paralelno izvršavanje velikog broja pokušaja autentikacije s liste mogućih lozinki.

Umjesto pokušaja pogađanja same lozinke, napadač ekstrahira NTLM/LM kriptografske sažetke vrijednosti lozinki s kompromitiranih sustava koristeći alat samdump2. Ove hash vrijednosti se mogu koristiti za autentikaciju na drugim sustavima bez potrebe za poznavanjem stvarne lozinke. To je poznato kao „pass-the-hash“ tehnika.

SCP (Secure Copy) je mrežni protokol za siguran prijenos datoteka između lokalnog i udaljenog sustava. Napadač koristi SCP naredbu za kopiranje osjetljivih datoteka s kompromitiranog sustava na svoj lokalni napadački sustav.

Uspostavljanje povratne ljuske je tehnika kojom se uspostavlja interaktivna sesija udaljenog pristupa od žrtvinog sustava prema napadačevom sustavu. Napadač koristi alat nc (netcat) na žrtvinoj strani za uspostavljanje povratne veze prema svom sustavu. Ovo omogućava kontinuirani prijenos podataka bez potrebe za SCP-om.

Postavljanje zlonamjernog cron zadatka je tehnika kojom napadač konfigurira cron zadatke koji će periodično izvršavati naredbu za otvaranje povratne ljuske na kompromitiranom sustavu. Ovo omogućava napadaču trajni pristup čak i ako se inicijalna ranjivost zakrpa.

Učitavanje zlonamjernog kernel modula je tehnika kojom napadač koristi naredbu `insmod` za učitavanje zlonamjernog kernel modula u jezgru operacijskog sustava. Ovaj modul djeluje kao *rootkit* i omogućava napadaču skrivanje aktivnosti te održavanje trajnog pristupa sustavu.

4. Tehnike generiranja napada

Generiranje mrežnog prometa u svrhu simulacije i testiranja sigurnosnih sustava jedan je od ključnih aspekata u suvremenoj kibernetičkoj sigurnosti. U ovom radu detaljno se razmatraju tehnike korištene za generiranje mrežnih napada, s posebnim naglaskom na promjenu parametara napada, skaliranje napada te upravljanje intenzitetom i s vremenom napada. Primarni alat koji koristimo za ove svrhe temelji se na korištenju lokalne baze uzoraka napada (engl. *sample*), koja uključuje raznolike tipove napada, definiranih scenarija te izvršavanje tih napada nad specifičnim mrežnim topologijama. Na temelju tih uzoraka generiramo mrežni promet koji simulira stvarne scenarije napada nad konkretnom topologijom.

Cjelokupan proces generiranja napada započinje čitanjem paketa iz ulaznih pcap datoteka. Analizom prvog paketa identificiraju se ključni parametri napadača i žrtve, nakon čega se paketi razdvajaju u dvije kategorije: paketi napadača i paketi žrtve. Parametri napada se zatim modificiraju prema zadanim vrijednostima scenarija.

Skaliranje napada uključuje generiranje dodatnih paketa na temelju faktora skaliranja. Ovi paketi se prilagođavaju novim portovima kako bi se izbjeglo ponovno korištenje istih resursa. Vremenske oznake paketa također se prilagođavaju kako bi se osigurala vjerodostojnost simulacije. Na kraju, svi paketi se sortiraju prema vremenu dolaska kako bi se stvorio kontinuirani i realističan napadni promet. Pojedini koraci prikazani su u nastavku.

4.1 Lokalna baza napada

U svrhu generiranja realističnih i raznolikih scenarija mrežnih napada, kreirana je lokalna baza napada koja sadrži male uzorke (engl. sample) raznolikih vrsta napada. Ova baza napada temelji se na kombinaciji javno dostupnih podataka o mrežnim napadima i ručno generiranih napada, čime se osigurava pokrivenost širokog spektra prijetnji i scenarija. Proces izrade lokalne baze napada uključivao je filtriranje napada iz javno dostupnih skupova podataka i ručno generiranje napada. Ti procesi su prikazani u nastavku rada.

4.1.1 Filtriranje iz javno dostupnih skupova podataka

Prvi korak u izradi baze napada bio je pregled i analiza javno dostupnih skupova podataka mrežnih prometa. Postoji nekoliko dobro poznatih izvora koji sadrže `PCAP` datoteke s raznim vrstama napada, poput CICIDS, UNSW-NB15, Kitsune Network Attack i sličnih.

Iz tih velikih i često pomiješanih skupova podataka, izdvojeni su pojedini napadi filtriranjem. Proces filtriranja uključivao je identifikaciju specifičnih uzoraka prometa koji odgovaraju određenim vrstama napada, poput DDoS, Nmap skeniranja, iskorištavanje ranjivosti, i sl.

Uzorak napada je minimalni skup mrežnih paketa koji sadrži reprezentativne karakteristike i obrasce određenog tipa mrežnog napada. Uzorci se izdvajaju iz stvarnih mrežnih snimki napada, a zatim spremaju u lokalnu bazu podataka. Glavne prednosti uzoraka napada su:

1. Modularnost - kombiniranje različitih uzoraka za složene scenarije napada,
2. Skalabilnost - skaliranje uzoraka za različite veličine i intenzitete napada,
3. Prilagodljivost - mogućnost prilagodbe parametara poput IP adresa, portova, vremena,
4. Učinkovitost - minimalan set paketa smanjuje potrebni prostor za pohranu.

Uzorci napada omogućuju sistematizirano generiranje realističnih mrežnih napada, ključnih za testiranje i evaluaciju sustava za detekciju i prevenciju prijetnji. Njihova modularnost i prilagodljivost ključne su za kreiranje širokog raspona scenarija napada.

4.1.2 Ručno generiranje napada

U slučajevima kada određeni tip napada nije mogao biti pronađen u javno dostupnim skupovima podataka, pristupilo se ručnom generiranju tih napada.

Ručno generiranje uključivalo je konfiguraciju specifičnih alata i skripti za simulaciju napada unutar kontroliranog mrežnog okruženja. Ovi alati omogućuju generiranje različitih vrsta napada s preciznom kontrolom nad parametrima kao što su IP adrese, portovi, protokoli i drugi relevantni atributi.

Generirani napadi su snimljeni s pomoću alata za snimanje mrežnog prometa Wireshark te su pohranjeni kao `PCAP` datoteke u lokalnu bazu napada.

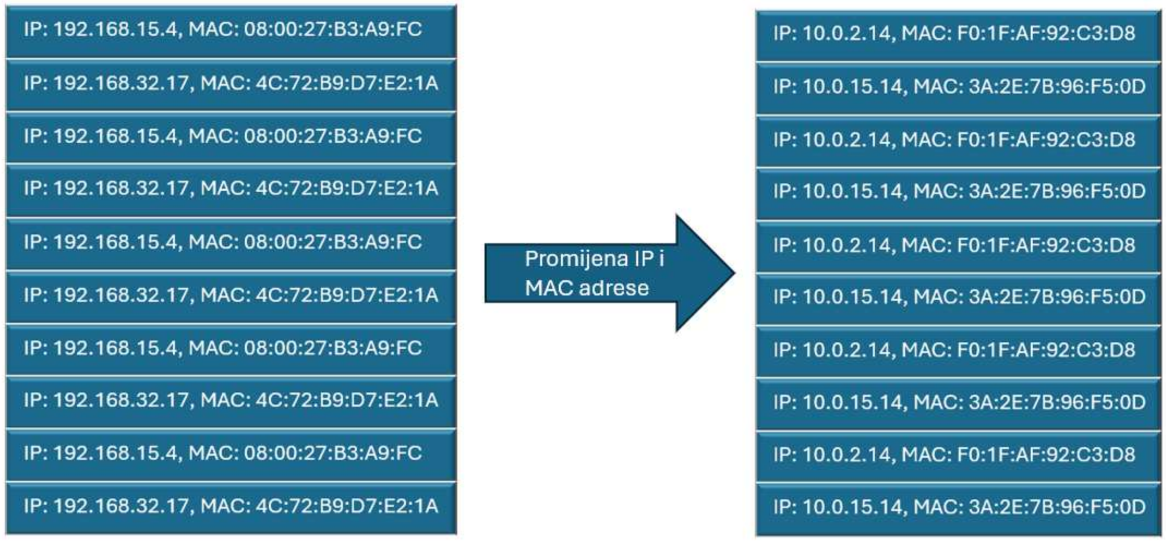
4.2 Modificiranje napada

Modificiranje napada ključno je za prilagođavanje napada specifičnim scenarijima napada. Ovaj proces sastoji se od četiri glavna koraka: promjene parametara napada, skaliranja napada, promjene intenziteta i upravljanja s vremenom napada. Promjena parametara napada omogućava prilagodbu osnovnih karakteristika napada kako bi odgovarale različitim mrežnim topologijama, što je ključno za testiranje napada u različitim mrežnim okruženjima. Skaliranje napada osigurava da se opseg napada može prilagoditi veličini i kapacitetu ciljanog sustava, omogućujući realističnu simulaciju koja odražava stvarne uvjete mreže. Promjena intenziteta napada omogućava kontrolu nad snagom i učestalošću napada, što je važno za testiranje otpornosti mrežnih sustava na različite razine napadačkog pritiska. *Upravljanje s vremenom napada uključuje planiranje i sinkronizaciju napadačkih aktivnosti*. Tako se simuliraju kompleksne napadačke kampanje, što je bitno za otkrivanje potencijalnih slabosti u sustavima obrane. Ove modifikacije zajedno omogućavaju fleksibilnu i detaljnu prilagodbu napada, simulirajući širok spektar napadačkih scenarija i testiranje različitih obrambenih strategija.

4.2.1 Promjena parametara napada

Promjena parametara napada ključna je za prilagodbu napada specifičnim uvjetima generiranja napada. Proces započinje identifikacijom osnovnih parametara napadača i žrtve analizom prvih paketa iz uzorka napada. Ovi osnovni parametri uključuju MAC adrese, IP adrese te brojeve portova napadača i žrtve. Nakon identifikacije, ovi se parametri mogu prilagoditi prema potrebama specifičnog scenarija.

Na primjer, promjene IP adresa napadača i žrtve omogućuju simulaciju napada iz različitih mrežnih segmenata ili čak različitih mreža. Promjene MAC adresa omogućuju simulaciju napada unutar istog lokalnog mrežnog segmenta, dok promjena TTL vrijednosti može simulirati napade s različitih udaljenosti unutar mreže. Dodatne promjene poput modifikacije TCP sekvencijskih brojeva ili brojeva potvrda (ACK) koriste se za simulaciju složenijih napada kao što su TCP hijacking ili replay napadi. Na **Slika 2** u nastavku grafički je prikazana promjena IP i MAC adresa paketa.

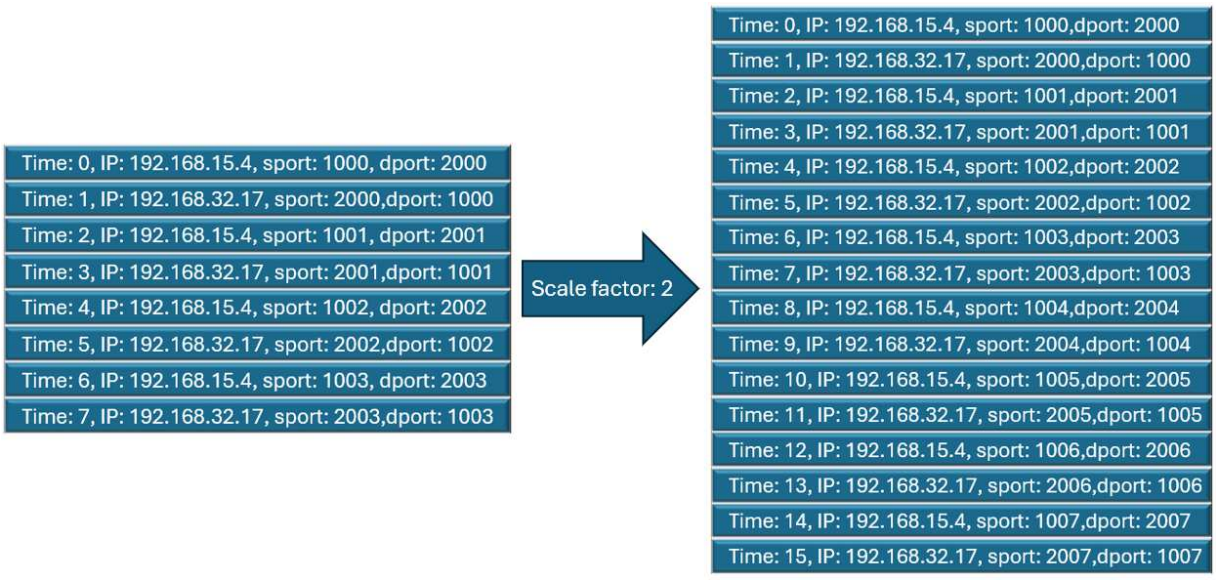


***Slika 2**: Grafički prikaz procesa promjene parametara napada*

4.2.2 Skaliranje

Skaliranje napada predstavlja proces prilagodbe trajanja napada generiranjem dodatnih paketa. Na temelju parametra `duration` u scenariju, određuje se željeno trajanje napada te se računa faktor skaliranja (engl. *scale factor*). Faktor skaliranja određuje koliko puta je potrebno povećati broj paketa u uzorku napada kako bi trajanje napada odgovaralo zadanom periodu.

Ako je zadano trajanje napada dulje od trajanja uzorka napada, broj paketa se multiplicira prema faktoru skaliranja. Pri tome je važno osigurati da se ne koriste isti portovi kako bi se izbjeglo ponovno korištenje istih resursa, što bi moglo otežati detekciju i analizu napada. Novi portovi se generiraju na temelju postojeće liste portova te se dodaju u promet kako bi se osigurala vjerodostojnost simulacije. Taj proces skaliranja vidi se na **Slika 3**.

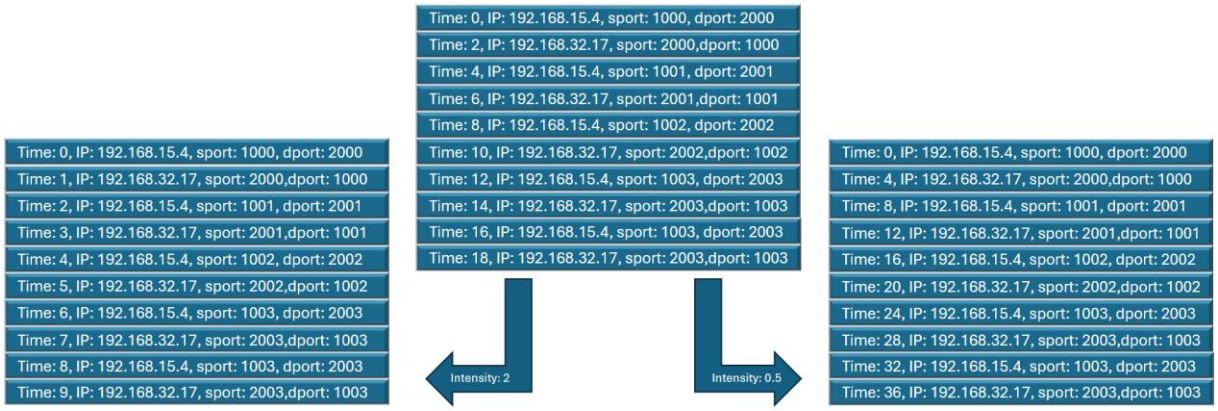


***Slika 3**: Grafički prikaz procesa skaliranja napada*

4.2.3 Intenzitet

Intenzitet napada definiran je parametrom `intensity` u scenariju, koji određuje brzinu generiranja i slanja paketa. Ovaj parametar je ključan za simulaciju stvarnog mrežnog opterećenja koje napad može izazvati. Visoki intenzitet napada može preopteretiti mrežne resurse, dok niski intenzitet omogućuje proučavanje suptilnijih napada koji mogu proći neopaženo kroz sigurnosne sustave.

Prilagođavanje intenziteta napada uključuje kontrolu vremenskih razmaka između generiranih paketa. Ovaj proces osigurava da napad simulira stvarno mrežno ponašanje te omogućuje testiranje performansi i otpornosti mreže pod različitim uvjetima opterećenja. Promjene intenziteta mogu se provoditi dinamički tijekom trajanja napada, što omogućuje simulaciju složenih napadačkih scenarija s varijabilnim opterećenjem. Takva promjena nad intenzitetom vizualno je vidljiva na **Slika 4**.



***Slika 4**: Grafički prikaz procesa promjene intenziteta*

4.2.4 Upravljanje s vremenom napada

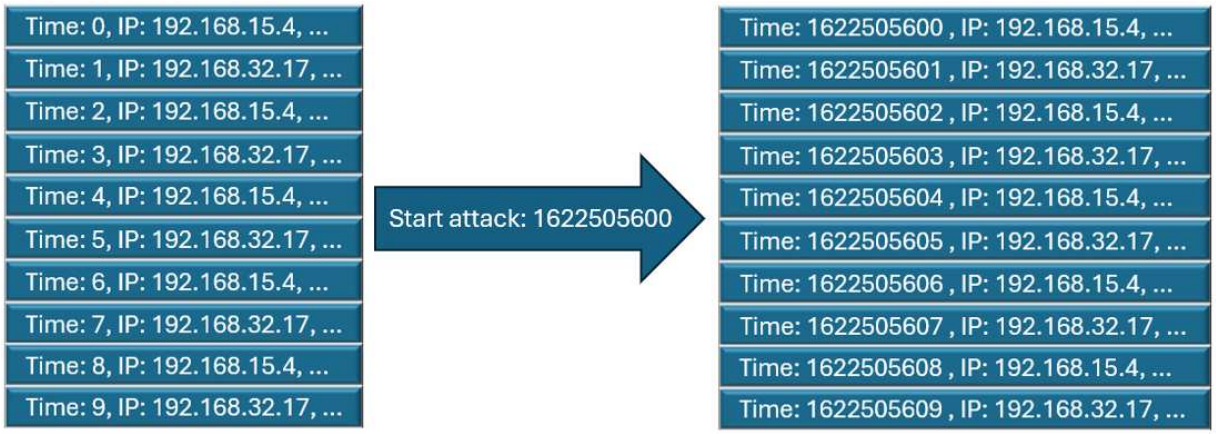
Upravljanje s vremenom napada je ključni korak u simulaciji realističnih mrežnih napada. Ovaj korak omogućava precizno određivanje trenutka kada napad počinje i prilagodbu vremenskih oznaka svakog paketa u skladu s tim. Nakon što su svi prethodni koraci (promjena parametara napada, skaliranje, promjena intenziteta) završeni, dolazi na red upravljanje s vremenom napada. U programu je omogućeno postavljanje specifičnog vremena početka napada. Ovo se postiže na sljedeći način.

Korisnik može definirati vrijeme kada želi da napad započne. Ovo vrijeme se izražava kao apsolutna vremenska oznaka (npr. Unix timestamp ili drugo odgovarajuće vremensko obilježje), a zapisuje se u polje `start_attack` datoteke scenarija. Prvi paket u uzorku napada obično ima vremensku oznaku koja se smatra početnom (npr. vrijeme 0). Sve vremenske oznake drugih paketa su relativne u odnosu na ovo inicijalno vrijeme.

Kako bi se svi paketi uskladili s novim vremenom početka napada, svaki paket mora biti ažuriran s novom vremenskom oznakom. Proces je sljedeći. Za svaki generirani paket dohvaća se trenutna vremenska oznaka. Zadanom vremenu početka napada dodaje se trenutna vremenska oznaka svakog paketa. Ovo osigurava da svaki paket ima točno vrijeme dolaska u mrežu. Na primjer, ako je zadano vrijeme početka napada 1622505600 (Unix timestamp), a vremenska oznaka prvog paketa je 0, vremenske oznake svih paketa će se prilagoditi dodavanjem 1622505600. Tako, paket koji je imao vremensku oznaku 10 sekundi nakon

početka uzorka, sada će imati vremensku oznaku 1622505610. Nakon što su nove vremenske oznake izračunate, one se primjenjuju na sve pakete u napadu. To se postiže iteracijom kroz sve pakete i ažuriranjem njihovih vremenskih polja s izračunatim vrijednostima. Time se osigurava da su svi paketi usklađeni s definiranim vremenom početka napada.

Upravljanje s vremenom napada omogućuje preciznu kontrolu nad simulacijom mrežnih napada, omogućavajući korisnicima da definiraju točan trenutak početka napada. Ova funkcionalnost je ključna za analiziranje sigurnosnih sustava u realnim uvjetima, gdje je vremensko usklađivanje napada od velike važnosti. Dodavanjem zadanog vremena početka napada svakom paketu, osigurava se konzistentnost i realizam u simuliranim napadima, što omogućava učinkovitiju analizu sigurnosnih mehanizama. Upravljanje s vremenom početka napada vizualno je prikazano u nastavku na **Slika 5**.



***Slika 5**: Grafički prikaz procesa upravljanja s vremenom napada*

4.2.5 Statički i dinamički napadi

U ovom pristupu, napadi se dijele na statičke i dinamičke. Statički napadi generiraju se s pomoću klase StaticGenerator, dok se dinamički napadi razdvajaju na dvije ključne komponente: generiranje napadačkih paketa s pomoću klase AttackGenerator i generiranje paketa žrtve s pomoću klase VictimGenerator.

Statički napadi su jednostavniji za implementaciju jer ne zahtijevaju dinamičko prilagođavanje parametara. Oni se generiraju na temelju početnog uzorka napada bez mogućnosti skaliranja ili promjene intenziteta, odnosno moguće je samo promjena parametara napada i upravljanje s vremenom napada. Razlog tome je što statički napadi spadaju u neku od kategorija napada koja su točno definirana te nije moguće mijenjati intenzitet ili skalirat napad. Neki od takvih napada su iskorištavanje ranjivosti (engl. *exploits*), stražnji ulaz (engl. *backdoor*), povratna ljuska (engl. *reverse shell*) i sl. Ova vrsta napada korisna je za osnovno testiranje i analizu sigurnosnih sustava.

Dinamički napadi, s druge strane, omogućuju detaljnu kontrolu nad svim aspektima napada. Generiranje napadačkih i žrtvinih paketa omogućuje simulaciju kompleksnijih scenarija u kojima se parametri napada mogu dinamički prilagođavati. Ovi napadi su pogodni za napredne analize sigurnosnih sustava koji zahtijevaju preciznu simulaciju različitih vrsta prijetnji.

5. Razvijeno programsko rješenje

U svrhu praktičnog dijela diplomskog rada razvijen je program `Universal Attack Generator` i to je program koji generira parametrizirani promet mrežnih napada temeljenih na zadanoj mrežnoj topologiji i definiranom scenariju napada. Program je dizajniran tako da omogućuje kontrolu nad vremenom generiranja prometa, njegovom intenzitetu, izvoru i odredištu te svim ostalim parametrima koji ovise o topologiji na kojoj se napad provodi. Topologije su definirane u formatu koji koristi CCS simulator. Program je modularno konstruiran tako da svaki dio može funkcionirati neovisno o ostalim komponentama. Glavne komponente programa uključuju `Topology Parser`, `Scenario Parser`, `Attack Generator` i `PcapMerger`. Program se pokreće iz naredbenog retka te mu se kao parametri predaju putanje do datoteke topologije i scenarija. Opis korištenja programa i svih dostupnih parametara dobiva se korištenjem zastavice `-h` ili `--help`. Opis korištenja je prikazan u **Izvorni kod 4**.

```
usage: main.py [-h] topology_file scenario_file

Generate network attack traffic based on given topology
and scenario files.

positional arguments:

  topology_file  The path to the topology file.

  scenario_file  The path to the scenario file.

options:

  -h, --help     show this help message and exit
```

Izvorni kod 4: Ispis opcija programa

5.1 Opis rješenja, korištenih alata i tehnologija

Python [15], verzija 3.12, odabran je kao glavni programski jezik zbog svoje fleksibilnosti, jednostavnosti korištenja te široke podrške za biblioteke koje su ključne za ovaj projekt.

Python je programski jezik visoke razine koji se često koristi za brzo prototipiranje, razvoj web aplikacija, analizu podataka i automatizaciju zadatka. Također, omogućuje brzo razvijanje i testiranje prototipova te ima bogat ekosustav alata za mrežnu analizu i obradu podataka. U kontekstu diplomskog rada, *Python* se koristi kao glavni jezik za razvoj programa za generiranje i analizu mrežnih napada.

NetworkX [16] je *Python* biblioteka koja pruža alate za analizu, manipulaciju i vizualizaciju složenih mreža, uključujući grafove i dijagrame. U diplomskom radu, NetworkX se koristi za modeliranje i analizu mrežnih topologija te za vizualizaciju rezultata. Omogućuje efikasno parsiranje i manipulaciju mrežnim topologijama, što je ključno za generiranje realističnih mrežnih napada u kontekstu Universal Attack Generatora. Ova sposobnost omogućuje precizno modeliranje napada i analizu potencijalnih sigurnosnih prijetnji unutar simuliranih mrežnih okruženja.

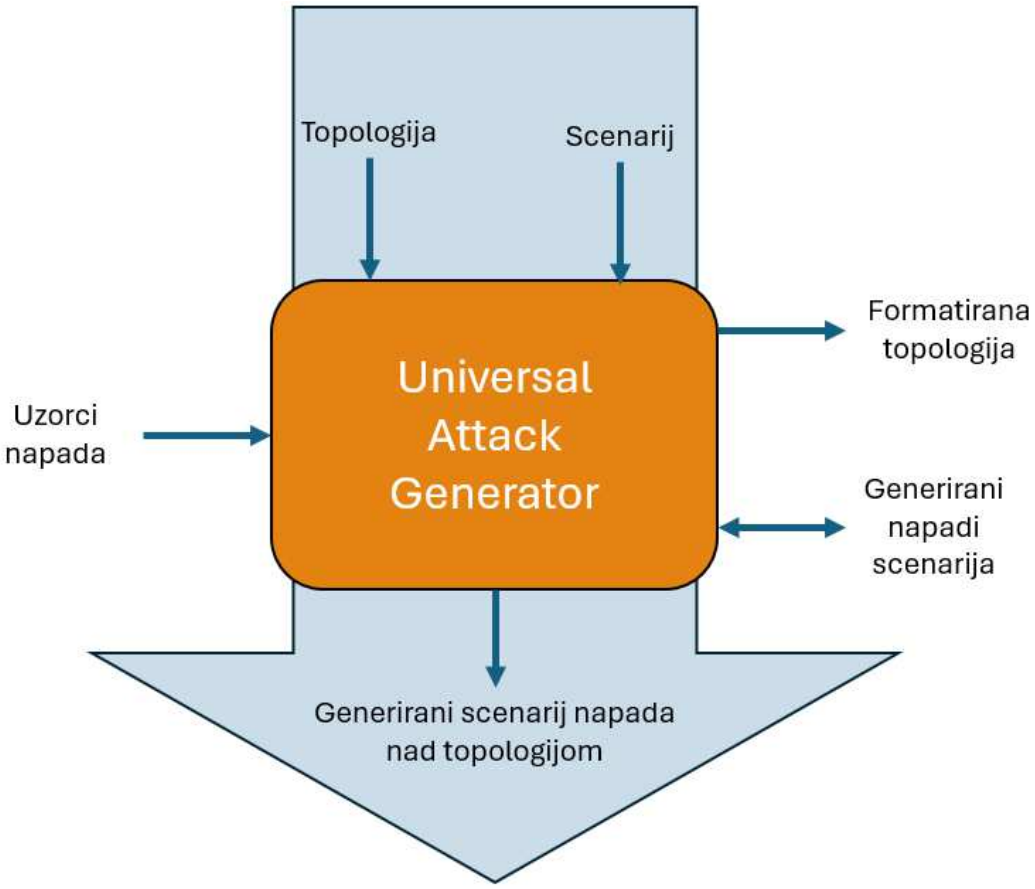
PyYAML [17] je *Python* biblioteka koja omogućuje parsiranje i generiranje YAML datoteka. YAML je lako čitljiv i jednostavan format za strukturirane podatke, često korišten za konfiguracijske datoteke i razmjenu podataka. U kontekstu diplomskog rada, PyYAML se koristi za definiranje scenarija napada i njihovih parametara u YAML formatu, što omogućuje jednostavniju konfiguraciju i upravljanje. PyYAML omogućuje lako čitanje i modificiranje različitih scenarija napada, čime se postiže fleksibilnost i prilagodljivost u generiranju mrežnih napada.

Scapy [18] je *Python* biblioteka koja omogućuje manipulaciju mrežnim paketima na visokom nivou, uključujući izradu, slanje, hvatanje i analizu paketa na raznim protokolima. U diplomskom radu, Scapy se koristi za generiranje i modificiranje mrežnih paketa koji simuliraju napade u simuliranim mrežnim okruženjima. Kao moćan interaktivni alat, Scapy omogućuje generiranje, manipulaciju i analizu mrežnih paketa na niskom nivou. U Universal Attack Generatoru, koristi se za čitanje, modificiranje i generiranje `PCAP` datoteka koje predstavljaju mrežne napade.

Korištenje ovih biblioteka omogućuje razvoj funkcionalnog i efikasnog alata za simulaciju mrežnih napada u okviru diplomskog rada.

5.2 Implementacija

U ovom odjeljku opisani su implementacijski detalji programa koji je razvijen. Također, opisana je arhitektura cijelog sustava što uključuje način rada programa s lokalnom bazom napada i kako radi pojedina komponenta. Kao što je detaljno opisano u prethodnom poglavlju, pri pokretanju program prima dvije ključne datoteke: datoteku koja opisuje topologiju mreže i datoteku scenarija napada. U svakom koraku procesa generiranja pojedinog napada iz scenarija, program učitava odgovarajući uzorak napada iz lokalne baze uzoraka. Izlazi programa uključuju formatiranu topologiju, odnosno datoteku koja sadrži definiranu i formatiranu CCS topologiju te generirane `PCAP` datoteke koje sadrže generirane napade za svaki korak scenarija, te konačnu `PCAP` datoteku koja objedinjuje sve napade u jedan cjelokupni scenario. Cijeli sustav prikazan je na **Slika 6**.



***Slika 6**: Ulazi i izlazi sustava*

Program Universal Attack Generator omogućava generiranje različitih vrsta mrežnih napada, odnosno scenarija nad definiranim mrežnim topologijama. Program se sastoji od nekoliko ključnih komponenti koje su prikazane na sekvencijskom dijagramu na **Slika 7**.

Rad programa započinje kada korisnik pokrene program navodeći željenu topologiju mreže i scenarij napada. Glavna komponenta, odnosno ulazna točka u program je `main.py` datoteka. Ona pri pokretanju prima ulazne parametre i prosljeđuje ih na daljnju obradu.

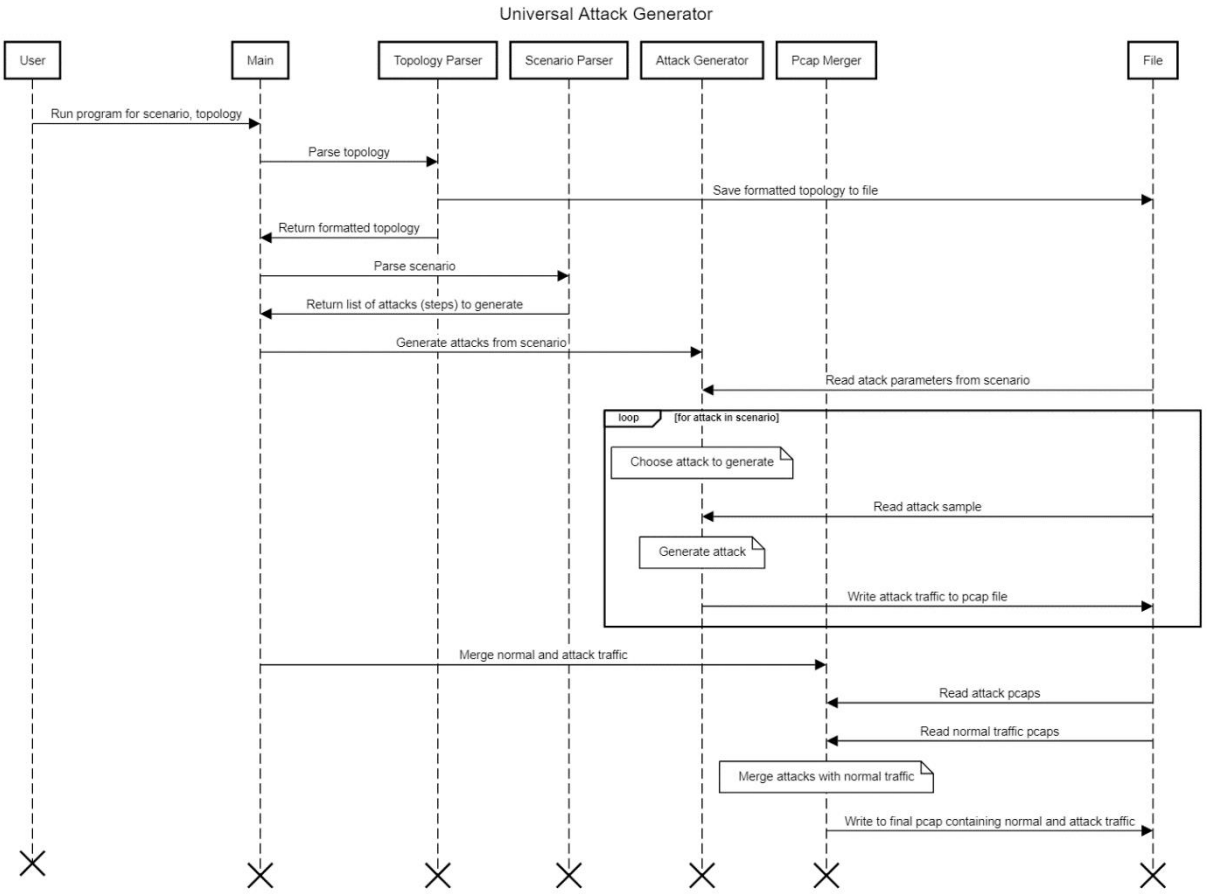
Komponenta Topology Parser prima datoteku koja opisuje topologiju mreže i formatira ju u odgovarajući format. Pošto CCS topologija ne sadrži mrežna sučelja, odnosno čvorovi nemaju dodijeljene IP adrese, Topology Parser analizira mrežu i traži usmjernike, odnosno podmreže i sukladno s pronađenim dodjeljuje svakom čvoru odgovarajuća mrežna sučelja i IP adrese. Formatirani prikaz topologije vraća se glavnoj komponenti te se također zapisuje u posebnu datoteku kako bi se dalje mogla koristiti i analizirati izvan ovog programa.

Scenario Parser prima datoteku koja sadrži detaljno opisani scenarij napada i parsira ga u popis napada koje treba generirati. Iz datoteke scenarija program Scenario Parser čita sve potrebne informacije kao što su ime pojedine faze te napade koji se izvršavaju unutar pojedine faze. Datoteka scenarija je dizajnirana tako da se cijelom scenariju dodjeljuju parametri koji vrijede za svaku fazu i napad, ali isto tako je moguće svakoj fazi dodijeliti drugačije parametre koji će nadjačati globalne parametre te će tako program uzeti parametre definirane za tu fazu. Nakon što je parser završio s obradom scenarija, on taj popis vraća glavnoj komponenti koja dalje kreće s generiranjem pojedinog napada što je opisano u nastavku.

Attack Generator je ključna komponenta koja generira stvarne napade na temelju scenarija i topologije. Za svaki napad iz popisa napada, ova komponenta učitava odgovarajuću konfiguraciju napada iz samog scenarija i uzorak mrežnog prometa za taj napad. Koristeći sve informacije o određenom napadu, komponenta mijenja pojedine parametre u uzorku napada, skalira ga, mijenja intenzitet i upravlja s vremenom početka napada. Tako program generira mrežni promet koji simulira definirani napad i sprema ga u zasebnu `PCAP` datoteku. Svaki od opisanih koraka se provodi za svaku pojedinu fazu, odnosno napad pojedine faze. Nakon što je program generirao sve napade definirane u scenariju napada sljedeća komponenta preuzima zadatak spajanja svih napada u jednu datoteku. Generirana datoteka se sastoji od normalnog mrežnog prometa neke mreže računala te ubačenog malicioznog prometa.

Pcap Merger je konačna komponenta koja spaja normalni mrežni promet s generiranim napadima. Ona učitava PCAP datoteke s normalnim prometom i PCAP datoteke s generiranim napadima, spaja ih u jednu cjelinu, sortira pakete po vremenu te sprema sve pakete svih napada u jednu rezultirajuću PCAP datoteku. Ta datoteka sadrži i normalni promet i simulirane napade, odnosno cjelokupni mrežni promet cjelovitog scenarija napada.

Konačna PCAP datoteka može se koristiti za testiranje sustava za detekciju i prevenciju upada (IDS/IPS), analizu mrežnog prometa ili druge svrhe vezane za sigurnost mreže.



Slika 7: Sekvencijski dijagram programa

5.2.1 Lokalna baza uzoraka napada

Lokalna baza uzoraka napada (Scenario_pcaps) sadrži male PCAP datoteke koje predstavljaju pojedine mrežne napade. Te se datoteke koriste kao osnovni uzorci za

generiranje napada definiranih u scenariju. Ti uzorci su pronađeni i filtrirani iz javno dostupnih skupova podataka ili su ručno generirani kao što je opisano u poglavlju 4.1 Lokalna baza napada.

Uzorci napada predstavljaju snimke mrežnog prometa koje sadrže karakteristična obilježja i ponašanja određenih vrsta napada. Svaki uzorak započinje s napadačkim paketom iz kojeg se izvlače informacije o napadaču, poput IP adrese i korištenih portova, što omogućava prilagodbu i modifikaciju napada tijekom procesa generiranja.

Lokalna baza sastoji se od više stotina ili tisuća takvih uzoraka spremnih za korištenje. Uzorci su kategorizirani u sljedeće skupine: Fuzzers, Analysis, Backdoors, DoS (Distributed Denial of Service), Exploits, Generic, Reconnaissance, Shellcode, Worms i ostale slične kategorije koje obuhvaćaju širok raspon napadačkih tehnika i prijetnji. Ova raznovrsna zbirka uzoraka omogućava generiranje realističnih simulacija mrežnih napada, što je ključno za testiranje i evaluaciju sustava za otkrivanje i prevenciju upada. Lokalnu bazu je moguće vrlo lako nadopunjavati i uređivati dodavanjem novih uzoraka, čime se osigurava neprekidno ažuriranje i proširivanje opsega napadačkih scenarija.

5.2.2 Proces formatiranja CCS topologije

U nastavku se detaljno opisuje proces formatiranja topologije u okviru diplomskog rada, korištenjem *Python* koda. Ovaj proces uključuje parsiranje početne topologije, dodjelu IP adresa i mrežnih sučelja te spremanje formatirane topologije.

1. Parsiranje topologije započinje učitavanjem podataka iz `JSON` datoteke koja sadrži opis mrežnih čvorova i njihovih međusobnih veza. Svaki čvor je predstavljen kao instanca klase Node, koja sadrži informacije o čvoru, uključujući njegov ID, ime, oznake, resurse i povezane čvorove. U ovoj fazi, svi čvorovi s oznakama `machine` ili `trust_zone` dodaju se u mrežu. Zatim se uspostavljaju veze između čvorova na temelju njihovih resursa. Izdvojeni kod koji filtrira čvorove na temelju zadanih oznaka vidljiv je na **Izvorni kod 5**.

```
node = Node(item)
    if "machine" in node.labels
            or "trust_zone" in node.labels:
        nodes[node.id] = node
```

***Izvorni kod 5**: Opis filtriranja čvorova*

Nakon što su čvorovi dodani u mrežu, uspostavljaju se veze između njih na temelju njihovih resursa.

2. Nakon što je mreža stvorena, sljedeći korak je konverzija u format pogodan za daljnju obradu. Ovaj korak uključuje stvaranje strukture podataka koja sadrži informacije o čvorovima, njihovim sučeljima i povezanim čvorovima. Struktura se sastoji od sljedećih polja: `name`, `id`, `type`, `interfaces`, `connects_to`.
3. U ovom dijelu koda, informacije o čvorovima, sučeljima i povezanim čvorovima dodaju se u odgovarajuće strukture podataka.
4. U ovoj fazi, čvorovima se dodjeljuju IP adrese i mrežna sučelja na temelju njihove povezanosti u mreži. Svakom sučelju dodjeljuje se IP adresa iz unaprijed definirane podmreže. Ovdje se za svaki čvor stvara nova podmreža i dodjeljuju se IP adrese njegovim sučeljima. Povezani čvorovi također dobivaju odgovarajuće IP adrese čime se osigurava pravilna povezanost u mreži (**Izvorni kod 6**).

```
for connected_node in formatted_network["nodes"]:
    if connection == connected_node["name"]:
        subnet_counter += 1
        ip_address = subnet.network_address + subnet_counter
        connected_node["interfaces"].append({
            "name": "eth0",
            "ip_address": str(ip_address),
            "subnet_mask": str(subnet.netmask)
        })
        break
```

***Izvorni kod 6**: Dodjela mrežnih sučelja i IP adresa*

5. Nakon dodjele IP adresa i sučelja, formatirana topologija sprema se u novu `JSON` datoteku. Ovaj korak omogućuje daljnju analizu i korištenje formatirane topologije u drugim dijelovima sustava. U glavnoj funkciji `run_formatter`, topologija se učitava iz datoteke, kreira se mreža, formatira se i dodjeljuju se IP adrese. Konačni rezultat sprema se u novu `JSON` datoteku za daljnje korištenje.

Proces formatiranja topologije u ovom radu uključuje parsiranje početne topologije, konverziju u format pogodan za daljnju obradu, dodjelu IP adresa i mrežnih sučelja te spremanje formatirane topologije. Korištenjem opisanih metoda i algoritama, osigurava se točna i učinkovita transformacija mrežne topologije, čime se omogućuje daljnja analiza i simulacija mrežnih aktivnosti.

5.2.3 Proces parsiranja scenarija

U ovom dijelu diplomskog rada opisuje se proces parsiranja scenarija napada korištenjem *Python* koda. Parsiranje scenarija uključuje učitavanje opisa napada iz `YAML` datoteke, izvršavanje faza i taktika napada te generiranje odgovarajućih mrežnih paketa.

1. Proces započinje funkcijom run_parser, koja učitava scenarij napada iz `YAML` datoteke. `YAML` format omogućava jednostavno definiranje strukture napada, uključujući faze i taktike. Ova funkcija učitava `YAML` datoteku s pomoću `yaml.safe_load` funkcije, što omogućava sigurno parsiranje podataka. Nakon učitavanja, poziva se funkcija `execute_attack_scenario` za izvršavanje učitanog scenarija.
2. Svaki scenarij napada sastoji se od više faza. Funkcija `execute_attack_scenario` iterira kroz faze definirane u scenariju i za svaku fazu poziva funkciju execute_phase. Svaka faza sadrži detalje o taktikama koje treba izvršiti. Funkcija `execute_phase` ispisuje naziv faze, prikazuje opće informacije ako su dostupne te iterira kroz taktike unutar faze i poziva `execute_tactic` za svaku taktiku.
3. Funkcija `execute_tactic` je odgovorna za izvršavanje pojedinačnih taktika unutar faze. Ona simulira izvršavanje taktike, definira ulazne i izlazne datoteke, stvara instance potrebnih generatora napada te modificira i skalira promet napada i žrtve. Ako

je taktika definirana kao statična (parametar `static`), koristi se `StaticGenerator`, koji modificira parametre bez skaliranja intenziteta napada. Tijek generiranja statičkih napada vidi se u **Izvorni kod 7**.

```
if tactic["static"]:
   static_gen = StaticGenerator(input_file, tactic['name'])
   static_gen.modify(parameters)
   generated_packets = list()
   generated_packets.extend(static_gen.victim_packets)
```

***Izvorni kod 7**: Proces modificiranja napada kod statičkih napada*

U suprotnom se koriste `AttackGenerator` i `VictimGenerator` za generiranje i skaliranje mrežnog prometa napada i žrtve kao što je vidljivo u **Izvorni kod 8**.

```
else:
   attack_gen = AttackGenerator(input_file, tactic['name'])
   victim_gen = VictimGenerator(input_file, tactic['name'])

   scale_factor = calc_scale_factor(attack_gen.packets,
                                    parameters.get("duration", 1))
   attack_gen.modify(parameters)
   attack_gen.scale_attack(scale_factor)
   victim_gen.modify(parameters)
   victim_gen.scale_attack(scale_factor)

   generated_packets = list()
   generated_packets.extend(victim_gen.victim_packets)
   generated_packets.extend(attack_gen.attacker_packets)
```

***Izvorni kod 8**: Proces modificiranja napada kod ne-statičkih napada*

4. Nakon što su svi paketi generirani i modificirani, oni se spajaju i spremaju u `PCAP` datoteku pomoću funkcije `generate_pcap`. Ova datoteka sadrži sve pakete generirane tijekom simulacije napada i može se koristiti za daljnju analizu mrežnog prometa.

Proces parsiranja scenarija napada opisuje korake potrebne za učitavanje, izvršavanje i generiranje mrežnog prometa na temelju definiranih faza i taktika u `YAML` datoteci. Korištenjem opisanih metoda, omogućava se precizna simulacija napada, što je ključno za analizu sigurnosnih aspekata mreža i sustava.

5.2.4 Proces generiranja pojedinog napada iz scenarija

Generiranje napada u mrežnim scenarijima je složen proces koji uključuje učitavanje, modificiranje, skaliranje i generiranje mrežnih paketa na temelju zadanih parametara. U ovom dijelu ćemo detaljno opisati ključne korake i metode korištene za generiranje mrežnih napada koristeći *Python* skriptu.

Klasa `AttackGenerator` centralni je dio procesa generiranja napada. Ova klasa učitava pakete iz `PCAP` datoteke, identificira informacije o napadaču i žrtvi, te pruža metode za modificiranje i skaliranje napada prema zadanim parametrima.

1. Prvi korak u generiranju napada je učitavanje uzorka napada iz `PCAP` datoteke. Ovo se postiže s pomoću metode `rdpcap` iz biblioteke `scapy.all`, koja učitava pakete iz specificirane datoteke
2. Metoda define_attacker analizira prvi paket u skupu paketa i ekstrahira informacije poput MAC adresa, IP adresa i portova napadača i žrtve, stvarajući rječnik attacker s tim podacima
3. Metoda modify iterira kroz pakete napadača i modificira različite parametre, poput MAC adresa, IP adresa, portova i zastavica, koristeći vrijednosti iz proslijeđenog rječnika parameters
4. Skaliranje napada uključuje generiranje dodatnih paketa kako bi se povećao intenzitet napada. Metoda `scale_attack` koristi nekoliko pomoćnih metoda za generiranje novih portova, izračunavanje vremenskih razlika između paketa, generiranje dodatnih paketa i modifikaciju vremenskih oznaka. Metoda `generiraj_portove` generira dodatne portove za skaliranje napada, uzimajući u obzir postojeće portove u prometu i zadani faktor skaliranja

5.2.5 Proces generiranja PCAP datoteke

Nakon modifikacije i skaliranja paketa, generirani paketi se spremaju u novu `PCAP` datoteku. Ovo omogućava daljnju analizu ili simulaciju napada u mrežnom okruženju. Glavni proces izvršavanja napadnog scenarija sastoji se od nekoliko koraka:

1. Učitavanje Scenarija: Učitava se scenarij napada iz `YAML` datoteke.
2. Izvršavanje Faza: Svaka faza unutar scenarija se izvršava, pri čemu se za svaku fazu poziva metoda `execute_phase`.
3. Izvršavanje Taktika: Unutar svake faze izvršavaju se pojedinačne taktike s pomoću metode `execute_tactic`.

Generiranje napada u mrežnim scenarijima zahtijeva detaljnu analizu i modifikaciju mrežnih paketa kako bi se simulirali različiti uvjeti napada. Korištenjem opisanih klasa i metoda, moguće je generirati sofisticirane napade koji mogu poslužiti za testiranje sigurnosti mrežnih sustava i otkrivanje potencijalnih slabosti.

5.2.6 Proces spajanja generiranih napada

U okviru analize mrežnih napada, važno je imati sveobuhvatnu i koherentnu zbirku mrežnih podataka koji omogućavaju detaljnu analizu i reprodukciju napadačkih scenarija. Nakon generiranja pojedinačnih segmenata napada, sljedeći korak je njihovo spajanje u jedinstvenu `PCAP` datoteku koja obuhvaća cijeli napad. Ovaj proces omogućava jednostavnije istraživanje i razvoj sigurnosnih mjera te simulaciju napada u kontroliranim uvjetima. U ovom dijelu rada detaljno ćemo opisati proces kombiniranja `PCAP` datoteka korištenjem klase `PcapMerger`.

1. Prvi korak u procesu je pretraživanje direktorija za sve `PCAP` datoteke koje završavaju s `_gen.pcap`. Ove datoteke predstavljaju pojedinačne segmente napada generirane u prethodnim fazama. Klasa `PcapMerger` inicijalizira se s direktorijem koji sadrži ove datoteke te (opcionalno) normalnim prometom koji se može uključiti u konačnu datoteku. Metoda `read_pcap_files` pretražuje direktorij i dodaje sve datoteke koje završavaju s `_gen.pcap` u listu `pcap_files`.

2. Nakon identifikacije svih relevantnih datoteka, metoda `merge_pcap_files` čita pakete iz svake datoteke i dodaje ih u listu packets. Također, ova metoda može uključiti normalni promet ako je dostupan.
3. Ključni korak u procesu je sortiranje paketa prema vremenu kako bi se osigurala kronološka točnost podataka. Metoda `sort_by_time` sortira pakete na temelju njihovog atributa `time`. Nakon sortiranja, svi paketi se zapisuju u konačnu `PCAP` datoteku pomoću metode `write_merged_pcap`. Ova datoteka omogućava sveobuhvatnu analizu napadačkog scenarija.
4. Metoda `run_merger` poziva sve potrebne metode za učitavanje, spajanje, sortiranje i zapisivanje `PCAP` datoteka, čineći proces kombiniranja jednostavnim i automatiziranim.

Kombiniranjem generiranih `PCAP` datoteka u jednu konačnu datoteku, omogućuje se detaljna analizu i reprodukcija napadačkih scenarija. Klasa `PcapMerger` olakšava ovaj proces pružajući jednostavne metode za učitavanje, spajanje, sortiranje i zapisivanje paketa. Ova sveobuhvatna datoteka ključna je za istraživanje i razvoj sigurnosnih mjera te simulaciju napada u kontroliranim okruženjima.

5.3 Primjeri korištenja i rezultati

Prvi korak za korištenje Universal Attack Generatora je preuzimanje izvornog koda s GitLab repozitorija. To se postiže s pomoću naredbe za kloniranje repozitorija:

```
git clone https://gitlab.com/VilimP/universal-attack-generator.git
```

Ova naredba preuzima cijeli repozitorij na lokalno računalo, uključujući sve potrebne datoteke i direktorije.

Nakon što je repozitorij kloniran, potrebno je instalirati sve potrebne biblioteke kako bi program mogao pravilno funkcionirati. Sve potrebne biblioteke navedene su u datoteci requirements.txt. Instalaciju je moguće obaviti s pomoću sljedeće naredbe:

```
pip install -r requirements.txt
```

Ova naredba koristi `pip`, *Pythonov* alat za upravljanje paketima, za automatsku instalaciju svih navedenih biblioteka.

Kada su svi preduvjeti ispunjeni, program se može pokrenuti. Ulazna točka programa je datoteka `main.py`, što znači da pokretanje programa započinje izvršavanjem ove datoteke. Prije pokretanja, važno je osigurati da su sve potrebne datoteke smještene na odgovarajućim mjestima:

- Datoteka topologije mora biti unutar direktorija `./Topology`.
- Datoteka scenarija mora biti unutar direktorija `./Scenario`.

Kada su sve datoteke na svojim mjestima, program se može pokrenuti sljedećom naredbom:

```
python main.py Global.json .\Scenario\Scenario.yml
```

U ovom primjeru, Global.json je naziv datoteke koja sadrži mrežnu topologiju, dok Scenario.yml predstavlja datoteku scenarija napada.

Nakon što je program uspješno pokrenut, on generira `PCAP` datoteke koje predstavljaju pojedinačne mrežne napade. Ove datoteke spremaju se u direktorij `Generated`. Na kraju procesa, sve generirane `PCAP` datoteke kombiniraju se u jednu konačnu `PCAP` datoteku koja sadrži cijeli napad. Ova konačna datoteka sprema se pod nazivom `Generated_scenario.pcap`.

Primjer direktorijske strukture nakon pokretanja programa mogao bi izgledati kao što je prikazano u **Izvorni kod 9**.

```
./Generated
      |- attack1_gen.pcap
      |- attack2_gen.pcap
      |- ...
      |- Generated_scenario.pcap
```

***Izvorni kod 9**: Struktura direktorija generiranih napada*

`Generated_scenario.pcap` je datoteka koja sadrži cjelokupni promet generiranih napada, objedinjena i spremna za daljnju analizu ili korištenje.

Pokretanje programa rezultira generiranjem sintetičkog mrežnog prometa koji simulira napadačke aktivnosti definirane u odabranom scenariju. Program se može pokretati nad beskonačno mnogo kombinacija scenarija, ovisno o broju i raznolikosti uzoraka dostupnih u lokalnoj bazi uzoraka napada. Napadi u scenariju moraju biti precizno definirani kroz globalne i lokalne parametre, nakon čega program generator preuzima daljnji proces generiranja napadačkog prometa.

Izlaz programa dizajniran je tako da u svakom trenutku pruža uvid u napredak generiranja i trenutnu fazu izvršavanja. Sažetiji prikaz izlaza programa prikazan je u **Izvorni kod 10**. Detaljne informacije ispisuju se u konzolu, uključujući opise faza i taktika, te parametre poput IP adresa napadača i žrtve, trajanja napada, intenziteta i vremena početka. Tijekom generiranja, program izvještava o ključnim koracima poput skaliranja, modifikacije, prilagodbe vremena napada, spajanja i zapisivanja generiranih paketa u `PCAP` datoteke. Na kraju se ispisuje ukupno vrijeme potrebno za generiranje scenarija napada, zajedno s putanjom do generirane `PCAP` datoteke koja sadrži sintetički mrežni promet.

Ključni dijelovi izlaza programa:

```
[1] Obrada topologije
      CCS topology -> .\Topology\Global.json
      Kreiranje mreže (povezivanje čvorova)
      Converting to format
      Asigning interfaces and IP addresses
      Formatted network saved to .\Topology\Global_formatted.json.
[2] Obrada scenarija
    Reading scenario file
[3] Attack scenario generation
    Executing phase: RECONNAISSANCE
        General Information:
            description: This phase involves gathering information
            about the target network and systems to identify
            potential vulnerabilities and entry for exploitation.
                  Attacker IP: 192.168.1.100
                  Victim IP: 192.168.1.200
                  Attacker MAC: 00:0a:95:9d:68:16
                  Victim MAC: 00:0a:95:9d:68:17
```

```
                duration: 1
                intensity: 1
                start_attack: 1717943141
        Executing tactic: Port Scan
            [+] Scaling
            [+] Modification
            [+] Adjusting attack time
            [+] Merging
            [+] Writing to pcap
                => Generated Port Scan tactic in PCAP file:
                                    ./Generated/Port Scan_gen.pcap
    Executing phase: INITIAL_COMPROMISE
         General Information:
            description: This phase focuses on exploiting
            vulnerabilities in target systems to gain initial access
            and establish a foothold within the network.
                  Attacker IP: 192.168.1.100
                  Victim IP: 192.168.1.200
        Executing tactic: Apache Struts2 code execution
            [+] Modification
            [+] Merging
            [+] Writing to pcap
                => Generated Apache Struts2 code execution tactic in
      PCAP file: ./Generated/Apache Struts2 code execution_gen.pcap
[4] Spajanje .pcap datoteka
    [+] Searching for _gen.pcap files
    [+] Reading and merging .pcap files
    [+] Writing packets to Generated.pcap
        Written to ./Generated/Generated.pcap
=> Attack scenario generated in 14.628965139389038s
```

***Izvorni kod 10**: Izlaz programa*

5.4 Ograničenja implementacijskog rješenja

Implementacija rješenja za generiranje mrežnih napada suočava se s nekoliko značajnih ograničenja koja utječu na njegovu učinkovitost, točnost i opseg primjene. Ova ograničenja

proizlaze iz prirode mrežnih scenarija, složenosti napada i tehničkih izazova povezanih s generiranjem i modifikacijom mrežnog prometa.

Jedno od glavnih ograničenja implementacijskog rješenja je ograničena baza uzoraka napada. Trenutna baza podataka sadrži samo ograničen broj `PCAP` datoteka koje predstavljaju uzorke različitih mrežnih napada. Ovo ograničenje utječe na raznovrsnost generiranih napada, što može rezultirati manje sveobuhvatnim testiranjem sigurnosnih mjera u mrežnim sustavima. Proširenje baze uzoraka napada povećalo bi raznovrsnost i omogućilo generiranje šireg spektra napadačkih scenarija. Međutim, prikupljanje dodatnih uzoraka može biti izazovno zbog ograničene dostupnosti javno dostupnih `PCAP` datoteka koje sadrže specifične i kompleksne napade.

Generiranje velikih napadačkih scenarija može biti vremenski zahtjevno. Optimizacija performansi je stoga ključna za smanjenje vremena potrebnog za generiranje i simulaciju velikih napadačkih scenarija. Trenutna implementacija može imati problema s obradom velikih količina podataka zbog računalnih ograničenja i složenosti algoritama koji se koriste za generiranje i modifikaciju paketa. Optimizacija algoritama i korištenje učinkovitijih tehnika obrade podataka mogla bi značajno poboljšati performanse i omogućiti bržu generaciju scenarija, što je posebno važno za dinamične i kompleksne mrežne topologije.

Točnost simulacije je još jedno važno ograničenje. Generirani promet možda neće uvijek točno oponašati stvarne napade zbog ograničenja uzoraka i modifikacija koje se primjenjuju na pakete. Ovaj problem je posebno izražen kod simulacije kompleksnih napada kao što su eksploatacije ranjivosti sustava, virusi, programi za stražnji ulaz (engl. *Backdoor*) i rootkitovi. Kako bi se poboljšala točnost simulacije, potrebno je dodatno istraživanje i razvoj koji bi omogućili preciznije generiranje prometa. Međutim, zbog vrlo velikog broja mogućnosti modifikacija napada i kompleksnosti određenih napada, ovo predstavlja značajan izazov.

Još jedan izazov leži u pronalaženju i modificiranju normalnog mrežnog prometa tako da simulira promet unutar specifične mrežne topologije koja se koristi za simulaciju napada. Čak i kada se pronađe odgovarajući promet, prilagodba tog prometa drugačijoj mrežnoj topologiji može biti izuzetno teška zbog različitih rasporeda mrežnih uređaja i kretanja paketa unutar mreže. Ovaj problem dodatno komplicira prilagodbu napada različitim mrežnim konfiguracijama i topologijama.

Postoji mogućnost da neki napadi koji se žele uključiti u scenarij nisu moguće generirati jer ne postoji odgovarajući mrežni promet tog napada dostupan na internetu te ako se želi nadomjestiti taj mrežni promet simuliranjem tog napada i snimanjem tijekom izvršavanja, postoji šansa da je napad previše kompleksni. Nedostupnost tih napada ograničava sposobnost rješenja da simulira određene vrste prijetnji i testira učinkovitost sigurnosnih mjera protiv njih.

6. Zaključak

U ovom diplomskom radu istražena je metodologija generiranja mrežnog prometa napada na mrežu korištenjem mrežnih topologija i scenarija napada. Kroz analizu različitih alata i tehnika, razvijen je programski alat koji omogućuje simulaciju različitih vrsta napada u kontroliranom okruženju.

Korištenjem definiranih scenarija napada, omogućeno je simuliranje složenih napada u različitim fazama, počevši od prikupljanja informacija o mreži, preko inicijalnog kompromitiranja sustava, sve do izvlačenja osjetljivih podataka i održavanja pristupa. Ova modularna struktura omogućuje fleksibilnost i prilagodljivost u simulaciji različitih scenarija napada, što je ključno za analizu sigurnosti mreže i identifikaciju ranjivosti.

Kroz implementaciju programa za generiranje mrežnog prometa napada, koristeći *Python* i relevantne biblioteke poput Scapy-a i NetworkX-a, postignuta je efikasna simulacija različitih napada u raznim mrežnim topologijama. Integracija s `YAML` formatom omogućuje jednostavno definiranje i konfiguraciju scenarija napada, čime se olakšava upravljanje parametrima napada i prilagodba specifičnim zahtjevima.

Pronalazak odgovarajućih javno dostupnih skupova podataka za simulaciju mrežnog prometa napada bio je izazovan zbog njihove ograničene dostupnosti i reprezentativnosti. Kao rezultat toga, neki scenariji napada zahtijevali su ručno generiranje, što je zahtijevalo dodatne napore i vrijeme te moglo dovesti do varijacija u kvaliteti simulacija.

Ovi izazovi naglašavaju potrebu za daljnjim istraživanjem i razvojem resursa za simulaciju mrežnih napada te za alatima i metodologijama koji olakšavaju generiranje autentičnih i raznovrsnih napada u kontroliranom okruženju.

Ograničenja implementacijskog rješenja za generiranje mrežnih napada, opisana u tekstu ističu potrebu za daljnjim istraživanjem i razvojem kako bi se poboljšala raznovrsnost uzoraka, performanse pri velikim scenarijima i točnost simulacije. Proširenje baze uzoraka napada, optimizacija algoritama za generiranje i modifikaciju paketa, te razvoj metoda za točniju simulaciju kompleksnih napada su ključni koraci ka unapređenju ovakvih rješenja. Unatoč trenutnim ograničenjima, ovakva rješenja pružaju vrijedne alate za testiranje sigurnosti mrežnih sustava i identifikaciju potencijalnih slabosti.

Kroz ovaj rad, istaknuta je važnost simulacija napada u testiranju i poboljšanju sigurnosti mrežnih sustava. Daljnje istraživanje u ovom području može doprinijeti razvoju naprednih alata i tehnika za obranu od napada te osigurati stabilnost i pouzdanost mrežnih infrastruktura u digitalnom dobu.

7. Literatura

[1] »Check Point,« 10. 1. 2022.. [Mrežno]. Available: https://blog.checkpoint.com/security/check-point-research-cyber-attacks-increased-50-year-over-year/. [Pokušaj pristupa 6. 6. 2024.].

[2] »Indeed,« 8. 12. 2023.. [Mrežno]. Available: https://ca.indeed.com/career-advice/finding-a-job/soc-analyst. [Pokušaj pristupa 6. 6. 2024.].

[3] »The UNSW-NB15 Dataset,« University of New South Wales, 2015. [Mrežno]. Available: https://research.unsw.edu.au/projects/unsw-nb15-dataset. [Pokušaj pristupa 6. 6. 2024.].

[4] »Keysight,« [Mrežno]. Available: https://www.keysight.com/us/en/products/network-test/network-test-hardware/perfectstorm.html. [Pokušaj pristupa 6. 6. 2024.].

[5] »UNSW Canberra,« 2024.. [Mrežno]. Available: https://www.unsw.edu.au/canberra/our-research/our-facilities/Australian-cyber-security-centre-labs. [Pokušaj pristupa 6. 6. 2024.].

[6] »Common Vulnerabilities and Exposures,« 2024.. [Mrežno]. Available: https://www.cve.org/. [Pokušaj pristupa 6. 6. 2024.].

[7] »Argus,« QOSIENT, LLC., 2024.. [Mrežno]. Available: https://www.openargus.org/. [Pokušaj pristupa 10. 6. 2024.].

[8] »Zeek,« 2024.. [Mrežno]. Available: https://zeek.org/. [Pokušaj pristupa 10. 6. 2024.].

[9] »CSE-CIC-IDS2018,« University of New Brunswick, 1. 2018. [Mrežno]. Available: https://www.unb.ca/cic/datasets/ids-2018.html. [Pokušaj pristupa 6. 6. 2024.].

[10] »Communications Security Establishment Canada,« [Mrežno]. Available: https://www.cse-cst.gc.ca/en. [Pokušaj pristupa 6. 6. 2024.].

[11] »Canadian Institute for Cybersecurity,« [Mrežno]. Available: https://www.unb.ca/cic/. [Pokušaj pristupa 6. 6. 2024.].

[12] »CICFlowMeter (formerly ISCXFlowMeter),« University of New Brunswick, 4. 2017.. [Mrežno]. Available: https://www.unb.ca/cic/research/applications.html. [Pokušaj pristupa 6. 6. 2024.].

[13] Rapid7, »Metasploitable 2,« [Mrežno]. Available: https://docs.rapid7.com/metasploit/metasploitable-2/. [Pokušaj pristupa 8. 6. 2024.].

[14] »Wireshark,« [Mrežno]. Available: https://www.wireshark.org/. [Pokušaj pristupa 6. 6. 2024.].

[15] »Python Official Website,« 6. 6. 2024. [Mrežno]. Available: https://www.python.org/.

[16] »NetworkX,« NetworkX developers, 2024.. [Mrežno]. Available: https://networkx.org/. [Pokušaj pristupa 6. 7. 2024.].

[17] PyPI, »PyYAML,« 2024.. [Mrežno]. Available: https://pypi.org/project/PyYAML/. [Pokušaj pristupa 7. 6. 2024.].

[18] »Scapy,« Scapy community, 2024.. [Mrežno]. Available: https://scapy.net/. [Pokušaj pristupa 7. 6. 2024.].

Parametrizacija napadačkog prometa i primjena u zadanim topologijama

Sažetak

Ovaj rad istražuje izazove treniranja algoritama strojnog učenja za detekciju mrežnih napada, posebno zbog nedostatka malicioznog prometa u dostupnim skupovima podataka. Većina mrežnog prometa je benigni, a identificiranje malicioznog prometa je izazovno. Tradicionalni pristupi snimanja prometa tijekom penetracijskih testova ili umjetnog generiranja napadačkog prometa često brzo zastarijevaju. U radu su analizirani javno dostupni skupovi podataka poput UNSW-NB15, CSE-CIC-IDS2018 i Kitsune Network Attack dataset, te ručno snimljeni skupovi podataka. Tehnike generiranja napada uključuju filtriranje javno dostupnih podataka i ručno generiranje napada, dok je modifikacija napada ključna za prilagodbu parametrima, skaliranje, promjenu intenziteta i upravljanje s vremenom napada. Razvijeno je programsko rješenje za generiranje napadačkog prometa prema topologijama formata CCS. Opisana je implementacija lokalne baze uzoraka napada, formatiranje CCS topologije, parsiranje scenarija, generiranje napada i kreiranje PCAP datoteka. Prikazani su primjeri korištenja, rezultati i ograničenja rješenja, uz smjerove za daljnje istraživanje.

Ključne riječi: SOC analitičar, skup podataka, simulacija, mrežna topologija, parametrizacija, CCS

Parametrization of Attack Traffic and Application in Given Topologies

Abstract

This paper explores the challenges of training machine learning algorithms for detecting network attacks, particularly due to the lack of malicious traffic in available datasets. Most network traffic is benign, and identifying malicious traffic is challenging. Traditional approaches of recording traffic during penetration tests or artificially generating attack traffic often quickly become outdated. This study analyzes publicly available datasets such as UNSW-NB15, CSE-CIC-IDS2018, and Kitsune Network Attack dataset, as well as manually recorded datasets. Attack generation techniques include filtering publicly available data and manually generating attacks, with attack modification being crucial for parameter adjustment, scaling, intensity variation, and timing control of attacks. A software solution was developed to generate attack traffic according to CCS topology formats. The implementation of a local attack sample database, CCS topology formatting, scenario parsing, attack generation, and PCAP file creation are described. Examples of usage, results, and solution limitations are presented, along with directions for further research.

Keywords: SOC analyst, dataset, simulation, network topology, parametrization, CCS