KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

Andrius Madeliauskas

DDOS ATAKŲ APTIKIMAS ANALIZUOJANT BGP MARŠRUTŲ ANOMALIJAS

Baigiamasis magistro darbas

Vadovas
Doc. dr. Algimantas Venčkauskas

KAUNAS, 2019

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
KOMPIUTERIŲ KATEDRA

DDOS ATAKŲ APTIKIMAS ANALIZUOJANT BGP MARŠRUTŲ ANOMALIJAS

Baigiamasis magistro darbas
Informacijos ir informacinių technologijų sauga (kodas 6211BX008)

Vadovas
(parašas) Doc. dr. Algimantas Venčkauskas
(data)

Recenzentas
(parašas) Doc. dr. Gedeiminas Činčikas
(data)

Projektą atliko
(parašas) Andrius Madeliauskas
(data)

KAUNAS, 2019

KAUNO TECHNOLOGIJOS UNIVERSITETAS

Informatikos fakultetas

(Fakultetas)

Andrius Madeliauskas

(Studento vardas, pavardė)

Informacijos ir informacinių technologijų sauga (6211BX008)

(Studijų programos pavadinimas, kodas)

„DDoS atakų aptikimas analizuojant BGP maršrutų anomalijas“

AKADEMINIO SĄŽININGUMO DEKLARACIJA

20 ___ m. _____ ____ d.

Kaunas

Patvirtinu, kad mano **Andriaus Madeliausko** baigiamasis projektas tema „DDoS atakų aptikimas analizuojant BGP maršrutų anomalijas“ yra parašytas visiškai savarankiškai, o visi pateikti duomenys ar tyrimų rezultatai yra teisingi ir gauti sąžiningai. Šiame darbe nei viena dalis nėra plagijuota nuo jokių spausdintinių ar internetinių šaltinių, visos kitų šaltinių tiesioginės ir netiesioginės citatos nurodytos literatūros nuorodose. Įstatymų nenumatytų piniginių sumų už šį darbą niekam nesu mokėjęs.

Aš suprantu, kad išaiškėjus nesąžiningumo faktui, man bus taikomos nuobaudos, remiantis Kauno technologijos universitete galiojančia tvarka.

(vardą ir pavardę įrašyti ranka)	*(parašas)*

Madeliauskas, A. „DDoS atakų aptikimas analizuojant BGP maršrutų anomalijas“. Magistro baigiamasis projektas / vadovas doc. dr. Algimantas Venčkauskas; Kauno technologijos universitetas, Informatikos fakultetas, Kompiuterių katedra.

Mokslo kryptis ir sritis: Informatikos inžinerija, Technologiniai mokslai

Reikšminiai žodžiai: *DDOS, BGP, maršrutų parinkimas, mašininis mokymas, „Traceroute“*

Kaunas, 2019. 44 p.

SANTRAUKA

Kokybiškas interneto tinklų darbas yra aktualus šių laikų verslui, pramonei, švietimui, mokslui, medicinai ir visuomenei. Interneto tinklai reguliariai susiduria su įvairiais trikdžiais – tarp jų ir DDoS atakos. Augant tinklams auga ir atakos, dėl to kyla poreikis jas aptikti ir apsisaugoti. Šio baigiamojo darbo tikslas yra sukurti sistemą, leidžiančią aptikti DDoS atakas stebint BGP tinklų maršrutus. Siekiant įvykdyti tikslą iškeliami uždaviniai išanalizuoti aktualias temas, reikalingas darbo realizavimui.

Darbe analizuojama BGP maršrutų ir tinklų veikimas ir DDoS bei kitų stambaus masto įvykių įtaka jų veiklai. Nagrinėjami aktualiausi tinklų analizavimo, maršrutų stebėjimo bei DDoS atakų aptikimo metodai. DDoS atakų aptikimui parenkamas tikrinamų duomenų lyginimas su standartinio bei atakų srauto pavyzdžiais. Lyginimams atlikti analizuojami populiariausi mašininio mokymosi algoritmai, skirti duomenų klasifikavimui. Darbo realizacijai parenkami 4 skirtingi algoritmai.

Realizavimo metu sukuriama sistema, leidžianti automatiškai rinkti duomenis apie adresus ir maršrutus iki jų ir po to, pagal poreikį analizuoti jų duomenis. Duomenų analizei paruošiama 100 adresų ir maršrutų. Galutiniam analizės modulio testavimui parenkami du maršrutai. Analizės modulis sėkmingai praneša apie galimą ataką, o užtikrintumo koeficientas svyruoja nuo 60% iki 100%, priklausomai nuo tiriamų duomenų.

Madeliauskas, A. Detection of DDOS Attacks by Analyzing BGP Route Anomalies: Master's thesis in Information and Information Technology Security / supervisor assoc. prof. Algimantas Venčkauskas. The Faculty of Informatics, Kaunas University of Technology.

Research area and field: Informatics Engineering, Technological Sciences

Key words: *DDOS, BGP, routing, machine learning, "Traceroute"*

Kaunas, 2019. 44 p.

SUMMARY

Quality of online networking is relevant to today's business, industry, education, science, medicine and society. Internet networks are regularly confronted with various disruptions - including DDoS attacks. As networks grow, attacks also grow, creating the need to detect and protect them. The goal of this final thesis is to create a system that detects DDoS attacks by monitoring the routes of BGP networks. In order to fulfill the goal, the tasks are set to analyze the relevant topics necessary for the realization of the system.

The work analyzes the operation of BGP routing and networking and the influence of DDoS and other large-scale events on their operation. The most current methods of network analysis, route tracking and DDoS attack detection are analyzed. For DDoS attack detection, the comparison of the data samples of standard traffic and attack traffic is chosen. The most popular machine learning algorithms for data classification are analyzed for this comparison. 4 different algorithms are selected for the realization of the work.

During implementation, a system is designed to automatically collect data on addresses and routes. Analyze module is developed to analyze and provide data on request. 100 addresses and routes are prepared for data analysis. Two routes are selected for the final test of the analysis module. The analysis module successfully reports about a potential attack, with a confidence factor ranging from 60% to 100%, depending on the data being investigated.

TURINYS

Įvadas 12
1. Interneto tinklų veikimo ir stebėjimo analizė 14
1.1. Analizės tikslas 14
1.2. Tyrimo objektas, sritis ir problema 14
1.3. DDoS atakos 14
1.3.1. DDoS veikimas 15
1.3.2. DDoS tipai 15
1.4. Kiti stambaus masto tinklų įvykiai 16
1.5. Tinklo stebėjimo metodai 17
1.5.1. Paketų skenavimas ir analizavimas 18
1.5.2. Srautų skenavimas 19
1.5.3. Tinklų maršruto parinkimas, BGP 19
1.6. Maršrutų stebėjimo metodai 22
1.6.1. RIB stebėjimas. 23
1.6.2. „Traceroute“ stebėjimas 23
1.6.3. BGP sesijų stebėjimas 24
1.7. Atakų atpažinimo metodai 25
1.7.1. Šaltinio IP adresų metodas 25
1.7.2. Bazinių srauto profilių metodas 26
1.7.3. Atstumo stebėjimo metodas 26
1.7.4. MULTOPS metodas 26
1.8. Tinklo analizės išvados 27
2. Tinklo duomenų analizė naudojant mašininį mokymąsi 29
2.1. Mašininis mokymas 29
2.2. Duomenų klasifikavimas ir jo algoritmai 30
2.2.1. Logistinės regresijos metodas 30
2.2.2. Naiviojo Bajeso metodas 31
2.2.3. KNN metodas 32
2.2.4. Sprendimų medžio metodas 32
2.2.5. Atsitiktinio miško metodas 33
2.2.6. Daugiasluoksnio perceptrono metodas 34
2.3. Metodo pasirinkimas 35
3. Sistemos struktūra 36
3.1. Sistemos platforma 36
3.2. Stebimas tinklas 37
3.3. Globali sistemos struktūra 37

3.4. Sistemos elementų veikimo specifika 38
3.4.1. Užklausų generavimo modulis 38
3.4.2. Duomenų bazės struktūra 39
3.4.3. Duomenų analizės modulis 40
3.5. Apibendrinimas 40
4. Sistemos realizacija 41
4.1. Maršrutų statistika 41
4.2. Užklausų generavimo modulis 42
4.3. Duomenų analizės modulis 45
5. Sistemos rezultatų analizė 48
5.1. Duomenų rinkinys 48
5.2. Statistinių duomenų analizė ir žymėjimas 48
5.3. Analizės modulio rezultatai 51
5.3.1. Maršrutas „howtogermany.com“ 51
5.3.2. Maršrutas „s-bahn-berlin.de“ 52
5.4. Išvados ir pastabos 53
Išvados 54
Literatūra 55
Priedai 56

LENTELIŲ SĄRAŠAS

3.1 lentelė. Duomenų lentelės pavyzdys 39
5.1 lentelė. Pagal vėlinimą rūšiuotos reikšmės 49
5.2 lentelė. 2019-04-23 09:04 s-bahn-berlin.de statistikos imtis 50
5.3 lentelė. Reguliarios s-bahn-berlin.de statistikos pavyzdys 50
5.4 lentelė. Galutinis atakų žymėjimo skaičius 50
5.5 lentelė. Maršruto „howtogermany.com“ analizės rezultatai 51
5.6 lentelė. Maršruto „s-bahn-berlin.de“ analizės rezultatai 53

PAVEIKSLŲ SĄRAŠAS

1.1 pav. „Botnet“ ataka 15
1.2 pav. „Wireshark“ vartotojo sąsaja [5] 18
1.3 pav. BGP būsenų schema 20
1.4 pav. „Traceroute“ veikimo schema 24
1.5 pav. „BGPplay“ vartotojo sąsaja 25
1.6 pav. MULTOPS srauto rinkimo schema 27
2.1 pav. Logistinė kreivė [17] 31
2.2 pav. Pasirinkimų medžio struktūra 32
2.3 pav. Atsitiktinio miško struktūra 33
2.4 pav. Trisluoksnio perceptrono pavyzdys 34
3.1 pav. Sistemos komponentai 37
3.2 pav. Planuojama duomenų rinkimo modulio struktūra 38
4.1 pav. Užklausų generavimo modulio sekos diagrama 42
4.2 pav. Užklausų generavimo modulio blokinė schema 43
4.3 pav. Tikrinimo funkcija 44
4.4 pav. Duomenų analizės modulio sekos diagrama 45
4.5 pav. Duomenų analizės modulio blokinė schema 46
4.6 pav. Vektorių paruošia funkcija 47

TERMINŲ IR SANTRUMPŲ ŽODYNAS

AS	Autonominė sistema (angl. *Autonomous System*)
ASN	Autonominės sistemos numeris (angl. *Autonomous System Number*)
BGP	Ribinio tinklų sietuvo protokolas (angl. *Border Gateway Protocol*)
DB	Duomenų bazė (angl. *Database*)
DBVS	Duomenų bazės valdymo sistema (angl. *Database*)
DDoS	Paskirstyta paslaugos trikdymo ataka (angl. *Distributed Denial of Service*)
DNS	Sričių vardų sistema (angl. *Domain Name System*)
eBGP	Išorinis ribinio tinklų sietuvo protokolas (angl. *Exterior Border Gateway Protocol*)
EGP	Išorinis tinklų sietuvo protokolas (*angl. Exterior Gateway Protocol*)
GB	Gigabaitas(ai) (angl. *Gigabyte(s)*)
HDFS	Paskirstyta „Hadoop“ failų sistema (angl. *Hadoop Distributed File System*)
HTTP	Hipertekstų persiuntimo protokolas (angl. *Hypertext Transfer Protocol*)
iBGP	Vidinis ribinio tinklų sietuvo protokolas (angl. *Interior Border Gateway Protocol*)
ICMP	Interneto kontrolės žinučių protokolas (angl. *Internet Control Message Protocol*)
IETF	Interneto inžinerijos darbo grupė (angl. *Internet Engineering Task Force*)
IGP	Vidinis tinklų sietuvo protokolas (angl. *Interior Gateway Protocol*)
IP	Internetinis protokolas (angl. *Internet Protocol*).
ISP	Interneto paslaugos tiekėjas (angl. *Internet Service Provider*)
IT	Informacinės technologijos
JSON	Atviro standarto formatas, perduodantis duomenų objektus, sudarytus iš atributo ir reikšmės porų (angl. *JavaScript Object Notation*)
KB	Kilobaitas(ai) (angl. *Kilobyte(s)*)
KNN	K artimiausių „kaimynų“ (angl. *K-Nearest Neighbours*)
MLP	Daugiasluoksnis perceptronas (angl. *Multilayer perceptron*)
MMSE	Minimali vidutinė kvadratinė paklaida (angl. *minimum mean square error*)
MTR	Angl. „My Traceroute“
MULTOPS	MUltiLevel Tree for Online Packet Statistics
NTP	Tinklo laiko protokolas (angl. *Network Time Protocol*)

OS Operacinė sistema (angl. *Operating System*).

OSI Lygmeninis tinklų modelis (angl. *Open Systems Interconnection Reference Model*)

RAM Kompiuterio operatyvioji atmintinė (angl. *Random Access Memory*)

RFC IETF organizacijos dokumentas, kuriame pateikiamos techninės ir organizacinės pastabos apie internetą (angl. *Requests for Comments*)

RIB Maršrutų parinkimo informacinė bazė (angl. *Routing Information Base*)

RIPE Regioninis registras, atsakingas už Europos ir dalies Azijos IP adresų valdymą (pranc. *Réseaux IP Européens*)

RIR Regioninė interneto registratūra (angl. *Regional Internet Registry*)

TCP perdavimo valdymo protokolas (angl. *Transmission Control Protocol*)

TTL Gyvavimo trukmė (angl. *Time to Live*)

UDP Vartotojo datagramos protokolas (angl. *User Datagram Protocol*)

ĮVADAS

Šio darbo tema yra iškelta studijuojant informacijos ir informacinių technologijų saugą.

Interneto paslaugos ir kompiuteriniai tinklai yra neatsiejama pasaulio dalis. Tai sistema, kuri neabejotinai svarbi įvairiose situacijose. Pagrindinė interneto funkcija yra bendravimas tarp skirtingų įvairiu atstumu nutolusių įrenginių, sistemų, asmenų. Niaudamiesi internetu, įvairūs vartotojai gali pasiekti milžinišką kiekį informacijos, tokios kaip mokslinės žinios, mokomoji medžiaga, naujienos, žiniasklaidos informacija ir t.t. Globalių tinklų pagalba įvairios organizacijos gali sujungti savo skyrius, darbuotojus iš viso pasaulio ir taip bendrauti su kitomis organizacijomis. Internetas yra svarbi pasaulinės ekonomikos ir bankininkystės dalis.

Globalūs tinklai yra svarbi mūsų pasaulio dalis, dėl to yra siekiama jų sklandžios veiklos. Be abejo, tokio mąsto sistemos sulaukia nuolatinių sutrikimų, sukeltų įvairų priežasčių. Neretai viena iš priežasčių yra kenkėjiška asmenų ar asmenų grupių atliekama veikla, kurios tikslas yra pakenti šiems tinklams, juos sutrikdyti. DDoS atakos yra viena iš tokio tipo kenkėjiškų veiklų.

Šiame darbe bus nagrinėjama DDoS atakų įtaka stambiems kompiuteriniams tinklams ir dėl to kilusioms anomalijoms, siekiant aptikti naujas atakas. Tokio tipo veikla yra aktuali įvairioms organizacijoms, teikiančioms IT paslaugas ar turinčioms didelius ir paskirstytus tinklus. Taip pat DDoS atakų aptikimas yra aktualus organizacijoms, stebinčioms bendrą interneto būklę, saugumą, statistikos kompanijoms.

Magistro darbo tikslas – suprojektuoti sistemą, leidžiančią aptikti DDoS atakas stebint BGP tinklų maršrutus.

Šio darbo uždaviniai:

- išnagrinėti BGP protokolo veikimo principą;
- išsiaiškinti DDOS atakų ir kitų stambaus masto įvykių įtaką tinklų maršrutams;
- išanalizuoti esamus maršrutų stebėjimo metodus ir sprendimus;
- sudaryti pasirinktų BGP maršrutų stebėjimo ir analizavimo sistemą;
- surinkti duomenis apie BGP maršrutų pasikeitimus ir juos konsoliduoti;
- išanalizuoti gautus rezultatus ir pateikti išvadas apie sistemos veiklą.

Sėkmingai įvykdžius projektą bus sukurta sistema, kuri leis atpažinti DDoS atakas, vykstančias dideliuose tinkluose. Tai būtų unikalus įrankis, kuris leistų įvairioms organizacijoms stebėti didelius, per šalis, žemynus ar visą pasaulį paskirstytus tinklus ir užfiksuoti, kad prasideda naujos atakos.

Darbas yra sudarytas iš 5 skyrių. Apžvelgsime kiekvieną iš jų.

Interneto tinklų veikimo ir stebėjimo analizė. Analizės metu nagrinėjama BGP tinklų veikimas ir DDoS atakų įtaka tinklų būklei. Siekiant geresnio supratimo, taip pat bus nagrinėjama ir kitų stambaus mąsto tinklo įvykių įtaka jiems. Prieš kuriant sprendimą, apžvelgsime kuo daugiau

metodų skirtų DDoS atakoms aptikti tinkluose ir apibendrinus pasirinksime tinkamą sprendimą, kurį galima tobulinti ir plėtoti.

Tinklo duomenų analizė naudojant mašininį mokymąsi. Mašininio mokymosi analizėje bus nagrinėjamas mašininio mokymosi veikimas, kokie mokymosi tipai tinka atakų aptikimui. Pagrindinė nagrinėjamų algoritmų grupė bus prižiūrimo mašininio mokymosi algoritmai, tokie kaip pasirinkimų medis, KNN, logistinė regresija, atsitiktinis medis.

Sistemos struktūra. Šis skyrius yra orientuotas į kuriamos sistemos planavimą ir projektavimą. Skyriuje bus nagrinėjama, kokią platformą reikėtų pasirinkti, analizuojama kokius tinklus stebėti geriausia. Taip pat bus projektuojama sistemos modulio struktūra ir veikimo principas.

Sistemos realizacija. Realizacijos skyrius yra praėjusio skyriaus tęsinys. Šiame skyriuje išsamiai nagrinėsime sistemos struktūrą, kaip ji bendrauja su kitomis sistemomis. Aprašysime duomenų rinkimo bei analizės modulių algoritmus.

Sistemos rezultatų analizė. Įvykdžius realizaciją, bus atliekama rezultatų analizė. Rezultatų analizė paskirstyta į du etapus: statistikos duomenų nagrinėjimas ir duomenų analizė, panaudojant mašininio mokymosi algoritmus.

1. INTERNETO TINKLŲ VEIKIMO IR STEBĖJIMO ANALIZĖ

Kadangi šio darbo tikslas yra DDoS ir kitų stambaus mąsto atakų aptikimas, analizuojant BGP maršrutų anomalijas, šiame skyriuje bus apžvelgiamas darbo aktualumas ir išanalizuojama esama situacija bei būdai problemai spręsti.

1.1. Analizės tikslas

Šio darbo analizės tikslas yra išnagrinėti BGP protokolo veikimą ir DDoS, bei kitų atakų įtaką jam.

Analizės tikslui pasiekti reikia įvykdyti šiuos uždavinius:

- išnagrinėti DDoS atakų veikimą;
- išnagrinėti BGP protokolo veikimo principą;
- išsiaiškinti DDoS atakų ir kitų stambaus masto įvykių įtaką tinklų maršrutams;
- išanalizuoti esamus maršrutų stebėjimo metodus ir sprendimus.

1.2. Tyrimo objektas, sritis ir problema

Iš analizės tikslo galima spręsti, kad tyrimo metu pagrindinė nagrinėjama sritis bus tinkų maršruto parinkimas. Į tai įeis BGP maršruto parinkimo protokolas, bei didelio mąsto įvykių, tokių kaip DDoS atakos, sąveika su BGP maršrutais.

Stambaus mąsto tinklų atakos yra sunkiai sustabdomos ir išvengiamos. Šios atakos sukelia didelio masto nuostolius ir problemas. Norint surasti būdą atpažinti atakas ir jų išvengti, šioje analizėje bus nagrinėjama, kokiais metodais galime stebėti tinklus ir atpažinti juose vykstančias atakas.

1.3. DDoS atakos

Prieš pradedant analizuoti DDoS atakų stebėjimą tinkle, pirmiausia išsamiai išanalizuosime pačias DDoS atakas. Trumpai peržvelgsime jų veikimo principą, žalą ir kitus bruožus.

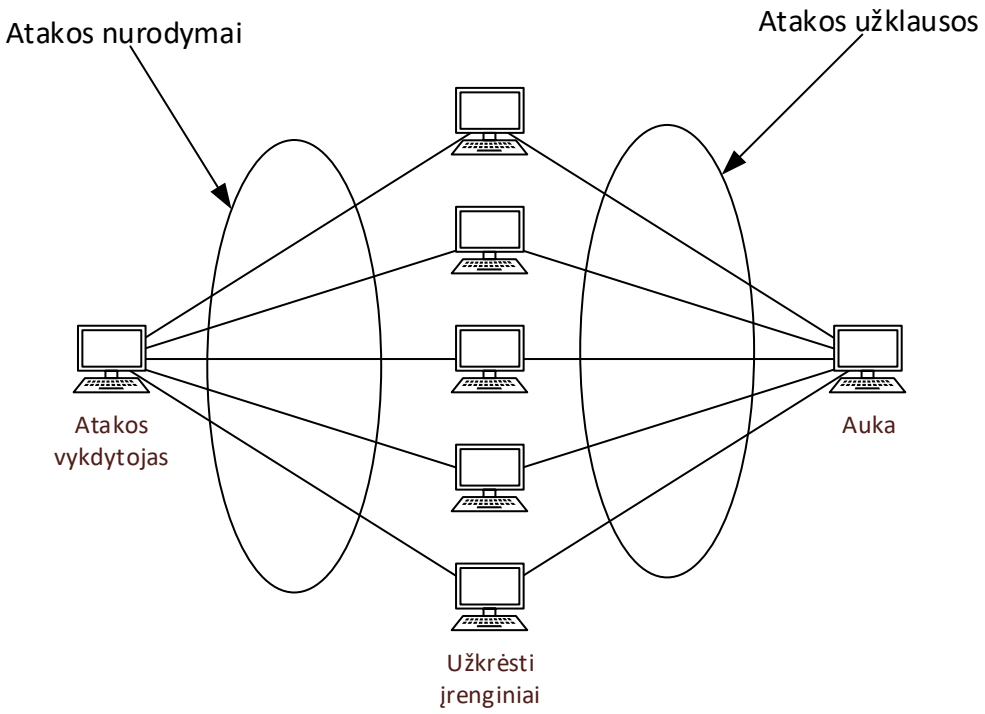
Paskirstyta paslaugos trikdymo ataka (angl. *Distributed Denial of Service, DDoS*) yra kibernetinė ataka, kurios tikslas yra sutrikdyti internetinės paslaugos ar sistemos darbą, apkraunant ją didžiuliu srauto kiekiu, ateinančiu iš įvairių šaltinių. Šis srauto kiekis gali „užkimšti“ interneto linijas ar visiškai užpildyti RAM, dėl to aukos sistema gali visiškai nustoti veikti.

DDoS gali būtų naudojamos keliems tikslams. Pirma, atakos vykdytojas gali prievarta išgauti pinigų iš atakos aukų. Taip pat atakos, nutaikytos į vieną įmonę, gali suteikti naudos konkurentams. Panašiai, priešiškai nusiteikę subjektai gali turėti naudos ir iš valstybinių sistemų sutrikdymo.

1.3.1. DDoS veikimas

Norint sėkmingai įvykdyti DDoS ataką, jos organizatorius turi turėti prie interneto prijungtų įrenginių kontrolę. Kompiuteriai, serveriai ir kiti įrenginiai yra užkrečiami kenkėjiška programine įranga, taip juos paverčiant „botnet“ tinklo dalimi.

Kai „botnet“ tinklas yra paruoštas atakai, atakos vykdytojas per atstumą gali tiesiogiai perduoti atnaujintas komandas tinklo nariams. Kai pasirenkamas aukos IP adresas, „botnet“ nariai siunčia užklausas į šį IP adresą ir taip apkrauna aukos sistemą ir tinklą (1.1 pav.).



1.1 pav. „Botnet“ ataka

1.3.2. DDoS tipai

DDoS atakas galima suskirstyti į tris tipus: srautinės atakos, programų lygmens atakos, protokolų atakos. [1] Peržvelgsime kiekvieną iš šių tipų su keletu pavyzdžių.

Srautinės atakos

Srautu paremtos atakos (angl. *volume-based attack*) yra skirtos panaudoti visą galimą tinklo pralaidumą tarp aukos ir išorinio interneto. Tam gali būti panaudojami skirtingų tipų paketai ir metodai. Kaip keletas pavyzdžių gali būti UDP protokolo antplūdis, ICMP („ping“) paketų antplūdis ar DNS paketų srautas, panaudojant nesaugius DNS serverius ir kita. Toliau išsamiau aprašomi keletas šių pavyzdžių.

DNS stiprinimas. Ataka, kurios metu atakos sukėlėjas išsiunčia daugybę užklausų apie DNS įrašą į DNS serverius, savo adresą suklastodamas kaip aukos. Tada gautas atsakymas iš DNS serverio keliauja tiesiai į aukos kompiuterį ir yra daug didesnis, nei atakuojančios pusės išsiųstas srautas.

ICMP antplūdis. Ataka, kuri nėra stiprinama protokolo, tačiau veikia panašiu principu, kaip ir kitos. Paketas yra siunčiamas iš netikro IP adreso ir tada atsakymai keliauja į tą adresą, kuris būna

aukos kompiuterio. „Smurf“ antplūdžio versija veikia panašiai, kaip paprastas ICMP antplūdis, tačiau yra ir atakos stiprinimas panaudojant transliavimo (angl. *broadcast*) tinklus.

NTP stiprinimas. Laiko sinchronizavimo protokolas yra vienas iš protokolų, kurį galima lengvai panaudoti atakos stiprinimui. NTP serveriui reikia labai nedidelės užklausos, o gaunamas atsakymas labai didelis.

„**Memcached“ ataka**. Dar viena, ganėtinai nauja, lengvai „auginama“ ataka. „Memcached“ yra vidinis serverių procesas atminties valdymui. Daugelis šių procesų, nesukonfigūravus tvarkingai, turi atvirą tinklo prievadą. Palyginus su kitomis atakomis, ši turi milžinišką augimo daugiklį – 51000. Tai reiškia, kad išsiuntus 15 baitų užklausą galima gauti 750 KB atsakymą. Tai leidžia panaudoti šį protokolą milžiniško mąsto atakoms. Palyginimui, DNS daugiklis yra 179, o NTP - 556.9. [2]

Programų lygmens atakos

Šių atakų, kitaip dar vadinamų 7-jo OSI lygmens atakomis, tikslas sunaudoti aukos įrangos resursus. Atakos yra nutaikytos į sistemos dalį, kurioje yra generuojamas saityno turinys. Pavienės HTTP užklausos šioms sistemoms daug problemų nekelia, tačiau generuojant didelį kiek užklausų į saityno turinį, galima greitai sutrikdyti sistemos darbą, ypač tada, kai užklausos susiję su duomenų bazėmis. Gali pasitaikyti ir kitokių metodų, pavyzdžiui „Slowloris“ ataka, kurios tikslas išlaikyti aktyvų prisijungimą kuo ilgiau ir tuo pačiu didinti atidarytų prisijungimų skaičių. [3]

Protokolų atakos

Protokolų atakos, kitaip dar žinomos, kaip išsekinimo atakos (angl. *state-exhaustion attack*) yra skirtos sutrikdyti paslaugą ar sistemą, pripildant būsenos lenteles saityno serveriuose arba sunaudojant užkardų ar srauto balansavimo sistemų resursus. Vienas iš pavyzdžių, dažnai naudojama „SYN“ paketų antplūdžio ataka. [4] Gavusi „SYN“ paketą, sistema siunčia „SYN-ACK“ paketą ir tikisi gauti „ACK“ atgal, tačiau atakuojantys įrenginiai neatsako ir sistema lieka su daug neužbaigtų užklausų. Taip atakos sukėlėjas gali užpildyti prisijungimų lentelę aukos sistemoje ir sutrikdyti tolimesnę tinklo veiklą.

1.4. Kiti stambaus masto tinklų įvykiai

Jau nagrinėjome, kokių tipų DDoS atakos yra vykdomos siekiant sutrikdyti tinklo įrenginių veiklą. Tačiau nepasiekiamas serveris ar tinklo dalis ne visada yra DDoS ataka. Interneto tinklai yra sudėtinga sistema, kuri gali būti ištikta įvairių problemų, sukeliančių ryšio sutrikimus. Toliau nagrinėsime galimus tinklo sutrikimus ir jų įtaką tinklo įrenginių pasiekiamumui.

Tinklų perpildymas

Tinklo perpildymas yra viena iš dažniausių tinklo sutrikimų priežasčių. Jis yra sukeliamas, kai per daug tinklo vartotojų bando prisijungti vienu metu. Tuo metu tam tikroje zonoje tinklo įranga nesugeba apdoroti sukuriamo srauto ir atsiranda didelis vėlinimas, paketų praradimas ar net visiškas

tinklo veiklos nutrūkimas. Pagal sugeneruojamo srauto vietą, tinklų perpildymą taip pat galima skirstyti į du tipus: sukeltas didelio išeinančio srauto ir sukeltas didelio ateinančio srauto.

Didelio išeinančio srauto perpildymas dažniausiai įvyksta, kai viešose vietose prisijungia daug vartotojų, pavyzdžiui, viešieji įrenginiai, konferencijos, bibliotekos, aukštųjų mokyklų bendrabučiai ir panašiai.

Didelio ateinančio srauto perpildymai pasižymi per dideliu srauto kiekiu ateinančiu į tam tikrą tinklą ar serverį. Perpildymai sukeliami atsiradus dideliam paslaugos ar serverio populiarumui. Tai gali būti naujų paslaugų ar produktų teikimas, akcijos internetinėse parduotuvėse, renginių registravimas ir panašiai. Reikia paminėti, kad didelio ateinančio srauto tinklo perpildymai yra labai panašūs į srautines arba taikomojo lygmens DDoS atakas. Taip yra dėl to, kad tiek vienas, tiek kitas trikdis yra sukeliamas darant daug užklausų iš skirtingų įrenginių į tą patį adresą.

Ryšio linijos nutrūkimas

Tokio tipo sutrikimai yra ganėtinai paprasti, tačiau reguliariai pasitaikantys. Ryšio linijų nutrūkimai dažniausiai pasireiškia fiziškai pažeidus ryšio kanalus. Dažniausiai pažeidžiamos linijos yra optiniai kabeliai vandenynuose, po žeme, duomenų centre ar netgi optikos kabelių jungtys. Ryšio linijų pažeidimai įvyksta tiek dėl stichinių nelaimių, tiek dėl žmonių veiklos, pavyzdžiui, kelių darbai, statybos, gali pasitaikyti ir kenkėjiškos, chuliganiškos veiklos. Nutrūkus ryšio linijoms, matomas visiškas tinklo nepasiekiamumas tam tikroje tinklo zonoje, tai gali būti ir keli serveriai, ir pusė šalies tinklo.

Įrangos gedimai

Dideli tinklai yra sudaryti iš daug įvairios tinklo įrangos. Kaip ir su visa kita įranga, tinklo įranga taip pat genda. Gedimai gali nutikti dėl įvairiausių priežasčių, pavyzdžiui, neteisingas įrangos naudojimas, įrengimas, įrangos nusidėvėjimas, katastrofos ir panašiai. Kaip ir įvykus ryšio linijų nutrūkimui, įrangos gedimas sukelia panašius padarinius: visiškas tinklo nepasiekiamumas tam tikroje tinklo zonoje.

Konfigūravimo klaidos

Dar viena dažna problema sukelianti ryšio sutrikimus yra žmonių padarytos klaidos. Tai gali būti įvairios konfigūravimo klaidos, prižiūrint tinklo įrenginius, pavyzdžiui, neteisingi IP adresai, užkardos konfigūravimas, blogai sukonfigūruoti prievadai, tinklo kilpos. Trikdžiai sukelti tokių klaidų gali būti labai įvairūs, tačiau dažniausi gali susidurti su visišku tam tikros tinklo dalies nepasiekiamumu arba dideliu paketų praradimu.

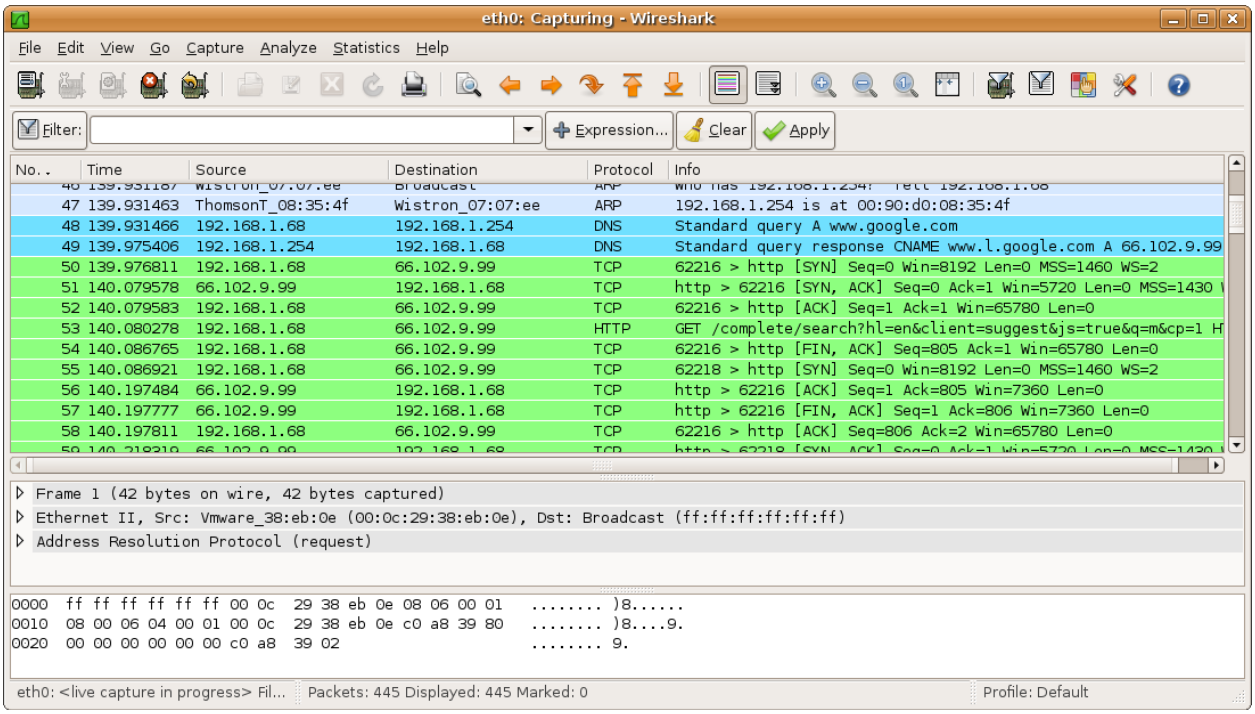
1.5. Tinklo stebėjimo metodai

Norint sėkmingai atrasti vykstančias DDoS atakas, reikalingas tinklo, kuriuo keliauja duomenys į sistemą, stebėjimas. Dažniausiai, nepaisant kokio tipo DDoS ataka vykdoma, jos įtaka daugiau ar mažiau atsispindi ir tinklo duomenyse, ir būsenoje.

Toliau bus analizuojami keli pagrindiniai tinklo stebėjimo metodai.

1.5.1. Paketų skenavimas ir analizavimas

Vienas iš klasikinių tinklo stebėjimo metodų yra paketų skenavimas ir analizavimas. Paketų skenavimą galima atlikti įvairiuose tinklo mazguose. Paketų skenavimui yra naudojami įvairūs programiniai arba net dedikuoti aparatiniai sprendimai, pavyzdžiui: „Ethereal“, „WinPcap“, „AirPcap“. Šios programinės įrangos sistemos leidžia skenuoti paketus ir analizuoti jų duomenis, pateiktus kadrais ir patogiai išskirstytus pagal OSI modelio lygmenis. Toliau pateikiama „WireShark“ analizavimo įrankio vartotojo sąsaja (1.2 pav.).



1.2 pav. „Wireshark“ vartotojo sąsaja [5]

Pats paketų skenavimas yra plačiai pritaikomas sprendimas, nes tinklo srauto stebėjimas antrame OSI lygmenyje yra paprastas. Šiuos duomenis galima matyti netgi esant paprastoms užkardoms, kadangi duomenų srautas vis vien keliauja iki įrenginio procesoriaus.

Kaip jau anksčiau minėta, didžiausias šio stebėjimo metodo privalumas yra plati pritaikymo galimybė, nereikalaujanti daug papildomos programinės ar aparatinės įrangos. Galima stebėti įvairias „UNIX“ arba „Windows“ sistemas.

Nepaisant to, šis metodas turi ir trūkumų. Pirma, skenavimo metu, fiksuojamas visas paketas, dėl to, norint laikyti ilgesnio laikotarpio duomenis (analizavimui ar statistikai), reikia adekvačiai galingos duomenų saugojimo sistemos, taip pat ir resursų duomenų apdorojimui. Taip pat stebėjimas šiuo metodu leidžia stebėti tik srautą, einantį per vieną įrenginį.

1.5.2. Srautų skenavimas

Srautų (angl. *flow*) skenavimas leidžia surinkti tinklu einančių paketų informaciją. Šio tyrimo metodo metu, labai panašiai, kaip ir paketų skenavimo metu, galima pastebėti įvairias tinklo anomalijas, kurios vyksta viename ar kitame įrenginyje.

Daugelis modernių maršruto parinktuvų ir komutatorių, veikiančių trečiame OSI lygmenyje turi galimybę eksportuoti srautų informaciją. Priklausomai nuo gamintojo, tai gali būti „NetFlow“, „sFlow“ ir „IPFIX“ standartai. Eksportuojant srauto paketų informaciją ir ją pateikiant tam tikrai programinei sistemai, galima matyti bendro pobūdžio informaciją, tokią kaip šaltinio ir tikslo adresai, prievadai, protokolas, TCP žymos (angl. *flags*), baitų ir paketų kiekis ir taip toliau.

Šis metodas leidžia efektyviau stebėti tą patį tinklą, lyginant su paketų skenavimu, kadangi surenkama tik reikalinga informacija, o ne visas paketas su pertekline informacija. Kadangi tai plačiai palaikomų standartų metodas, srautų skenavimas labai dažnai yra naudojamas kaip tinklo stebėjimas, ginantis nuo DDoS atakų. Srautų stebėjimą naudoja daugybė aukštos klasės ir atvirojo kodo sprendimų bei įrankių, skirtų apsaugoti nuo DDoS. Keletas pavyzdžių būtų „Andrisoft“ „Wanguard“, „FlowTraq“, atvirojo kodo „FastNetMon“ ir t.t.

Lyginant su paketų skenavimu, šis metodas yra daug efektyvesnis resursų sunaudojimo atžvilgiu. Atsikratant paketo turinio, atiduodama tik reikalinga informacija, taigi nereikia apdoroti perteklinės informacijos. Dėl šios priežasties galima šį metodą ir geriau vertinti privatumo ir saugos klausimu. [6]

Nepaisant to, kad srauto informacijos funkcijos ir standartai yra gan plačiai pritaikomi, įrenginių palaikančių šias funkcijas tikrai mažiau, lyginant su paketų skenavimu. Taip pat šis metodas, kaip ir anksčiau minėtas paketų skenavimas, leidžia stebėti tik tai, kas vyksta vidinio tinklo ribose.

1.5.3. Tinklų maršruto parinkimas, BGP

Toliau bus nagrinėjamas tinklų analizavimas, stebint maršruto parinkimo pakitimus. Prieš aiškinantis maršrutų stebėjimo metodus, būtina išanalizuoti BGP protokolo funkcijas ir veikimą.

Šiuo metu BGP yra pagrindinis protokolas, kurio pagrindu veikia viso interneto maršrutų valdymas. Pagrindinė ribinio tinklų sietuvo protokolo (angl. *Border Gateway Protocol*) funkcija yra dalintis tinklo pasiekiamumo informacija tarp BGP protokolą palaikančių sistemų. Į šią tinklo pasiekiamumo informaciją įeina autonominių sistemų (angl. *autonomous system, AS*) maršrutai, pasirinktinis kelias, įvairios žymos. Naudojantis BGP protokolu, šia informacija keičiasi interneto tiekėjai (ISP) arba kitos stambios IT struktūros (BGP tarp skirtingų AS vadinamas eBGP). Kiekvienas toks stambus tinklas turi savo individualų autonominės sistemos numerį (ASN), kuris yra paskiriamas regioninės interneto registratūros (angl. *Regional Internet Registry*), trumpai vadinamos RIR. Ši institucija taip pat reguliuoja IP adresų prisikyrimą tinklams.

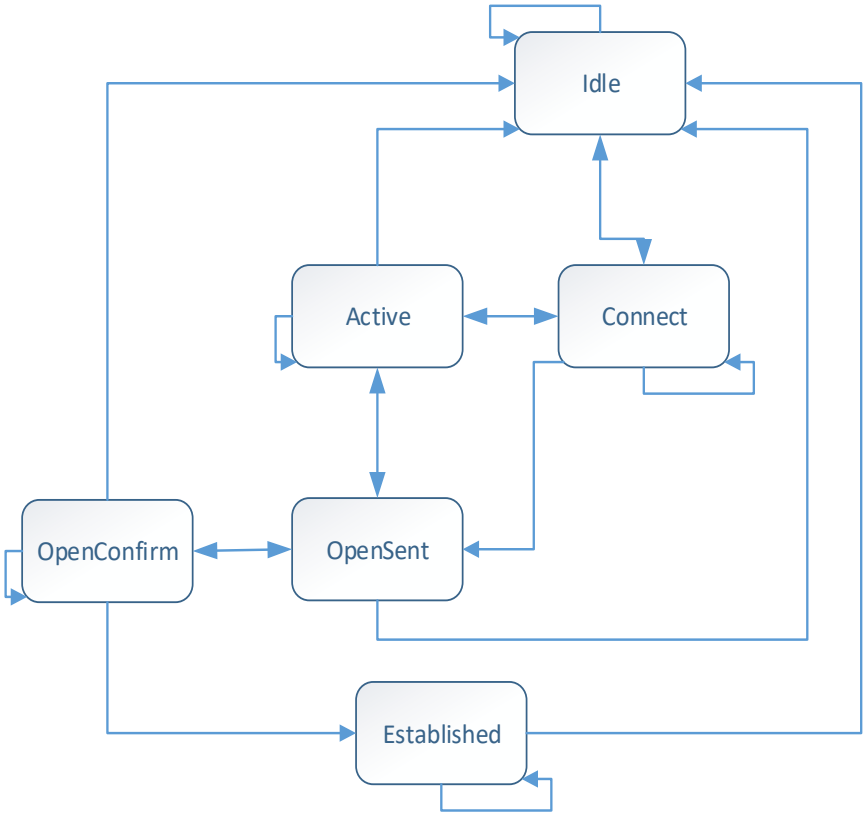
Kiekviena autonominė sistema dalinasi maršruto parinkimo informacija, naudodama užmegztą TCP BGP sesiją tarp pagrindinių AS įrenginių, pavyzdžiui, maršruto parinktuvų. Tai pat reikėtų paminėti, kad BGP protokolas gali būti naudojamas ir maršruto parinkimo autonominės sistemos viduje. Tokiu atveju šis vidinis BGP vadinamas iBGP (angl. *Interior Border Gateway Protocol*). Įrenginius, naudojančius šį protokolą, priimta vadinti BGP „kaimynais“ (angl. *neighbors*).

Šie „kaimynai“ rankiniu arba savarankiškai automatizuotu būdu užmezga BGP sesijas, kurios kaip jau minėta, naudoja TCP kaip perdavimo protokolą. Tai pašalina reikmę kurti papildomą fragmentavimo, persiuntimo ar patvirtinimo sistemą. BGP naudoja 179 TCP prievadą. BGP siuntėjas palaiko aktyvią sesiją numatytą laiko periodą, siųsdamas 19 baitų „keep-alive“ žinutes. [7]

BGP būsenos

Atliekant operacijų vykdymo sprendimus, naudojamas paprastas baigtinis automatas (angl. *„Finite-state machine“*), kuris susideda iš šešių būsenų (1.3 pav.):

- laisva būsena (angl. *Idle*);
- sujungimo (angl. *Connect*);
- aktyvi (angl. *Active*);
- „OpenSent“;
- „OpenConfirm“;
- užmegzta (angl. *Established*)



1.3 pav. BGP būsenų schema

Kiekvienai sesijai BGP naudoja būsenos kintamąjį, kuris leidžia sekti kurioje būsenoje yra sesija.

Pirmoji yra laisva būsena. Šioje būsenoje BGP inicijuoja visus resursus, atmeta ateinančias BGP ryšio užklausas ir tada bando inicijuoti TCP sesiją į reikiamą įrenginį. Sekanti būsena yra sujungimo būsena. Šioje būsenoje maršruto parinktuvas laukia sėkmingo TCP sujungimo užbaigimo ir tada pereina į „OpenSent“ būseną. Jei TCP sujungimas nesėkmingas, tada pradedamas „ConnectRetry“ laikmatis, kuris nusako pakartotinio prisijungimo bandymų laiką. Jei laikmatis pasiekia 0, grįžtama prie sujungimo būsenos. „OpenSent“ būsenoje maršruto parinktuvas siunčia atidarymo žinutę ir laukia atsakymo, kad galėtų pereiti į „OpenConfirm“ būseną. „OpenConfirm“ laukia „keep-alive“ pranešimo. Apsikeitus „keep-alive“ žinutėmis BGP sesija pereina į užmegzto (*angl. Established*) ryšio būseną.

BGP Atributai

Maršrutai, išmokti panaudojant BGP, turi parametrus, kurie yra naudojami nustatyti geriausiam keliui, kai tarp galutinio ir pradinio mazgo yra keletas kelių pasirinkimų. Šie parametrai vadinami BGP atributais (*angl. attributes*). Šių atributų yra iš tiesų nemažai. Darbe apžvelgsime jų skirstymą ir keletą pagrindinių pavyzdžių.

Pirma, maršruto atributai skirstomi į dvi pagrindines kategorijas [8]:

- gerai žinomi atributai;
- papildomi atributai.

Gerai žinomi (*angl. Well-known*) atributai yra tie atributai, kuriuos palaiko visos BGP sistemos. Tai yra IETF pateikti atributai. Šie atributai papildomai skirstomi į būtinus visoms sistemoms pagal BGP standartą ir nebūtinus, tai yra, pagal BGP sistemos sutarimą. Visos BGP sistemos turi palaikyti tiek vieno, tiek kito potipio atributus.

Papildomi (*angl. Optional*) atributai nebūtinai turi būti palaikomi sistemos. Šiuos atributus skirsto į tranzitinius ir ne tranzitinius. Tranzitiniai atributai yra tie, kuriuos BGP procesas radęs „vėliavėlę“ pažyminčią tranzitinę būseną, turi persiųsti kitiems BGP nariams. Ne tranzitiniai atributai, kaip galima spręsti iš pavadinimo, yra tie, kuriuos BGP sistema gali ignoruoti ir nesiųsti kitiems nariams.

Kadangi BGP turi nemažai atributų, išnagrinėsime kelis pagrindinius. Tai yra gerai žinomi būtinieji atributai: „Origin“, „As_Path“, „Next_Hop“.

„Origin“. Šis atributas nurodo maršruto atnaujinimo kilmę. Kilmės atributas gali turėti šias reikšmes:

- IGP
- EGP
- nebaigtas (*angl. Incomplete*)

Jei BGP turi keletą maršrutų, tada „Origin“ atributas yra vienas iš faktorių pasirenkant tinkamą maršrutą. Aukščiausią prioritetą turi IGP reikšmė, po to EGP ir mažiausią prioritetą turi „Incomplete“ reikšmė.

IGP reikšmė yra gaunama iš vidinio maršruto parinkimo protokolo, esančio AS viduje.

EGP reikšmė yra gaunama iš EGP (*angl. Exterior Gateway Protocol*) protokolo, kuris naudojamas dalinantis maršruto parinkimo informacija tarp skirtingų AS.

„Incomplete“ reikšmė. Ši reikšmė dažnai laikoma sugadinta, tačiau taip nėra. Tai tiesiog reiškia, kad kilmės informacija nėra užbaigta.

„As_Path“ atributas nusako kelią iki tikslo, sudarytą iš tarpinių AS. Ši reikšmė duoda AS numerių sąrašą, per kuriuos keliaujama pereinant visą maršrutą. Kiekvienas BGP narys, skelbdamas maršrutą į tikslą, turi prie maršruto pridėti savo AS numerį, po to tai atlieka sekantis narys ir taip toliau. „As_Path“ atributas prasideda paskutiniu praeitu AS ir baigiasi kilmės AS.

Yra du informacijos pateikimo variantai, jei prefiksas yra kilęs iš BGP nario, kuris siunčia informaciją. Jei informacija siunčiama eBGP nariams, tada siuntėjas prideda savo AS numerį į „As_Sequence“ tipo segmentą, kuris yra „As_Path“ atribute. Jeigu informacija siunčiama iBGP nariams, tada „As_Path“ atributas yra tuščias.

Jei prefiksas yra kilęs iš kito BGP siuntėjo, persiunčiamas „As_Path“ taip pat kaip ir ankstesniu variantu, priklauso nuo to ar siunčiama iBGP ar eBGP. Jei informacija siunčiama į vietinį iBGP narį, tada „As_Path“ nėra modifikuojamas.

Jei siunčiama į eBGP, tada informacija priklauso nuo pirmo segmento esančio „As_Path“. „As_Sequence“ atveju sistema tiesiog prideda į eilę savo AS numerį. Tuo tarpu, kai pirmas segmentas yra „As_Set“, sistema prideda „As_Sequence“ segmentą į „As_Path“ ir įrašo savo AS numerį.

„Next_Hop“. Šis atributas skirtas nurodyti sekantį šuolio IP adresą, reikalingą pasiekti galutiniam tikslui. Šis atributas turi kelis scenarijus.

Kai informacija siunčiama iš vidinio nario ir siunčiamas maršrutas nėra vietinis, tada BGP siuntėjas nemodifikuoja „Next_Hop“ atributo, nebent to sistema yra specialiai sukonfigūruota siųsti savo IP adresą. Jei maršruto kilmė vietinis narys, tada BGP siuntėjas panaudoja adresą sąsajos, kuria pasiekiamas skelbiamas tinklas.

Jeigu maršrutas yra tiesiogiai prijungtas prie siunčiančio nario, tada siuntėjas „Next_Hop“ atributui naudoja savo paties IP adresą.

1.6. Maršrutų stebėjimo metodai

Toliau bus nagrinėjami keli maršrutų stebėjimo metodai, kurių pagalba būtų galima stebėti tinklo pakitimus.

1.6.1. RIB stebėjimas.

Kiekvienas BGP narys, gavęs informaciją iš kitų „kaimynų“, ją apdoroja ir paruošia lokaliam naudojimui. Tai leidžia gautus maršrutus atitinkamai skelbti kitiems BGP nariams, naudojant tam tikras nustatytas taisykles. Informacija, skirta visoms šioms procedūroms atlikti yra saugoma specialioje duomenų bazėje, vadinamoje maršruto parinkimo informacijos baze (*angl. Routing Information Base*), trumpai RIB.

Giovanni Comarela ir Mark Crovella darbe „Identifying and Analyzing High Impact Routing Events with PathMiner“ duomenys renkami būtent naudojant RIB [9]. Darbe informacija buvo renkama 9 metus, kiekvieną dieną, kaip šaltinį naudojant BGP maršruto parinkimo informacijos bazes, viešai pateikiamas RIPE organizacijos, bei pateiktų „Route Views project“.

Iš RIB renkamą informaciją sudarė visi datos, prefikso ir AS kelio įrašai, susiję su IPv4 adresais. Renkama informacija buvo saugoma 12 mazgų HDFS klasteryje. RIPE teikiami RIB turėjo daug maršrutų, tačiau tai nebuvo visiškai visi keliai, kadangi ne visi maršrutai stebimi. Iš RIPE teikiamų prefiksų buvo pasirinkta stebėti 20000.

Apibendrinus šį variantą, galima išskirti keletą pastebėjimų. Pirma, naudojant duomenis, pateikiamus RIPE organizacijos, reikia mažiau rūpintis, kokius taškus arba maršrutus reikia stebėti. Tačiau nuo to laiko, kai buvo pradėtas vykdyti šis projektas „Route Views project“ nebeteikia tokių galimybių, kokios minėtos projekte.

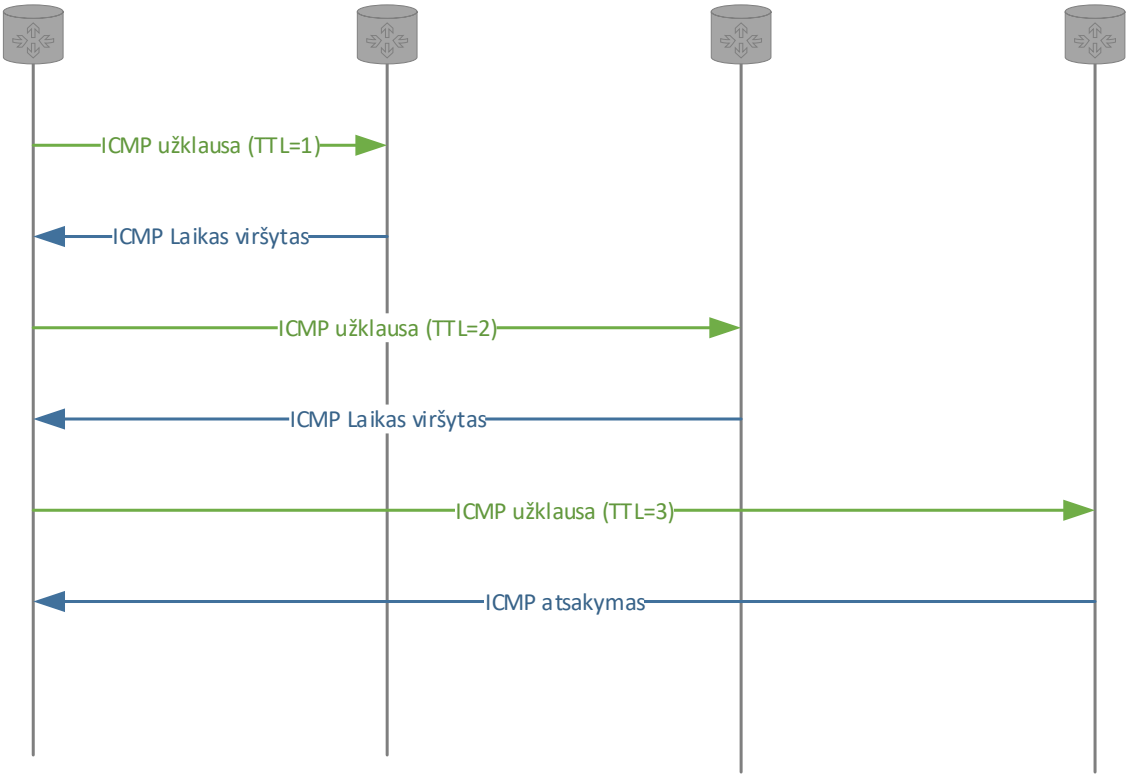
Kitas pastebėjimas yra informacijos kiekio problema. Šiame darbe informacija buvo renkama kartą per dieną. Šis variantas magistro projektui netinka, kadangi didžioji dalis atakų trunka daug trumpiau, nei dieną laiko. Taip magistro darbo metu nebus galimybės kurti tokio kompleksinio klasterio.

1.6.2. „Traceroute“ stebėjimas

„Traceroute“ yra įrankis, kuris leidžia matyti kelią iki nustatyto IP adreso.

Norint gerai suprasti „Traceroute“, reikia trumpai išanalizuoti ir ICMP protokolą, bei jo funkciją. ICMP yra interneto kontrolės žinučių protokolas (*angl. Internet Control Message Protocol*). [10] Šį protokolas naudoja įvairūs tinklo įrenginiai, pavyzdžiui, maršruto parinktuvai, siųsti informacines žinutes. Žinutės gali nurodyti, kad įrenginys yra nepasiekiamas, nežinomas ar panašiai. Šios ICMP protokolo žinutės yra formuojamos pagal RFC 1122 standartą ir siunčiamos į pradinį IP adresą, kuris daro užklausą. [11] „Traceroute“ darbas yra susijęs su šių ICMP paketų atsakymais.

Ši programa siunčia tiriamuosius (*angl. probe*) paketus su maža gyvavimo trukme (TTL), tada klauso ICMP paketo atsakymo "time exceeded" iš sietuvo (*angl. Gateway*). Po to TTL yra didinamas ir taip daroma tol, kol gaunama ICMP "port unreachable". Tai pasako, kad pasiektas reikiamas mazgas (1.4 pav.).



1.4 pav. „Traceroute“ veikimo schema

Naudojant „Traceroute“ atmainas, tokias kaip „MTR“ galima daryti papildomas užklausas, kurios iš karto gali išversti IP adresus į AS numerį ir panašiai.

Atliekant tyrimą apie incidentą, nutikusį Kinijos Telekomui („Characterizing Large-scale Routing Anomalies: A Case Study of the China Telecom Incident“) duomenų rinkimui buvo naudojamas būtent „Traceroute“ įrankis, maršrutų informacijai gauti. [12]

Tyrimo metu, dėl jo specifikos, buvo siekiama stebėti konkrečius potinklius. Kokie potinkliai buvo stebimi, šiuo atveju nebuvo svarbu. Tačiau šių pasirinktų potinklių stebėjimas buvo vykdomas pasirinkus atitinkamą IP adresą ir jį stebint „Traceroute“ programa.

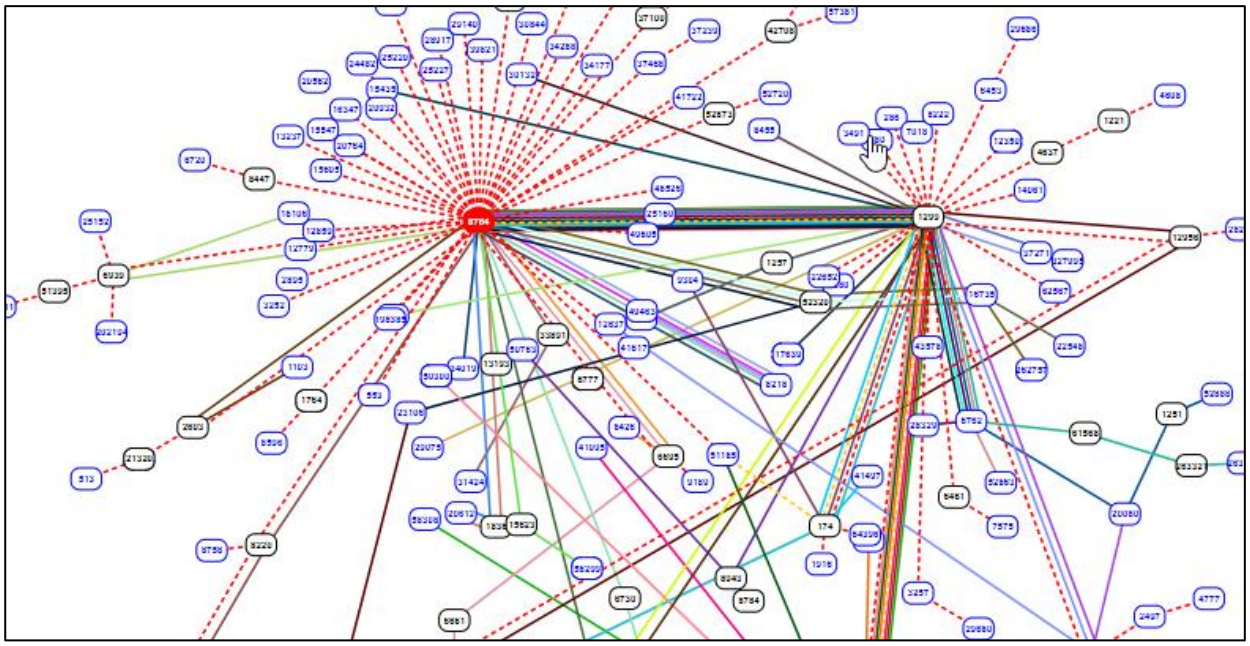
Šis stebėjimo metodas turi trūkumą, kuris yra ir paminėtas tyrime apie Kinijos telekomą. Kadangi nėra stebimas pats AS ar netgi jo skelbiami maršrutai, bet tik paprastas IP adresas, tai gali sukelti papildomų netikslumų. Tai gali įvykti, dėl IP adreso kelių pasikeitimų dėl jo vartotojo ar tiekėjų kaltės. Nepaisant to, šis metodas yra ganėtinai paprastas ir lengvai naudojamas, naudojant „Traceroute“ nereikia prisirišti prie trečių šalių teikiamos informacijos, tereikia turėti reikalingų IP adresų sąrašą ir bus galima gauti maršrutus iki jų.

1.6.3. BGP sesijų stebėjimas

Apie pačias BGP sesijas ir jų veikimą buvo kalbama praėjusiame skyriuje, dėl to bus koncertuojamasi į patį stebėjimą.

Tiriant BGP sesijų užgrobimą arba dažną kaitaliojimąsi (*angl. flapping*) pačių BGP sesijų stebėjimas yra normalus tyrimo metodas. Šiam tyrimui reikia turėti savo BGP narį, sujungtą su kitu

BGP „kaimynu“. Ir netgi tada matomas ribotas informacijos kiekis, labiausiai susijęs su vietiniu tinklu. Tačiau tiriant BGP sesijų pasikeitimus, taip pat gali padėti RIPE teikiamos paslaugos. RIPE „BGPlay“ (1.5 pav.) yra sistema, kuri leidžia matyti interaktyvų BGP maršrutų žemėlapį pagal nustatytą datą. Šio žemėlapio teikiama informacija yra gerai atvaizduota ir aiški. Tačiau siekiant pastebėti atakas, šis



1.5 pav. „BGPplay“ vartotojo sąsaja

žemėlapis gali nepadėti. Pirma, mes matome tik kelią iki AS, o ne iki konkretaus adreso, kas nėra blogai, tačiau visas AS gali būti mažiau jautrus atakoms. Taip pat, tai yra trečiosios šalies įrankis, dėl to atsiranda priklausomybė nuo organizacijos, kuri šį įrankį prižiūri, jį gali būti sunkiau pritaikyti unikaliose sistemose, kitaip nei kuriant savo įrankį.

1.7. Atakų atpažinimo metodai

Norint pastebėti DDoS arba kitas atakas, vien tinklo stebėjimo neužtenka. Sėkmingam atakų aptikimui reikalingas geras atpažinimo metodas. Per daugelį metų buvo pasiūlyta nemažai metodų, kaip galima stebėti atakas. Iš daugybės metodų tikrai nėra daug pritaikytų praktiškai ir / arba pateikiama vieša informacija apie tai. Toliau bus apžvelgiama keletas DDoS atpažinimo metodų, kurie pasirinkti siekiant ištirti kuo įvairesnius atpažinimo būdus.

1.7.1. Šaltinio IP adresų metodas

Pradėsime nuo atakų atpažinimo, naudojant šaltinio IP adresus. Tokio tipo sistema stebi naujų šaltinių adresų atsiradimą, vietoj pačio srauto stebėjimo. Šis metodas paremtas Jung tyrimu, kuris parodo, kad atakos metu didžioji dalis ataką sukeliančių IP adresų būna nauji. [13] Tai netgi neturėtų sukelti problemų, atsiradus staigiems tinklo apkrovos šuoliams, kadangi neretai tokie šuoliai kyla iš jau žinomų IP adresų.

Pagrindinis šio stebėjimo trūkumas yra tai, kad tokią aptikimo sistemą atakuojanti pusė gali nesunkiai apeiti, panaudojant žinomus (ne naujus) adresus, vykdant ataką. Atakos sukėlėjas gali

naudoti naujus adresus normaliam sujungimui, o po to atakai. Taip pat galima maišyti naujus ir jau žinomus adresus.

1.7.2. Bazinių srauto profilių metodas

Kim siūlomas DDoS atakų atpažinimo metodas yra paremtas stabilių bazinių srautų profilių kūrimu. [14] Pagal nuokrypius nuo šių bazinių profilių, galima atpažinti vykstančias atakas. Metodo autorius atliko tyrimą, kuriuo buvo siekiama patikrinti tinklo srauto stabilumą, atsižvelgiant į tam tikrus tinklo parametrus. Šių parametrų reikšmių negalima tiksliai įvardinti, kadangi pastebėta, kad šios reikšmės gali labai skirtis, priklausomai nuo stebimos sistemos. Šios bazinės profilio reikšmės gali būti gyvavimo trukmė (TTL), TCP paketų žymos (angl. *flags*), protokolų tipai, IP adresai, paketų dydžiai ir panašiai. Be abejo, šiuos parametrus atakos sukėlėjas gali bandyti nuspėti. Nepaisant to, nuspėti tikslius profilio parametrus ganėtinai sudėtinga, kadangi šie parametrai, kaip anksčiau minėta, nebūtų universalūs ir skirtųsi priklausomai nuo sistemos. Ši metodas turi keletą trūkumų. Stebėjimas naudojant bazinį profilį turi sąlyginai didelį klaidingai teigiamų atakų aptikimų kiekį, taip pat reikalinga galingesnė įranga, palyginus su kitais metodais, pavyzdžiui, šaltinių adresų stebėjimas.

1.7.3. Atstumo stebėjimo metodas

Dar vienas iš DDoS aptikimo metodų yra paremtas atstumo stebėjimu. Stebimas vidutinis sujungimo atstumas ir srautas pagal atitinkamą atstumą. [15] Vidutiniam ryšio sujungimo atstumo apskaičiavimui naudojama TTL reikšmė. Tariamas normalus srautas nustatomas nuspėjant vidutinę atstumo reikšmę, kuri yra gauta naudojant eksponentinio išlyginimo skaičiavimo metodą. Metodas pagal atitinkamą atstumą bando nuspėti, kiek tinklo srauto atkeliaus iš skirtingų atstumų. Nuspėjamos ateinančio srauto reikšmės skaičiuojamos naudojant minimalią vidutinę kvadratinę paklaidą, trumpai – MMSE (angl. *minimum mean square error*).

Galima pabrėžti keletą šio metodo trūkumų. Pirma, atakų atpažinimas yra paremtas TTL reikšme. Atakos sukėlėjas, žinodamas apie sistemos veiklą, gali pasirinkti kelią, kuris leistų apeiti aptikimo sistemą. Taip pat atakuojant galima „sureguliuoti“ paketų TTL. Ši sistema gali aptikti ir klaidingų teigiamų rezultatų, jeigu stipriai pasikeistų globalūs interneto maršrutai.

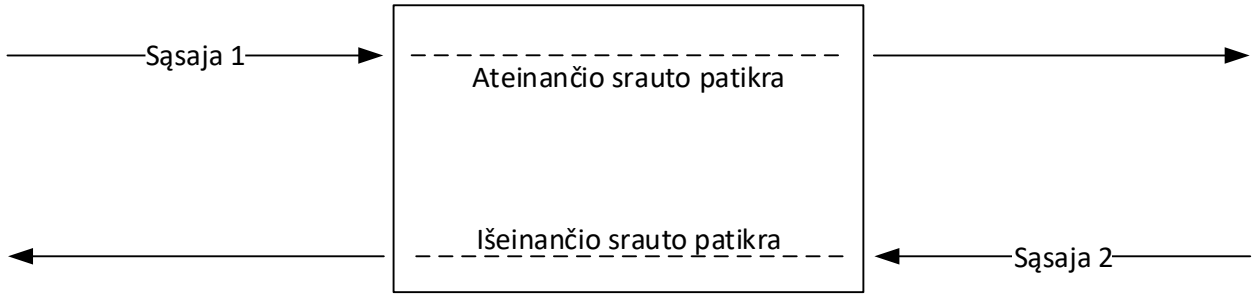
1.7.4. MULTOPS metodas

MULTOPS (MUltiLevel Tree for Online Packet Statistics) metodas pasiūlytas Gil ir Poletto. Tai maršruto parinktuvuose naudojamas euristinis metodas, paremtas struktūrizuotais duomenimis. Atakų aptikimui MULTOPS naudoja duomenis apie iš arba į adresą ar potinklį einantį neproporcingą srautą. Srauto duomenys yra renkami medžio formos duomenų struktūroje. Medžio mazgai laiko informaciją apie adresus ir potinklius ir tam tikros šakos gali keisti savo dydį priklausomai nuo srauto pokyčių. [16]

MULTOPS metode renkama paketų srauto statistika tarp dviejų adresų arba potinklių. Dėl šios priežasties šiame metode galima rasti aukos adresą arba atakuojančios pusės adresą. Būtent dėl

šios priežasties MULTOPS metodas išskiriamas į du režimus: orientuotas į auką ir orientuotas į atakuojantį. Į auką orientuotame režime bandoma aptikti aukos IP adresą, o orientuotame į atakuojantį bandoma rasti kenkiantį IP adresą. Šis skirtumas yra labai pabrėžiamas dėl to, nes nuo to priklauso kokį srautą reikia blokuoti: ateinantį į aukos IP ar išeinantį iš atakuojančio IP adreso ar adresų.

MULTOPS metodas turi gauti du paketų srautus (1.6 pav.), kurių kiekvienas ateina per atskirą tinklo sąsają. Per vieną sąsają gaunama ateinančio srauto paketai, per kitą išeinančio srauto paketai.



1.6 pav. MULTOPS srauto rinkimo schema

Priklausomai nuo metodo režimo (orientuoto į atakuojamą ar į atakuojantį IP adresus), stebimos viena iš šių sąsajų. Sąsajoje stebimos paketų srauto vertės ir tuo pat metu lyginamos su vidurkio reikšme. Jei esama reikšmė per daug nukrypsta nuo vidurkio, tada tai atpažįstama kaip ataka.

Šis metodas, kūrėjų teigimu, turi keletą trūkumų. Pirma, taip negalima atpažinti atakos, jeigu IP adresai bus parenkami atsitiktinai, tačiau šį trūkumą galima pastebėti ir kituose metoduose. Taip pat kaip ir profiliavimo metode, dideli netikėti, tačiau nekenkėjiškos kilmės srauto šuoliai, gali būti klaidingai nustatyti kaip atakos.

1.8. Tinklo analizės išvados

Analizės metu buvo išnagrinėta DDoS atakos ir jų veikimo metodai. Dėl to nustatyta, kad šių atakų įtaka geriausiai matosi interneto tinklų ir jų mazgų būsenoje. Kadangi DDoS atakos atsispindi tinklų būsenoje, norint pastebėti šias atakas, logiškas sprendimas yra stebėti internetą ir sistemų tinklus.

Išnagrinėti trys pagrindiniai tinklo stebėjimo metodai, kuriais būtų galima pastebėti stambias DDoS atakas. Srautų stebėjimo ir paketų skenavimo metodai turi savų privalumų, tačiau netinka pagrindiniam šio darbo tikslui – didelių srautinių DDoS atakų stebėjimui, kadangi šie metodai tinka tik savo infrastruktūros stebėjimui. Tuo tarpu maršrutų stebėjimas leidžia stebėti didelio mąsto viešojo interneto dalies pokyčius. Reikia pabrėžti, kad ne visi nagrinėti maršrutų metodai, leidžia stebėti visą internetą. Geriausiai šiam darbui tinkantis yra maršrutų mazgų stebėjimas. Šiam metodui nereiks jokios papildomos didelės tinklo infrastruktūros.

Tiriant aptikimo metodus pastebėta, kad yra sukurta tikrai nemažai įvairių būdų, atpažinti DDoS atakas. Tačiau daugelis metodų yra ganėtinai seni ir nepaplitę. Nemaža dalis šių metodų yra paremta profiliavimu ar srauto lyginimu su vidutine reikšme. Iš to galima spręsti, kad tai paprastas ir ganėtinai efektyvus metodas. Tačiau siekiant efektyvesnio stebėjimo tolesnėje darbo veikloje bus

siekiama pritaikyti mašininį mokymą. Mašininio mokymo panaudojimas siekiant atpažinti atakas turės panašumų su paprastu duomenų profiliavimu, tačiau leis profilių reikšmes nustatyti automatiškai ir jos galės keistis dinamiškai, pagal pasikeitusias mazgų savybes.

Norint toliau tęsti darbą, reikia suprojektuoti sistemą, kuri leistų rinkti maršrutų į pasirinktus adresus pasikeitimus. Šią sistemą turi sudaryti trys dalys: duomenų rinkimo, duomenų saugojimo ir duomenų analizavimo bei pateikimo. Tolimesnėje darbo eigoje turi būti įvykdyti šie uždaviniai:

- išnagrinėti ir pasirinkti tinkamą mašininio mokymosi algoritmą ar algoritmus duomenų analizavimui;
- suprojektuoti tinklo stebėjimo sistemą paremtą vėlinimo statistika ir maršrutų pokyčiais;
- suplanuoti kaip bus saugomi statistiniai duomenys;
- suprojektuoti duomenų analizavimo modulį.

2. TINKLO DUOMENŲ ANALIZĖ NAUDOJANT MAŠININĮ MOKYMĄSI

Darbo metu, analizuojant tinklų statistikos duomenis ir aptinkant DDoS atakas, planuojama naudoti mašininį mokymą. Mašininio mokymo algoritmai bus naudojami siekiant nustatyti ar stebimas srautas yra ataka, ar ne, pagal prieš tai surinktus duomenis. Šiame skyriuje bus nagrinėjama, kokie mašininio mokymosi algoritmai yra naudojami ir rekomenduojami tokio tipo užduotims, kurie labiau tinka DDoS aptikimui ir srauto klasifikavimui.

2.1. Mašininis mokymas

Mašininis mokymas (angl. *machine learning*) yra algoritmų ir statistinių modelių mokslo sritis, kurios tikslas yra efektyviai atlikti kompiuterinėms sistemoms paskirtas užduotis, remiantis įvairiais modeliais ir išvadomis. Matematiniai mašininio mokymo skaičiavimai gali būti naudojami įvairiose srityse, tarp jų ir duomenų analizei. Šio projekto atveju, reikės nagrinėti reikšmes, kurios atspindi kompiuterinių tinklų būsenas.

Šiuo metu išskiriami du pagrindiniai mašininio mokymosi tipai:

- neprižiūrimas mokymasis (angl. un*supervised learning*);
- prižiūrimas mokymasis (angl. *supervised learning*).

Neprižiūrimas mokymasis

Tai yra tokio tipo mašininis mokymasis, kai algoritmui nėra paskiriama jokios konkretaus išvedamo rezultato baigtys. Toks algoritmas gali padėti grupuoti tam tikras reikšmes ir atlikti tokius uždavinius, kaip klientų segmentavimas ar tam tikrų požymių atradimas.

Prižiūrimas mokymasis

Prižiūrimo mokymosi metu, sistemai pateikiami įvesčių pavyzdžiai ir norimos išvestys. Turint šiuos duomenis, sistema paima įvestis ir priskiria jas prie atitinkamų išvesčių. Modelio mokymas vyksta tol, kol būna pasiekiamas tinkamas jo tikslumas. Prižiūrimą mokymąsi taip pat galima paskirstyti į kelis atskirus pogrupius:

- pusiau prižiūrimas mokymasis (angl. *semi-supervised learning*), tai toks mokymasis, kai sistema gauna nepilną informacijos rinkinį ir dalis išvesčių nėra pažymėta;
- aktyvus mokymasis (angl. *active learning*) yra mokymasis, kurio metu sistemos vartotojo reguliariai klausiama, koks turi būti išvesties žymėjimo pasirinkimas;
- mokymasis su pastiprinimu (angl. *reinforcement learning*) yra toks kurio, algoritmas keičia elgsenos strategiją, priklausomai nuo aprašytų veiksnių.

Prižiūrimo mokymosi pavyzdžiai yra tokie algoritmai, kaip pasirinkimų medis, regresija, KNN, logistinė regresija, atsitiktinis medis ir kt.

Kaip jau minėta, mašininis mokymasis gali būti pritaikomas labai įvairiose srityse, tačiau kompiuterinių tinklų stebėjimo sritis nėra labai išvystyta. Tinklo stebėjimas turi šiek tiek niuansų. Pirma, kompiuteriniai tinklai yra labai įvairūs ir sunku pritaikyti vieną visiems tinkantį modelį. Tai reiškia, kad kiekvienam tinklui algoritmus reikės „mokyti“ atskirai. Kita problema yra tai, kad tinklai yra dinamiški ir jų savybės keičiasi. Tačiau pastaroji problema nėra labai rimta, didžiausias to trūkumas, kad tinklams keičiantis, išmoktas algoritmas irgi turės keistis, nes kitaip gali atsirasti daugiau klaidingų rezultatų, nei prieš pasikeitimus.

Šio darbo metu nagrinėsime prižiūrimo mašininio mokymosi klasifikavimo algoritmus. Jų pagalba bus siekiama atpažinti, ar tam tikras adresas yra kenčiantis nuo srautinės DDoS atakos, analizuojant jau surinką statistiką apie tą adresą ir maršruto į jį mazgus.

2.2. Duomenų klasifikavimas ir jo algoritmai

Kaip jau minėta, sėkmingam projekto atlikimui reikia išanalizuoti ir pasirinkti tinkamus klasifikavimo algoritmus. Toliau nagrinėsime šiuos prižiūrimo mokinimosi algoritmus:

- logistinės regresijos metodas (angl. Logistic Regression);
- naiviojo Bajeso metodas (angl. Naïve Bayes);
- KNN metodas (angl. K-Nearest Neighbours);
- sprendimų medžio metodas (angl. Decision Tree);
- atsitiktinis miško metodas (angl. Random Forest);
- daugiasluoksnio perceptrono metodas (angl. multilayer perceptron).

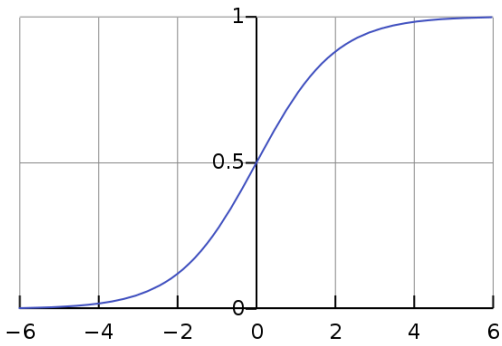
Algoritmai pasirinkti iš dažniausiai siūlomų ir populiariausių algoritmų sąrašų [17, 18, 19].

2.2.1. Logistinės regresijos metodas

Logistinė regresija yra plačiai naudojamas statistinis modelis, kuris yra pritaikytas binarinių reikšmių (0-1) klasifikavimui. Tokio klasifikavimo pavyzdžiai gali būti įvykio nutikimo nuspėjimas. Logistinės regresijos metodo pavadinimas kyla nuo logistinės funkcijos, kuri yra naudojama šiame algoritme:

$$f(x) = \frac{1}{1 + e^{-x}}$$

Loginė funkcija pasižymi tuo, kad jos išvesčių aibė yra tarp 0 ir 1 (2.1 pav.). Logistinė funkcija pasižymi S formos grafiku. Y reikšmės yra gaunamos logaritmiškai transformuojant X reikšmes. Po to reikšmių aibei pritaikoma riba, kuri atsakymus klasifikuoja į dvi grupes.



2.1 pav. Logistinė kreivė [17]

Logistinės regresijos privalumas yra tas, kad ši funkcija skirta klasifikavimui. Tai leidžia patogiai priimti daug įvairių reikšmių ir gauti vienareikšmį atsakymą. Tuo pačiu šios savybės yra ir trūkumas, kadangi yra tik dvi galimos reikšmės. Tai tuo pačiu turi dar vieną trūkumą – šis modelis daro prielaidą, kad visi kintamieji yra nepriklausomi vienas nuo kito ir kad reikšmių aibėje nėra trūkstamų reikšmių.

2.2.2. Naiviojo Bajeso metodas

Šis metodas naudojamas siekiant apskaičiuoti, ar įvykis nutiks, atsižvelgiant į prieš tai nutikusius įvykius. Metodas pagrįstas Bajeso teorema, darant prielaidą, kad nuspėjamos reikšmės yra nepriklausomos viena nuo kitos. Bajeso klasifikavimo metu daroma prielaida, kad visos reikšmės priklausančios tam tikrai klasei yra nepriklausomos ir nesusijusios su kitomis to klasės reikšmėmis. Pavyzdžiui, jei skirstytume gyvūnus ir turėtume pelę, kurios savybės yra kelių centimetrų ilgis, pilkas kailis ir uodega, Bajeso algoritmas nekreiptų dėmesio, kad šios savybės apibūdina pelę ir vertintų kiekvieną reikšmę ar savybę nepriklausomai.

Bajeso teoremos formulė yra:

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$

Šioje formulėje:

- P(c|x) yra prognozuojamos klasės galutinė tikimybė;
- P(c) yra išankstinė klasės tikimybė;
- P(x|c) yra galimumas;
- P(x) yra išankstinė prognozės tikimybė.

Naiviojo Bajeso metodas yra lengvai sudaromas ir ypač lengvai panaudojamas dideliems duomenų rinkiniams. Šis metodas žinomas kaip paprastas, tačiau sugebantis našumu lenkti netgi daug sudėtingesnius algoritmus. Aišku, dažniausiai šis protokolas naudojamas siekiant klasifikuoti duomenis su skirtingomis klasėmis.

2.2.3. KNN metodas

KNN arba K- artimiausių „kaimynų“ mašininio mokymosi metodas gali būti naudojamas tiek regresijos, tiek klasifikavimo uždaviniams spręsti. Tačiau dažniausiai šis metodas taikomas klasifikavimui. KNN yra ganėtinai paprastas algoritmas, kuris saugo visas turimas atvejų reikšmes ir tada klasifikuoja naujas atvejus pagal didžiausią K „kaimynų balsų“ skaičių. Pakyrimas vyksta panaudojant atstumo funkciją ir priskiriant naują atvejį panašiausiai klasei.

K artimiausio „kaimyno“ atstumo funkcijos gali būti šios:

- Euklidinė;
- Manheteno;
- Minkovskio;
- Hemingo.

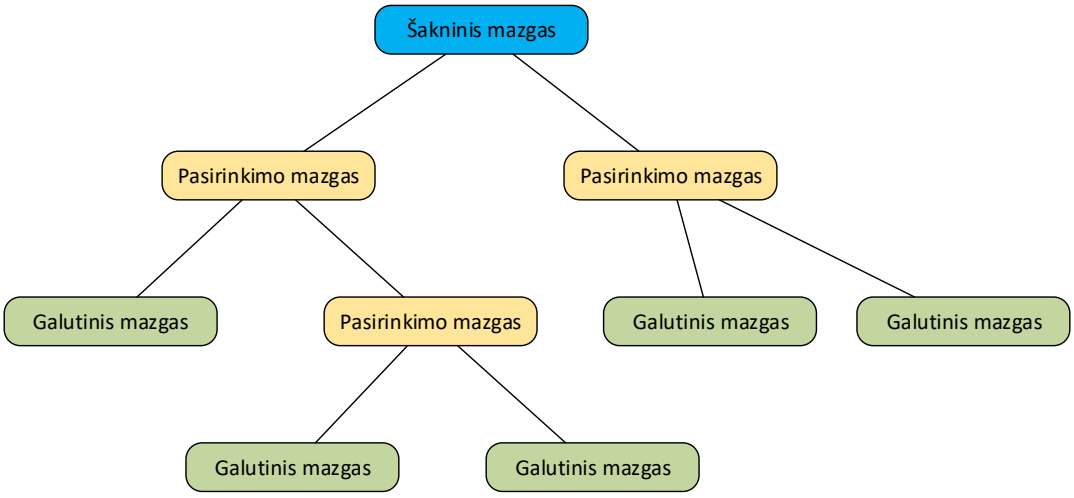
KNN yra ganėtinai lengvai pritaikomas algoritmas, kuris yra atsparus duomenų triukšmams ir efektyviai veikia su dideliais duomenų kiekiais.

Tiesa, šis metodas turi keletą trūkumų. Pirma, nustatyti K reikšmę ir išspręsti algoritmą reikia daug skaičiavimo resursų. Taip pat šis metodas turėtų gauti normuotas reikšmes, kitaip jos gali stipriai paveikti galutinį atsakymą. Dėl to neretai šis algoritmas naudojamas tarpiniam duomenų apdorojimui, norint išgryninti reikšmes.

2.2.4. Sprendimų medžio metodas

Kaip pasako pavadinimas, sprendimų medžio metode, nuspręsti atsakymą naudojami sprendimų medžiai. Šis metodas dalina objektų aibes į du arba daugiau homogeninių rinkinių. Tai padaroma remiantis atskirų ir unikalių reikšmių bei atributų vertėmis (2.2 pav.).

Kaip pavyzdį, galima imti transporto priemonių klasifikavimą ar jos tinkamos važiuoti. Kaip šakninis mazgas, gali būti klausimas, ar transporto priemonė turi visus sveikus ratus. Sekantys pasirinkimo klausimai gali būti tokie, kaip ar veikia variklis, ar perduodamas sukimo momentas ir t.t. Galutiniuose mazguose būtų konstatuojama, ar transporto priemonė tinkama važiuoti, ar ne.



2.2 pav. Pasirinkimų medžio struktūra

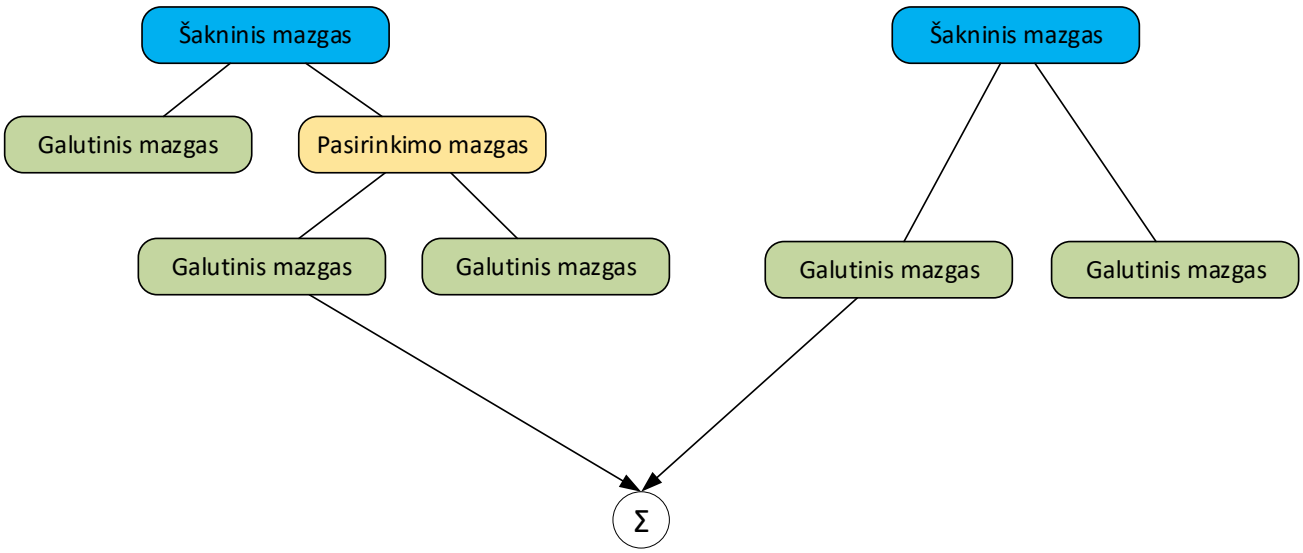
Sprendimų medžio algoritmas turi daug privalumų. Pirma, šį algoritmą lengva suprasti net ir neturint duomenų analizės žinių, vartotojai gali lengvai interpretuoti gautus duomenis ir pritaikyti juos savo tyrimuose.

Kitas privalumas yra tai, kad šis algoritmas gali pasitarnauti, kaip tarpinis tikrinimas. Sprendimų algoritmas gali pagelbėti rasti labiausiai naudingas ir išvestį įtakojančias reikšmes, taip galima išgryninti reikšmių rinkinius ir nereiktų naudoti šimtų atributų. Tuo pačiu šis algoritmas, palyginus su kitais metodais reikalauja mažiau valymo ir reikšmių aibės gali turėti daugiau triukšmo.

Nors šis algoritmas yra ganėtinai atsparus triukšmui, turi problemų dėl perdėto objektų priskyrimo (angl. *overtfit*). [20] Šią problemą gali padėti išspręsti atsitiktiniai „miškai“ apie kuriuos netrukus kalbėsime.

2.2.5. Atsitiktinio miško metodas

Atsitiktinio miško metodas yra labai panašus pasirinkimų medžio algoritmą. Šis algoritmas iš tiesų yra sudarytas iš pasirinkimų medžių rinkinių (2.3 pav.), todėl jo pavadinimas ir yra „miškas“. Šis metodas taip pat, kaip ir pasirinkimo medžio metodas, tinka naudoti klasifikavimui ir regresijoms.



2.3 pav. Atsitiktinio miško struktūra

Atsitiktinio miško metode pasirinkimų medžių rinkinys naudojamas kaip klasifikavimo „balsai“, tai yra, kiekvienas medis atiduoda savo balsą už savo tirto objekto klasę. Jau anksčiau minėjome, kad sprendimų medžiai linkę perdėtai priskirti reikšmes, dėl savo didelio augimo. Atsitiktinio miško modelis išsprendžia šią problemą ir reikšmes išlygina, dėl jų sumavimo.

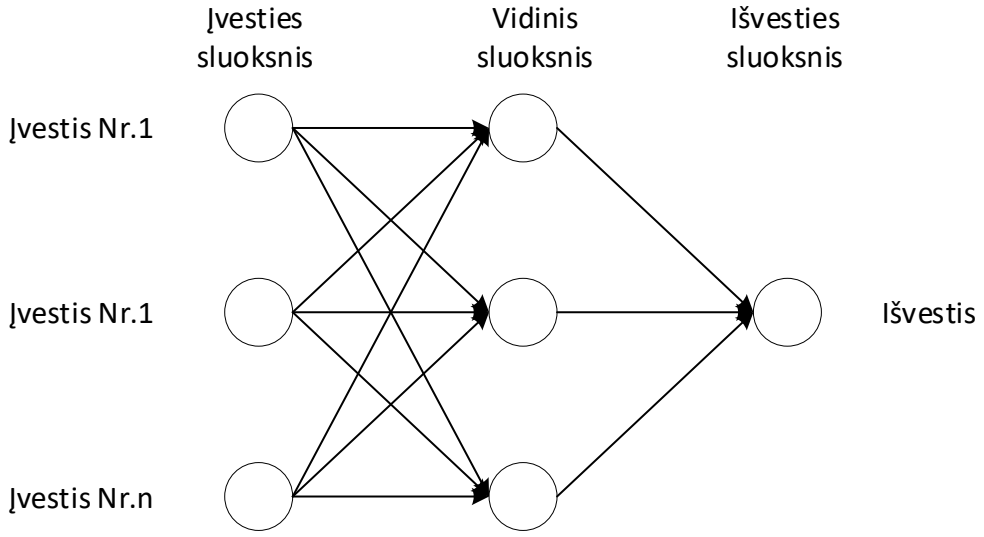
Atsitiktinė dalis šiame metode yra reikšmių paėmimas. Metodas sukuria daug pasirinkimo medžių, pagal skirtingus įvesčių rinkinius. Pasirinkimų medžio algoritmai paleidžiami tam tikrą kiekį kartų. Galų gale, šios tarpinės medžių išvestys sujungiamos į vieną, naudojant vidutinės sumos reikšmę.

2.2.6. Daugiasluoksnio perceptrono metodas

Kalbant apie daugiasluoksnį perceptrono metodą, iš pradžių reikia išsiaiškinti kas yra perceptronas. Perceptronas yra linijinio klasifikavimo algoritmas, turintis vieną ar daugiau įvesčių, skaičiavimo procesorių ir vieną išvestį. Vykdymo metu yra dauginami svoriai ir pridedama tendencijos. Radus klasifikavimo klaidų, svoriai atnaujinami pagal tokią formulę:

$$svoris = svoris + mokymosi\ greits * (tikimasi - nuspėta) * x$$

Daugiasluoksniame perceptrono metode yra daugiau nei vienas perceptronas. Paprastame trijų sluoksnių pavyzdyje (2.4 pav.) turime įvesties sluoksnį, vidinį sluoksnį ir išvesties sluoksnį. Dažniausiai šie sluoksniai veikia padavimo į priekį tinklą. Kiekvienas sluoksnis turi savo funkcijas, kurių tikslas priartinti prie klasifikatorių ir išmokti geriausių tam reikiamų parametrų. Visi metodo sluoksniai yra sujungti tarpusavyje su kiekvienu skaičiavimo mazgu, bet mazgai yra nepriklausomi vienos nuo kito tame pačiame sluoksnyje.



2.4 pav. Trisluoksnio perceptrono pavyzdys

Dažniausiai šių algoritmų naudojimo strategija yra pradėti su atsitiktiniais svoriais ir tada juos gryninti su kiekviena iteracija. Dėl šios priežasties, svarbu algoritmams parinkti tinkamą mokymosi greitį, kad būtų galima nustatyti, kiek jie mokosi per kiekvieną iteraciją.

Daugiasluoksnis perceptronas yra ganėtinai lankstus ir universalus metodas, kuris leidžia priimti daug įvesčių ir turėti daug vidinių sluoksnių, tai leidžia šiam algoritmui dirbti greitai. Algoritmas gali dirbti su dideliais kiekiais duomenų.

Tačiau dėl savo sluoksnių įvairovės šis modelis yra reikalaujantis daug daugiau resursų, nei minėti anksčiau. Taip pat sluoksniai sukelia kitą problemą, vidinių sluoksnių veiklos darbo metu matyti negalima, todėl šis metodas veikia kaip „juodoji dėžė“.

2.3. Metodo pasirinkimas

Išnagrinėjome populiariausius mašininio mokymosi algoritmus. Pastebėta, kad didžioji dalis algoritmų ir metodų turi panašias savybes ir sugebėjimus, nors jų veikimo principai žymiai skiriasi. Tolesnėje darbo veikloje bus naudojami šie metodai:

- sprendimų medžio metodas (angl. Decision Tree)
- KNN metodas (angl. K-Nearest Neighbours)
- atsitiktinis miško metodas (angl. Random Forest)
- daugiasluoksnio perceptrono metodas (angl. multilayer perceptron)

Šie protokolai pasirinkti dėl ganėtinai skirtingo veikimo principo, o tai leis turėti tikrinimo įvairovę ir mažiau paveiktas galutines bendras išvestis. KNN metodas pasirenkamas dėl savo atsparumo triukšmui. Tačiau taip pat reikia pabrėžti, kad šis algoritmas reikalauja normuotų reikšmių, dėl to reikės jo išvesčiai taikyti mažesnį galutinio atsakymo koeficientą. Atsitiktinis miško metodas pasirenkamas dėl bendro našumo ir efektyvumo, bei pranašumo prieš sprendimų medį. Daugiasluoksnio perceptrono metodas pasirinktas dėl įvairovės, kadangi labiausiai skiriasi nuo kitų.

3. SISTEMOS STRUKTŪRA

Siekiant sukurti efektyviai veikiančią sistemą, reikia turėti planą ar projektą, kurį būtų galima realizuoti. Šiame skyriuje pagrindinis dėmesys bus skiriamas sistemos projektavimui. Skyrius sudarytas taip, kad būtų išsprendžiami analizės metu iškelti uždaviniai.

3.1. Sistemos platforma

Pradedant projektuoti bet kokią programinę sistemą, reikia suplanuoti, kokia platforma bus naudojama kaip pagrindas. Žemiausias lygmuo IT srityje yra fizinis lygmuo. Į fizinį lygmenį šiuo atveju įeina fizinė vietovė ir aparatinė įranga.

Dirbant su tinklo duomenimis, fizinė lokacija gali turėti labai daug įtakos renkamai informacijai ir pačiam projektui. Šis projektas yra taikomas srautinių DDoS atakų stebėjimui globaliu mastu. Dėl šios priežasties, geriau pasirinkti vietovę, iš kurios patogiai būtų galima pasiekti daug skirtingų adresų. Šiuo atveju pasirenkama Europa. Europos sąjunga yra geriau žinoma teritorija, todėl galima lengviau pasirinkti stebimus adresus. Kitas svarbus faktorius yra tai, kad Europa turi gerą tinklo infrastruktūrą, bei didelius srauto kiekius. Tikslios lokacijos pasirinkimui logiška naudoti miestą, turintį gerą tinklo infrastruktūrą ir esantį kuo arčiau geografinio stebimos teritorijos centro. Tam šiuo atveju tinka Frankfurtas.

Išsirinkus fizinę lokaciją, reikia suprojektuoti, kokia aparatinė įranga ar jos atitikmuo bus naudojamas. Kadangi nuspręsta vietovė yra Frankfurtas, naudoti savo nuosavą aparatinę įrangą galimybės nėra. Tam tikslui bus pasirenkama debesijos nuoma. Projekto metu bus nuomojamas virtualus serveris. Virtualus serveris yra pigesnis variantas, kurio resursų pilnai užtenka projekto reikmėms. Serverio resursų pasirinkimas labiausiai yra įtakojamas duomenų saugojimo platformos. Kadangi manomai skaičiavimų darbo metu nebus daug, pasirenkamas dviejų branduolių procesorius. Atminties reikės šiek tiek daugiau, kadangi platformoje planuojama naudoti duomenų bazę, o pastaroji veikia geriau, esant daugiau operatyviosios atminties. Kietojo disko talpa turi būti ganėtinai didelė, kadangi reikės laikyti daug statistinių duomenų. IP adreso užteks vieno, kadangi ši sistema neturėtų pasiekti tokio srauto lygio, kuris įtakotų IP adreso blokavimą.

Planuojamos sistemos specifikacijos:

- Virtuali x64 bitų sistema;
- 2 procesoriaus branduoliai;
- 4 GB operatyviosios atminties;
- 100 GB disko talpa;
- 1 IPv4 adresas.

Paskutinis svarbus sistemos platformos elementas yra operacinė sistema. Operacinės sistemos pasirinkimą įtakoja kelios priežastys. Pirma, siekiama vengti papildomų išlaidų, dėl to renkamasi

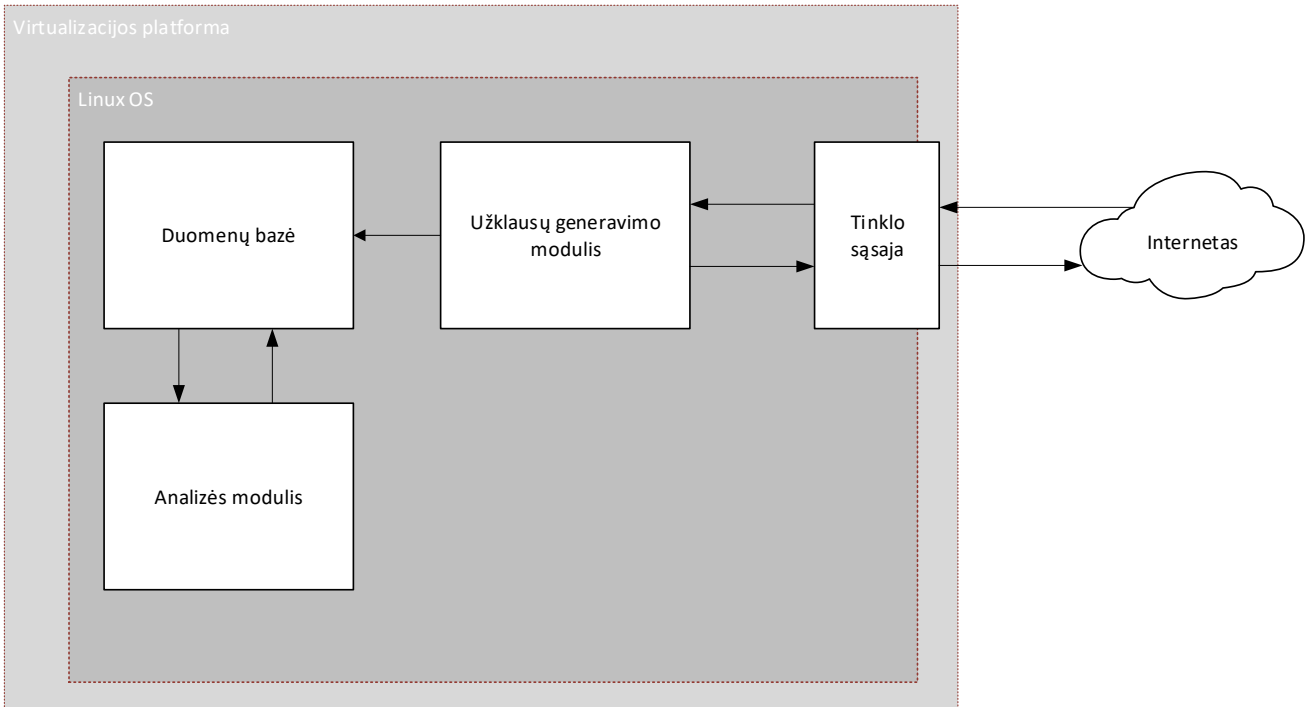
atvirojo kodo operacinė sistema. Kitas faktorius yra sistemos resursų panaudojimas. Turint ribotus platformos resursus nėra reikalo naudoti didelę, išpūstą OS, vietoj to geriau kuo lengvesnė sistema. Taip pat svarbu pasirinkti OS, kuri būtų pažįstama ir suprantama. Paskutinis faktorius yra sistemos suderinamumas su kitais moduliais ir programavimo kalbomis, kuriuos planuojama naudoti sistemoje. Beveik visus šiuos reikalavimus atitinka daugelis „Linux“ atmainų, tačiau naujausios ir reikalingos įrankių versijos yra „Ubuntu“ OS.

3.2. Stebimas tinklas

Pasirinkus geografinę zoną, kurioje bus stebimas tinklas, sekantis žingsnis yra išsirinkti adresus kuriuos stebės sistema. Šie adresai turi būti pakankamai patikimi, kitaip projekto vykdymo metu jų pasiekiamumas gali būti prarastas ir taip bus trukdoma sėkmingai atlikti darbą. Projektui parinkta 100 populiarių svetainių iš Europos ir taip pat keletas papildomai parinktų svetainių iš Lietuvos, Ispanijos, Švedijos ir Vokietijos. Visi puslapiai buvo surinkti iš „Amazon Alexa“ analizės platformos.

3.3. Globali sistemos struktūra

Suplanavus sistemos platformą, po to reikia suprojektuoti, iš kokių elementų bus sudaryta sistema. Kaip jau minėta analizės išvadose, DDoS aptikimo sistema turi atlikti duomenų rinkimą, duomenų saugojimą, analizavimą ir atvaizdavimą. Dėl to logiška, kad sistema turi būti sudaryta iš atitinkamų komponentų (3.1 pav.).



3.1 pav. Sistemos komponentai

Sistemos darbas prasidės nuo užklausų generavimo modulio. Šis modulis bus atsakingas už duomenų rinkimą iš tinklo ir jų įkėlimą į duomenų bazę. Kaip matome šis modulis tiesiogiai abipusiu

ryšiu bendrauja su tinklo sąsaja, kadangi turi daryti užklausas apie adresų pasiekiamumą ir gauna iš adresų atsakymus. Tuo tarpu šio modulio ryšys su duomenų baze yra vienpusis, dėl to, kad moduliui nereikia jokios informacijos apie buvusias stebimo tinklo mazgų būsenas.

Sekantis sistemos elementas yra duomenų bazė. Kaip jau anksčiau minėta, duomenų bazė vienpusiu ryšiu gauna informaciją iš užklausų generavimo modulio. Tačiau su analizės moduliu duomenų bazė bendrauja abipusiu ryšiu, nes analizės modulis nuskaito statistinę informaciją, esančią duomenų bazėje. Grįžtamasis ryšys yra pažymėtas, dėl to, kad atlikus analizę, rezultatus reikia kažkur įrašyti ir jau turint duomenų bazę, nėra reikalo papildomai kurti dar vienos atskiros duomenų talpos.

Analizės modulis šioje sistemoje atlieka didžiausią darbą. Šis modulis nuskaito duomenis iš DB ir tada pagal pasirinktą algoritmą apskaičiuoja ar tam tikru metu, tikrinamoje vietoje vyksta DDoS ataka.

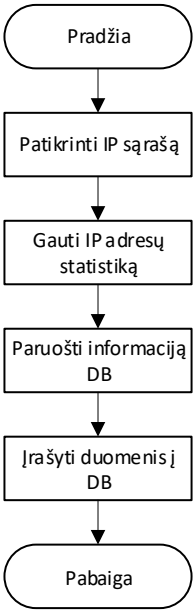
3.4. Sistemos elementų veikimo specifika

Praeitame skyriuje buvo aprašyta, ką daro kiekvienas elementas ir kaip jie susieti vienas su kitu. Toliau bus nagrinėjama kiekvieno elemento veikimo specifika. Į tai įeis modulių veikimo algoritmai, gaunamos duomenų struktūros, duomenų bazės struktūra ir kita.

3.4.1. Užklausų generavimo modulis

Šis modulis, generuodamas užklausas į nustatytus IP adresus, renka informaciją apie jų pasiekiamumą. Be abejo, prieš renkant informaciją apie IP adresus, reikia žinoti kokius IP adresus reikia stebėti. Šio projekto atveju, stebimų adresų sąrašas bus nustatytas iš anksto ir įrašomas į failą.

Darbo pradžioje modulis nuskaitys IP adresų sąrašą (3.2 pav.) ir surinks reikalingus duomenis, paruoš juos įrašyti į DB ir tada įrašys. Šis modulis bus reguliariai aktyvuojamas operacinės sistemos, pagal nustatytą grafiką.



3.2 pav. Planuojama duomenų rinkimo modulio struktūra

Sekantis žingsnis yra gauti IP adresų pasiekiamumo duomenis. Šie duomenys bus renkami sąlyginai dažnai ir iš didelio adresų skaičiaus, dėl to reikės modulį daryti taip, kad jis galėtų sėkmingai kurti keletą atskirų, vienu metu dirbančių procesų. Duomenys bus renkami naudojant analizės metu pasirinktą „Traceroute“ komandos atmainą „MTR“. Šis programa gali atiduoti duomenis įvairiu formatu, kas žymiai palengvina programos integravimą į užklausų modulį. Planuojama duomenis rinkti JSON formatu, reikalaujant grąžinti IP adresą, šuolių skaičių, AS numerį, laiką, paketų praradimą ir be abejo vėlinimą

Surinkus reikalingą informaciją apie IP adresą, ją modulis turės atitinkamai apdoroti ir paruošti įkėlimui į duomenų bazę. Tai bus struktūra, tinkanti standartinėms DB užklausoms. Sekančiame žingsnyje šie, jau paruošti duomenys, bus siunčiami į duomenų bazę kartu su kitų adresų duomenimis.

3.4.2. Duomenų bazės struktūra

Šio projekto duomenų bazės struktūra yra ganėtinai paprasta. Realizacijos metu duomenų bazėje bus naudojama viena lentelė. Lentelė bus panaši į užklausų modulio gautą informaciją iš „MTR“ (3.1 lent.)

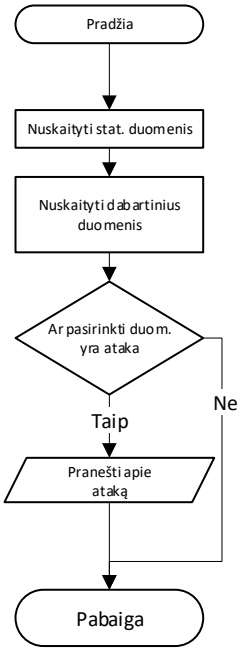
3.1 lentelė. Duomenų lentelės pavyzdys

index	timestamp	ip_address	ASN	hop_count	final_hop	ping
1	1529782710	154.16.202.1	AS61317	1	0	237
2	1529782710	62.115.12.14	AS1299	2	0	376
3	1529782710	62.115.120.7	AS1299	3	0	896
4	1529782710	62.115.137.167	AS1299	4	0	731
5	1529782710	62.115.144.199	AS1299	5	0	892
6	1529782710	1.1.1.1	AS13335	6	1	687

Šioje lentelėje, kaip ir „traceroute“ duomenyse, matome laiką, kada buvo padarytas testas („timestamp“), IP adresas („ip_address“), AS numeris („ASN“), šuolių skaičius iki pasirinkto mazgo („hop_count“) ir vėlinimas („ping“). Šie duomenys, kaip jau anksčiau minėta, reikalingi sėkmingam projektuojamos sistemos darbui. Reikia atkreipti dėmesį, kad šiame lentelės pavyzdyje yra kelias iki vieno adreso, bet lentelėje įrašyti 6 IP adresai. Taip yra dėl to, kad analizuojant ir atvaizduojant atakas ir anomalijas, bus reikalinga informacija, apie visus mazgus iki iš anksto nustatytų adresų. Dėl šios priežasties, lengvam galutinių mazgų atskyrimui nuo tarpinių mazgų, yra pridėtas stulpelis, pasakantis ar adresas yra paskutinis mazgas, ar ne („final_hop“). Ši reikšmė yra paprasta – 0 arba 1. Taip pat lentelė turi indeksavimo reikmėms skirtą unikalią reikšmę – „index“

3.4.3. Duomenų analizės modulis

Duomenų analizės modulis yra svarbiausia DDoS atakų aptikimo sistemos dalis. Kaip galima spręsti iš pavadinimo, šio modulio darbas yra nuskaityti surinktus statistinius duomenis ir apskaičiuoti ar pasirinktu momentu vyksta ataka (3.3 pav.).



Modulio darbas prasideda nuo ribinės reikšmė skaičiavimo. Tam planuojama panaudoti vidutines reikšmes pagal nustatytą laiko periodą. Taip pat projekto metu bus bandoma pritaikyti mašininį mokymąsi (angl. *machine learning*), siekiant pasiekti geresnių projekto rezultatų. Vidutiniai duomenys bus paremti visais renkamais duomenimis: šuolių skaičiumi, vėlinimu, kelio pasikeitimais ir kita.

Tolimesniame etape momentiniai duomenys yra lyginami su vidutinių reikšmių arba mašininio mokymosi pagalba sukurtomis reikšmėmis. Modulis, sulyginęs reikšmes, nuspręs ar pranešti apie rąstą ataką, ar ne.

3.5. Apibendrinimas

Šiame skyriuje buvo suplanuota, iš kokių elementų turi būti sudaryta sistema. Aprašyta kokios projekto dalys turi dirbti kartu ir kaip informacija eina nuo mazgo iki vartotojo. Informacijos rinkimas prasideda nuo analizės modulio, toliau informacija keliauja į duomenų bazę. Iš DB analizės modulis surinkęs duomenis, pateikia vartotojui visą reikalingą informaciją. Išskyrus sistemos modulius ir jų svarbiausius darbo aspektus, toliau galima realizuoti sistemą su tikra programine įranga ir tikslia struktūra, bei informacija.

4. SISTEMOS REALIZACIJA

Šiame skyriuje aprašoma srautinių DDoS atakų aptikimo sistemos realizavimas. Bus apžvelgiama kiekvieno atskiro sistemos elemento veikla bei bendras jų darbas. Realizacijos aprašymo tvarka orientuojama pagal duomenų kelią, t.y. pradedant nuo išorinio statistikos rinkimo įrankio iki analizės modulio rezultato.

4.1. Maršrutų statistika

Pirmas žingsnis realizuojant įrankį, yra duomenų rinkimas. Duomenų rinkimą galima išskirstyti į dar keletą skyrių. Arčiausiai tinklo ir OS yra išorinis įrankis „MTR“. Maršrutų statistika renkama naudojant „Linux OS“ maršrutų stebėjimo įrankį. Mūsų atveju, įrankis atiduoda reikšmes į standartinę „Linux“ išvestį. Toks būdas yra tinkamas dėl to, kad šį įrankį iškviečia duomenų rinkimo modulis, apie kurį bus kalbama vėliau.

Maršrutų rinkimo užklausą, iškviestą duomenų rinkimo scenarijaus, atspindi ši komanda:

mtr -zn --json -c 5 -i 0.2 Y.Y.Y.Y

Užklausos parametrai:

- -z – ASN rodymas. Ši dalis naudojama maršruto aprašymui, kuris būtų nepriklausomas nuo IP adresų ir taip analizuojant duomenis atakoms atpažinti;
- -n – nerodyti domeno. Norime matyti tik IP adresus;
- --json – gražinti duomenis JSON formatu. JSON patogus ir universalus metodas duomenims aprašyti ir apdoroti;
- -c – atlikti n ICMP užklausų, šiuo atveju 5;
- -i – laiko tarpas tarp užklausų, šiuo atveju 0,2 sekundės;
- Y.Y.Y.Y – tikrinamo maršruto galutinis IP adresas.

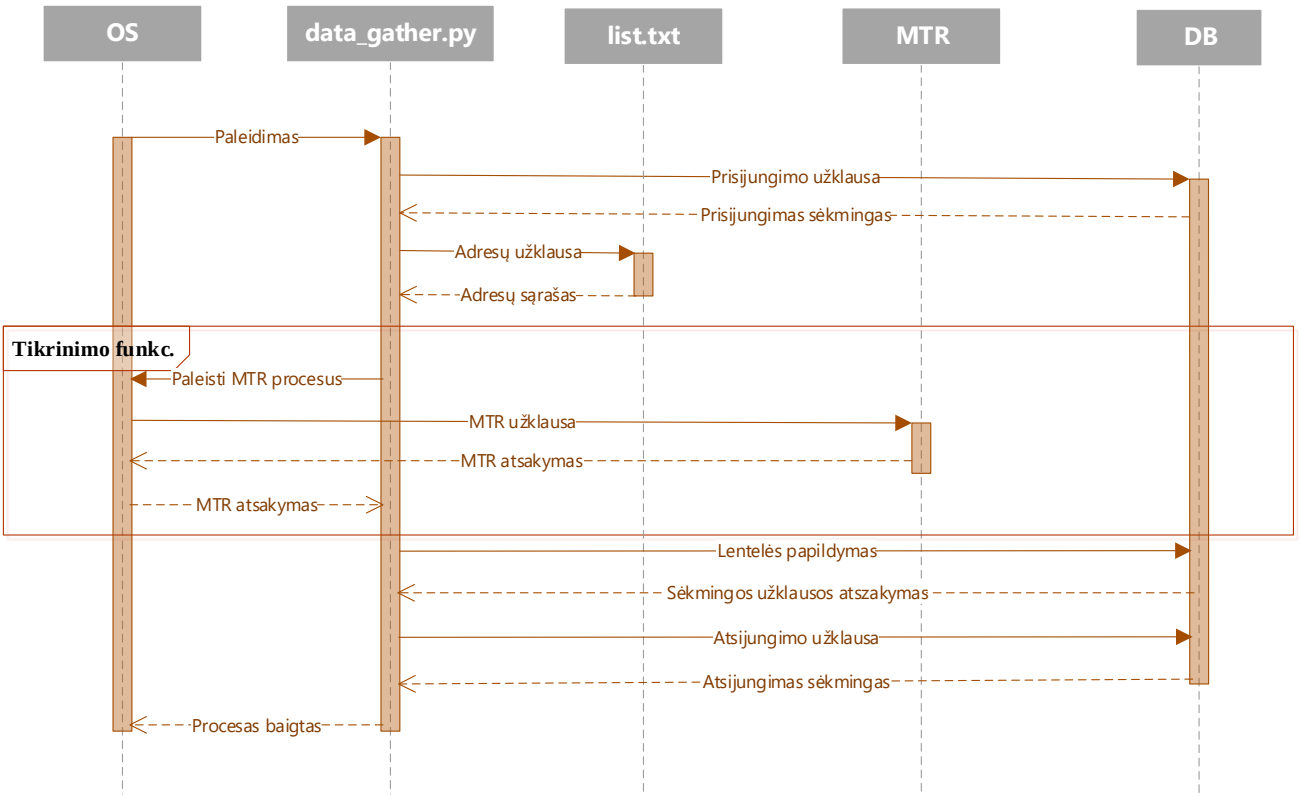
Duomenys atiduodami JSON formatu. Duomenų masyve yra kiekvieno mazgo, per kurį pereina informacija, pavyzdžiui, IP adresas, vėlinimas, paketų praradimas ir t.t.

„MTR“ atiduodamų reikšmių yra daug daugiau, negu sistemoje yra renkama. Duomenų rinkimo modulis pasirenka šias reikšmes:

- "count": "11" - hop_count, maršruto mazgo numeris, pradedant nuo serverio;
- "host": "195.2.21.58" – ip, Mazgo IP adresas;
- "ASN": "AS1273" – ASN, Mazgo ASN numeris;
- "Loss%": 0.00 – loss, Mazgo ICMP paketų praradimas;
- "Avg": 46.89 – ping, Mazgo atsako greitis.

4.2. Užklausų generavimo modulis

Kaip jau anksčiau minėta, maršrutų rinkimo modulis yra „Python“ kalba parašytas scenarijus, kurio pagrindinė užduotis yra surinkti duomenis apie maršrutą ir įvesti juos į DB. Scenarijus yra paprastas, sudarytas viso labo iš kelių funkcijų, tačiau tuo pačiu vykdymo metu jis bendrauja su keliomis skirtingomis sistemomis. Tai galima matyti toliau pateikiamoje sekos diagramoje (4.1 pav.).



4.1 pav. Užklausų generavimo modulio sekos diagrama

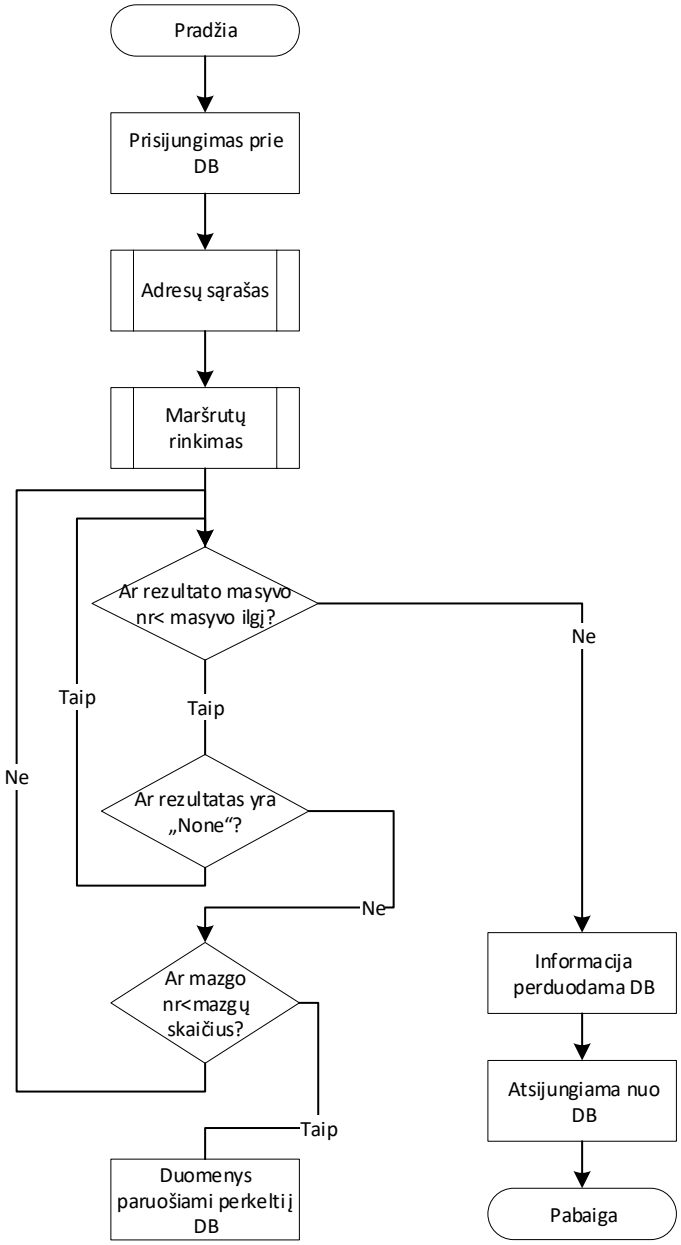
Modulio pavadinimas yra „data_gather“. Modulis yra paleidžiamas OS programos „Cron“. „Cron“ modulį paleidžia kartą į minutę. Paleistas scenarijus iš pradžių prisijungia prie „MySQL“ DBVS, kad atsiradus poreikiui, būtų galima atlikti reikiamus veiksmus.

Sekantis žingsnis yra adresų, kurių maršrutų informaciją reikia surinkti, sąrašas. Sąrašas nuskaitomas iš failo.

Turint sąrašą, modulis gali pradėti rinkti duomenis. Svarbią duomenų rinkimo dalį atlieka tikrinimo funkcija, apie kurią netrukus kalbėsime išsamiau. Sekos diagramos atžvilgiu, tai yra tiesiog kreipimasis į OS, kad ši paleistų tam tikrą kiekį MTR komandų ir po to jas įvykdžius, gražintų jų standartinę išvestį.

Surinkus duomenis, modulis juos įrašo į DB ir uždaro DBVS sesiją.

Norint tiksliau nagrinėti „data_gather“ modulio veikimą, galima pasitelkti blokinę schemą (4.2 pav.).



4.2 pav. Užklausų generavimo modulio blokinė schema

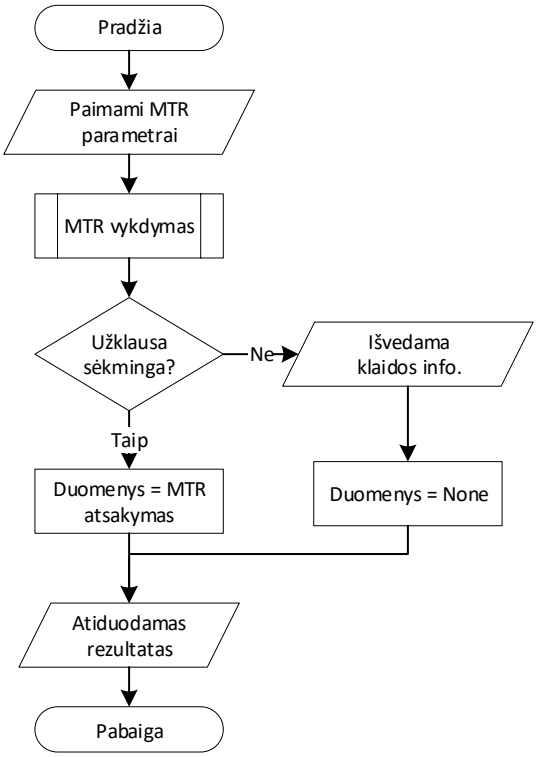
Iš sekos diagramos jau susidarėme supaprastintą veikimo aprašymą. Prisijungimas prie DBVS vyksta panaudojant darbo sistemai sukurtą MySQL vartotoją ir vietinį IP adresą. Iš karto po to, naudojama trumpa funkcija skirta apdoroti maršrutų adresų sąrašą. Sekančiame žingsnyje yra vykdoma maršrutų rinkimo funkcija, apie kurią kalbėsime atskirai.

Maršrutų funkcija pagrindiniam programos ciklui grąžina visas jau apdorotas reikšmes, sudėtas į vieną masyvą. Deja, masyvas turi ir reikšmes, gautas iš MTR programos, kurių mes nenaudojame. Kadangi duomenų masyvas yra kelių lygių (kiekvienas maršrutas turi masyvą sudarytą

iš maršruto mazgų) jo apdorojimui naudojamas ciklas cikle. Blokinėje diagramoje tai atspindi visi trys pasirinkimo blokai. Be minėtų adresų masyvo ir mazgų masyvų apdorojimo, trečias pasirinkimas atspindi patikrą, skirtą pašalinti, nepavykusius skenavimus, pavyzdžiui ištrintas DNS adresas.

Duomenų paruošimo cikle duomenys paruošiami atiduoti į DB kiekvieną adreso tikrinimo iteraciją. Paruošus visą masyvo informaciją persiuntimui į DB, duomenys išsiunčiami ir „MySQL“ sesija uždaroma. Uždarius sesiją, modulio darbas baigtas.

Svarbi modulio dalis yra tikrinimo funkcija (4.3 pav.).



4.3 pav. Tikrinimo funkcija

Pagrindinė šios funkcijos paskirtis yra paleisti tam tikrą nustatytą MTR procesų skaičių vieną šalia kito ir po to surinkti iš jų duomenis. Taip nuspręsta daryti dėl lėto MTR darbo. Kadangi šis įrankis tikrina maršrutus ICMP „Ping“ metodu, šis procesas turi dirbtines laiko pertraukas tarp užklausų. Pertraukas galima sumažinti, bet tai gali neigiamai paveikti rezultatus, kadangi tinklo įrenginiai neretai neatsako į ICMP užklausas, jeigu jų būna pateikiama neįprastai daug. Patikrinti 100 adresų iš eilės modulis užruktų apie 8 minutes, tuo tarpu paleidus 30 lygiagrečių procesų darbas atliekamas per mažiau nei 30 sekundžių.

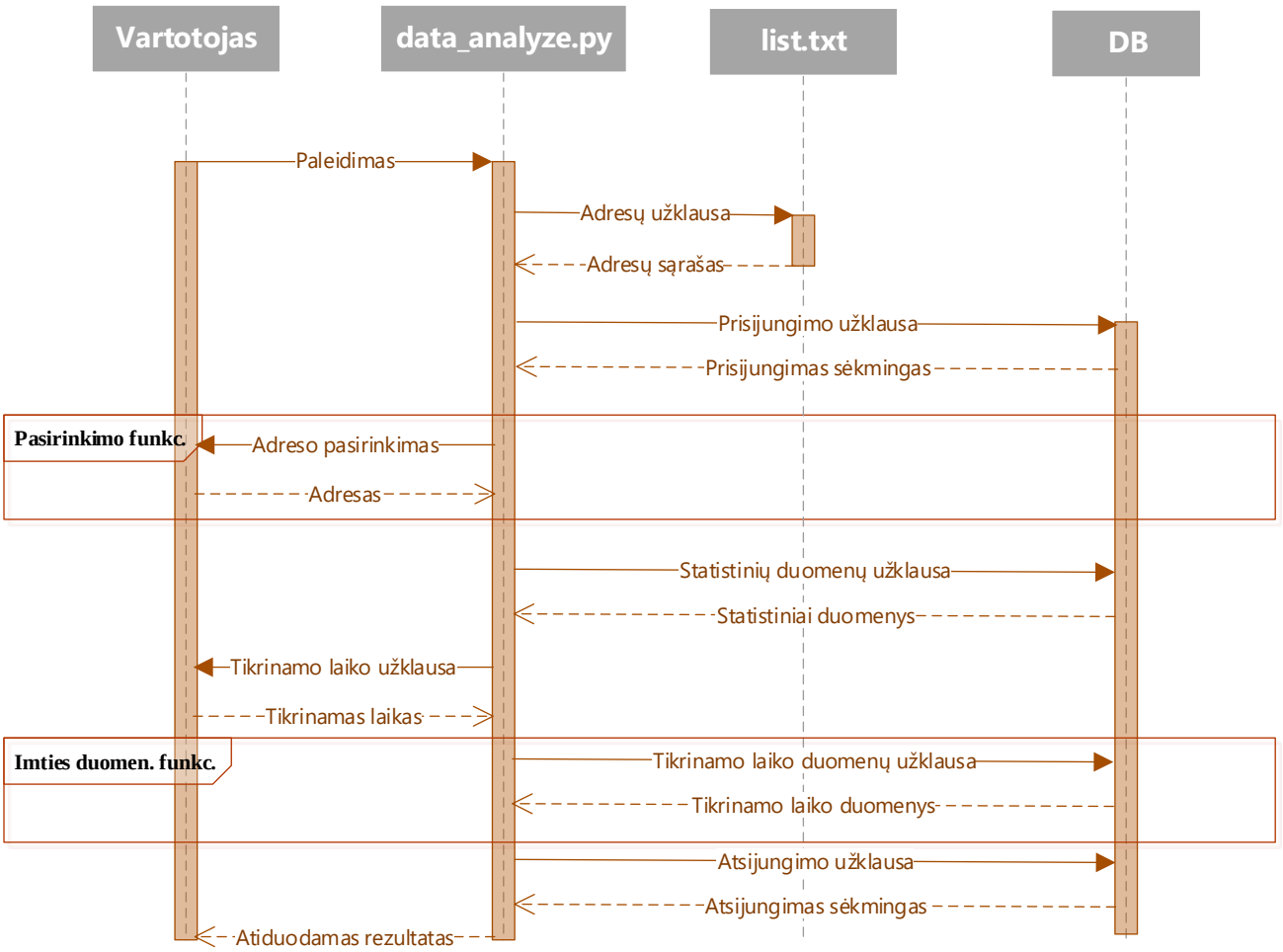
Projekto modulyje funkcija gauna MTR parametrus: adresus, ICMP paketų kiekį, laiko periodą tarp užklausų. Toliau funkcija iškviečia MTR procesą per OS valdymo bibliotekas. Jei užklausa sėkminga, į duomenis atiduodamas rezultatas, jeigu nesėkminga atiduodamas tuščias masyvas.

Reikia paminėti, kad daugelio procesų paleidimą kontroliuoja ne ši funkcija, o pagrindinis scenarijus, tačiau siekiant paprastesnio blokinės schemos atvaizdavimo šis procesas nebuvo žymimas.

4.3. Duomenų analizės modulis

Kita esminė sistemos dalis yra duomenų analizės modulis. Šis modulis veikia nepriklausomai nuo duomenų rinkimo modulio, kadangi duomenų rinkimo modulis yra paleidžiamas OS pagal nustatytą grafiką, o analizės modulis yra paleidžiamas rankiniu būdu.

Kaip ir duomenų rinkimo modulis, analizės modulis turi bendrauti su pora sistemų ir taip pat vartotoju (4.4 pav.).



4.4 pav. Duomenų analizės modulio sekos diagrama

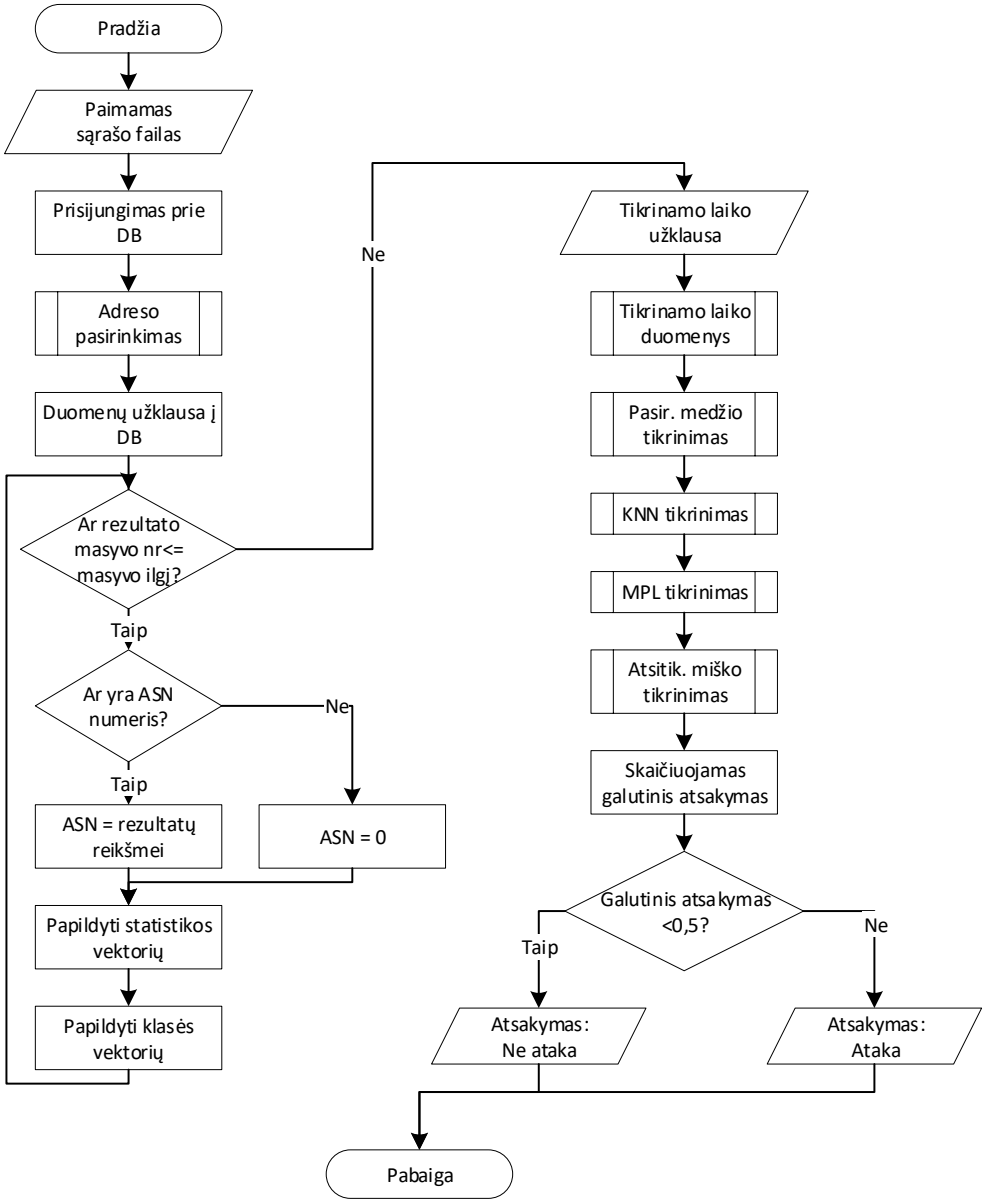
Modulio darbas pradedamas, kai vartotojas rankiniu būdu per OS paleidžia „data_analyze.py“ scenarijų. Kaip ir duomenų rinkimo modulis, šis taip pat nuskaito adresų sąrašą. Sekančiu žingsniu taip pat prisijungiama prie DBVS.

Po to trumpa funkcija atlieka tikrinamo adreso pasirinkimą. Vartotojui pasiūloma pasirinkti analizuojamą maršrutą ir tada renkami šio maršruto statistiniai duomenys. Duomenys surenkami iš duomenų bazės. Tai yra du duomenų vektoriai: statistika ir statistikos klasifikavimas. Surinkus duomenis apie maršrutą, vartotojo yra klausiama, kokią duomenų imtį jis nori patikrinti. Imties

duomenų rinkimas yra labai panašus kaip statistinių duomenų rinkimas, tačiau yra atliekamas atskiros funkcijos ir paima tik vieną statistikos vektorių, kadangi ši reikšmė dar nėra klasifikuota.

Iš DB surinkus visus reikalingus duomenis, modulis atsijungia. Toliau vykdomas duomenų tikrinimas ir vartotojui išvedamas rezultatas.

Toliau pateikiama blokinė modulio schema (4.5 pav.).



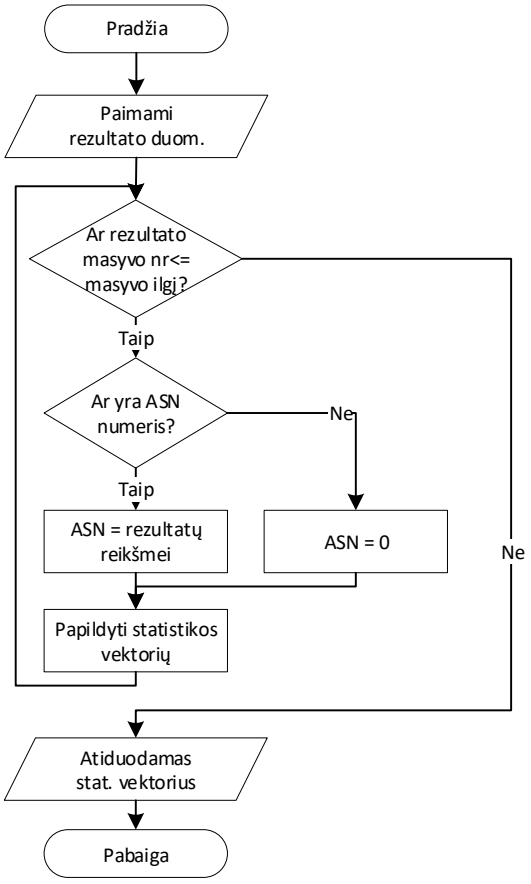
4.5 pav. Duomenų analizės modulio blokinė schema

Kaip jau matėme sekos diagramoje, sistemos darbas prasideda nuo tų pačių žingsnių, t.y. paimamas maršrutų sąrašo failas ir prisijungiama prie DBVS. Sekantis žingsnis yra adreso pasirinkimas. Čia darbą atlieka nesudėtinga funkcija, kuri vartotojui pateikia maršrutų sąrašą, kurį gavo iš sąrašo failo ir tada duoda pasirinkti, kokį adresą norima tikrinti.

Pasirinkto adreso duomenų ieškoma duomenų bazėje. Maršrutų duomenys turi turėti du požymius, kad būtų paimti į duomenų masyvą: turi turėti atakos ar reguliaraus srauto žymėjimą ir turi turėti reikalingą galutinį adresą. Kadangi visi duomenys paimami, kaip vienas dviejų lygių masyvas,

reikia atlikti duomenų apdorojimą. Blokinėje schemoje ši vieta supaprastinta, du blokai: „Ar rezultato masyvo nr. <= masyvo ilgį?“ ir „Ar yra ASN numeris?“ atspindi šį procesą. Trumpai tariant, ciklo metu yra sumuojami du vektoriai: statistikos ir statistikos tipo (klasės). Vektoriai sumuojami, nes mašininio skaičiavimo algoritmams reikia paprasto reikšmių rinkinio, o ne kelių lygių duomenų masyvų.

Perėjus visą statistikos masyvą, gautą iš DB, modulis pereina prie tikrinamų duomenų pasirinkimo. Pasirinkimo metu vartotojui siūloma įrašyti laiką, kurio statistikas norima tikrinti. Toliau tikrinamąjį vektorių paruošia funkcija, labai panaši į dalį pagrindinio scenarijaus (4.6 pav).



4.6 pav. Vektorių paruošia funkcija

Esminis funkcijos skirtumas yra tai, kad pastaroji nerenka klasės vektoriaus, kadangi šie duomenys dar jos neturi. Sekantis žingsnis yra vektorių atidavimas mašininio skaičiavimo algoritmams. Paeiliui informaciją patikrina pasirinkimo medžio, KNN, daugiasluoksnio perceptrono ir atsitiktinio miško algoritmai. Kiekvienas algoritmas atiduoda 0 arba 1, kurie atitinka klases: reguliarus srautas arba ataka. Galų gale, kiekviena iš reikšmių dauginama iš koeficiento ir sumuojama pagal formulę:

$$Rezultatas = knn * 0{,}15 + mpl * 0{,}25 + ats.mišk.* 0{,}3 + spr.med * 0{,}3$$

Jeigu gaunamas rezultatas yra mažiau nei 0,5, imtis traktuojama kaip reguliarus srautas, jei daugiau nei 0,5 – ataka.

5. SISTEMOS REZULTATŲ ANALIZĖ

Šiame skyriuje analizuosime realizuotos sistemos veikimą. Iš pradžių bus tikrinami surinkti duomenys ir toliau jie bus paruošiami sistemos analizės moduliui. Darbo metu buvo tikrinama 100 adresų ir maršrutų į juos. Atlikus pradinę statistikos analizę, bus pasirenkamas adekvatus adresų skaičius analizės modulio testavimui.

Analizės modulio testavimo metu pasirinktus ir sužymėtus adresus tikrinsime su mašininių algoritmų komplektu. Pagal komplekto rezultatus spręsime, kaip analizės modulis atliko paskirtą užduotį.

5.1. Duomenų rinkinys

Realizavus sistemą, ji buvo paleista kaupti duomenis. Duomenys apie 100 populiarių Europos adresų ir maršrutų iki jų, buvo kaupiami dvi savaites. Po dviejų savaičių duomenų rinkimas buvo sustabdytas. Duomenų rinkimą buvo nuspręsta sustabdyti dėl kelių priežasčių. Pirma, duomenų bazėje buvo 16 milijonų įrašų. Tai nėra didelis duomenų kiekis, tačiau, kai naudojamos užklausos filtruoja visus duomenis pagal įvairius filtrus, o ne indeksą, duomenų apdorojimas užtrunka jaučiamą laiko tarpą (30s. – 1min.). Taip pat, siekiant apmokyti algoritmus atpažinti atakas, iš pradžių duomenis reikia sužymėti rankiniu būdu. Tokį darbą lengviau atlikti su mažesniu duomenų kiekiu.

Išsamesnė informacija apie surinktus duomenis:

- 100 adresų ir maršrutų;
- 16025164 įrašų;
- 22319 statistikos komplektų (imta kartą į minutę);
- užimta vieta – 1,6 GB.

5.2. Statistinių duomenų analizė ir žymėjimas

Kaip jau anksčiau minėjome, statistinių duomenų yra iš tiesų nemažai ir norint juos analizuoti su sistemos analizės moduliu, pirma reikės duomenis sužymėti į atskirus vektorius: atakų statistika ir normalaus darbo statistika.

Deja, nėra paprasto būdo nuspręsti, ar tam tikri duomenys paimti vykdant ataką, ar ne. Darbo analizės metu nepavyko rasti šaltinių ar duomenų bazių, kuriuose būtų viešai skelbiama, kad tam tikru metu, tam tikras adresas buvo puolamas DDoS ataka.

Tačiau nagrinėjant esamus atakų atpažinimo metodus, daugelis tai daro profiliuojant turimus duomenis, kaip ir analizės metu minėtas Yoohwan Kim metodas. Rankinio žymėjimo metu bus remiamasi panašia logika, t.y. ieškoma įrašų, kurių statistiniai duomenys būtų stipriai nukrypę nuo standartinių reikšmių.

Atakų ieškojimas pagal statistinių reikšmių, tokių kaip ryškus vėlinimas ar didelis prarandamų paketų kiekis, turi vieną trūkumą. Šie statistinių reikšmių pasikeitimai ne visada yra įtakojami srautinių DDoS atakų. Tai gali sukelti ir kiti stambaus mąsto tinklo įvykiai, kaip ryšio linijos nutraukimas, įrangos gedimai, bloga konfigūracija ar kitos žmonių padarytos klaidos. Tačiau, kaip jau analizės metu aptarėme, visos šios problemos turi savitus tinklo pasikeitimo bruožus. Vienintelis įvykis, kurio neina atskirti pagal turimus duomenis nuo srautinės DDoS atakos, yra didelio ateinančio srauto linijos perpildymas. Šio darbo metu tokius įvykius traktuosime kaip DDoS ataką, kadangi jų veikimas aukos atžvilgiu yra identiškas.

Atsižvelgiant į anksčiau nagrinėtą kitų tinklo įvykių poveikį tinklo būklei, žymint DDoS atakas bus ieškoma tokių požymių:

- žymiai didesnis atsako vėlinimas – 2,5 karto ir daugiau (išskyrus pasikeitus maršrutui);
- 20% arba didesnis paketų praradimas;
- anksčiau minėtų požymių egzistavimas bent 2 – 3 paskutiniuose maršruto mazguose;
- paketų praradimas nesiekia 100% (pašalinti neveikiančio arba neaktyvaus serverio galimybę);
- anksčiau minėtų požymių pasikartojimas keliuose duomenų rinkiniuose iš eilės.

Pradžioje atakų paieška vykdoma rūšiuojant visus paskutinių mazgų įrašus pagal vėlinimą nuo didžiausio iki mažiausio. Tokioje (5.1 lentelė) išvestyje galima aptikti įvykius, kurie potencialiai yra DDoS atakos.

5.1 lentelė. Pagal vėlinimą rūšiuotos reikšmės

id	timestamp	ip	destination_ip	ASN	hop count	last count	ping	loss	anomaly
7003965	2019-04-28 10:15	207.150.217.136	howtogermany.com	AS3064	15	1	3138	70	
7005683	2019-04-28 10:17	207.150.217.136	howtogermany.com	AS3064	15	1	2608	90	
7004823	2019-04-28 10:16	207.150.217.136	howtogermany.com	AS3064	15	1	2347	60	
688057	2019-04-23 09:04	93.92.135.251	s-bahn-berlin.de	AS29014	8	1	2045	40	
4119994	2019-04-26 02:47	93.92.135.251	s-bahn-berlin.de	AS29014	8	1	1304	40	
5238521	2019-04-27 00:19	52.29.143.106	peli.com	AS16509	11	1	1252	0	
1034964	2019-04-23 15:47	193.219.137.110	smm.lt	AS5479	7	1	1093	0	
980176	2019-04-23 14:44	93.92.135.251	s-bahn-berlin.de	AS29014	8	1	1074	0	
952626	2019-04-23 14:12	193.219.137.110	smm.lt	AS5479	7	1	1003	40	
953490	2019-04-23 14:13	193.219.137.110	smm.lt	AS5479	7	1	990	0	
2021871	2019-04-24 10:42	217.114.85.70	swegon.com	AS30811	7	1	983	0	
1033219	2019-04-23 15:45	193.219.137.110	smm.lt	AS5479	7	1	962	0	
1034090	2019-04-23 15:46	193.219.137.110	smm.lt	AS5479	7	1	918	0	
1032344	2019-04-23 15:44	193.219.137.110	smm.lt	AS5479	7	1	885	0	

Šioje išvestyje galima matyti, kad howtogermany.com, s-bahn-berlin.de, swegon.com, smm.lt galimai buvo paveikti srautinių DDoS atakų. Reikia nepamiršti, kad siekiant atrinkti nereikalingus

duomenis, šioje lentelėje matomi tik galutinių adresų įrašai. Tolesnėje veikloje nagrinėjamas kiekvieno atvejo maršrutas. 2019-04-23 09:04 s-bahn-berlin.de pavyzdys (5.2 lentelė):

5.2 lentelė. 2019-04-23 09:04 s-bahn-berlin.de statistikos imtis

id	timestamp	ip	destination_ip	ASN	hop count	last count	ping	loss	anomaly
688050	2019-04-23 09:04	154.16.202.1	s-bahn-berlin.de	AS61317	1	0	4	0	
688051	2019-04-23 09:04	62.115.12.14	s-bahn-berlin.de	AS1299	2	0	1	0	
688052	2019-04-23 09:04	62.115.120.7	s-bahn-berlin.de	AS1299	3	0	19	0	
688053	2019-04-23 09:04	62.115.139.7	s-bahn-berlin.de	AS1299	4	0	18	0	
688054	2019-04-23 09:04	85.158.2.12	s-bahn-berlin.de	AS29014	5	0	2080	40	
688055	2019-04-23 09:04	85.158.2.11	s-bahn-berlin.de	AS29014	6	0	2069	40	
688056	2019-04-23 09:04	93.92.134.19	s-bahn-berlin.de	AS29014	7	0	2056	40	
688057	2019-04-23 09:04	93.92.135.251	s-bahn-berlin.de	AS29014	8	1	2045	40	

Šioje lentelėje aiškiai matome, kad nuo 5 maršruto mazgo yra aiškus vėlinimo padidėjimas, bei paketų praradimas. Daug tikimybės, kad tai yra DDoS ataka. Tuo tarpu, reguliaraus srauto pavyzdyje tokių vėlinimų nėra (5.3 lentelė):

5.3 lentelė. Reguliarios s-bahn-berlin.de statistikos pavyzdys

id	timestamp	ip	destination_ip	ASN	hop count	last count	ping	loss	anomaly
701042	2019-04-23 09:19	154.16.202.1	s-bahn-berlin.de	AS61317	1	0	1	0	
701043	2019-04-23 09:19	62.115.12.14	s-bahn-berlin.de	AS1299	2	0	1	0	
701044	2019-04-23 09:19	62.115.120.7	s-bahn-berlin.de	AS1299	3	0	19	0	
701045	2019-04-23 09:19	62.115.139.7	s-bahn-berlin.de	AS1299	4	0	19	0	
701046	2019-04-23 09:19	85.158.2.12	s-bahn-berlin.de	AS29014	5	0	16	0	
701047	2019-04-23 09:19	85.158.2.11	s-bahn-berlin.de	AS29014	6	0	17	0	
701048	2019-04-23 09:19	93.92.134.19	s-bahn-berlin.de	AS29014	7	0	16	0	
701049	2019-04-23 09:19	93.92.135.251	s-bahn-berlin.de	AS29014	8	1	17	0	

Atlikus duomenų analizę, buvo atrinkti šie adresai ir maršrutai į juos tolesniems tyrimams:

- howtogermany.com
- s-bahn-berlin.de

Kiti peržiūrėti adresai ir jų maršrutai, turintys anomalijų, nepatenkino pakankamai anksčiau minėtų požymių. Daugelis pasižymėjo tik paskutinio mazgo vėlinimu, kuris gali įvykti dėl įvairių priežasčių. Kita dalis turėjo ir paketų praradimų, tačiau taip pat tik paskutiniame mazge.

Tolesnėje darbo veikloje bus siekiama išbandyti analizės modulio veikimą. Duomenų žymėjimas ir dėjimas į klasifikavimo vektorius bus atliekamas palaipsniui, siekiant išbandyti daugiau variacijų, tačiau galutinis žymėjimas, palikus 10 – 20% atakų vėlesniam aptikimui, yra toks (5.4 lentelė):

5.4 lentelė. Galutinis atakų žymėjimo skaičius

Adresas	Pažymėtų kaip ataka	Pažymėtų kaip ne ataka
howtogermany.com	4	400
s-bahn-berlin.de	13	624

5.3. Analizės modulio rezultatai

Kaip jau anksčiau minėta, toliau bus analizuojami statistiniai duomenys, kurie yra priskirti atakos arba reguliaraus srauto vektoriams. Siekiant ištirti sistemos veikimą ir efektyvumą, rezultatai tikrinami tarpiniais variantais.

Kadangi atakos, kurios stipriai įtakotų visą maršruto kelią nevyksta dažnai, atakų ir reguliaraus srauto vektorių duomenų kiekis skiriasi dešimtis kartų. Tokios savybės gali sukelti sunkumų mašininio mokymosi algoritmams sėkmingai atpažinti atakas. Patikrinti algoritmų efektyvumą tam tikrais atvejais, bus pradedama analizuoti su panašiu pavyzdžių skaičiumi tiek atakos, tiek reguliaraus srauto vektoriuose, o po to tikrinama pakopomis, didinant reguliaraus srauto vektorių.

5.3.1. Maršrutas „howtogermany.com“

Kaip jau minėta, tikrinant algoritmų veikimą, žymėjimas atliekamas palaipsniui. Su adresu howtogermany.com atliekami trys variantai su trimis imtimis. Tikrinamų duomenų imtys buvo pasirinktos pagal stebimas savybes. Imtis Nr. 1 buvo duomenimis panaši į atakos vektoriui priklausančius duomenis. Imtis Nr. 2 buvo tarpinis variantas tarp reguliaraus srauto ir atakos vektorių. Kaip imtis Nr. 3 buvo pasirinktas tipinio srauto pavyzdys. Išsamesnė informacija apie imtis pateikiama priede Nr. 1.

Kiekviena imtis buvo analizuojama su mašininio mokymosi algoritmais, išbandant visus tris vektorių variantus. Gauti rezultatai (5.5 lentelė):

5.5 lentelė. Maršruto „howtogermany.com“ analizės rezultatai

Imties nr.	Pažymėta ataka	Pažymėta ne ataka	Ar ataka?				Bendras koeficientas
			Pasir. medžio	KNN	MLP	Atsitikt. miško	
1	4	4	Taip	Taip	Ne	Taip	0.75
	4	40	Taip	Ne	Ne	Taip	0.6
	4	400	Taip	Ne	Ne	Taip	0.6
2	4	4	Taip	Ne	Ne	Ne	0.3
	4	40	Taip	Ne	Ne	Ne	0.3
	4	400	Taip	Ne	Ne	Ne	0.3
3	4	4	Ne	Ne	Ne	Ne	0
	4	40	Ne	Ne	Ne	Ne	0
	4	400	Ne	Ne	Ne	Ne	0

Pagal šio maršruto rezultatus matome, kad analizės modulis pasiekia siekiamą tikslą. Svarbiausias testas yra pirmos imties. Esant vienodiems atakos ir reguliaraus srauto vektoriams, analizės modulis sąlyginai geru rezultatu pripažino imtį kaip ataką. Iš keturių algoritmų, kaip ataka šios imties nepripažino tik daugiasluoksnis perceptronas (MLP). Pakeitus vektorių dydį į 1:10 santykį (atakos vektorius : reguliaraus srauto vektorius) K artimiausio „kaimyno“ (KNN) algoritmas taip pat nusprendė, kad nagrinėjami duomenys nėra ataka. Nepaisant to galutinis sprendimas ir toliau buvo

lygus atakai (koeficientas didesnis nei 0,5). Padidinus santykį iki 1:100 reikšmės išliko tokios pat kaip ir su 1:10 santykiu.

Antroji imtis šio maršruto tyrime, manomai buvo kitas ryšio sutrikimas, kadangi tai nebuvo panašu į ataką, bet turėjo daugiau tipinio srauto požymių su didesniu vėlinimu galiniame adrese. Kaip buvo galima tikėtis, lentelėje matome, kad imtis nebuvo pripažinta ataka. Visais vektorių dydžio santykio variantais gautas vienodas atsakymas – pasirinkimų medis pripažino ataką, tuo tarpu kiti algoritmai jos nerado. Galutinis koeficientas buvo 0,3. Ši reikšmė yra arti ribos (0,5), kuri nusako ar imtis yra ataka ar ne, tačiau taip pat pakankamai toli.

Trečioji tiriama duomenų imtis buvo reguliarus srautas. Šio maršruto reguliaraus srauto duomenys yra sąlyginai panašūs, dalis beveik identiški, todėl šios imties atsakymas buvo vienareikšmiškas: ne ataka.

5.3.2. Maršrutas „s-bahn-berlin.de“

Šis maršrutas turi daugiau duomenų, turinčių srautinės atakos požymių. Dėl šios priežasties, lyginant su praėjusiu maršrutu, darbe bus pateikiama daugiau imčių ir bandymų.

Tikrinant s-bahn-berlin.de buvo pažymėti 13 duomenų pavyzdžių, kad jie priklauso atakos vektoriui. Atitinkamai, kaip ir anksčiau reguliaraus srauto vektorius buvo daromas 1:1, 1:10 ir 1:100 santykiu, arba kitaip tariant: 13 imčių, 130 imčių ir 1300 imčių.

Šiame teste buvo analizuojamos 5 imtys. Visos paskirstytos pagal parametrus. Pirmoji, kaip ir anksčiau, labai panaši į jau pažymėtas atakas. Antroji ir trečioji yra šiek tiek mažesnius atakos rodiklius turinčios imtys. Paprastas imčių palyginimas gali būti pateiktas galinio adreso vėlinimu:

- imties nr. 1 vėlinimas apie 790 ms.;
- imties nr. 2 vėlinimas apie 280 ms.;
- imties nr. 3 vėlinimas apie 120 ms.;
- imties nr. 4 vėlinimas apie 39 ms.;
- imties nr. 5 vėlinimas apie 17 ms.

Ketvirtoji imtis yra reguliarus srautas su šiek tiek pakilusiu vėlinimu. Penktoji imtis yra tipinis šio maršruto pavyzdys. Išsamesnė imčių informacija pateikta priede Nr. 2.

Atlikus atpažinimo modulio testavimus, gavome 15 rezultatų variacijų (5.6 lentelė). Nagrinėjant vektorių santykį 1:1 visi algoritmai vieningai atsakydavo, kad 1,2,3 imtys yra atakos. Tai rodė geresnius rezultatus, nei anksčiau nagrinėto maršruto analizė. Tačiau 1:1 vektorių dydžio santykis neleido tinkamai apsispręsti daugiasluoksnio perceptrono (MLP) algoritmui. Šis metodas net ir reguliarų srautą palaikė ataka.

Antra ir trečia maršruto tyrimo dalys (1:10 ir 1:100 santykiai) atidavė identiškus rezultatus, todėl nagrinėsime juos bendrai. Pasirinkimo medžio, KNN ir atsitiktinio miško algoritmai šiuose metoduose grąžino tuos pačius atsakymus, kaip ir analizuojant imtis su vienodo dydžio vektoriais.

Pasikeitimai atsirado daugiasluoksnio perceptrono rezultatuose. Kaip prieš tai MLP gražino išvestį, kad visos imtys yra atakos, pakeitus santykį į 1:10 ir 1:100 šis rezultatas tapo priešingas ir buvo grąžintos išvestys, kad visos imtys nėra atakos.

5.6 lentelė. Maršruto „s-bahn-berlin.de“ analizės rezultatai

Imties nr.	Pažymėta ataka	Pažymėta ne ataka	Ar ataka?				Bendras koeficientas
			Pasir. medžio	KNN	MLP	Atsitikt. miško	
1	13	13	Taip	Taip	Taip	Taip	1
	13	130	Taip	Taip	Ne	Taip	0.75
	13	1300	Taip	Taip	Ne	Taip	0.75
2	13	13	Taip	Taip	Taip	Taip	1
	13	130	Taip	Taip	Ne	Taip	0.75
	13	1300	Taip	Taip	Ne	Taip	0.75
3	13	13	Taip	Taip	Taip	Taip	1
	13	130	Taip	Taip	Ne	Taip	0.75
	13	1300	Taip	Taip	Ne	Taip	0.75
4	13	13	Ne	Ne	Taip	Ne	0.25
	13	130	Ne	Ne	Ne	Ne	0
	13	1300	Ne	Ne	Ne	Ne	0
5	13	13	Ne	Ne	Taip	Ne	0.25
	13	130	Ne	Ne	Ne	Ne	0
	13	1300	Ne	Ne	Ne	Ne	0

Apžvelgiant bendrąjį koeficientą galime matyti, kad yra nemažai tarpinių reikšmių tarp 0 ir 1, kurias įtakojo MLP algoritmas, tačiau visi galutiniai atsakymai gaunami teisingi.

5.4. Išvados ir pastabos

Srautinių DDoS atakų aptikimo tyrimo metu buvo susidurta su keliais nesklandumais. Kaip jau anksčiau minėta, trūko galimybės patvirtinti ar tarp statistinių duomenų rastos anomalijos iš tiesų buvo DDoS atakos. Tai buvo sprendžiama, pagal analizės metu padarytas išvadas. Kita problema buvo tai, kad turimos informacijos kiekis neleido atlikti labai išsamių tyrimų. Įvykiai, kurių ieškome statistikoje, įvyksta retai ir normaliomis sąlygomis nėra reguliarūs, todėl iš 100 stebėtų adresų tik apie 10 turėjo maršrutų anomalijų, iš kurių viso labo pora atitiko keliamus reikalavimus.

Nagrinėjant šiuos du maršrutus pastebėta, kad mažas vektorių dydis gali stipriai iškraipyti rezultatus. Pavyzdžiui howtogermany.com turėjome viso labo 4 imtis, kurios galėjo rodyti vykusią ataką. Tuo tarpu antro maršruto atakos vektoriaus dydis buvo šiek tiek didesnis ir tai padėjo gauti geresnius ir tolygesnius tyrimo rezultatus.

Nepaisant nesklandumų buvo pasiektas patenkinamas atpažinimo koeficientas (>50%). Tiek 5.5 lentelėje, tiek 5.6 lentelėje galima pastebėti, kad atakų imčių (pažymėta rusva spalva) atpažinimo koeficientai svyruoja tarp 60% ir 100% arba 0,6 ir 1.

IŠVADOS

1. Darbo metu buvo išnagrinėtas BGP protokolo veikimas bei DDoS ir kitų stambaus mąsto įvykių įtaka BGP tinklų veiklai. BGP analizavimo metu buvo išsiaiškinta, kad BGP tinklai yra pagrindinis jungiamasis tinklų protokolas. Nagrinėjat DDoS atakas pastebėta, kad atakos, kurios įtakoja BGP tinklų veiklą yra srautinės DDoS atakos. Jų įtaka tinklams yra pastebima, kadangi daugelis kitų atakų yra programų lygmenyje, kurio pasikeitimai neįtakoja žemesniuose OSI lygmenyse dirbančių tinklo įrenginių.

2. Analizuojant maršrutų ir atakų stebėjimo būdus, veikla buvo išskirstyta į kelis lygmenis. Analizė pradėta nuo tinklų stebėjimo metodų. Nors darbe buvo išsamiai gilinamasi į BGP maršrutų ir atributų stebėjimą, galų gale buvo pasirinkta tinklus stebėti nenaudojant BGP parametrų, o remiantis maršrutų informacija, renkama naudojant ICMP protokolą. Tai buvo mažiausiai resursų reikalaujantis sprendimas, leidžiantis stebėti didelio mąsto tinklus. Atakų aptikimui buvo nuspręsta naudoti keturis mašininio mokymosi algoritmus, reikšmėms klasifikuoti.

3. Suplanavus projekto vykdymo metodus, buvo pradėtas sistemos projektavimas. Sistema buvo sudaryta iš dviejų modulių: duomenų rinkimo ir duomenų analizės. Modulius sėkmingai pavyko sukurti naudojant „Python“ programavimo kalbą. Duomenys buvo renkami ir saugomi į duomenų bazę. Realizuojant sistemą buvo susidurta su keliomis problemomis. Pirma, maršrutų stebėjimas ICMP protokolu buvo vykdomas labai ilgai (100 adresų – 8 minutės). Tam išspręsti buvo pasitelktas daugelio procesų darbas. Antra, didelis duomenų kiekis, kurį reikėjo apdoroti kuriamos sistemos moduliams, taip pat lėtino sistemos darbą, dėl to buvo apsiribota iki 16 mln. įrašų DB.

4. Gautos statistikos buvo analizuojamos dviem etapais. Pradėta nuo rankinio statistikos tikrinimo. Tikrinant statistiką, iš 100 stebėtų adresų rasta tik apie 10 adresų su tam tikromis anomalijomis ir iš jų tik pora atitiko keliamus reikalavimus. Tokiai problemai išspręsti reikėjo plėsti stebimų adresų skaičių. Analizuojant pasirinktus rezultatus išsiaiškinta, kad sistema gali dirbti ir su nedideliu atakų pavyzdžių duomenų vektoriumi, tačiau daug geresni rezultatai pasiekiami turint didesnį manomų atakų kiekį. Imčių, pasižyminčių atakos požymiais, žymėjimo kaip atakos užtikrintumo koeficientas svyruoja nuo 60% iki 100%, priklausomai nuo tiriamų duomenų.

Literatūra

1. Cloudflare. What Is a Distributed Denial-of-Service (DDoS) Attack? [Tinkle] Cloudflare. https://www.cloudflare.com/learning/ddos/what-is-a-ddos-attack/.
2. —. Memcached DDoS Attack. [Tinkle] Cloudflare. https://www.cloudflare.com/learning/ddos/memcached-ddos-attack/.
3. Cisco. What Is a DDoS Attack? Distributed Denial of Service. [Tinkle] Cisco. https://www.cisco.com/c/en/us/products/security/what-is-a-ddos-attack.html.
4. Imperva. What is Slowloris? [Tinkle] Imperva. https://www.imperva.com/learn/application-security/slowloris/?utm_campaign=Incapsula-moved.
5. *DDoS Attack detection method and mitigation using pattern of the flow.* Ahmad Sanmorino, Setiadi Yazid. Bandung, Indonesia : s.n., 2013. ICoICT 2013.
6. SF007. Screenshot of Wireshark 1.0 on ubuntu. [Tinkle] Wikipedia. https://commons.wikimedia.org/wiki/File:Wireshark_screenshot.png.
7. Rekhter, Y. RFC 4271 - A Border Gateway Protocol 4 (BGP-4). *Tools.ietf.org.* [Tinkle] 2006 m. January. https://tools.ietf.org/html/rfc4271.
8. Osama, Wael. BGP Attributes List. [Tinkle] 2012 m. 05 15 d. http://www.networkers-online.com/blog/2012/05/bgp-attributes/.
9. *Identifying and Analyzing High Impact Routing Events with PathMiner.* Giovanni Comarela, Mark Crovella. Vancouver, BC, Canada : Internet Measurement Conference, 2014. 978-1-4503-3213-2.
10. traceroute(8). [Tinkle] die.net. https://linux.die.net/man/8/traceroute.
11. Braden, R. Requirements for Internet Hosts -- Communication Layers. [Tinkle] 1989 m. https://tools.ietf.org/html/rfc1122.
12. *Characterizing Large-Scale Routing Anomalies: A Case Study of the China Telecom Incident.* Rahul Hiran, Niklas Carlsson, Phillipa Gill. 2013 m., Lecture Notes in Computer Science, T. 7799.
13. J. Jung, B. Krishnamurthy, M. Rabinovich. *Flash crowds and denial of service attacks: Characterization and implications for CDNs and web sites.* 2002.
14. *Baseline profile stability for network anomaly detection.* Y. Kim, J. Y. Jo, K. K. Suh. 2008 m., International Journal of Network Security, T. 6.
15. Y. You, M. Zulkernine, and A. Haque. *Detecting flooding-based DDoS attacks.* 2007.
16. *MULTOPS: a data-structure for bandwidth attack detection.* Poletto, T. M. Gil and M. 2001. 10th Usenix Security Symposium.
17. Shaw, Reena. Top 10 Machine Learning Algorithms for Data Science Beginners. [Tinkle] 2018 m. 05 9 d. https://www.dataquest.io/blog/top-10-machine-learning-algorithms-for-beginners/.
18. Ray, Sunil. Essentials of Machine Learning Algorithms (with Python and R Codes). [Tinkle] 2017 m. 09. https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/.
19. Garg, Rohit. 7 Types of Classification Algorithms - Analytics India Magazine. [Tinkle] https://www.analyticsindiamag.com/7-types-classification-algorithms/ .
20. Graph of Logistics Curve. [Tinkle] Wikipedia. https://en.wikipedia.org/wiki/File:Logistic-curve.svg.
21. Decision Tree. [Tinkle] Saedsayad. https://www.saedsayad.com/decision_tree_overfitting.htm.

PRIEDAI

Priedas Nr. 1 - howtogermany.com tyrimo imtys

Imtis Nr. 1:

id	timestamp	ip	destination_ip	ASN	hop_count	last_count	ping	loss	anomaly
7306968	2019-04-28 16:04	154.16.202.1	howtogermany.com	AS61317	1	0	3	0	
7306969	2019-04-28 16:04	149.6.43.145	howtogermany.com	AS174	2	0	1	0	
7306970	2019-04-28 16:04	130.117.1.118	howtogermany.com	AS174	3	0	1	0	
7306971	2019-04-28 16:04	154.54.58.238	howtogermany.com	AS174	4	0	11	0	
7306972	2019-04-28 16:04	154.54.27.169	howtogermany.com	AS174	5	0	82	0	
7306973	2019-04-28 16:04	154.54.40.110	howtogermany.com	AS174	6	0	91	0	
7306974	2019-04-28 16:04	154.54.24.222	howtogermany.com	AS174	7	0	100	0	
7306975	2019-04-28 16:04	154.54.28.110	howtogermany.com	AS174	8	0	107	0	
7306976	2019-04-28 16:04	154.24.22.125	howtogermany.com	AS174	9	0	110	0	
7306977	2019-04-28 16:04	154.24.19.130	howtogermany.com	AS174	10	0	116	0	
7306978	2019-04-28 16:04	154.24.5.230	howtogermany.com	AS174	11	0	116	0	
7306979	2019-04-28 16:04	38.104.151.202	howtogermany.com	AS174	12	0	115	0	
7306980	2019-04-28 16:04	10.206.253.16	howtogermany.com	AS???	13	0	118	20	
7306981	2019-04-28 16:04	10.206.0.44	howtogermany.com	AS???	14	0	118	20	
7306982	2019-04-28 16:04	207.150.217.136	howtogermany.com	AS3064	15	1	760	50	

Imtis Nr. 2:

id	timestamp	ip	destination_ip	ASN	hop_count	last_count	ping	loss	anomaly
7307838	2019-04-28 16:05	154.16.202.1	howtogermany.com	AS61317	1	0	6	0	
7307839	2019-04-28 16:05	149.6.43.145	howtogermany.com	AS174	2	0	1	0	
7307840	2019-04-28 16:05	130.117.1.118	howtogermany.com	AS174	3	0	1	0	
7307841	2019-04-28 16:05	154.54.58.238	howtogermany.com	AS174	4	0	11	0	
7307842	2019-04-28 16:05	154.54.27.169	howtogermany.com	AS174	5	0	82	0	
7307843	2019-04-28 16:05	154.54.40.110	howtogermany.com	AS174	6	0	91	0	
7307844	2019-04-28 16:05	154.54.24.222	howtogermany.com	AS174	7	0	100	0	
7307845	2019-04-28 16:05	154.54.28.110	howtogermany.com	AS174	8	0	107	0	
7307846	2019-04-28 16:05	154.24.22.125	howtogermany.com	AS174	9	0	110	0	
7307847	2019-04-28 16:05	154.24.19.130	howtogermany.com	AS174	10	0	116	0	
7307848	2019-04-28 16:05	154.24.5.230	howtogermany.com	AS174	11	0	116	0	
7307849	2019-04-28 16:05	38.104.151.202	howtogermany.com	AS174	12	0	116	0	
7307850	2019-04-28 16:05	10.206.253.16	howtogermany.com	AS???	13	0	118	20	
7307851	2019-04-28 16:05	10.206.0.44	howtogermany.com	AS???	14	0	118	20	
7307852	2019-04-28 16:05	207.150.217.136	howtogermany.com	AS3064	15	1	268	0	

Imtis Nr. 3:

id	timestamp	ip	destination_ip	ASN	hop_count	last_count	ping	loss	anomaly
632818	2019-04-23 08:01	154.16.202.1	howtogermany.com	AS61317	1	0	1	0	
632819	2019-04-23 08:01	149.6.43.145	howtogermany.com	AS174	2	0	1	0	
632820	2019-04-23 08:01	130.117.1.118	howtogermany.com	AS174	3	0	1	0	
632821	2019-04-23 08:01	154.54.58.238	howtogermany.com	AS174	4	0	10	0	
632822	2019-04-23 08:01	154.54.27.169	howtogermany.com	AS174	5	0	82	0	
632823	2019-04-23 08:01	154.54.40.110	howtogermany.com	AS174	6	0	90	0	
632824	2019-04-23 08:01	154.54.24.222	howtogermany.com	AS174	7	0	100	0	
632825	2019-04-23 08:01	154.54.28.130	howtogermany.com	AS174	8	0	114	0	
632826	2019-04-23 08:01	154.54.81.86	howtogermany.com	AS174	9	0	122	0	
632827	2019-04-23 08:01	154.54.81.89	howtogermany.com	AS174	10	0	135	0	
632828	2019-04-23 08:01	154.24.5.230	howtogermany.com	AS174	11	0	135	0	
632829	2019-04-23 08:01	38.104.151.202	howtogermany.com	AS174	12	0	137	0	
632830	2019-04-23 08:01	10.206.253.16	howtogermany.com	AS???	13	0	137	0	
632831	2019-04-23 08:01	10.206.0.44	howtogermany.com	AS???	14	0	136	0	
632832	2019-04-23 08:01	207.150.217.136	howtogermany.com	AS3064	15	1	135	0	

Priedas Nr. 2 - howtogermany.com tyrimo imtys

Imtis Nr. 1:

id	timestamp	ip	destination_ip	ASN	hop_count	last_count	ping	loss	anomaly
6107507	2019-04-27 17:02	154.16.202.1	s-bahn-berlin.de	AS61317	1	0	0	0	
6107508	2019-04-27 17:02	62.115.12.14	s-bahn-berlin.de	AS1299	2	0	1	0	
6107509	2019-04-27 17:02	62.115.120.7	s-bahn-berlin.de	AS1299	3	0	19	0	
6107510	2019-04-27 17:02	62.115.139.7	s-bahn-berlin.de	AS1299	4	0	19	0	
6107511	2019-04-27 17:02	85.158.2.12	s-bahn-berlin.de	AS29014	5	0	528	40	
6107512	2019-04-27 17:02	85.158.2.11	s-bahn-berlin.de	AS29014	6	0	520	40	
6107513	2019-04-27 17:02	93.92.134.19	s-bahn-berlin.de	AS29014	7	0	512	40	
6107514	2019-04-27 17:02	93.92.135.251	s-bahn-berlin.de	AS29014	8	1	797	20	

Imtis Nr. 2:

id	timestamp	ip	destination_ip	ASN	hop_count	last_count	ping	loss	anomaly
1.2E+07	2019-05-02 01:47	154.16.202.1	s-bahn-berlin.de	AS61317	1	0	5	0	
1.2E+07	2019-05-02 01:47	62.115.12.14	s-bahn-berlin.de	AS1299	2	0	2	0	
1.2E+07	2019-05-02 01:47	62.115.120.7	s-bahn-berlin.de	AS1299	3	0	19	0	
1.2E+07	2019-05-02 01:47	62.115.139.7	s-bahn-berlin.de	AS1299	4	0	19	0	
1.2E+07	2019-05-02 01:47	85.158.2.12	s-bahn-berlin.de	AS29014	5	0	318	0	
1.2E+07	2019-05-02 01:47	85.158.2.11	s-bahn-berlin.de	AS29014	6	0	304	0	
1.2E+07	2019-05-02 01:47	93.92.134.19	s-bahn-berlin.de	AS29014	7	0	292	0	
1.2E+07	2019-05-02 01:47	93.92.135.251	s-bahn-berlin.de	AS29014	8	1	280	0	

Imtis Nr. 3:

id	timestamp	ip	destination_ip	ASN	hop_count	last_count	ping	loss	anomaly
8348103	2019-04-29 12:02	154.16.202.1	s-bahn-berlin.de	AS61317	1	0	1	0	
8348104	2019-04-29 12:02	62.115.12.14	s-bahn-berlin.de	AS1299	2	0	1	0	
8348105	2019-04-29 12:02	62.115.120.7	s-bahn-berlin.de	AS1299	3	0	19	0	
8348106	2019-04-29 12:02	62.115.139.7	s-bahn-berlin.de	AS1299	4	0	19	0	
8348107	2019-04-29 12:02	85.158.2.12	s-bahn-berlin.de	AS29014	5	0	137	20	
8348108	2019-04-29 12:02	85.158.2.11	s-bahn-berlin.de	AS29014	6	0	132	20	
8348109	2019-04-29 12:02	93.92.134.19	s-bahn-berlin.de	AS29014	7	0	128	20	
8348110	2019-04-29 12:02	93.92.135.251	s-bahn-berlin.de	AS29014	8	1	123	20	

Imtis Nr. 4:

id	timestamp	ip	destination_ip	ASN	hop_count	last_count	ping	loss	anomaly
6259986	2019-04-27 19:57	154.16.202.1	s-bahn-berlin.de	AS61317	1	0	5	0	
6259987	2019-04-27 19:57	62.115.12.14	s-bahn-berlin.de	AS1299	2	0	3	0	
6259988	2019-04-27 19:57	62.115.120.7	s-bahn-berlin.de	AS1299	3	0	19	0	
6259989	2019-04-27 19:57	62.115.139.7	s-bahn-berlin.de	AS1299	4	0	19	0	
6259990	2019-04-27 19:57	85.158.2.12	s-bahn-berlin.de	AS29014	5	0	47	60	
6259991	2019-04-27 19:57	85.158.2.11	s-bahn-berlin.de	AS29014	6	0	45	60	
6259992	2019-04-27 19:57	93.92.134.19	s-bahn-berlin.de	AS29014	7	0	42	60	
6259993	2019-04-27 19:57	93.92.135.251	s-bahn-berlin.de	AS29014	8	1	39	60	

Imtis Nr. 5:

id	timestamp	ip	destination_ip	ASN	hop_count	last_count	ping	loss	anomaly
16019713	2019-05-05 14:35	154.16.202.1	s-bahn-berlin.de	AS61317	1	0	1	0	
16019714	2019-05-05 14:35	62.115.12.14	s-bahn-berlin.de	AS1299	2	0	1	0	
16019715	2019-05-05 14:35	62.115.120.7	s-bahn-berlin.de	AS1299	3	0	19	0	
16019716	2019-05-05 14:35	62.115.139.7	s-bahn-berlin.de	AS1299	4	0	18	0	
16019717	2019-05-05 14:35	85.158.2.12	s-bahn-berlin.de	AS29014	5	0	16	0	
16019718	2019-05-05 14:35	85.158.2.11	s-bahn-berlin.de	AS29014	6	0	16	0	
16019719	2019-05-05 14:35	93.92.134.19	s-bahn-berlin.de	AS29014	7	0	17	0	
16019720	2019-05-05 14:35	93.92.135.251	s-bahn-berlin.de	AS29014	8	1	17	0	

Priedas Nr. 3 - programos kodas

Duomenų rinkimo modulis „data_gather.py“

```
#!/usr/bin/python3
import multiprocessing
import itertools
import subprocess
import json
from pprint import pprint
import mysql.connector

#ip_list_file='list_short.txt'
ip_list_file='/opt/list_avg.txt'

##-MTR Settings-----------
```

```
icmp_count=10
icmp_interval=0.2
##------------------------

##-MYSQL Settings-----------
sql_user='admin'
sql_pass='615sd-hAfbgr8fdf'
sql_host='127.0.0.1'
sql_db='net_data'
##------------------------

def worker(dst,icmp_count,icmp_interval):
    p = subprocess.Popen("mtr -zn --json -c " + str(icmp_count) + " -i " +
str(icmp_interval) + " " + dst, stdout=subprocess.PIPE, shell=True)
    (output, err) = p.communicate()
    if output:
        data = json.loads(output)
    else:
        data=None
    return data

def add_list(ip_list_file):
    with open(ip_list_file) as f:
        dest_list = [line.rstrip() for line in f]
    return dest_list

if __name__ == '__main__':

    ##-MYSQL Connect-----------
    cnx = mysql.connector.connect(user=sql_user, password=sql_pass,
host=sql_host, database=sql_db)

    cursor = cnx.cursor()

    add_output = ("INSERT INTO trace_data "
                  "(ip, destination_ip, ASN, hop_count, last_count, ping, loss) "
                  "VALUES ( %(data_host)s, %(data_d_host)s, %(data_asn)s,
%(data_count)s, %(last_count)s, %(data_avg)s, %(data_loss)s)")
    ##------------------------



    dest_list=add_list(ip_list_file)
    print(dest_list)

    with multiprocessing.Pool(processes=30) as pool:
        results = pool.starmap(worker, zip(dest_list,
itertools.repeat(icmp_count), itertools.repeat(icmp_interval)))


    for n in range(len(results)):
      if results[n]!=None:
         data=results[n]
         i=0
         last_count=0
         count=data['report']['hubs']
         for x in count:
           data_host=data['report']['hubs'][i]['host']
           data_asn=data['report']['hubs'][i]['ASN']
           data_count=data['report']['hubs'][i]['count']
           data_avg=data['report']['hubs'][i]['Avg']
           data_loss=data['report']['hubs'][i]['Loss%']
           data_d_host=data['report']['mtr']['dst']
           i=i+1
```

```
            if i>(len(count)-1):
              last_count = 1
            data_hop = {
              'data_host': data_host,
              'data_d_host': data_d_host,
              'data_asn': data_asn,
              'data_count': data_count,
              'last_count': last_count,
              'data_avg': data_avg,
              'data_loss': data_loss,
            }
            cursor.execute(add_output, data_hop)

    # Make sure data is committed to the database
    cnx.commit()
    cursor.close()
    cnx.close()
```

Duomenų analizės modulis „data_analyze.py“

```
#!/usr/bin/python2.7
import json
from pprint import pprint
import mysql.connector
import re
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.ensemble import RandomForestClassifier

#ip_list_file='list_short.txt'
ip_list_file='/opt/list_avg.txt'


def add_select(ip_list_file):
    with open(ip_list_file) as f:
        dest_list = [line.rstrip() for line in f]
    for n in range(len(dest_list)):
        print n+1, dest_list[n]
    o = raw_input("Please choose the destination address number: ")
    return dest_list[int(o)-1]

def check_val(myresult):
    data=[]
    date=0
    data_inner=[]
    data_middle=[]

    for x in myresult:
      striped_x=re.sub("\D", "", x[1])
      if not striped_x:
        x1_conv=0
      else:
        x1_conv=int(striped_x)
      data_inner.extend([x[2],x1_conv,x[4],x[5]])
      if x[3]==1:
        data_middle.append(data_inner)
        data_inner=[]
    return data_middle

cnx = mysql.connector.connect(user='admin', password='615sd-hAfbgr8fdf',
host='127.0.0.1', database='net_data')
cursor = cnx.cursor()
```

```
dst_ip=add_select(ip_list_file)
print dst_ip

cursor.execute('SELECT timestamp, ASN, hop_count, last_count, ping, loss, anomaly
FROM trace_data where destination_ip="' + dst_ip + '" and anomaly is not NULL')
myresult = cursor.fetchall()
data=[]
date=0
data_inner=[]
data_middle=[]
status=[]

for x in myresult:
  striped_x=re.sub("\D", "", x[1])
  if not striped_x:
    x1_conv=0
  else:
    x1_conv=int(striped_x)
  data_inner.extend([x[2],x1_conv,x[4],x[5]])
  if x[3]==1:
    data_middle.append(data_inner)
    status.extend([x[6]])
    data_inner=[]


X = data_middle
Y = status
Y = [str(i) for i in Y]


filter_str = raw_input("Please enter Mysql filter for value (timestamp): ")
cursor.execute('SELECT timestamp, ASN, hop_count, last_count, ping, loss, anomaly
FROM trace_data where destination_ip="' + dst_ip + '" and timestamp="' +
filter_str + '" ')
val_to_check = cursor.fetchall()
P = check_val(val_to_check)

#{Decision Tree Model}
clf = DecisionTreeClassifier()
clf = clf.fit(X,Y)
print "\n1) Using Decision Tree Prediction is " + str(clf.predict(P))

#{K Neighbors Classifier}
knn = KNeighborsClassifier()
knn.fit(X,Y)
print "2) Using K Neighbors Classifier Prediction is " + str(knn.predict(P))

#{using MLPClassifier}
mlpc = MLPClassifier()
mlpc.fit(X,Y)
print "3) Using MLPC Classifier Prediction is " + str(mlpc.predict(P))

#{using MLPClassifier}
rfor = RandomForestClassifier()
rfor.fit(X,Y)
print "4) Using RandomForestClassifier Prediction is " + str(rfor.predict(P))
+"\n"

final=int(knn.predict(P))*0.15 + int(mlpc.predict(P))*0.25 +
int(rfor.predict(P))*0.3 + int(clf.predict(P))*0.3
if final<0.5:
    print 'Final answer is it is: not attack. Value: ' + str(final)
else:
  print 'Final answer is it is: attack. Value: ' + str(final)
```