VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS
FUNDAMENTINIŲ MOKSLŲ FAKULTETAS
INFORMACINIŲ SISTEMŲ KATEDRA

Elonas Ševiakovas

ĮSILAUŽIMO Į KOMPIUTERĮ APTIKIMAS: VARTOTOJO ELGSENOS ANALIZĖ GILIOJO MOKYMO METODAIS
HOST-LEVEL INTRUSION DETECTION: ANALYSIS OF USER BEHAVIOUR BY DEEP LEARNING METHODS

Baigiamasis magistro darbas

Informacijos ir informacinių technologijų saugos studijų programa
valstybinis kodas 6211BX014
Informatikos inžinerijos studijų kryptis

Vadovas prof. dr. Dalius Mažeika

Vilnius, 2021

VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS
FUNDAMENTINIŲ MOKSLŲ FAKULTETAS
INFORMACINIŲ SISTEMŲ KATEDRA

TVIRTINU
Katedros vedėjas

_____
(Parašas)

_____
(Vardas, pavardė)

_____
(Data)

Elonas Ševiakovas

ĮSILAUŽIMO Į KOMPIUTERĮ APTIKIMAS: VARTOTOJO ELGSENOS ANALIZĖ GILIOJO MOKYMO METODAIS
HOST-LEVEL INTRUSION DETECTION: ANALYSIS OF USER BEHAVIOUR BY DEEP LEARNING METHODS

Baigiamasis magistro darbas

Informacijos ir informacinių technologijų saugos studijų programa
valstybinis kodas 6211BX014
Informatikos inžinerijos studijų kryptis

Vadovas prof. dr. Dalius Mažeika _____ _____
(Moksl. laipsnis, vardas, pavardė) (Parašas) (Data)

Lietuvių kalbos konsultantas dr. Vaida Buivydienė _____ _____
(Moksl. laipsnis, vardas, pavardė) (Parašas) (Data)

Vilnius, 2021

VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS
FUNDAMENTINIŲ MOKSLŲ FAKULTETAS
INFORMACINŲ SISTEMŲ KATEDRA

Informatikos inžinerija ……studijų kryptis

Informacijos ir informacinių technologijų sauga ……..studijų programa, valstybinis kodas ……

……..……..…specializacija

TVIRTINU
Katedros vedėjas

_____
(Parašas)
Prof .dr. Dalius Mažeika
(Vardas, pavardė)
_____
(Data)

BAIGIAMOJO MAGISTRO DARBO UŽDUOTIS

…….....Nr. .....
Vilnius

Studentui (ei) Elonas Ševiakovas
(Vardas, pavardė)

Baigiamojo darbo tema: Įsilaužimo į kompiuterį aptikimas: vartotojo elgsenos analizė giliojo mokymo metodais
Host-level intrusion detection: analysis of user behaviour by deep learning methods

patvirtinta 201…m. ……..…… d. dekano potvarkiu Nr. …….

Baigiamojo darbo užbaigimo terminas 201…m. ……..…… d.

BAIGIAMOJO DARBO UŽDUOTIS:

Baigiamojo darbo užduotys:

– Atlikti įsilaužimo aptikimo (IDS) sistemų ir juose naudojamų metodų apžvalgą.
– Apžvelgti mašininio mokymosi metodus, kurie galėtų būti naudojami vartotojo elgsenai tirti.
– Surinkti vartotojų veiksmų duomenis ir atlikti jų analizę, siekiant išsiaiškinti jų veiksmų teisėtumą.
– Pasiūlyti metodą, kaip galėtų būti patobulinamos IDS.
– Atlikti pasiūlyto metodo eksperimentą.

Baigiamojo darbo rengimo konsultantai:

……
(Moksl. laipsnis/pedag. vardas, vardas, pavardė)

Vadovas ..... Prof .dr. Dalius Mažeika
(Parašas) (Moksl. laipsnis/pedag.vardas, vardas, pavardė)

Užduotį gavau

……..
(Parašas)
……Elonas Ševiakovas……..
(Vardas, pavardė)
……2019 10 22 ….....
(Data)

Vilniaus Gedimino technikos universitetas
Fundamentinių mokslų fakultetas
Informacinių sistemų katedra

ISBN ISSN
Egz. sk. .....
Data .....-.....-.....

Antrosios pakopos studijų **Informacijos ir informacinių technologijų saugos** programos magistro baigiamasis darbas

Pavadinimas **Įsilaužimo į kompiuterį aptikimas: vartotojo elgsenos analizė giliojo mokymo metodais**

Autorius **Elonas Ševiakovas**

Vadovas **Dalius Mažeika**

Kalba: lietuvių

Anotacija

Baigiamajame magistro darbe išnagrinėtos įsilaužimo aptikimo sistemos, jų veikimo būdai bei naujausi įsilaužimo aptikimo metodai. Aprašytas mašininis mokymasis bei įvairūs jo tipai. Taip pat aptarti duomenų surinkimo būdai bei bendras jų surinkimo procesas. Trumpai apžvelgtos komercinės ir nekomercinės įsilaužimo aptikimo sistemos. Pasiūlytas metodas įsilaužimams aptikti, kuris paremtas vienu iš giluminių neuroninių tinklų. Pasirinktam metodui parašytas duomenų surinkimo įrankis, duomenys surinkti nesimuliuotoje aplinkoje. Galiausiai suprogramuotas pasiūlytas įsilaužimo aptikimo metodas, įvertintas jo efektyvumas lyginant jį su kitais giluminiais neuroniniais tinklais ir duomenimis. Darbą sudaro 4 dalys: įvadas, analitinė dalis, pasiūlyto metodo kartu su duomenų surinkimu ir vertinimu dalis, išvados. Darbo apimtis – 61 p. teksto be priedų, 40 pav., 5 lent., 30 bibliografinių šaltinių.

Prasminiai žodžiai: LSTM, Autoenkoderis, Giluminis neuroninis tinklas, IDS, Įsilaužimų aptikimas, Anomalijų aptikimas

Vilnius Gediminas Technical University
Faculty of Fundamental Sciences
Department of Information Systems

ISBN ISSN
Copies No. .....
Date .....-.....-.....

Master Degree Studies **Information and Information Technologies Security** study programme Master Graduation Thesis

Title	**Host-Level Intrusion Detection: Analysis of User Behavior by Deep Learning Methods**
Author	**Elonas Ševiakovas**
Academic supervisor	**Dalius Mažeika**

Thesis language: Lithuanian

Annotation

The final master thesis examines intrusion detection systems and newest intrusion detection methods. It also contains explanation of machine learning and various types of learning algorithms. In addition, data collection methods are reviewed along with general steps for data collection. Furthermore there is a short review of both commercial and non-commercial intrusion detection systems. A method for intrusion detection is proposed which is based on a type of deep neural network. A data collection tool was programmed specifically for the chosen method and data was collected in an unsimulated environment. Last of all a prototype of selected method was programmed and it's effectiveness was evaluated by comparing it to other deep learning methods and data. Structure: introduction, analytical part, suggested intrusion detection method with data collection and evaluation of results, conclusion. Thesis consists of 61 pages of text, 40 pictures, 5 tables, 30 sources.

Keywords: LSTM, Autoencoder, Deep Neural Network, IDS, Intrusion Detection, Anomaly Detection

Turinys

Paveikslų sąrašas 3
Lentelių sąrašas 4
1. Įvadas 6
2. Įsilaužimo aptikimo sistemų ir jose naudojamų metodų analizė 8
2.1. Įsilaužimo aptikimo sistemos 8
2.1.1. Kompiuterio IDS 9
2.1.2. Tinkle veikiančios IDS 10
2.2. IDS įsilaužimo aptikimo metodų tipai 11
2.2.1. Elgesiu paremtos IDS 11
2.2.2. Žiniomis paremtos IDS 12
2.3. Duomenų surinkimo ir apdorojimo būdai 13
2.4. Įsilaužimo aptikimo metodai 14
2.4.1. Mašininis mokymasis 15
2.4.2. Bendri įsilaužimo aptikimo metodų bruožai 22
2.4.3. Scenarijumi paremtas anomalijų aptikimas 23
2.4.4. Pasitikėjimu paremtas anomalijų aptikimas 24
2.4.5. Negatyvia atranka paremtas anomalijų aptikimas 24
2.4.6. Paprastas neuroninis tinklas anomalijoms aptikti 25
2.4.7. Genetinis neuroninis tinklas anomalijoms aptikti 26
2.4.8. Periferine įranga paremti anomalijų aptikimo metodai 27
2.5. Populiariausių HIDS apžvalga 28
2.6. Skyriaus apibendrinimas 30
3. Siūlomas anomalijų aptikimo metodas 31
3.1. Duomenys 31
3.1.1. Duomenų surinkimas 32
3.1.2. Duomenų analizė 35
3.1.3. Duomenų paruošimas 40
3.2. Anomalijos aptikimo metodas 42
3.2.1. LSTM autoenkoderis 43
3.2.2. Metodo veikimo principas 44
3.2.3. Naudojamos technologijos 46
3.3. Rezultatų analizė 46
3.3.1. Metodo įvertinimas su surinktais duomenimis 46
3.3.2. Metodo įvertinimas su surinktais duomenimis 52
3.4. Skyriaus apibendrinimas 52
4. Išvados 54
5. Literatūros sąrašas 55

Paveikslų sąrašas

1 pav. NIDS ir HIDS diagramos (pagal Auso, 2015) 11
2 pav. Hibridinio IDS veikimo pavyzdys 13
3 pav. Atsitiktinio miško struktūros schema 16
4 pav. Duomenų su trimis dimensijomis klasifikacija 17
5 pav. Tiesinio neuroninio tinklo diagrama (pagal Templeton, 2015) 18
6 pav. Grįžtamojo ryšio ir tiesinio neuroninių tinklų skirtumai (pagal Roos, 2019) 19
7 pav. Autoenkoderio pavyzdys 20
8 pav. VAE neuroninio tinklo diagrama 21
9 pav. Sugeneruota negatyvios paieškos erdvė 25
10 pav. RBF neuroninis tinklas 26
11 pav. Genetinio neuroninio tinklo mokymo schema anomalijoms aptikti 27
12 pav. Pelytės būsenų paveiksliukai 28
13 pav. Prisijungimų duomenų failo pavyzdys 33
14 pav. Procesų duomenų failo pavyzdys 33
15 pav. Siunčiamų duomenų failo pavyzdys 33
16 pav. Sisteminių įvykių failo pavyzdys 33
17 pav. Duomenų surinkimo procesas 34
18 pav. Apmokymo procesų duomenys 36
19 pav. Puolimo procesų duomenys 36
20 pav. Apmokymo tinklo duomenys 37
21 pav. Puolimo tinklo duomenys 37
22 pav. Apmokymo tinklo duomenys 39
23 pav. Puolimo tinklo duomenys 39
24 pav. "feature hashing" pseudokodas 40
25 pav. Duomenų apdorojimo diagrama 42
26 pav. Grįžtamojo ryšio sluoksnio veikimo principas (pagal Olah, 2015) 43
27 pav. LSTM sluoksnio veikimo principas (pagal Olah, 2015) 43
28 pav. Anomalijų aptikimo veikimo diagrama 45
29 pav. LSTM aptikimo dažnio ir klaidingų pranešimų dažnio santykis kai grupuojama po 9 įrašus 49
30 pav. LSTM apmokymo įvertinimas kai grupuojama po 9 įrašus 49
31 pav. LSTM aptikimo dažnio ir klaidingų pranešimų dažnio santykis kai grupuojama po 3 įrašus 49
32 pav. LSTM apmokymo įvertinimas kai grupuojama po 3 įrašus 49
33 pav. LSTM aptikimo dažnio ir klaidingų pranešimų dažnio santykis kai įrašai negrupuojami 50
34 pav. LSTM apmokymo įvertinimas kai įrašai negrupuojami 50
35 pav. VAE aptikimo dažnio ir klaidingų pranešimų dažnio santykis 51
36 pav. VAE apmokymo įvertinimas 51
37 pav. Paprasto autoenkoderio aptikimo dažnio ir klaidingų pranešimų dažnio santykis 51
38 pav. Paprasto autoenkoderio apmokymo įvertinimas 51
39 pav. LSTM aptikimo dažnio ir klaidingų pranešimų dažnio santykis su „ADFA-LD“ 52
40 pav. LSTM apmokymo įvertinimas su „ADFA-LD“ 52

Lentelių sąrašas

1 lentelė. Populiariausių HIDS palyginimas.....29
2 lentelė. 5 rečiausi ir dažniausi procesai 36
3 lentelė. 5 rečiausi ir dažniausi tinklu siunčiami duomenys 37
4 lentelė. 5 rečiausi ir dažniausi sisteminiai įvykiai 39
5 lentelė. Maišos matricos sudarymas 47

Santrumpos

IDS (angl. *intrusion detection system*) – įsilaužimo aptikimo sistema.

NIDS (angl. *network-based intrusion detection system*) – įsilaužimo aptikimo sistema, kuri stebi tinklą.

HIDS (angl. *host-based intrusion detection system*) – įsilaužimo aptikimo sistema, kuri stebi kompiuterį.

IPS (angl. *intrusion prevention system*) – įsilaužimo prevencijos sistema.

VPN (angl. *virtual private network*) – virtualus privatus tinklas.

OS – operacinė sistema.

API (angl. *application programming interface*) – programuojama aplikacijos sąsaja.

RBF (angl. *radial basis function*) – radialinė bazinė funkcija.

LSTM (angl. *long short-term memory*) – ilgalaikė, trumpalaikė atmintis.

VAE – variacinis autoenkoderis.

1. Įvadas

Nuolat tobulėjant informacinių sistemų apsaugos priemonėms įsilaužėliams tampa vis sunkiau atrasti sistemų spragų. Tačiau tai reiškia, kad taip pat yra sunkiau apsisaugoti nuo mažai žinomų atakų. Dėl to be prevencinių priemonių reikalingos sistemos, kurios sugebėtų aptikti jau įsilaužusius programišius. Įsilaužėliai, turėdami prieigą prie normalios sistemos paskyros, gali toliau bandyti ieškoti spragų ar ieškomos informacijos jau iš sistemos vidaus, nes organizacijų vidinis tinklas dažniausiai būna mažiau apsaugotas ir stebimas, nei prieiga iš išorės. Paprasčiausia būtų tikrinti, ar paskyros nenaudoja kokių nors pavojingų komandų, tačiau įsilaužėliai žino apie įvairius aptikimo būdus, todėl jie gali užsimaskuoti ir rinkti informaciją bei vykdyti kenksmingą veiklą per ilgą laiką, kol originalus paskyros savininkas atlieka savo darbus. Žinoma, pasinaudoti savo paskyros privilegijuotomis teisėmis neteisėtiems veiksmams atlikti gali ne tik įsilaužėliai, bet ir pačios organizacijos darbuotojai. Dėl to reikalinga įsilaužimų aptikimo sistema, kuri gebėtų išmokti normalų elgesį paskyrose, gebėtų prisitaikyti prie natūralių darbuotojų elgesio pasikeitimų ir aptiktų anomalią veiklą, kuri galėtų reikšti, kad prie paskyros prieigą gavo įsilaužėlis.

Tyrimo objektas – įsilaužimų aptikimo metodai.

Darbo tikslas – pasiūlyti vartotojų veiksmų kompiuteryje analize paremtą įsilaužimo į kompiuterius aptikimo būdą.

Darbo uždaviniai:

1. Atlikti įsilaužimo aptikimo (IDS) sistemų ir juose naudojamų metodų apžvalgą.
2. Apžvelgti mašininio mokymosi metodus, kurie galėtų būti naudojami vartotojo elgsenai tirti.
3. Surinkti vartotojų veiksmų duomenis ir atlikti jų analizę, siekiant išsiaiškinti jų veiksmų teisėtumą.
4. Pasiūlyti metodą, kaip galėtų būti patobulinamos IDS.
5. Atlikti pasiūlyto metodo eksperimentą.
6. Pateikti atliktos analizės ir gautų rezultatų išvadas.

Temos naujumas – apskritai įvairių mašininių metodų taikymas skirtingose srityse yra vis dar aktyvus, ypač medicinoje, atpažįstant veidus, suvokiant natūralią kalbą ir pan. Įsilaužimo aptikimas, ypač stebint įtartiną vartotojo elgesį, nėra taip aktyviai tyrinėjamas. Dėl to liko dar nemažai įvairių mašininio mokymosi metodų, kurie pasiteisino kitose srityse, ir kuriuos galima būtų išbandyti įtartinam elgesiui aptikti.

Temos aktualumas – šiuo metu apie įsilaužimo ir įtartinų darbuotojų aptikimo metodus yra rašoma nemažai įvairių straipsnių (Cooper, 2019; Grimmer, Röhling, Kreusel, & Ganz, 2019; Zhang, Chen, Yang, Zhao, & Yan, 2019). Didžiausias iššūkis yra sukurti tokius metodus, kurie galėtų išmokti normalų darbuotojų elgesį, prisitaikyti prie nuolatinės elgesio kaitos ir tuo pačiu sugebėtų aptikti sudėtingus ir per ilgą laiko tarpą trunkančius įsilaužimus, ir duomenų nutekinimus.

Nors daug metodų demonstruoja neblogus rezultatus, panašu, kad jie dar nelabai naudojami tikrose IDS. Taip pat dauguma naujų metodų yra išbandomi su viešai prieinamais sintetiniais duomenimis iš universitetų virtualiose „Linux“ sistemose, o ne realiose organizacijose.

Tyrimo metodika – analitinei daliai atlikti naudojama su tema susijusių mokslinių straipsnių literatūros analizė.

Mokslinė darbo vertė – pasiūlytas ir įgyvendintas naujas metodas įsilaužimui aptikti analizuojant įtartiną naudotojo elgesį.

Darbo rezultatai:

1. Atlikta literatūros šaltinių, susijusių su IDS veikimo metodais, analizė.
2. Surinkti duomenys eksperimentui atlikti.
3. Pasiūlytas anomalijų aptikimo metodas.
4. Pateikti anomalijų aptikimo modulio prototipo rezultatai.

Darbo struktūra:

1. Pirmame skyriuje pateikiamas įvadas, tyrimo objektas, darbo tikslas bei uždaviniai, temos naujumas ir aktualumas.
2. Antrame skyriuje apibrėžiama įsilaužimo aptikimo sistemos sąvoka, jos tipai ir veikimo metodai. Taip pat apžvelgiami nauji analitiniai įsilaužimo bei keisto vartotojo elgesio aptikimo metodai. Galiausiai palyginamos modernios IDS.
3. Trečiame skyriuje aprašomas naudojamas duomenų rinkinys bei jo surinkimo būdas, siūlomas įtartino elgesio aptikimo metodas, palyginamas pasirinkto metodo efektyvumas su skirtingais duomenimis ir algoritmais bei pateikiami eksperimentų rezultatai.

2. Įsilaužimo aptikimo sistemų ir jose naudojamų metodų analizė

Šiame skyriuje apibrėžiama įsilaužimų aptikimo sistemos sąvoka, aprašomi jų veikimo ir naudojamų metodų tipai. Trumpai paaiškinama, kas yra mašininis mokymas. Aptariama duomenų, reikalingų įsilaužimo aptikimo sistemoms, gavyba. Apžvelgiami naujesni įsilaužimams ir keistam vartotojų elgesių aptikti naudojami metodai. Galiausiai išvardinamos ir palyginamos populiariausios HIDS.

2.1. Įsilaužimo aptikimo sistemos

Įsilaužimo aptikimo sistema (angl. *Intrusion Detection System* – toliau IDS) – tai prietaisas arba programinė įranga, kuri stebi tinklą arba pačias sistemas siekiant aptikti kenkėjišką veiklą ar kitus pažeidimus. Dažniausiai bet kokia įtartina veikla yra pranešama arba administratoriui, arba surenkama ir perduodama į saugumo informacijos ir įvykių valdymo sistemą, kuri iš įvairių šaltinių surenka informaciją apie skirtingus įvykius ir panaudojant filtravimo metodus bando atskirti tikrus įsilaužimus nuo klaidingų pranešimų (Maurizio Martellini, 2017).

IDS pagrindinės funkcijos: incidentų aptikimas įvairiose sistemose arba tinkle, incidentų ir kitos informacijos žurnalizavimas bei mokymasis iš įprasto sistemos darbo. Šios sistemos palengvina saugumo specialistų darbą, nes nebereikia rankiniu būdu stebėti žurnalo įrašų, viską automatiškai stebi ir apie incidentus įspėja IDS (8 Best HIDS Tools—Host-Based Intrusion Detection Systems, 2019). Kai kurios IDS gali atlikti ir prevencines funkcijas pagal tam tikras taisykles, tačiau tam yra skirtos atskiros sistemos.

Be įsilaužimų IDS taip pat galima naudoti kaip apsaugą nuo vidinių grėsmių, tarkim organizacijos darbuotojų, kurie tyčia ar netyčia bando nutekinti informaciją arba kitaip pakenkti organizacijai.

Pagal veikimo vietą IDS skirstomos į dvi pagrindines kategorijas (Yeung & Ding, 2003):

- Stebinčios kompiuterį (angl. *Host IDS*, toliau – HIDS).
- Stebinčios tinklą (angl. *Network IDS*, toliau – NIDS).

2.1.1. Kompiuterio IDS

HIDS gali veikti tiek serveriuose, tiek asmeniniuose kompiuteriuose. Jų tikslas – aptikti bandymus įsilaužti į individualius kompiuterius arba įtartinus veiksmus su kompiuteriu.

Pagrindinis HIDS veikimo principas – žurnalo (angl. *log*) analizė. Žurnaluose saugoma tokia informacija:

- sėkmingi ar nesėkmingi prisijungimai prie sistemos;
- programų veikimo pranešimai;
- naudotojo iškviečiamos funkcijos;
- OS informaciniai arba klaidų pranešimai ir t. t.

Tačiau be žurnalų gali būti pasitelkta kita informacija, kuri labiau priklauso nuo kiekvienos organizacijos, kompiuterio ar operacinės sistemos. Pavyzdžiui, gali būti stebimi naudojimosi klaviatūra ir pelyte ypatumai: rašymo greitis, pelės judinimo greitis, pelytės judinimo diapazonas ir t. t. Taip pat gali būti stebimi įvairūs OS atminties regionai: ar nebuvo pakeisti registrai, ar neįrašytos naujos programos, ar nebuvo bandoma kviesti pavojingų funkcijų.

HIDS dažnai susideda iš dviejų dalių: viena pagrindinė administracinė programa, kuri gauna visus pranešimus, ir po duomenų analizės programą kiekviename stebimame kompiuteryje, kurios siunčia įtartinus įvykius pagrindinei programai (Fuchsberger, 2005).

Įsilaužimo atveju HIDS yra pažeidžiamos, nes jos veikia tuose pačiuose kompiuteriuose, kuriuose įvyko incidentas. Dėl to HIDS turi būti papildomai apsaugotos, kad įsilaužėliai neblokuotų žinučių apie įvykusius incidentus arba iš vis jos neatjungtų. Šios sistemos gali būti apsaugomos užšifruojant informaciją apie incidentus, informaciją įrašant tik į skaitymui skirtas laikmenas, t. y., kas buvo įrašyta, nebegali būti ištrinta, taip pat turėtų būti naudojamas VPN tam, kad informacija nebūtų siunčiama kompromituotu tinklu. Galiausiai gali būti naudojami papildomi prie kompiuterio prijungiami moduliai, kurie turi įsilaužimo aptikimo sistemas ir yra apsaugoti nuo modifikacijos iš išorės.

2.1.2. Tinkle veikiančios IDS

Tinkle veikiančios įsilaužimo aptikimo sistemos analizuoja patį tinklą ir jame siunčiamus paketus. NIDS dažnai yra atskiri prietaisai, kurie yra prijungti prie organizacijos tinklo. Tai gali būti arba atskiri būtent paketų analizei skirti prietaisai, kurie dažnai būna brangoki, bet geriau susitvarko su dideliais duomenų srautais, arba įprasti kompiuteriai su pigesniu paketų aptikimo moduliu. Šie prietaisai pasislepia tam, kad nebūtų aptinkami iš išorės, ir tokiu būdu apsisaugo nuo įsilaužėlių.

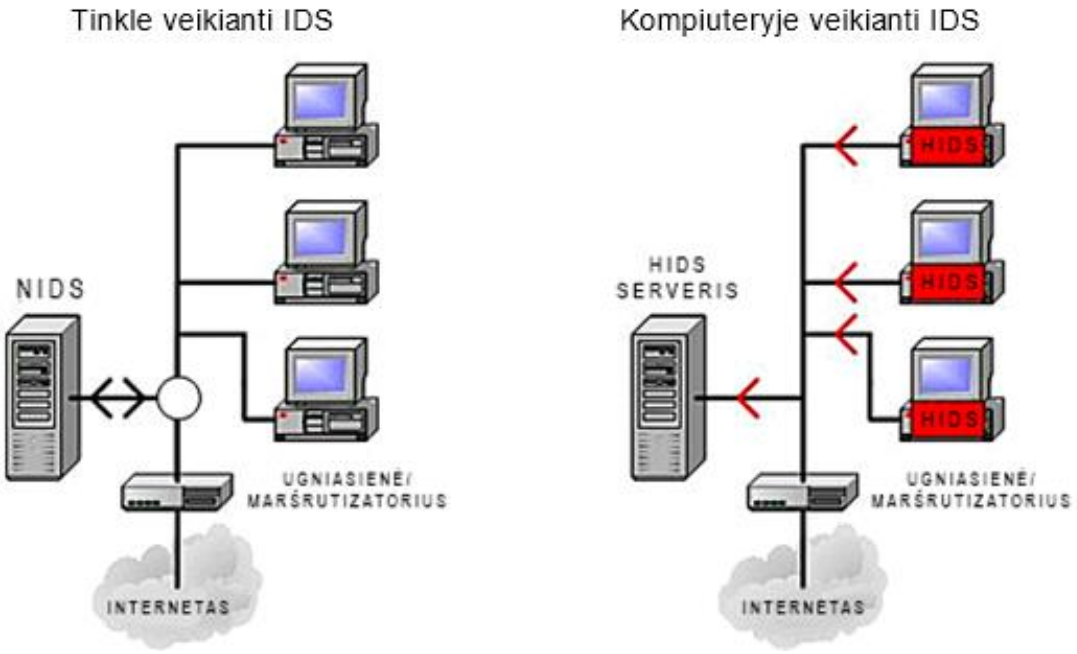
NIDS tinkle keliaujančius paketus kopijuoja į savo atmintį. Paketų antraštės, turinys ar šiaip bendras formatas yra analizuojamas bandant aptikti anomalijas arba yra lyginamas su duomenų bazėje esančiais parašais. Aptikus ataką šio tipo IDS išsiunčia žinią atitinkamiems asmenims ir IPS, kuri gali automatiškai imtis prevencinių veiksmų, pavyzdžiui, perkonfigūruoti ugniasienę bei užblokuoti puolamus prievadus (Fuchsberger, 2005).

NIDS sugeba aptikti tokias atakas:

- prievadų skenavimą;
- bandymus prisijungti į jautrius IP adresus;
- srauto perpildymo atakas ir t. t.

Dažniausiai tinklo srautai yra labai dideli ir NIDS gali nesugebėti su viskuo susidoroti. Todėl sistemą galima sukonfigūruoti taip, kad ji analizuotų paketus su tam tikromis savybėmis, tarkim visus paketus, kurie bendrauja HTTP protokolu ir paketai, kurie keliauja į arba iš nepatikimų serverių.

Palyginus su HIDS, NIDS veikia realiu laiku ir gali iš karto reaguoti į iškilusias grėsmes, o HIDS bando aptikti sėkmingų ar nesėkmingų įsilaužimų požymius daugiausia iš žurnalų įrašų. Idealiu atveju organizacija turėtų naudoti abiejų tipų IDS. Abiejų IDS veikimo principai pavaizduoti **1 pav.**



1 pav. NIDS ir HIDS diagramos (pagal Auso, 2015)

2.2. IDS įsilaužimo aptikimo metodų tipai

IDS aptikimo metodai būna dviejų tipų (Fuchsberger, 2005):

- Elgesiu paremti.
- Žiniomis paremti.

Dar kitaip elgesiu paremti metodai vadinami anomalijų aptikimo metodais, o žiniomis paremti – piktnaudžiavimo aptikimo (angl. *misuse detection*) metodais (Yeung & Ding, 2003). Tiek HIDS, tiek NIDS gali naudoti abiejų tipų metodus.

2.2.1. Elgesiu paremtos IDS

Statistinių anomalijų aptikimas arba elgesiu paremtas aptikimas veikia tikrinant sistemos ar paskyros darbą lyginant su įprastu. Iš anksto yra nustatoma, kas yra normalu šiai sistemai ir paskyrai. Jei IDS aptinka, kad vyksta kažkas statistiškai neįprasto, sistema įspėja atitinkamus asmenis.

Tačiau yra sunku numatyti, kas tiksliai yra normalus sistemos ar paskyros elgesys. Dažniausiai tam yra naudojamas mašininis mokymasis. Taip pat turi būti numatytas mechanizmas, kuris sugeba IDS pritaikyti prie pasikeitusios normos. O norma gali pasikeisti, jei vartotojui buvo

įrašyta nauja programinė įranga, jeigu buvo atnaujintas tinklas, arba tiesiog pakito vartotojo įpročiai, gal jis atrado naujesnį ir geresnį būdą savo užduočiai atlikti. Bet dinamiškas prisitaikymas irgi turi rizikos, nes įsilaužėlis gali tuo pasinaudodamas klaidingai išmokyti IDS, kas yra normalu.

Elgesiu paremtos IDS yra mažiau patikimos, nes naudoja mašininio mokymosi metodus, kurie gali būti prastai arba nepakankamai išmokyti, bei gali dažnai klaidingai aptikti įsilaužymus, taip sumažindami pasitikėjimą visa IDS. Iš kitos pusės elgesiu paremtos IDS gali aptikti dar nematytus įsilaužimo incidentus.

2.2.2. Žiniomis paremtos IDS

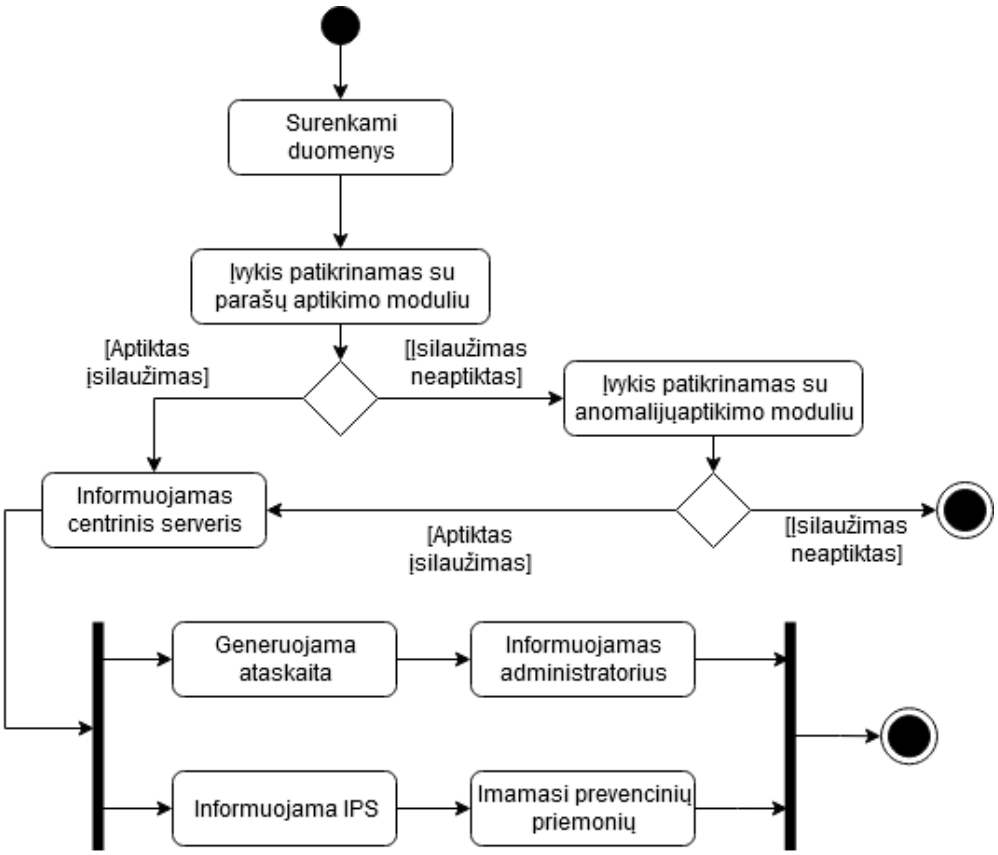
Žiniomis arba parašais paremtos IDS ieško tam tikrų įsilaužimo požymių, kurie yra iš anksto žinomi. Šie požymiai arba suprogramuojami tiesiai į IDS arba gaunami iš duomenų bazių, kurios yra nuolat atnaujinamos sužinojus apie naujas atakas. Veikimo principu šios IDS yra panašios į antivirusines programas (8 Best HIDS Tools—Host-Based Intrusion Detection Systems, 2019).

Pavyzdiniai požymiai gali būtis:

- didelis nesėkmingų prisijungimų prie jautrios informacijos kiekis;
- tam tikra bitų struktūra IP paketuose, kuri gali reikšti perpildymo atakų bandymą;
- pavojingos ir įtartinos komandos;
- bandymas peržiūrėti jautrią informaciją ir t. t.

Žiniomis paremtų IDS duomenų bazės yra nuolat atnaujinamos, bet parašai yra ganėtinai konkretūs ir gali neaptikti net žinomų atakų, jei jos truputi modifikuotos, vykdomos neįprasta tvarka arba slepiamos tarp kitų komandų. Taip pat šio tipo IDS nesugeba apsisaugoti nuo visiškai naujų, dar nematytų atakų. Negana to įsilaužėliai gali atlikti kenkėjiškus veiksmus labai lėtai, tarkim kas kelias dienas, jei šie parašai yra pritaikyti tam tikroms veiksmų sekoms, jie niekaip neaptiks šių veiksmų, nes jie yra per daug panašūs į įprastą darbuotojų elgesį.

Nepaisant žiniomis paremtų IDS problemų ir palyginus jas su elgesiu paremtomis IDS, šios sistemos turi mažiau klaidingų incidentų, greičiau veikia ir yra paprasčiau įgyvendinamos. Dėl to prieš 15 metų dažniausiai pasitaikančios komercinės IDS būdavo žiniomis paremtos (Fuchsberger, 2005), tačiau dabar jos naudojamos kartu su elgesiu paremtais metodais. Hibridinio IDS pavyzdinis veikimo principas parodytas **2 pav**.



2 pav. Hibridinio IDS veikimo pavyzdys

Norint, kad IDS teisingai veiktų, reikia tinkamos strategijos duomenims surinkti. Dėl to verta išsiaiškinti, kaip duomenys yra surenkami bei kokiu būdu jie paruošiami analizei. Toliau aptariami duomenų surinkimo ir susisteminimo būdai.

2.3. Duomenų surinkimo ir apdorojimo būdai

Pirmiausia duomenų surinkimo metodas priklauso nuo to, kur IDS veikia. NIDS duomenis renka tiesiai iš tinklo, t. y. sistema stebi ir analizuoja kiekvieną tinklu keliaujantį paketą. HIDS renka informaciją tiktai vieno kompiuterio kontekste, net jei surinkti duomenys siunčiami į centralizuotą serverį. Kompiuteryje duomenys gali būti renkami tiesiogiai arba netiesiogiai (Spafford & Zamboni, 2000).

Netiesiogiai renkami duomenys dažniausiai yra žurnalų įrašai. Iš vienos pusės juos paprasta gauti, nes operacinės sistemos be papildomos konfigūracijos juos kaupia automatiškai. Blogai yra tai, kad šie duomenys dažnai yra pertekliniai, ne visi, kurių reikia įsilaužimams aptikti ir dėl didelio kiekio juos sunku apdoroti.

Kitas, tiesioginis duomenų rinkimo būdas yra žymiai sunkiau įgyvendinamas, ypač, jei reikia duomenų iš kitų programų. Sisteminiai duomenys: procesoriaus apkrova, tinklo srautai iš ir į kompiuterį, atmintis bei kiti OS resursai yra gan lengvai prieinami. Tačiau aplikacijoms reikia rašyti atskirus modulius, kurie reikiamus duomenis rinktų iš išorės stebint programos elgesį. Kai kuriais atvejais galima įsiterpti tarp sisteminių bibliotekų ir programos bei taip stebėti, su kokiais parametrais yra kviečiamos bibliotekos funkcijos. Taip pat būna, kad pačios programos analizei reikiamą informaciją renka ir teikia per API arba atskirai saugo sisteminiame žurnale. Šis atvejis yra pats idealiausias, nes patiems nereikia kurti duomenų rinkimo programų, tačiau ganėtinai retas.

Jei įmanoma, siūloma kuo daugiau duomenų rinkti tiesioginiu būdu. Tada į incidentus galima reaguoti iš karto, užpuolikas negali pakeisti žurnalo įrašų bei taip sutrikdyti IDS darbą, ir sistemai reikės skirti mažiau kompiuterio resursų žurnalų apdorojimui. Žurnalai patogūs tais atvejais, jei informacijos kitaip išgauti neįmanoma arba reikalauja per daug investicijų.

Surinkus duomenis kitas žingsnis yra jų paruošimas analizės metodams. Dauguma įsilaužimo analizės algoritmų veikia su skaičių matricomis ar vektoriais. Duomenis renkant tiesioginiu būdu, jie greičiausiai iš karto bus gaunami skaitiniu pavidalu, o analizuojant žurnalus gali prireikti teksto analizės metodų, tačiau kadangi žurnalų įrašai yra generuojami aplikacijų ir turi aiškią, nekintančią struktūrą, turėtų užtekti paprastų taisyklių, kurios interpretuoja tekstinę informaciją ir paverčia ją į skaitinę. Gauta skaitinė informacija apjungiama į vektorių: jei reikšmė yra kiekybinė, jai priskiriama viena dimensija, jei reikšmė kokybinė, sukuriamos dimensijos kiekvienai galimai reikšmei. Galiausiai duomenų vektorius turėtų būti normalizuojamas tam, kad dimensijų svarba būtų apytiksliai vienoda. Vėliau dimensijų svarbą galima koreguoti padauginus arba padalinus reikšmes iš svarbos koeficiento.

Toliau apžvelgiami įvairūs anomalijų aptikimo metodai.

2.4. Įsilaužimo aptikimo metodai

Didžiausia įsilaužimo aptikimo metodų dalis remiasi anomalijų aptikimu. Šiam metodui reikalingas pradinis normalus sistemos ar vartotojo elgesio modelis, pagal kurį vėliau bus lyginami įvairūs veiksmai. Šį pradinį modelį įmanoma apsirašyti rankiniu būdu, tačiau tai yra nepraktiška, nes reikia iš anksto nuspėti visus galimus normalius naudotojų veiksmus, tai reikia atlikti kiekvienam vartotojui ir galiausiai šį modelį reikia nuolatos atnaujinti. Dėl to anomalijų analizei yra naudojami pažangūs algoritmai, kurie sugeba mokytis iš esamos sistemos būsenos, ir po to

pasitelkus surinktas žinias gali nustatyti, kurie veiksmai yra normalūs, o kurie įtartini bei galimai kenkėjiški.

Prieš analizuojant įsilaužimo aptikimo metodus, pirmiausia reikia išsiaiškinti, kas yra mašininis mokymasis, nes dauguma anomalijomis paremtų algoritmų naudoja būtent jį sprendimams priimti.

2.4.1. Mašininis mokymasis

Mašininis mokymasis – algoritmai ir statistiniai modeliai, kurie sugeba pasimokę iš tam tikrų duomenų rinkinių, nenaudojant atskirai tai užduočiai parašytų taisyklių, atlikti specifines užduotis. Mašininis mokymasis laikomas viena iš dirbtinio intelekto rūšių.

Pirmiausiai šie algoritmai gali būti apmokomi keliais būdais (Brownlee, 2019):

1. Prižiūrimas mokymasis – mokymosi duomenys yra iš anksto suklasifikuojami ir perduodami mokymosi algoritmui. Sprendžiamos problemos: klasifikacija ir regresija.
2. Neprižiūrimas mokymais – mokymosi duomenys nėra niekaip sužymėti ir algoritmas turi pats suprasti, kaip jie tarpusavy susiję. Sprendžiamos problemos: klasterizacija, dimensijų redukcija, asociacijos taisyklių mokymasis.
3. Iš dalies prižiūrimas mokymasis – tarp mokymosi duomenų yra ir sužymėtų, ir nesužymėtų duomenų. Dažniausiai naudojamas neprižiūrimo mokymosi rezultatams pagerinti.
4. Mokymasis su palaikymu (angl. *reinforcement*) – su dalimi duomenų pasimoko, o po to bando save įsivertinti su kita duomenų dalimi. Naudojamas žaidimų dirbtiniui intelektui kurti, simuliacijoms, genetiniams algoritmams ir kitoms problemoms, kurias galima nesunkiai automatiškai įvertinti.

Mašininio mokymosi algoritmų taipogi yra daug ir įvairių (Brownlee, 2019):

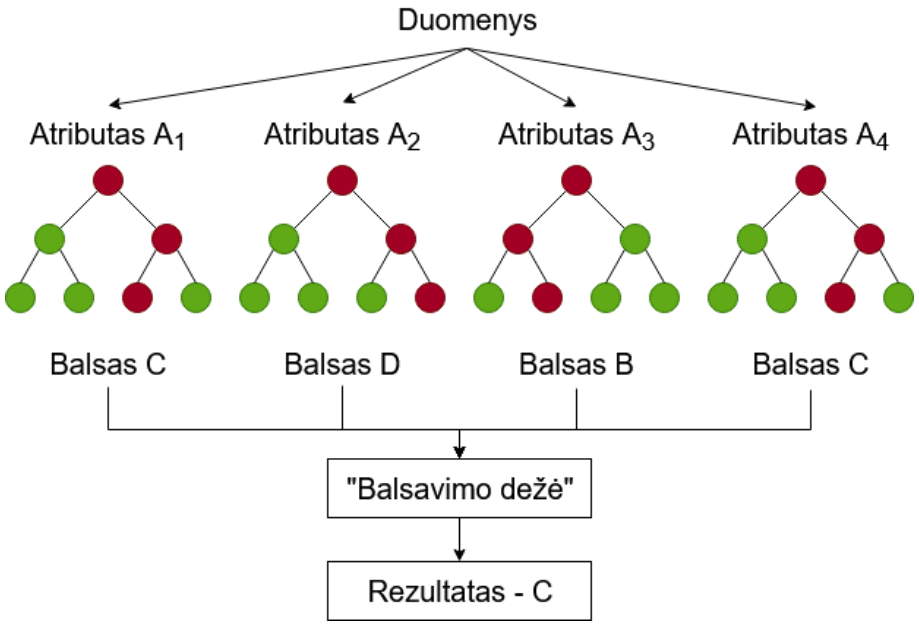
1. Regresijos algoritmai.
2. Sprendimų medžiai.
3. Bajeso algoritmai.
4. Klasterizacijos algoritmai.
5. Asociacijos taisyklės.
6. Dirbtiniai neuroniniai tinklai.

7. Giluminio mokymosi algoritmai.

Toliau plačiau apžvelgiami vieni iš populiaresnių mašininio mokymosi algoritmų (Goyal, 2019).

Atsitiktiniai miškai

Atsitiktinis miškas (angl. *random forest*) – pakankamai paprastas, bet galingas prižiūrimas algoritmas, kuris susideda iš nepriklausomai apmokomų sprendimų medžių aibės. Kiekvienas sprendimų medis yra apmokomas su skirtinga dalimi duomenų arba medžiams parenkami skirtingi atributai, todėl jie tarpusavyje skiriasi. Sugeneravus visus medžius sprendimai priimami pagal kiekvieno medžio balsą, jei dirbama su kiekybiniais duomenimis, paimamas balsų vidurkis, jei dirbama su kokybiniais duomenimis, parenkamas rezultatas su daugiausiai vienodų balsų. Atsitiktinio miško veikimo schema pavaizduota **3 pav**.



3 pav. Atsitiktinio miško struktūros schema

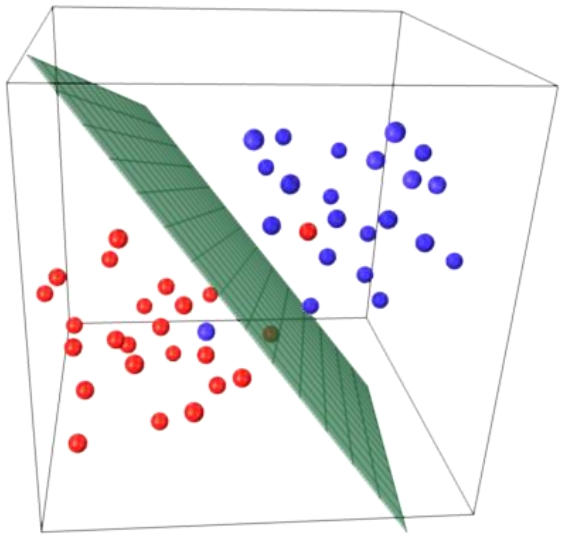
Sprendimų medžiai apjungiami į atsitiktinį mišką dėl to, kad atskirai jie yra labai jautrūs duomenų pasikeitimams, todėl nelabai patikimi. Tačiau kai medžių yra daug ir sprendimai priimami statistiniu metodu, spėjimo tikslumas smarkiai padidėja.

Gradiento pastiprinimo mašinos

Gradiento pastiprinimo mašina (angl. *gradient boosting machine*) – dar vienas algoritmas paremtas sprendimų medžių aibe. Medžiai nuolat yra stiprinami tikrinant paklaidas, t. y. kiekvienas modelis yra treniruojamas ieškant negatyvaus gradiento krypties. Pirmiausia sugeneruojamas vienas medis, tada panaudojus mokymosi duomenis patikrinama, ar spėjimo rezultatai geri. Pagal buvusio medžio rezultato kokybę sugeneruojamas naujas medis ir tikrinama, ar rezultatas tapo tikslesnis. Visi sugeneruoti medžiai yra pridedami į aibę ir jų spėjimo reikšmės yra atitinkamai dauginamos iš svorių, kuo geriau medis pasirodė su mokomaisiais duomenimis, tuo didesnį svorį jis turės. Taip modelis yra iteratyviai tobulinamas tol, kol bus pasiektas norimas medžių kiekis arba norimas tikslumas.

Palaikymo vektorių mašinos

Palaikymo vektoriaus mašina (angl. *support vector machine*) – algoritmas, kurio tikslas duomenis padalinti į dvi dalis. Duomenims su viena dimensija parenkama viena reikšmė, dviejose dimensijose parenkama tiesė, trijose dimensijose – plokštuma, pavaizduota **4 pav**. Palaikymo vektorius kitaip nei kiti panašūs klasifikavimo algoritmai, ignoruoja pasirinktą skaičių per daug nuo normos nutolusių duomenų, taip pat riba tarp duomenų parenkama ne pagal arčiausiai krašto esančius duomenis, o truputi gilesnes reikšmes.



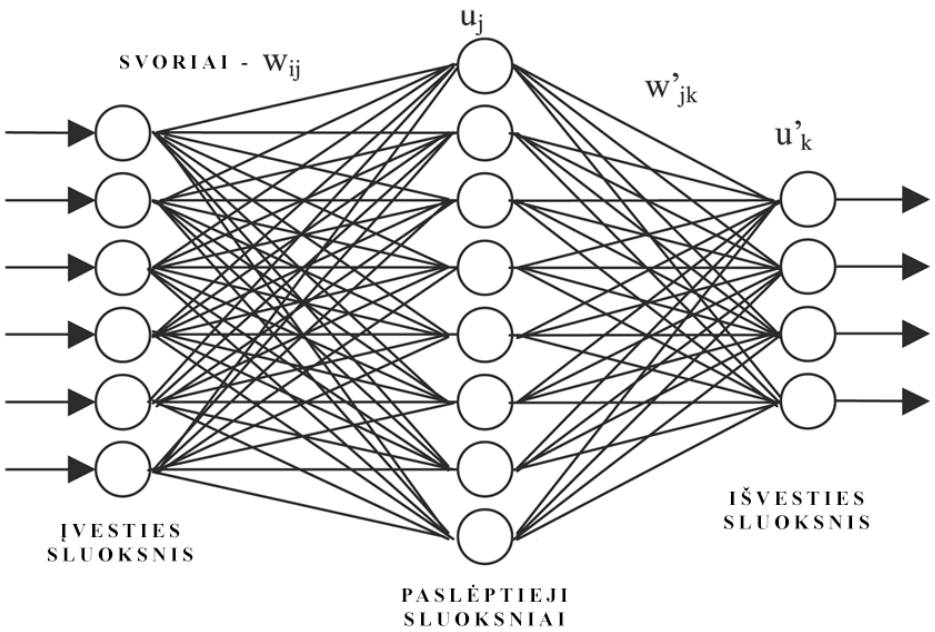
4 pav. Duomenų su trimis dimensijomis klasifikacija

Neuroniniai tinklai

Neuroniniai tinklai naudojami sudėtingoms problemoms spręsti: vaizdų klasifikavimui, kalbos supratimui, daiktų atpažinimui. Šiuose algoritmuose įvesties duomenys yra perduodami į tarpusavyje sujungtų mazgų, kitaip vadinamų neuronais, tinklą. Šie neuronų tinklai yra padalinti į kelis sluoksnius. Perduodant duomenis iš vieno sluoksnio į kitą atliekamos tam tikros matematinės operacijos.

Paprastų neuroninių tinklų architektūros:

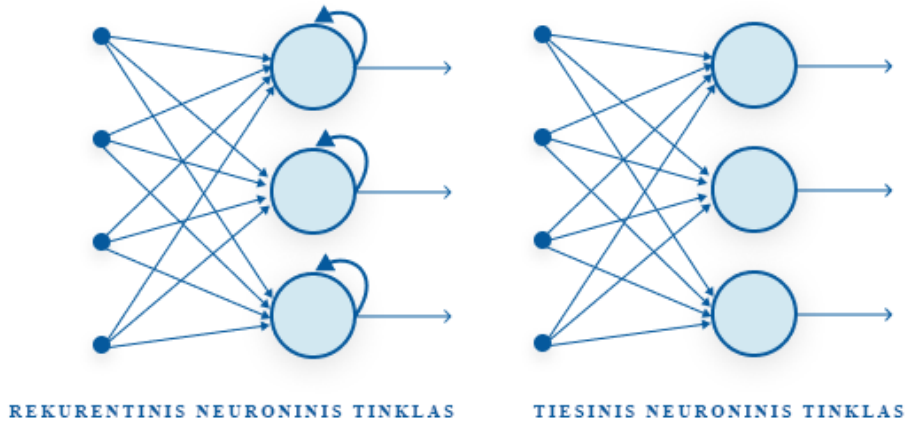
1. Tiesinis (angl. *feed-forward*) neuroninis tinklas – paprasčiausias tipas. Jis susideda iš įvesties sluoksnio, galimų vieno ar daugiau paslėptų sluoksnių ir vieno išvesties sluoksnio (Ussath, Jaeger, Cheng, & Meinel, 2017). Priklausomai nuo užduoties sudėtingumo paslėptų sluoksnių ir mazgų gali būti daugiau arba mažiau. Tiesinio neuroninio tinklo diagrama pavaizduota **5 pav.**



5 pav. Tiesinio neuroninio tinklo diagrama (pagal Templeton, 2015)

2. Grįžtamojo ryšio (angl. *recurrent*) neuroninis tinklas – kiek sudėtingesnis tipas, naudojamas, kai įvesties duomenys yra seka. Šis neuroninis tinklas panašus į tiesinį, tiesiog išvestis kiekvienam sekos elementui yra skaičiuojamas atskirai ir paslėptų sluoksnių reikšmės yra išsaugomos kitai iteracijai. Tai reiškia, kad be pirmo elemento visų likusių sekos elementų išvestims apskaičiuoti yra naudojamos

buvusių sekos elementų reikšmės (Ussath et al., 2017). Grįžtamojo ryšio neuroninio tinklo skirtumas nuo tiesinio neuroninio tinklo pavaizduotas **6 pav.**



6 pav. Grįžtamojo ryšio ir tiesinio neuroninių tinklų skirtumai (pagal Roos, 2019)

Tarp visų skirtingų lygių neuronų yra ryšiai, kurie turi savo reikšmes – svorius. Svoriai prieš mokymą yra parenkami atsitiktinai arba bandymo metodu, o besimokant keičiasi priklausomai nuo gaunamų duomenų.

Taip pat neuronai turi įvairias aktyvacijos funkcijas (Boukerche & Annoni Notare, 2002), kurios padeda filtruoti rezultatus arba optimizuoti neuroninį tinklą sumažinant reikiamų neuronų kiekį. Aktyvacijos funkcijos būna:

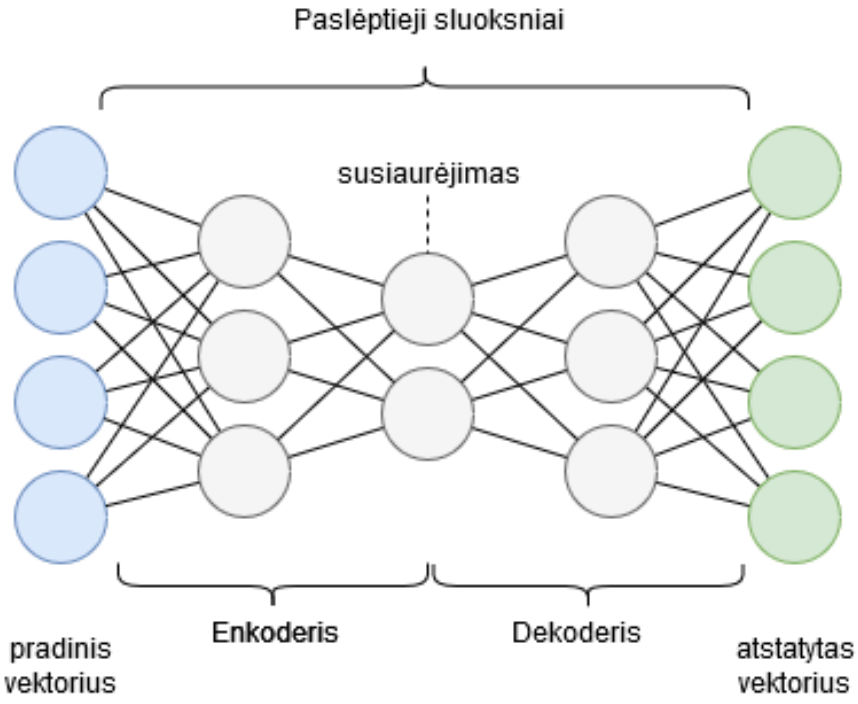
- tiesinės;
- Gauso;
- sigmoidinės;
- binarinės ir t. t.

Anomalijoms aptikti paprasti neuroniniai tinklai nėra praktiški, nes jiems reikia klasifikuotų duomenų. Toliau aprašomas vienas iš neapmokomų neuroninių tinklų – autoenkoderis.

Autoenkoderiai

Autoenkoderis – specifinis neuroninio tinklo tipas, kuris naudojamas neprižiūrimoms mašininio mokymosi užduotims atlikti (Subasi, 2020). Pagrindinis šio modelio tikslas yra išmokti apytiksles duomenų reprezentacijas. Autoenkoderiai gali būti naudojami: duomenų suspaudimui, išplėtimui, triukšmo mažinimui bei anomalijų aptikimui.

Tinklas susideda iš enkoderio ir dekoderio, jie abu yra tokios pat struktūros, tik apkeisti vietomis. Enkoderis mokosi, kaip įvesties vektorių suspausti į mažesnę duomenų reprezentaciją, neprarandant esminės informacijos, o dekoderis mokosi, kaip kuo tiksliau atstatyti originalų vektorių. Šis veikimo būdas pasiekiamas padarius susiaurėjimą viduriniame neuronų sluoksnyje, tinklo susiaurėjimas pavaizduotas **7 pav.** Mokymasis vyksta optimizuojant nuostolio (angl. *loss*) funkcijos rezultatą, t. y. parenkama funkcija, su kuria palyginami įvesties bei išvesties vektoriai, ir mokant modelį stengiamasi gauti kuo mažesnį nuostolio funkcijos rezultatą.



7 pav. Autoenkoderio pavyzdys

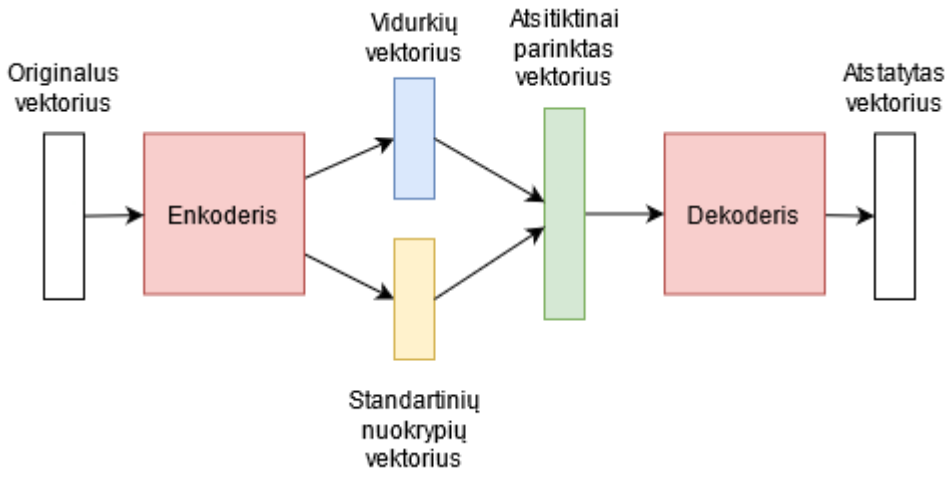
Anomalijų aptikimo veikimo principas naudojant autoenkoderį: modelis apmokomas apie normalią stebimos sistemos būseną, pasibaigus mokymui autoenkoderiui duodami nauji duomenys, jis duomenis apdoroja su apmokytu tinklu ir lygina gautą rezultatą su originaliais duomenimis. Jei skirtumas tarp duomenų viršija nustatytą ribą, gauti duomenys klasifikuojami kaip anomalūs.

Aprašytas autoenkoderis yra paprasčiausias jo variantas, yra ir daug sudėtingesnių, kurie atliekant kai kurias užduotis duoda geresnių rezultatų. Vienas iš jų, kuris dažnai sutinkamas naujesniuose moksliniuose straipsniuose, yra variacinis autoenkoderis, toliau VAE.

Variacinis autoenkoderis

Variacinis autoenkoderis – autoenkoderis, kurio mokymo procesas yra reguliarizuotas tam, kad apmokytas tinklas būtų kuo bendresnis, ir kad tinklo viduryje esantis suspaustas sluoksnis būtų tinkamai paruoštas naujų duomenų generavimui (Rocca & Rocca, 2019). Lyginant su įprastu autoenkoderiu VAE pagrindinis skirtumas tas, kad duomenys suspaudžiami ne į vieną mažesnės dimensijos vektorių, o į suspaustą skirstinį. Taip pat VAE be įprastam autoenkoderiui įkandamų užduočių sugeba sukurti naujus duomenis pagal tai, kaip buvo apmokytas. Pavyzdžiui, modelis sugeba žmogaus veido profilio nuotrauką pasukti keliais laipsniais ir gan įtikinamai atkurti, kaip veidas turėtų atrodyti iš naujos pozicijos. VAE neuroninio tinklo diagrama pavaizduota **8 pav.** Apmokant modelį:

- pirmiausia įvesties vektorius užkoduojamas į skirstinį;
- tada iš skirstinio paimamas vienas atsitiktinis taškas;
- paimtas taškas dekoduojamas ir gaunamas rezultatas gali būti lyginamas su originaliu vektoriumi.



8 pav. VAE neuroninio tinklo diagrama

Kaip pavaizduota diagramoje, suspaustas skirstinys susideda iš vidurkių ir standartinių nuokrypių vektoriaus ir praktikoje pasiskirstęs pagal normalųjį skirstinį.

Dauguma įsilaužimo metodų turi nemažai bendrų bruožų, tad prieš apžvelgiant įvairius anomalijų aptikimo metodus, pirma bus išvardinti pagrindiniai įsilaužimo aptikimo žingsniai.

2.4.2. Bendri įsilaužimo aptikimo metodų bruožai

Pirmiausia reikia surinkti teisingą informaciją iš žurnalų ir kitų šaltinių. Tai gali būti:

1. Sėkmingi, nesėkmingi prisijungimai – akivaizdžiai daug nesėkmingų prisijungimų gali reikšti bandymus atspėti slaptažodį.
2. Kompiuterio prisijungimo prie tinklo laikai – prisijungimas naktį arba po darbo gali reikšti, kad jungiamasi iš kitos laiko zonos, t. y. į kompiuterį įsilaužta iš kitos šalies.
3. Failų judėjimas – failų įrašymas į atmintines arba diskus bei jų įkėlimas į debesis. Tai gali reikšti duomenų nutekėjimą, ypač, jei darbuotojas to įprastai nedaro.
4. Naudojimosi el. paštu duomenys – kokiems žmonėms siųsti laiškai, ar laiškai siųsti toje pačioje kompanijoje, kokio dydžio laiškai, ar yra prisegta prie jų failų. Laiškų siuntimas su prisegtais failais į kitą nei kompanijos domeną gali reikšti duomenų nutekėjimą.
5. Internetu pasiektų svetainių sąrašas – tarp jų gali būti juodajam sąraše esančių svetainių, kurios tarkim turi kenksmingos programinės įrangos, arba į kurias galima įkelti duomenis kaip „Dropbox“ ar „Onedrive“.
6. Psichometriniai darbuotojų duomenys – psichologiniai testai, kurie galėtų padėti nustatyti, kurie darbuotojai mažiau lojalūs kompanijai, įsilaužimams jie neaktualūs.

Antra, reikia visus duomenys vektorizuoti. Tarkim, jei tai yra sąrašas svetainių, prie kurių buvo prieita, jas galima sugrupuoti pagal lankomumą ir tada duomenų vektoriuje atsirastu, pavyzdžiui, trys dimensijos: dažniausiai, vidutiniškai ir retai lankomų svetainių kiekis, retai lankomos svetainės jau yra anomalija ir galėtų reikšti lankymąsi po užkrėstus ar nesaugius puslapius. Taip pat šį duomenų vektorių gali tekti normalizuoti.

Galiausiai duomenis reikia perduoti kažkokiam algoritmui, kuris sugeba įvertinti, ar naudotojo veiksmas yra anomalija, ar ne. Prieš tai algoritmą reikėtų apmokyti su normalaus ir patikimo naudotojo veiksmų rinkiniu.

Papildomai dar galima įrengti medaus puodynę, kuri aptikus anomalijas visas užklausas siųstų į netikrą servisą ir grąžintų medaus žetonus. Medaus žetonai (angl. *honeytoken*) yra netikri, sugeneruoti duomenų objektai, kurie yra itin panašūs į tikrus duomenis (Qdo, Ri, Wkuhdw, Xvlqj, & Rulhqwhg, 2017). Tokiu būdu užpuolikas neįtaria, kad yra aptiktas, ir duomenys nėra nutekinami į išorę. Be to naudojantis medaus puodyne galima įsitikinti, kad įsilaužėlis bando atlikti įtartinus veiksmus saugioje aplinkoje.

2.4.3. Scenarijumi paremtas anomalijų aptikimas

Scenarijus įsilaužimo arba duomenų nutekinimo kontekste yra tam tikrų veiksmų seka, pagal kurią yra tyčia arba netyčia pasinaudojant organizacijos sistemomis, tinklais, duomenimis ar kitais resursais padaroma žala (Chattopadhyay, Wang, & Tan, 2018). Tai gali būti:

- sabotažas;
- reputacijos gadinimas;
- slaptų duomenų nutekinimas;
- nuolatinis šnipinėjimas;
- klientų duomenų pasisavinimas ir t. t.

Šiems kenksmingiems scenarijams aptikti daugiausia yra bandoma naudoti neprižiūrimus mašininio mokymosi algoritmus, kurie patys mokosi iš normalaus vartotojo elgesio ir sugeba aptiki anomalijas. Tačiau didžiausia problema tokiuose algoritmuose yra normalių veiksmų nustatymas. Jei darbuotojo darbas yra labai besikeičiantis, gali būti, kad šio žmogaus paskyroje anomalijos bus aptinkamos per dažnai arba per retai. Visa tai priklauso nuo to, kaip algoritmas išmokytas. Daug naujesnių algoritmų bando išspręsti būtent šią problemą.

Pavyzdžiui, vienas iš sprendimo būdų yra bandyti lyginti įvykius su žinomais kenksmingų scenarijų žingsniais. Pirmiausia yra apskaičiuojamas anomalinio elgesio koeficientas kiekvienai dienai, o po to bandoma šiuos koeficientus lyginti su žinomų įsilaužimo scenarijų koeficientais (Chattopadhyay et al., 2018). Šis algoritmas pranašesnis už kitus tuo, kad kiekviena diena analizuojama atskirai. Daroma prielaida, kad tą pačią dieną veiksmai iš esmės nesiskiria, dėl to toks požiūris leidžia apeiti problemą kylančią dėl po truputi besikeičiančio elgesio. Tačiau problema ta, kad kenkėjiškus scenarijus reikia žinoti iš anksto.

Kitas sprendimo būdas yra padalinti įsilaužimo aptikimo metodą į dvi dalis (Yang, Cai, Yu, & Meng, 2018):

1. Pirmoji dalis veikia kaip įprastas neprižiūrimas mašininio mokymosi algoritmas, kuris sugeba pats išanalizuoti veiksmus ir įvertinti, ar jie yra normalūs, ar įtartini.
2. Antra dalis yra kaip atskiras papildomas filtras, kuris gautą rezultatą dar patikrina pagal sukonfigūruotus scenarijus.

2.4.4. Pasitikėjimu paremtas anomalijų aptikimas

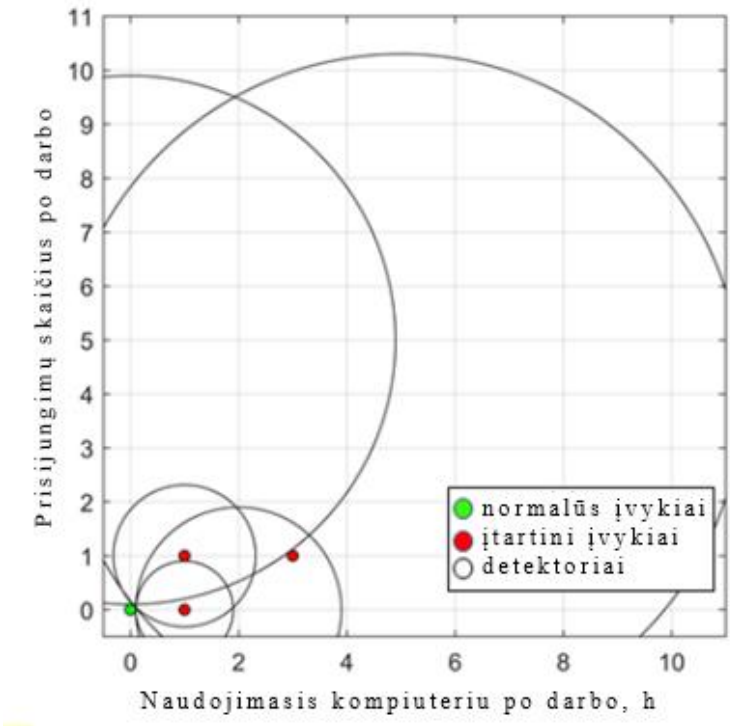
Pasitikėjimu paremtas įsilaužimo aptikimo metodas atskirai analizuoja tiek kiekvienos dienos veiksmus, tiek susumuotus savaitinius ar mėnesinius veiksmus. Taip pat šis metodas be įprastinių duomenų šaltinių kaip sistemų žurnalai pasitelkia ir darbuotojų psichometrinius duomenis. Tada skaičiuojant naujos dienos ar savaitės anomalijos koeficientą paimamas koeficientas iš praeitų skaičiavimų, kuris taip pat vadinamas pasitikėjimo rodikliu. Galiausiai koeficientai iš skirtingų periodų yra apjungiami ir analizuojami bendrai (Aldairi, Karimi, & Joshi, 2019).

Pasitikėjimu paremtas metodas labiau skirtas nelojaliems vietiniams darbuotojams aptikti negu išoriniams įsilaužėliams, nes į analizė įtraukiami psichometriniai duomenys, o įsilaužėliai gali apsimesti ir lojaliais bei patikimais darbuotojais.

2.4.5. Negatyvia atranka paremtas anomalijų aptikimas

Negatyvus atrankos algoritmas yra paremtas žmogaus imunine sistema (Igbe & Saadawi, 2018). Pirmas šio algoritmo žingsnis yra atsitiktinis detektorių generavimas. Šie detektoriai prilygsta žmogaus B ląstelėms ir yra sukuriami atsitiktinai parenkant normalius darbuotojo veiksmus. Tada šie detektoriai bando aptikti panašiausią į save normalaus darbuotojo veiksmą ir iki aptikto veiksmo apskaičiuojamas spindulys. Šie detektoriai yra generuojami tol, kol jie neužpildys didžiosios dalies erdvės, kurioje nėra normalių darbuotojų veiksmų. Ploto užimtumas apskaičiuojamas pagal detektorių spindulius.

Pats anomalijų aptikimas vyksta apskaičiuojant naujo veiksmo spindulį iki artimiausio detektoriaus. Jei atstumas iki artimiausio detektorius yra mažesnis nei to detektoriaus spindulys, veiksmas pažymimas kaip įtartinas, jei iki artimiausio detektoriaus yra toliau negu jo spindulys, tada veiksmas yra normalus. Detektorių aprėpiama erdvė pavaizduota **9 pav**.



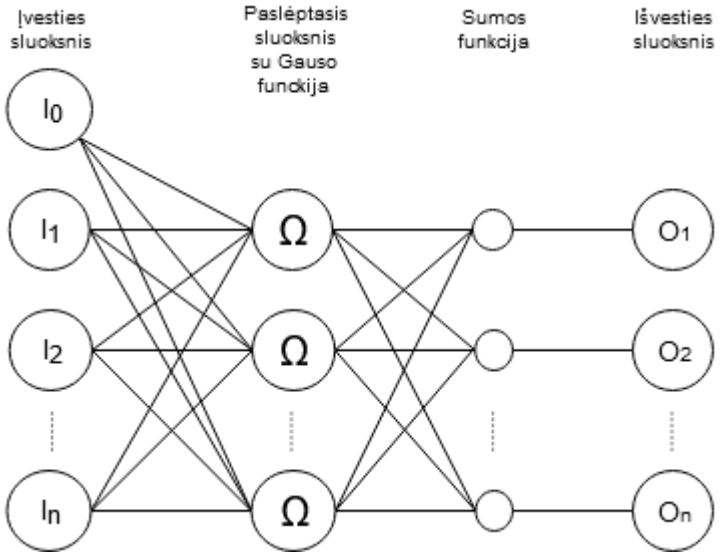
9 pav. Sugeneruota negatyvios paieškos erdvė

2.4.6. Paprastas neuroninis tinklas anomalijoms aptikti

Vieni iš mašininio mokymosi metodų anomalijoms aptikti yra neuroniniai tinklai, jie gali būti tiek prižiūrimi, tiek neprižiūrimi. Prižiūrimų privalumas tas, kad anomalijų aptikimas turėtų būti tikslesnis, nes duomenys yra patikrinami bei rankiniu būdu surūšiuojami į nepavojingus ir į kenksmingus. Jei darbuotojų veikla patikima ir pakankamai pastovi, neuroninį tinklą galima apmokyti tiesiog panaudojus jų elgesio duomenis, tariama, kad visa veikla yra vienodai neįtartina, kitu atveju duomenis reikia klasifikuoti rankiniu būdu ir tai gali užtrukti itin daug laiko. Dėl to praktiškiau yra naudoti kažkokį protingą būdą jau suklasifikuotiems duomenims generuoti arba naudoti neprižiūrimą neuroninį tinklą, kuris pagal paprastas taisykles pats viską išmoktų. Duomenų generavimas nėra labai patikimas būdas neuroniniams tinklams mokyti, nes realūs scenarijai įvairesni ir sudėtingesni, o investuoti daug laiko įmantriems duomenims generuoti irgi nepraktiška.

Vienas iš panaudotų paprastų neuroninių tinklų anomalijoms aptikti yra RBF neuroninis tinklas (Boukerche & Annoni Notare, 2002). Jo architektūra paprasta: įvesties sluoksnis, vienas paslėptasis sluoksnis ir išvesties sluoksnis. Mazgų reikšmėms apskaičiuoti paslėptasis sluoksnis naudoja Gauso funkciją ir atstumą iki kažkokio bendro taško, kurį reprezentuoja visas neuroninio

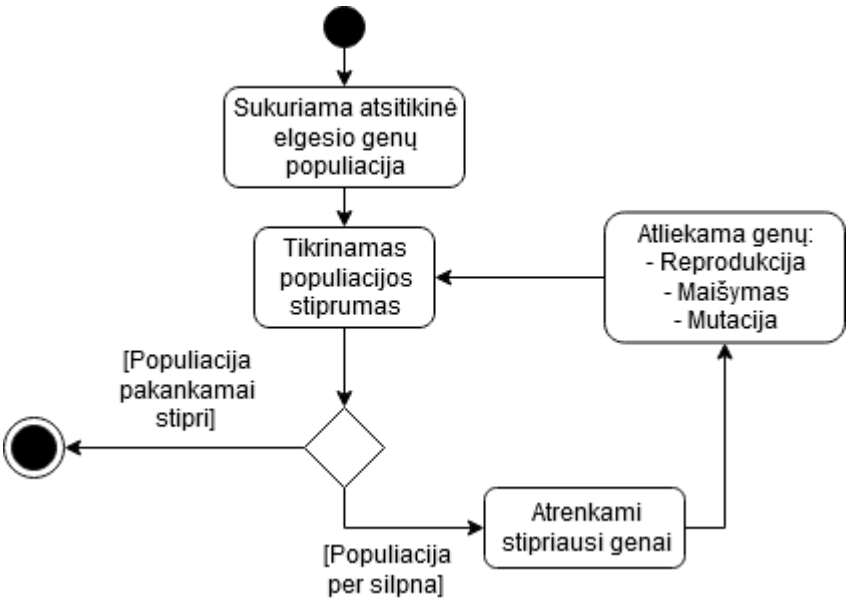
tinklo modelis. Modelio taškas turi tiek pat dimensijų, kiek yra įvesties mazgų, o atstumas iki taško apskaičiuojamas naudojant pasirinktą euristinę funkciją ir palyginus su kitais neuroniniais tinklais atlieka svorio vaidmenį. RBF neuroninio tinklo struktūra pavaizduota **10 pav**.



10 pav. RBF neuroninis tinklas

2.4.7. Genetinis neuroninis tinklas anomalijoms aptikti

Neuroniniams tinklams gali būti pasitelktos įvairios mokymo metodologijos, viena iš jų yra genetiniai algoritmai, kurie bando simuliuoti natūralią gamtos atranką ir genetiką (Balajinath & Raghavan, 2001). Šiuo atveju genai yra naudotojų veiksmai. Naudotojo veiksmų rinkiniai, kurie tiksliau atitinka tikrų vartotojų veiksmus, yra tvirtesni, todėl parenkami tolimesniam mokymuisi, o tie, kurie pasirodo prastai, yra silpnesni ir pašalinami. Patys veiksmų rinkiniai yra sugeneruojami atsitiktiniu būdu iš didelio veiksmų sąrašo, užkoduoto skaitinėmis reikšmėmis, o vėliau evoliucionuojami su mutacijom ir maišymu. Stipriausi rinkiniai yra maišomi vieni su kitais ir gaunami nauji rinkiniai, kurie turi veiksmų tiek iš vieno, tiek iš kito tėvinio rinkinio. Taip pat nauji rinkiniai turi šansą mutuoti, t. y. atsitiktinai pakeisti kokį nors geną. Mokymosi procesas yra kartojamas tol, kol randami rinkiniai su pakankamai geru tvirtumu arba tvirtumas nebesikeičia į gerąją pusę nei mutuojant, nei maišant. Tinklo mokymosi schema pavaizduota **11 pav**.



11 pav. Genetinio neuroninio tinklo mokymo schema anomalijoms aptikti

Taip išmokytiems neuroniniams tinklams nereikia turėti iš anksto suklasifikuotų duomenų rinkinių ir tinklas naudotojų elgesį išmoksta neprižiūrimas. Be to mokymasis gali būti įjungiamas vėl kas kelias savaites ar mėnesius, jei darbuotojo veiksmai dažnai keičiasi.

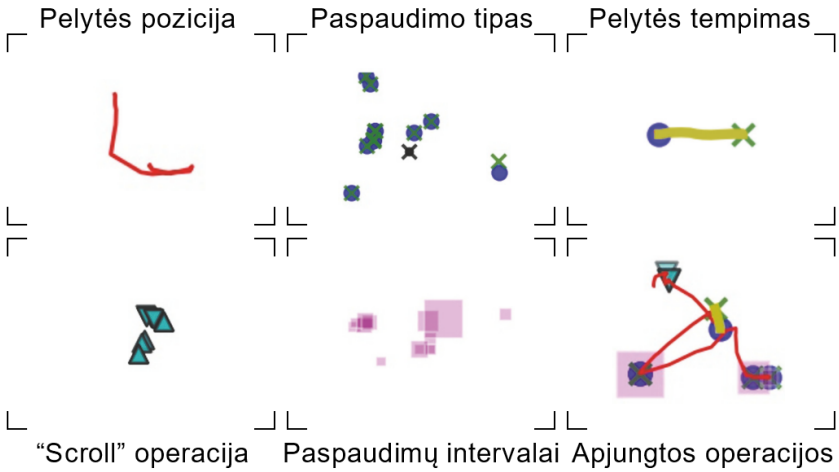
2.4.8. Periferine įranga paremti anomalijų aptikimo metodai

Be veiksmų sistemoje stebėjimo yra anomalijų aptikimo metodų, kurie pasikliauja periferinės įrangos naudojimo analize. Stebima įranga gali būti:

- pelytė;
- klaviatūra;
- vaizdo kamera.

Vaizdo kamera su įdiegta veido atpažinimo įranga gali būti naudojama pastoviai veidui stebėti. Tai reiškia, kad norint daryti kažkokius veiksmus su kompiuteriu, prie jo būtinai turi sėdėti konkretus darbuotojas. Jei bandoma aptikti ne tik įsilaužimus, bet ir keistą darbuotojų elgesį, dar galima stebėti žmogaus emocijas – iš veido galima pamatyti, ar žmogus yra susijaudinęs ir bando kažką nuslėpti.

Stebint pelytę galima analizuoti jos būsenas skirtingais laiko momentais būsenas paverčiant į paveiksliukus (Hu et al., 2019). Taip pat galima atskirai stebėti naudojimosi pelyte dinamiką: judinimo dažnumą, judėjimo krypties pasikeitimus, mygtukų paspaudimų dažnumą bei paspaudimo ilgumą, naudojimosi vidurinio mygtuku ypatybes ir t. t. (Hashem, Takabi, & Dantu, 2017). Pelytės būsenų paveiksliukai pademonstruoti **12 pav**.



12 pav. Pelytės būsenų paveiksliukai

Galiausiai galima stebėti naudojimo klaviatūra rodiklius: bendrus klaviatūros mygtukų paspaudimo duomenis, naudojimąsi trynimo mygtukais „Backspace“ ir „Delete“ bei naudojimąsi skaičių mygtukais (Hashem et al., 2017).

Išanalizavus žmogaus naudojimosi pelyte ir klaviatūra dinamiką iš šių duomenų galima sukurti darbuotojo unikalų profilį. Įsilaužėliams būtų labai sunku atkartoti tokį veiksmų profilį, tad IDS iš karto aptiktų, kad kažkas yra negerai ir informuotų atitinkamus asmenis apie aptiktą anomaliją.

2.5. Populiariausių HIDS apžvalga

Daugelis paminėtų įsilaužimo aptikimo metodų yra nauji ir išbandyti su viešai prieinamais, dažniausiai įvairių universitetų duomenimis. Tai reiškia, kad šie algoritmai veikiausiai nebuvo išbandyti didesnėse pelno siekiančiose organizacijose arba valstybinėse institucijose, kurios gali sulaukti daugiau ir įmantresnių bandymų įsilaužti. Dėl to tyrėjų gauti rezultatai gali būti netikslūs. Norint išsiaiškinti, ar ši prielaida teisinga, reikia patikrinti, kokius metodus naudoja dabartinės

populiariausios IDS. Populiariausių (8 Best HIDS Tools—Host-Based Intrusion Detection Systems, 2019; Cooper, 2019) HIDS palyginimas pademonstruotas **1 lentelėje**.

1 lentelė. Populiariausių HIDS palyginimas

Produktas	Prieinamumas	OS, kuriose veikia	Veikimo tipas
SolarWinds Security Event Manager	Mokamas su 30 d. pabandymo laikotarpiu	Windows, Linux, Mac OS	-
Papertrail	Ribota nemokama versija	Windows	Anomalijomis ir parašais paremtas
ManagerEngine Event Log Analizer	Ribota demonstracinė versija	Windows ir Linux	-
OSSEC	Ribota nemokama versija	Windows, Linux, Mac OS	Anomalijomis paremtas
Sagan	Nemokama	Linux, Mac OS	Anomalijomis ir parašais paremtas
Splunk	Nemokama	Linux, Mac OS	Anomalijomis paremtas

Tarp populiariausių HIDS produktų yra tiek didelėms korporacijoms priklausančių programų, kurios turi pinigų brangesnei įrangai išleisti, tiek mažoms organizacijoms ar net privatiems asmenims. Taip pat visos šiame sąraše esančios mokamos IDS turi bandomąsias versijas. Windows operacinėje sistemoje veikiančių nemokamų IDS yra nedaug ir jos skirtos didelėms organizacijoms. Šis Windows HIDS trūkumas egzistuoja greičiausiai dėl to, kad privatiems asmenims ar mažoms kompanijoms užtenka stebėti serverius, o stebėti kiekvieną darbuotojo kompiuterį, ypač, jei jų vos keli, neapsimoka.

Dauguma IDS produktų viešai neatskleidžia savo veikimo principų. Dažniausiai paminima, ar naudojami parašais, ar anomalijomis paremti metodai. Matosi, kad anomalijomis paremti metodai yra būtini ir visi produktai šiais metodais naudojasi, dalis produktų apjungia anomalijomis paremtus ir parašais paremtus metodus.

Kartais produktų veikimo metodai plačiau būna aprašomi patentuose. Splunk sistema vienintelė turi patentą ir pasak jo ši IDS naudoja kažką panašaus į negatyvia atranka paremtą anomalijų aptikimą (Patent No. US 2015/0341246A1, 2015).

2.6. Skyriaus apibendrinimas

Šiame skyriuje buvo apibrėžta įsilaužimų aptikimo sistemos sąvoka, aprašyti jų veikimo ir naudojamų metodų tipai. Trumpai apibūdintas mašininis mokymas. Apžvelgti naujesni įsilaužimams ir keistam vartotojų elgesiu aptikti naudojami metodai. Galiausiai išvardintos ir palygintos populiariausios HIDS.

Apžvelgus IDS paaiškėjo, kad keistam vartotojo elgesiui aptikti geriausiai tiktų kompiuteryje veikianti sistema, kuri veikia anomalijų aptikimo principu. Dabartinėse įsilaužimo aptikimo sistemose naudojami tiek įprasti statistiniai metodai, tiek mašininio mokymosi algoritmai. Tačiau dauguma anomalijų aptikimo metodų išbandyti su viešai prieinamais duomenimis ir realiose organizacijose nebuvo diegiami. Taip pat iki šiol eksperimentai buvo atlikti su paprastais neuroniniais tinklais, o tai reiškia, kad galima bandyti pritaikyti sudėtingesnius, konvoliucinius neuroninius tinklus arba kitokio tipo mašininio mokymosi metodus, kurie buvo sėkmingai panaudoti kitose mokslo srityse.

Toliau bus renkami ir susisteminami vartotojų elgesio duomenys. Bus parinktas ir įgyvendintas kitoks neuroninis tinklas anomalijų aptikimo rezultatui pagerinti. Galiausiai su suprogramuotu aptikimo metodu bus atliktas eksperimentas ir pateikti gauti rezultatai su išvadomis.

3. Siūlomas anomalijų aptikimo metodas

Šiame skyriuje aprašomi darbui su anomalijų aptikimo metodu naudojami duomenys, jų surinkimas, apdorojimo žingsniai, pasirinktas metodas, sukurtas analizės prototipas bei prototipo bandymo metu gauti rezultatai.

3.1. Duomenys

Anomalijų aptikimo metodui patikrinti naudojami du duomenų rinkiniai: vienas surinktas būtent šiam darbui kaip būdas aptikimo tikslumui padidinti, o kitas viešai prieinamas „ADFA-LD“ duomenų rinkinys. Viešas rinkinys naudojamas norint palyginti surinkto duomenų rinkinio efektyvumą.

„ADFA-LD“ duomenų rinkinys susideda iš įvairių atakų, kurios buvo sugeneruotos saugioje aplinkoje (Creech, 2014). Šis rinkinys pasirinktas dėl to, kad tie patys tyrėjai taip pat yra paruošę „Windows“ OS skirtą duomenų rinkinį, tačiau jis viešai nebeprieinamas. Kitų „Windows“ OS skirtų duomenų rinkinių neišėjo surasti, dėl to „Linux“ OS skirtas „ADFA-LD“ yra artimiausias reikalingam rinkiniui.

Rinkinys susideda iš: mokomųjų, validacijai skirtų ir puolimo failų. Mokomieji failai skirti algoritmo apmokymui, su validacijai skirtais ir puolimo duomenimis tikrinama, ar modelis teisingai apmokomas. Kiekvienas tekstinis failas susideda iš eilės skaičių, kur kiekvienas skaičius reiškia kažkokią užkoduotą komandą ar komandų seką operacinės sistemos branduoliui. Užkodavimui buvo pasitelktas dviejų žingsnių procesas. Pirmuoju nuskaityti visi surinkti komandų rinkiniai ir sukurtas visų užfiksuotų komandų žodynas. Tam, kad žodynas nebūtų per didelis, žodžiai, kurie neatitiko tyrėjų pasirinkto filtro, buvo išmesti. Antro žingsnio metu sukuriamas frazių žodynas, kur frazės yra nuo 1 iki 5 komandų, sugrupuotų kartu. Galiausiai duomenų rinkinys dar kartą skenuojamas ir aptiktus vieną iš išsaugotų frazių, ją identifikuojantis skaičius įrašomas į failą, kurį jau galima naudoti apmokant įvairius modelius.

„ADFA-LD“ atakų tipai yra šie:

1. Naujo vartotojo pridėjimas – vartotojas pridedamas atidarius užkrėstą failą.
2. „Hydra FTP“ ir „Hydra SSH“ atakos – nuotoliniu „bruteforce“ metodu atrandami slaptažodžiai. Pakankamai triukšmingi, tad turėtų būti lengvai aptinkami.
3. „Java Meterpreter“ ir „Linux Meterpreter“ atakos – įsiterpia į sistemos modulius ir leidžia interaktyviai atlikti kenksmingus veiksmus.
4. „C100 Web Shell“ ataka.

Atakoms įgyvendinti pasitelkti įvairūs būdai: socialinės inžinerijos simuliacija, iš kliento pusės paleidžiama ataka, paleidžiami kompromituoti failai, per internetą atliekama ataka.

Toliau bus aprašytas rašto darbo metu surinktų duomenų surinkimo procesas.

3.1.1. Duomenų surinkimas

Norint pasiekti didesnį anomalijų aptikimo metodų efektyvumą pasirinkta surinkti savus duomenis. Savi duomenys leidžia parinkti konkrečius puolimo scenarijus ir labiau atitinka kompiuterio naudotojo veiksmus, nes viešai prieinami duomenų rinkiniai, naudojami anomalijų aptikimui, yra simuliuoti. Taip pat bent jau „ADFA-LD/WD“ naudoja ganėtinai sudėtingus ir brangius būdus duomenims surinkti, kurie galbūt nebūtų tinkami praktiniam naudojimui realioje IDS sistemoje. Palyginus su „Linux“, „Windows“ OS labai sudėtinga surinkti informaciją apie į branduolį siunčiamas komandas. Reikia surinkti visų procesų sąrašą ir tarp kiekvieno iš jų įsiterpti. Įsiterpimas irgi nėra paprastas, nes nesunkiai galima sugadinti arba sulėtinti programos darbą, į kurią buvo įsiterpta. Taip pat įsiterpti tarp procesų gali ir kenksmingas programinis kodas. Galima naudoti jau sukurtas programas, bet iš jų sunku ištraukti informaciją realiu laiku ir visų programų bendrai. Taip pat kai kurios programos yra mokamos. Norint išvengti paminėtų problemų buvo ieškoma alternatyvų duomenims surinkti realiu laiku.

Rašto darbo metu paruoštas duomenų rinkinys susideda iš pagal laiko intervalus sugrupuotų duomenų. Kas pasirinktą laiko intervalą yra daromas dalies sisteminių duomenų atvaizdas, kuris susideda iš:

- šiuo metu prisijungusių paskyrų kiekio **13 pav.**;
- vykstančių procesų pavadinimų (procesai su tuo pačiu pavadinimu yra agreguojami) **14 pav.**;

- nuo paskutinio atvaizdo persiųstų baitų kiekio kiekvienam susijungusiam IP adresui **15 pav.**;
- nuo paskutinio atvaizdo įvykusių sisteminių įvykių kiekio **16 pav.**

```
time,loggedoncount
2021-01-23 00:00:00,0
2021-01-23 00:00:10,0
2021-01-23 00:00:20,0
```

13 pav. Prisijungimų duomenų failo pavyzdys

```
time,name,occurences
2021-01-23 10:17:20,Idle,100
2021-01-23 10:17:20,System,100
2021-01-23 10:17:20,Secure System,100
2021-01-23 10:17:20,Registry,100
2021-01-23 10:17:20,smss,100
2021-01-23 10:17:20,csrss,200
```

14 pav. Procesų duomenų failo pavyzdys

```
time,ip,bytes
2021-01-23 10:17:20,193.200.209.12,24390
2021-01-23 10:17:20,192.168.1.9,1897341
2021-01-23 10:17:20,155.133.230.50,112
2021-01-23 10:17:20,239.255.255.250,330
2021-01-23 10:17:20,255.255.255.255,200
2021-01-23 10:17:20,216.58.215.78,119
2021-01-23 10:17:30,255.255.255.255,704
2021-01-23 10:17:30,8.8.8.8,349
2021-01-23 10:17:30,192.168.1.9,37280
```

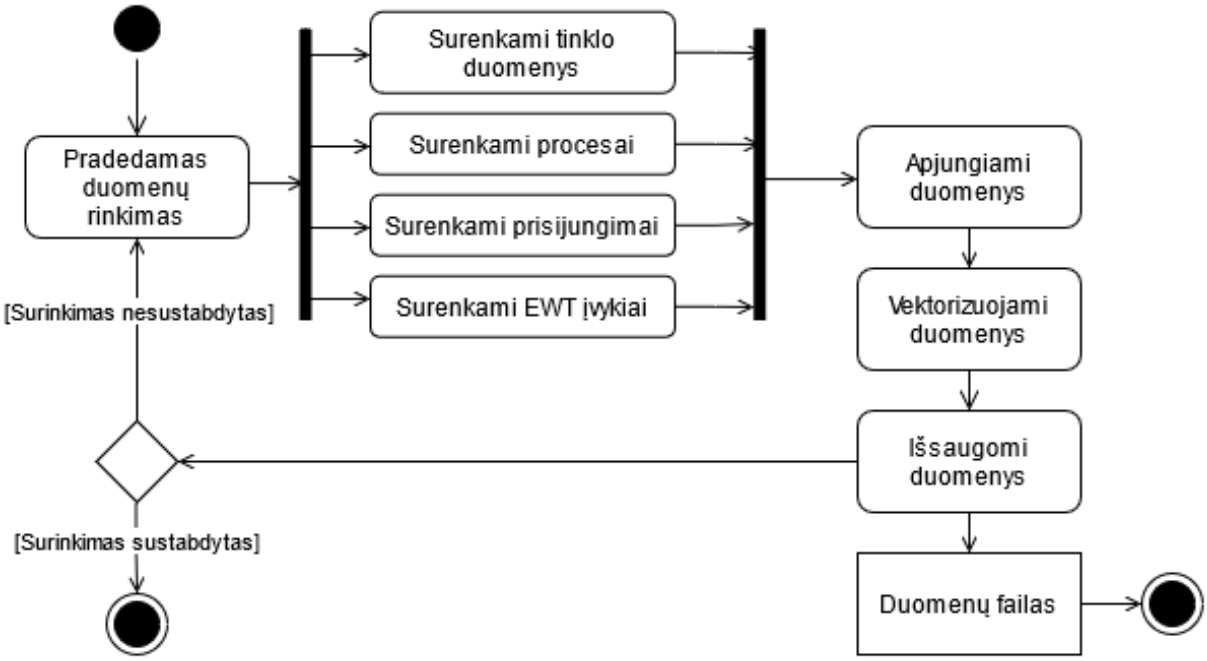
15 pav. Siunčiamų duomenų failo pavyzdys

```
time,name,occurences
2021-01-23 10:17:20,Registry/Query,121
2021-01-23 10:17:20,FileIO/Delete,26
2021-01-23 10:17:20,Memory/HardMemory,1100
2021-01-23 10:17:20,Thread/Stop,3
2021-01-23 10:17:20,VirtualMem/Alloc,831
2021-01-23 10:17:20,Registry/Close,249
2021-01-23 10:17:20,SystemConfig/Network,1
2021-01-23 10:17:20,Registry/SetInformation,11
2021-01-23 10:17:20,FileIO/MapFile,1175
2021-01-23 10:17:20,DiskIO/DriverCompleteRequest,37190
```

16 pav. Sisteminių įvykių failo pavyzdys

Duomenys realiu laiku įrašomi į 5 atskirus failus, keturi iš jų skirti skirtingiems renkamiems atributams išsaugoti. Paskutinis failas sudarytas iš vektorizuotų duomenų. Duomenų failai grupuojami į aplankus pagal surinkimo dieną, atėjus naujai dienai programa sukuria naują aplanką su naujais failais ir toliau duomenis rašo į juos.

Duomenys surinkti „Windows 10“ operacinėje sistemoje. Ši OS pasirinkta dėl to, kad „Windows“ vis dar išlieka daugiausiai naudojama operacinė sistema (NetMarketShare, 2020). Be to daug egzistuojančių IDS labiau skirtos kompiuteriams, kurie naudojami „Linux“ serveriams, o serveriai neatspindi vartotojų elgesio. Tinklo duomenys renkami pasitelkus „PCAP“ biblioteką, sisteminiams įvykiams naudojami „Windows“ OS prieinami „EWT“ įvykiai, procesai paimami tiesiogiai iš aplinkos ir prisijungimų laikai iš „Windows Event Viewer“. Įrankis kas 10 sekundžių surinktus duomenis įrašo į failus. Duomenų surinkimo procesas pavaizduotas **17 pav.**



17 pav. Duomenų surinkimo procesas

Duomenims surinkti buvo panaudoti keli scenarijai, kurie atspindi normalų vartotojo elgesį ir įsilaužimo elgesį. Sukurti scenarijai:

- Įprastas darbas.
- FTP ir SSH ataka.
- Naujo vartotojo sukūrimas.

Įprastas darbas atspindi naudotojo kasdienį elgesį, kuris gali keistis, tačiau daugiausia aktyviai naudoja kelias programas. Duomenų rinkimo metu buvo paleistos kelios programos fone: „VSCode“, „Skype“, „Steam“, „MS Word“, „Explorer“, o kelios aktyviai naudojamos: „Mozilla Firefox“, „Valheim“. Taip pat veikė įvairūs operacinės sistemos ir kiti procesai.

Naujo vartotojo sukūrimas įvykdomas tiesiog per „Windows“ konsolę. Pridedamas naujas, administratorių grupei priklausantis vartotojas.

FTP ir SSH atakos esmė prisijungti prie puolamo kompiuterio ir pasiimti kokį nors failą. Šiuo atveju bandoma aptikti ne kažkokią spragą, su kuria įsilaužta į kompiuterį, o veiksmus po įsilaužimo. Kadangi duomenys renkami ne simuliuotoje aplinkoje, nenorėta padaryti sunkiai atstatomos arba iš vis neatstatomos žalos, todėl pasirinktas paprastas scenarijus. Pirmiausia puolamame kompiuteryje paleidžiamas „FreeSHHd“ SSH serveris, tada prie tinklo prijungiamas kompiuteris, į kurį bus siunčiami pavogti duomenys, puolikas prisijungia per SSH prie puolamo kompiuterio kaip administratorius, pasiima failą ir jį per FTP nusiunčia į savo kompiuterį.

Duomenų surinkimo įrankis reikalauja pakankamai nedaug resursų: apie 40 MB operatyvinės atminties ir iki 3% Intel Core i7 6700k procesoriaus atminties. Taip pat įrankio nereikia paleisti kiekvieną kartą įjungiant kompiuterį, programa užregistruota kaip „Windows“ servisas, kuris automatiškai paleidžiamas įjungiant kompiuterį. Perkrovus kompiuterį programa be problemų tęsia duomenų surinkimą į tą patį failą.

Toliau atliekama surinktų duomenų analizė.

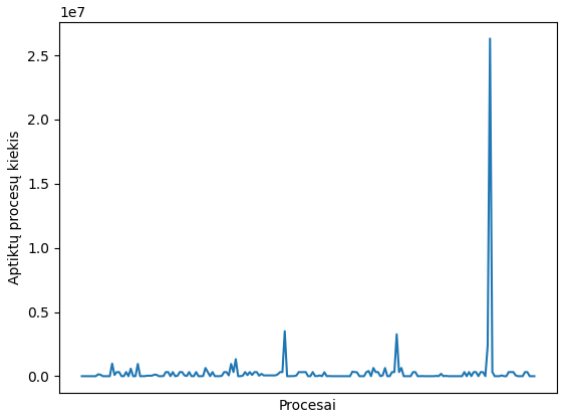
3.1.2. Duomenų analizė

Norint tinkamai apdoroti surinktus duomenis visi per pasirinktą laiko tarpą surinkti įrašai buvo sugrupuoti pagal proceso pavadinimą, sisteminį įvykį arba IP adresą, priklausomai, koks failas analizuotas.

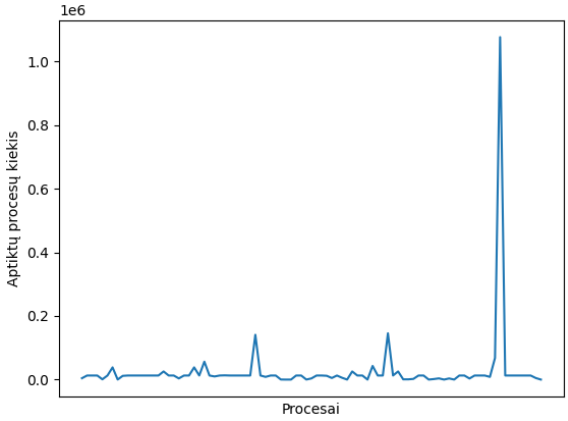
Iš viso surinkti kelių savaičių duomenys. Vienos dienos vektorizuoti duomenys susideda iš 3958 įrašų, kurio atributai yra laikas, prisijungusių paskirų skaičius, 1024 procesų duomenų kategorijų, 1024 tinklo duomenų kategorijų ir 1024 sisteminių įvykių duomenų kategorijų. Puolimo duomenų iš viso surinkta 21 min. arba 126 įrašai. Nevektorizuoti vienos dienos duomenys turi po apie 250000 įvykių, 42000 tinklo ir 270000 procesų įrašų.

Procesų duomenys

Pirmiausia patikrinti procesų duomenys. Apmokymui skirtoje dalyje unikalių procesų aptikta 195, o puolimo duomenyse aptikta 91 unikalių procesų pavadinimų. Duomenys palyginami **18 pav.** ir **19 pav.**



18 pav. Apmokymo procesų duomenys



19 pav. Puolimo procesų duomenys

Vizualiai apmokymo kartu su puolimo duomenimis atrodo panašiai ir didelių skirtumų akimi pamatyti neišeina. Matosi tik, kad puolimo duomenyse procesų mažiau ir jie pasikartoja rečiau, bet tai dėl žymiai trumpesnio laiko, iš kurio paimti duomenys. Taip pat matosi, kad kai kurios programos truputi iškreipia duomenis, nes turi daug procesų su tuo pačiu pavadinimu, bet renkant duomenis procesų kiekis vienai programai irgi yra svarbu, nes kai kurios kenksmingos programos veikia sukurdamos daug gijų arba savo kopijų. Taip pat kenksmingos programos gali bandyti pasislėpti pasivadinant taip pat kaip nekenksminga programa.

2 lentelė. 5 rečiausi ir dažniausi procesai

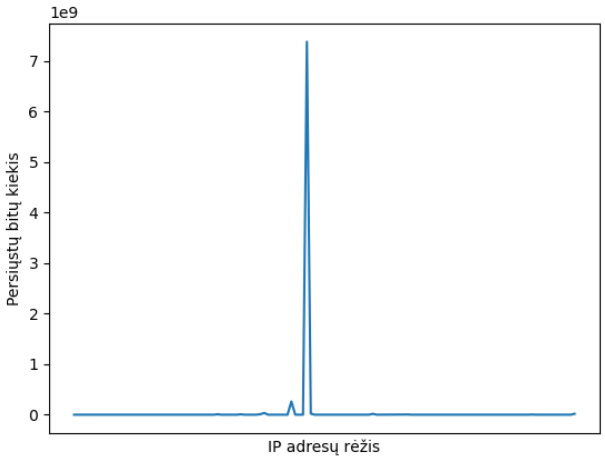
Apmokymo duomenys		Puolimo duomenys	
Pavadinimas	Kiekis	Pavadinimas	Kiekis
OriginLegacyCLI	1	SystemSettingsAdminFlows	1
DDConfigCA	1	default-browser-agent	2
DropboxCrashHandler	1	pingsender	10
schtasks	2	msoia	12
UsoClient	3	wodOther	30
RuntimeBroker	1324016	RuntimeBroker	56286
steamwebhelper	2386682	steamwebhelper	68000
firefox	3266004	Skype	140800

Skype	3505570	firefox	145535
svchost	26310752	svchost	1076675

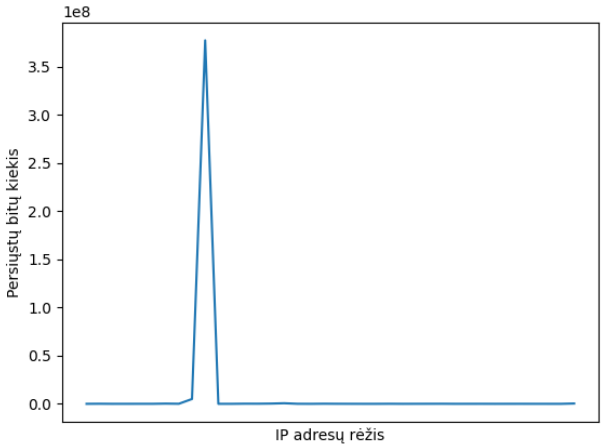
Iš **2 lentelės** matosi, kad dažniausiai pasikartojantys procesai yra tie, kurie turi daug gijų, rečiausių procesų neieškant internete, kas jie yra, neišeina atpažinti.

Tinklo duomenys

Antri analizuoti duomenys – tinklo. Apmokymui skirtoje dalyje unikalių IP adresų aptikta 7934, o puolimo duomenyse aptikta 144 unikalių IP adresų. Kadangi IP adresų labai daug, diagramoje vaizduojami IP adresai sugrupuoti pagal pirmą skaičių, pavyzdžiui, 192.*.*.*. Tinklo duomenys palyginami **20 pav.** ir **21 pav.**



20 pav. Apmokymo tinklo duomenys



21 pav. Puolimo tinklo duomenys

Su tinklo duomenimis matomas vienas skirtumas, kad puolimo duomenys neturi antro iškylimo, yra tik vienas. Abiejose lentelėse vienas IP adresų rėžis smarkiai iškreipia duomenis.

3 lentelė. 5 rečiausi ir dažniausi tinklu siunčiami duomenys

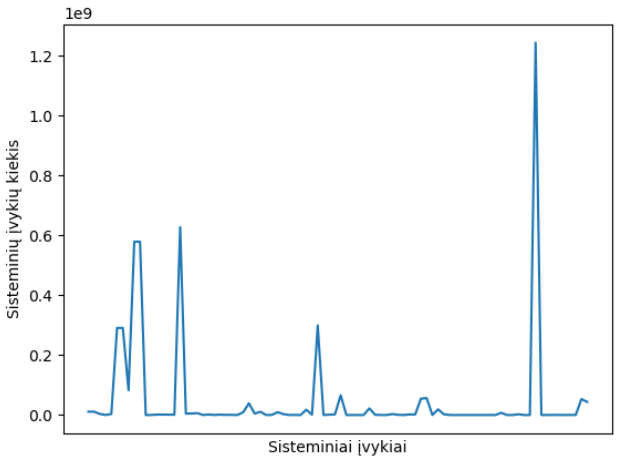
Apmokymo duomenys	Puolimo duomenys

IP	Kiekis	Paaiškinimas	IP	Kiekis	Paaiškinimas
185.179.203.234	36B	i3D.net	13.89.202.241	48B	Microsoft
188.122.64.138	36B	i3D.net	31.13.72.12	223B	Facebook
188.122.83.22	36B	i3D.net	44.239.41.173	237B	Amazon
188.122.65.134	36B	i3D.net	172.217.20.206	285B	Google
213.179.210.236	36B	i3D.net	216.58.215.118	342B	Google
173.194.164.188	14,69MB	Google	216.58.208.206	0,53MB	Google
99.181.65.53	16,11MB	Twitch	185.42.205.24	4,78MB	Twitch
185.252.108.67	0,23GB	Tikriausiai interneto tiekėjas	192.168.1.7	4,8MB	Užpuoliko IP
192.168.1.9	3,09GB	Kompiuterio IP	192.168.1.11	8,24MB	Kompiuterio IP
192.168.1.11	3,79GB	Kompiuterio IP	192.168.1.9	0,39GB	Kompiuterio IP

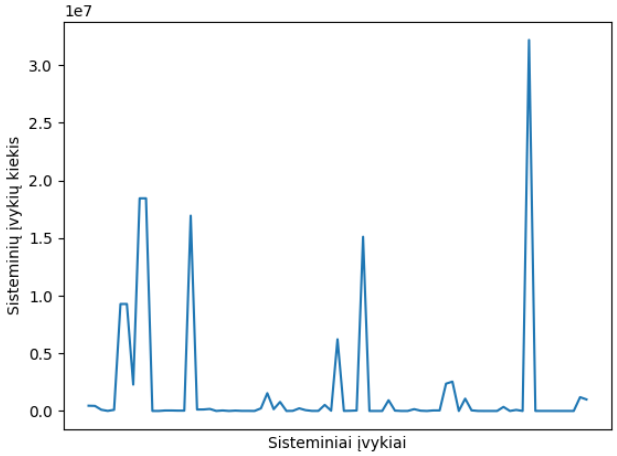
3 lentelėje pirmiausia aiškiai matosi, lokalaus kompiuterio du skirtingi IP adresai, kadangi duomenys iš skirtingų dienų, IP adresas spėjo pasikeisti. Lokalus IP adresas turi tokį didelį kiekį baitų, nes visi paketai ateinantys į kompiuterį skaičiuojami kaip į tą patį adresą, o paketai einantys iš kompiuterio turi įvairius IP adresus. Taip pat aiškiai matomas užpuoliko IP adresas ir jam siunčiamas pavogtas failas.

Sisteminių įvykių duomenys

Treti analizuoti duomenys – sisteminių įvykių. Apmokymui skirtoje dalyje unikalių įvykių aptikta 88, o puolimo duomenyse aptikta 79 unikalių įvykių. Įvykių duomenys palyginami **22 pav.** ir **23 pav.**



22 pav. Apmokymo tinklo duomenys



23 pav. Puolimo tinklo duomenys

Kaip ir su kitais duomenimis įvykių puolimo ir apmokymo duomenys vizualiai beveik nesiskiria. Puolimo duomenys turi vieną dažniau pasitaikantį įvykį, bet taip gali atrodyti ir dėl to, kad puolimo duomenų žymiai mažiau.

4 lentelė. 5 rečiausi ir dažniausi sisteminiai įvykiai

Apmokymo duomenys		Puolimo duomenys	
Pavadinimas	Kiekis	Pavadinimas	Kiekis
TcpIp/ConnectIPV6	3	TcpIp/Retransmit	3
TcpIp/AcceptIPV6	4	Registry/Flush	6
TcpIp/SendIPV6	10	TcpIp/Accept	13
LostEvent	11	Registry/ QueryMultipleValue	23
Memory/VirtualRotate	12	UdpIp/SendIPV6	71
Memory/DemandZeroFault	298942647	Memory/TransitionFault	15114918
DiskIO/ DriverMajorFunctionReturn	578776049	Dispatcher/ReadyThread	16943960
DiskIO/ DriverMajorFunctionCall	578776074	DiskIO/ DriverMajorFunctionCall	18447850
Dispatcher/ReadyThread	627542399	DiskIO/ DriverMajorFunctionReturn	18447852
Thread/CSwitch	1243296440	Thread/CSwitch	32185688

4 lentelėje dažniausių įvykių vienintelis skirtumas – tarp „Memory/TransitionFault“ ir „Memory/DemandZeroFault“, tačiau pažiūrėjus vien į pavadinimą, kažko neįprasto nesimato. Vėliau patikrinus modelį paaiškėjo, kad iš įvyko duomenų įsilaužimo pamatyti neišeina, įvykiai per daug bendriniai. Renkant duomenis antrą kartą galima būtų bandyti grupuoti įvykius pagal procesų pavadinimus. Taip būtų gaunami detalesni duomenys, bet taip pat būtų daugiau unikalių duomenų ir mašininio mokymosi algoritmas būtų labiau apkraunamas.

Prieš pradedant mokyti modelį su naudotojo duomenimis, reikia juos paruošti.

3.1.3. Duomenų paruošimas

Didžiausias iššūkis su tokiais duomenimis kaip komandos ir jų argumentai – iš anksto nežinomas galimų variantų rinkinys. Dauguma gilaus mokymosi algoritmų tikisi gauti fiksuoto dydžio duomenų vektorius. Šiuo atveju tai yra sudėtinga, nes į stebimą sistemą labai paprasta įrašyti naujus, dar nematytus įrankius. Galima būtų bandyti apriboti sistemoje naudojamų komandų kiekį, tačiau tai nėra labai geras sprendimas, nes vis tiek reikėtų visiškai iš naujo permokyti modelį pasikeitus leidžiamų įrankių kiekiui. Taip pat įrankių gali būti labai daug, kas reiškia daug vektoriaus dimensijų. Tai gali sulėtinti duomenų apdorojimo bei mokymosi greitį ir padidinti naudojamos atminties kiekį. Galiausiai įrankiai turi begalę argumentų, kurie gali turėti nenuspėjamas reikšmes. Dėl to vienas iš sprendimų yra tekstą apdoroti su vienkrypte funkcija, kurios rezultato galimos reikšmės apribotos iki nustatyto intervalo. Angliškai šis metodas vadinamas „feature hashing“ arba „hashing trick“, kurio veikimo principo pseudokodas pavaizduotas **24 pav.**

```
function hashing_vectorizer(features : array of string, N : integer):
     x := new vector[N]
     for f in features:
         h := hash(f)
         x[h mod N] += 1
     return x
```

24 pav. "feature hashing" pseudokodas

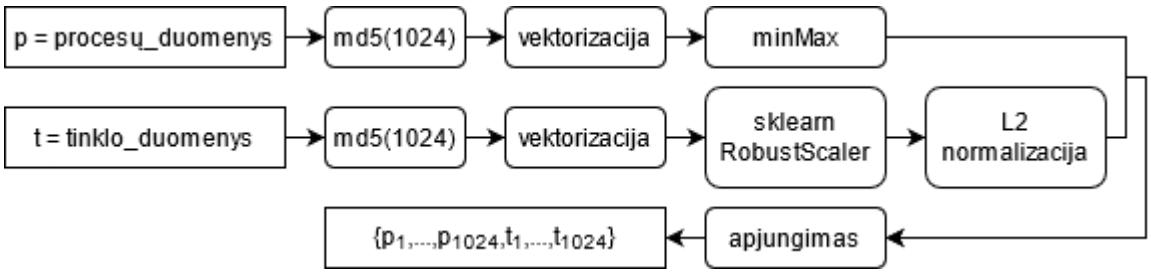
Viena iš didžiausių „hashing trick“ problemų yra kolizijos. Įvairių žodžių arba mūsų atveju komandų bei jų argumentų gali būti labai daug, todėl keliems skirtingiems žodžiams gali būti priskiriamos tos pačios reikšmės. Dėl to reikia parinkti tokį vienkryptės funkcijos rezultato intervalą, kad kolizijų būtų kuo mažiau, tačiau tuo pačiu intervalas neturėtų būti per didelis. Didelis intervalas pailgina modelio mokymosi procesą bei išnaudoja daugiau nei reikia atminties. Taip pat galima parinkti kuo geresnę vienkryptę funkciją, kuri tolygiau paskirsto gaunamas reikšmes.

Anomalijoms aptikti iš savo rinkinio paimti tik veikiančių procesų ir tinklo duomenys. Bandant prototipą paaiškėjo, kad prisijungimų duomenys bei EWT įvykiai anomalijų aptikimo rezultatus tik pablogino. Tekstiniai duomenys apdoroti su „MD5“ vienkrypte funkcija, kuri grąžina sveikąjį skaičių nurodytame rėžyje. Rezultatai panaudojus modulį apriboti iki tam tikro rėžio, šiuo atveju abu požymiai apriboti iki reikšmės nuo 0 iki 1024. Jei vektorizacijos metu atrasta kolizijų, reikšmės susumuotos kartu. Po apdorojimo su „MD5“ visi laukai tapo kategoriniais, todėl juos paprasta vektorizuoti, tiesiog reikia kiekvienai kategorijai sukurti po naują dimensiją. Vektorizavus požymius sekantis žingsnis – vektorius normalizuoti. Procesų pavadinimų duomenims gautas reikšmes užtenka suspausti į intervalą tarp 0 ir 1. Tinklo duomenims reikia kitokio apdorojimo, nes kiekvienam IP adresui persiunčiama labai skirtingi kiekiai duomenų, skirtumai tarp skirtingų reikšmių gali būti tūkstančiais kartų. Pirmiausia duomenys perskirstomi pagal 25% ir 75% kvantilius be centravimo pasinaudojus „Sklearn RobustScaler“, perskirstyti duomenys tada normalizuojami pasitelkiant L2 arba kitaip Euklido normą:

$$L^2 = \sqrt{\sum_{i=1}^{n} {x_i}^2} \tag{3.1}$$

čia L^2– gauta norma, x_i – vektoriaus dimensijos reikšmė.

Galiausiai visi požymiai yra apjungiami į tą pačią matricą grupuojant pagal laiką. Duomenų įrašo apdorojimo diagrama parodyta **25 pav.**



25 pav. Duomenų apdorojimo diagrama

Apdorojus visus duomenis gaunama matrica, kuri sudaryta iš *n* eilučių ir *m* stulpelių, kur *n* – įrašų kiekis duomenų rinkinyje, o *m* – apdorotų požymių gautų stulpelių suma, kuri šiuo atveju 2048.

Toliau aprašomas pasirinktas anomalijų aptikimo metodas.

3.2. Anomalijos aptikimo metodas

Anomalijoms aptikti siūloma gilųjį naudoti neuroninį tinklą – LSTM autoenkoderį. Šio tipo neuroninis tinklas jau sėkmingai panaudotas tinkle veikiančioje IDS, dėl to galima daryti prielaidą, kad šis metodas turėtų pasiteisinti ir kompiuterį analizuojančioje IDS (Meng, Lou, Fu, & Tian, 2018).

Kadangi duomenis apie vartotojo ir sistemos darbą galima apdoroti ir iš jų gauti skaitinių reikšmių vektorių, reiškia, kad įsilaužimo aptikimui galima naudoti neuroninius tinklus ir kitus mašininio mokymosi algoritmus.

Prižiūrimų ar iš dalies prižiūrimų mašininio mokymosi algoritmų naudoti įsilaužimų aptikimui nelabai galima, nes įsilaužimų būdai bei sistemų spragos būna labai įvairios ir iš anksto jų visų žinoti neįmanoma. Taip pat nuolat atrandama naujų spragų bei įsilaužimo būdų, tai reiškia, kad su klasifikuotais duomenimis apmokytas modelis sugebėtų aptikti tik jau matytus įsilaužimus. Dėl šių priežasčių pasirinktas LSTM autoenkoderis anomalijoms aptikti. Jis gan populiarus ir turi palaikymą daugelyje programavimo kalbų ir bibliotekų. Pasirinktas LSTM autoenkoderio variantas, nes jis labiau tinka sekoms ir periodiškiems duomenims.

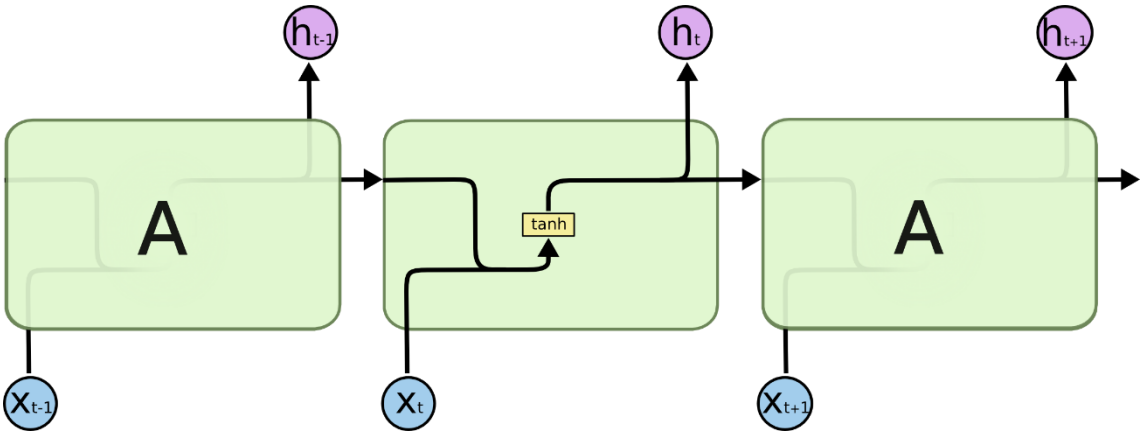
3.2.1. LSTM autoenkoderis

Ilgalaikės trumpalaikės atminties autoenkoderis, toliau LSTM, yra autoenkoderio variantas, kurio giluminiai sluoksniai turi atmintį. LSTM yra grįžtamojo ryšio neuroninio sluoksnio tipas, kuris sugeba susieti ne tik artimas laike reikšmes, bet ir tolimas (Olah, 2015). Skirtumai tarp LSTM ir paprasto grįžtamojo ryšio sluoksnio pavaizduoti **26 pav.** ir **27 pav.**, esminis skirtumas tas, kad LSTM turi du lygiagrečiai einančius duomenų srautus kartu su užmaršties bei įsiminimo galimybėmis, kurios veikia pasitelkus sigmoidines funckijas **σ**:

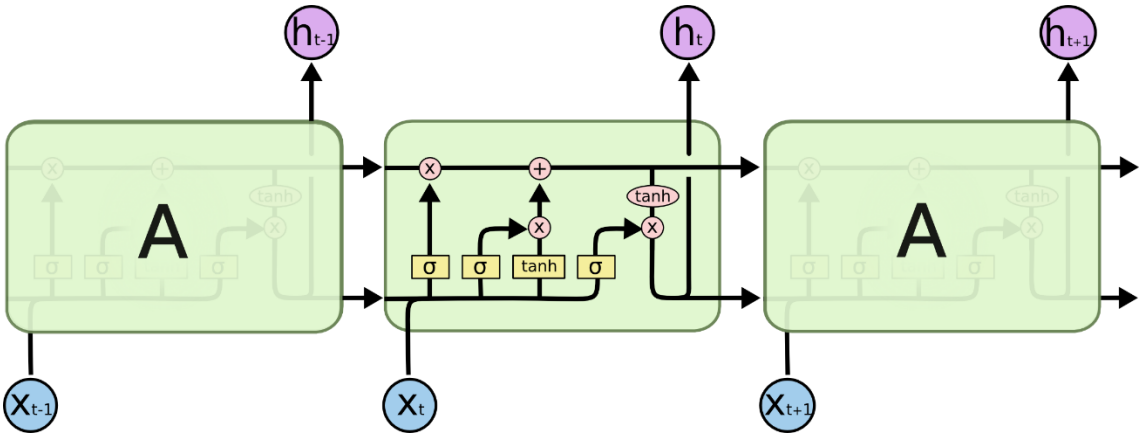
$$\sigma = \frac{1}{1 + e^{-x}} \tag{3.2}$$

čia σ – sigmoidinės funkcijos rezultatas, x – funkcijos argumentas.

LSTM autoenkoderis parinktas dėl to, kad vartotojo atliekami veiksmai dažnai priklauso nuo vienas kito, be to reikalinga ilgalaikė atmintis, nes kai kurie veiksmai yra periodiniai.



26 pav. Grįžtamojo ryšio sluoksnio veikimo principas (pagal Olah, 2015)



27 pav. LSTM sluoksnio veikimo principas (pagal Olah, 2015)

Matematiškai LSTM mazgą galima išreikšti šiomis formulėmis:

$$f_t = \sigma_g\left(W_t x_t + U_f h_{t-1} + b_f\right) \quad (3.3)$$

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \quad (3.4)$$

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \quad (3.5)$$

$$\tilde{c}_t = \sigma_g(W_c x_t + U_c h_{t-1} + b_c) \quad (3.6)$$

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \quad (3.7)$$

$$h_t = o_t * \sigma_h(c_t) \quad (3.8)$$

čia b_q – nuokrypio vektorius, W_q ir U_q yra tinklo svorių matricos, kur q yra arba i įvesties vartai, arba o išvesties vartai, arba f užmaršties vartai, arba c atminties ląstelė. Taip pat x_t yra įvesties į LSTM vektorius, f_t – užmaršties vartų aktyvacijos vektorius, i_t – įvesties/atnaujinimo vartų aktyvacijos vektorius, o_t – išvesties vartų aktyvacijos vektorius, $\tilde{c}_t$ – ląstelės įvesties aktyvacijos vektorius, c_t – ląstelės būsenos vektorius, h_t – LSTM išvesties vektorius.

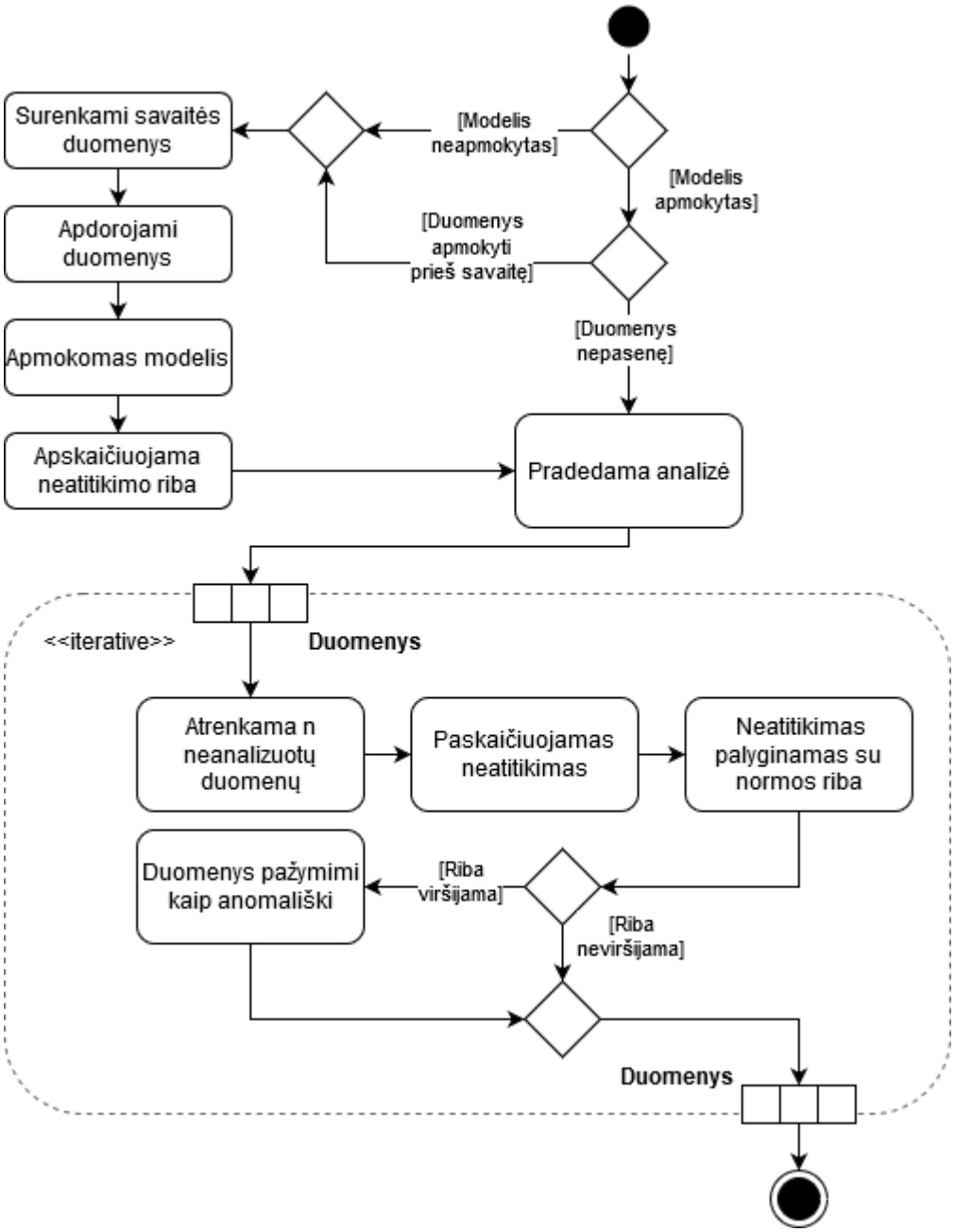
Norint aptikti elgesio anomalijas LSTM autoenkoderis pirmiausia turi būti apmokomas su įprastu vartoto elgesiu ar sistemos darbu. Turėtų užtekti surinkti kelių dienų duomenis, jų klasifikuoti nebūtina, nes tikimybė, jog įsilaužimas sutaptų su duomenų surinkimu, nedidelė. Apmokymas trunka iki kelių minučių ir gali būti atliekamas su procesoriumi, tikslus laikas priklauso nuo konkretaus įrenginio. Su vaizdo plokšte apmokymas trumpesnis, tačiau ne visi kompiuteriai turi atskiras vaizdo plokštes. Į apmokytą LSTM autoenkoderį toliau paduodami naujai realiu laiku renkami duomenys, modelis juos apdoroja ir atstatytą duomenų vektorių lygina su originaliu. Jei skirtumas tarp atstatyto duomenų vektoriaus ir originalo yra didesnis nei tikimasi, duomenų įrašas ar jų grupė pažymima kaip įtartina.

Vien neuroninio tinklo įsilaužimams aptikti neužtenka, reikia į procesą įtraukti ir duomenų surinkimą. Toliau aprašomas pilnas įsilaužimo aptikimo metodas.

3.2.2. Metodo veikimo principas

Surenkami savaitės vieno vartotojo duomenys, gauti įrašai paverčiami į duomenų vektorius, kurių visos reikšmės skaitinės. Tada treniruojamas modelis su savaitės duomenimis. Iš dalies normalaus elgesio duomenų apskaičiuojama įtartinų įvykių neatitikimo riba. Ši riba bus naudojama norint patikrinti, ar įvykiai įtartini, ar ne.

Pasibaigus modelio apmokymui arba jau turint iš anksto apmokytą modelį prasideda duomenų analizė. Duomenis grupuojami po *n* įrašų ir neatitikimai skaičiuojami bendrai visai grupei, o ne kiekvienam įrašui atskirai. Jei grupės neatitikimo vidurkis viršija normalių duomenų neatitikimo ribą, reiškia tarp grupėje esančių įrašų yra įtartinų veiksmų. Metodo veikimo diagrama atvaizduota **28 pav.**



28 pav. Anomalijų aptikimo veikimo diagrama

3.2.3. Naudojamos technologijos

Analitiniam moduliui įgyvendinti naudojama „Python“ programavimo kalba dėl savo didelės mašininiui mokymuisi ir duomenų mokslui skirtų bibliotekų įvairovės.

Pagrindinės naudojamos bibliotekos: „Tensorflow“, „Pandas“, „Sklearn“ ir „Numpy“.

„Tensorflow“ yra mašininio mokymosi biblioteka, kuri padeda lengvai sumodeliuoti, apmokyti ir įvertinti neuroninius tinklus bei kitus modelius. Ši biblioteka taip pat gali dirbti su kompiuterio vaizdo plokšte, o tai suteikia greitesnį modelių apmokymą.

„Pandas“ padeda paprastai pakrauti, paruošti ir analizuoti duomenis, kurių reikia apmokant mašininio mokymosi modelį.

„Sklearn“ turi įvairius metodus duomenų apdorojimui ir normalizavimui.

„Numpy greičiui parašyta biblioteka, kuri leidžia atlikti įvairias operacijas su vektoriais ir matricom. Didelis efektyvumas pasiekiamas dėl to, kad didžioji dalis bibliotekos parašyta su C kalba.

3.3. Rezultatų analizė

3.3.1. Metodo įvertinimas su surinktais duomenimis

Modelio efektyvumas tikrinamas su savais duomenimis. Rezultatų palyginimui parinkti trys skirtingi autoenkoderiai: paprastas, variacinis ir LSTM. Taip pat su geriausiai pasirodžiusiu modeliu išbandytas „ADFA-LD“ duomenų rinkinys.

Efektyvumui įvertinti naudojamas: aptikimo dažnio ir klaidingų pranešimų dažnio santykis ir MCC rodiklis. Šie rodikliai paskaičiuojami gaunant maišos matricą (angl. *confusion matrix*). Maišos matrica susideda iš keturių dalių: teisingai teigiama, klaidingai teigiama, teisingai neigiama ir klaidingai neigiama (*angl. „True Positive“*, toliau TP, *„False Positive“*, toliau FP, *„True Negative“*, toliau TN, *„False Negative“*, toliau FN). Kaip sudaroma maišos matrica, pavaizduota **5 lentelėje.**

5 lentelė. Maišos matricos sudarymas

Nuspėtas įvykio tipas / Tikras įvykio tipas	Kenksmingas įvykis	Nekenksmingas įvykis
Kenksmingas įvykis	TP	FN
Nekenksmingas įvykis	FP	TN

Iš maišos funkcijos gaunamas aptikimo dažnio ir klaidingų pranešimų dažnio santykis:

$$DR = \frac{TP}{TP + FN} \tag{3.9}$$

$$FAR = \frac{FP}{FP + TN} \tag{3.10}$$

čia DR – aptikimo dažnis, FAR – klaidingų pranešimų dažnis. Ši diagrama dažnai naudojama kituose straipsniuose įvairiems IDS modeliams įvertinti. Taip pat iš maišos funkcijos gaunamas Matthews koreliacijos koeficientas (angl. *Matthews corellation coeficcient*, toliau MCC):

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{3.11}$$

MCC skirtas modelio efektyvumui įvertinti ir siūlomas naudoti vietoj dažnai matomo F1 rodiklio, nes MCC atsižvelgia į klaidingų pranešimo dažnį, ne tik į aptikimo dažnį.

Kiekvienam rodikliui apskaičiuojama po 50 reikšmių pasirinktame ribų intervale. Riba tai nuostolio funkcijos reikšmė, pagal kurią nustatoma, ar įrašas įtartinas, ar ne. Intervalas parenkamas pasinaudojus:

$$l_{min} = l - \frac{l}{2} \tag{3.12}$$

$$l_{max} = l + \frac{l}{2} \tag{3.13}$$

čia l_{min} – intervalo žemutinė riba, l_{max} – intervalo aukštutinė riba, l – apmokyto modelio nuostolio funkcijos reikšmė.

Bandymai atlikti su stacionariu kompiuteriu, kuris turi:

- „Intel Core i7-6700K“ 4.00GHz procesorių su 4 branduoliais
- „Nvidia GTX 970“ vaizdo plokštę
- 8 GB 2667 MHz operatyvinės atminties
- SSD pastovią atmintį

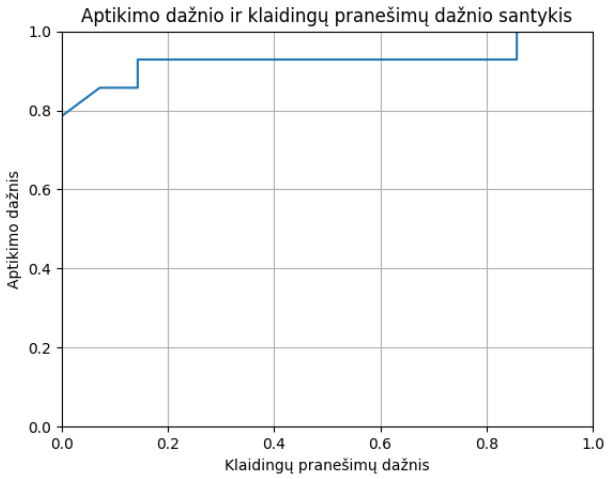
Pasirinkto anomalijų aptikimo metodui patikrinti pirmiausia suprogramuotas LSTM autoenkoderio prototipas. Jis susideda iš 2 LSTM sluoksnių enkoderyje, pasikartojančio sluoksnio ir 2 LSTM sluoksnių dekoderyje. LSTM sluoksnių dydžiai iš eilės: 124, 16, 16, 16, 124. Visi sluoksniai naudoja „ReLU“ aktyvacijos funkciją, kuri tiesiog nepraleidžia reikšmių mažesnių už 0. Svorių optimizacijai mokymosi metu naudojamas „Adam“ algoritmas. Neatitikimui tarp pradinio ir atstatyto vektoriaus apskaičiuoti naudojama vidutinė kvadratinė paklaida (angl. *mean sqared error*, toliau MSE):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \tilde{Y}_i)^2 \tag{3.14}$$

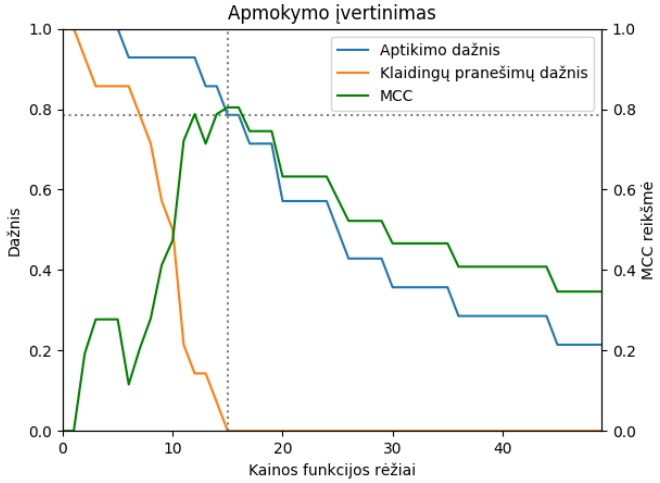
čia n spėjimų kiekis, Y – originalių reikšmių vektorius, $\tilde{Y}$ – spėjamų reikšmių vektorius. MSE naudojama kaip nuostolio funkcija.

Modelis apmokytas pasirinkus 50 epochų ir partijos dydį – 256. Apmokymui naudojami 2 dienų įprasto elgesio duomenys. Aptikimo tikslumo palyginimui naudojami trečios dienos duomenys, kurių paimta tiek pat, kiek puolimo duomenų, o modelio įvertinimui panaudoti visi puolimo duomenys.

Pirmiausia tikrinamas LSTM autoenkoderis ir jo rezultatai pavaizduoti **29 pav.** ir **30 pav.**



29 pav. LSTM aptikimo dažnio ir klaidingų pranešimų dažnio santykis kai grupuojama po 9 įrašus

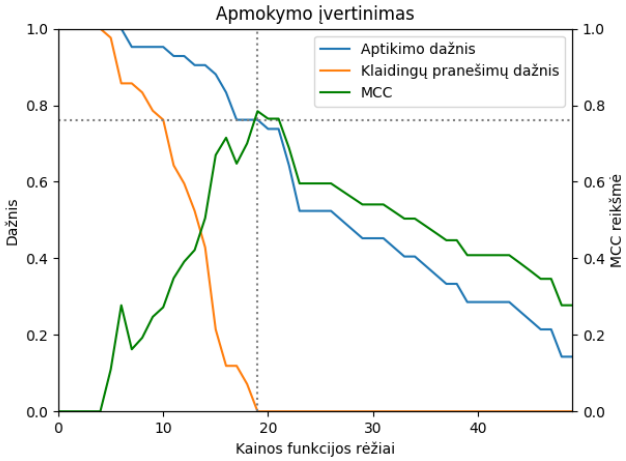


30 pav. LSTM apmokymo įvertinimas kai grupuojama po 9 įrašus

Geriausiai pasirodė LSTM autoenkoderis, kuriam duodami įrašai sugrupuoti po 9. Šiuo būdu pasiektas 78,57% aptikimo dažnis su 0 neteisingai pažymėtų įrašų grupių. Tačiau kadangi sugrupavus duomenis jų gavosi nedaug (14 sugrupuotų puolimo duomenų įrašų), gautų rezultatų patikimumas atitinkamai mažesnis.

Palyginimui modelis išbadytas ir su mažesnėmis grupėmis. LSTM autoenkoderio, kuris naudoja po 3 įrašus rezultatai pavaizduoti **32 pav.** ir **33 pav.**

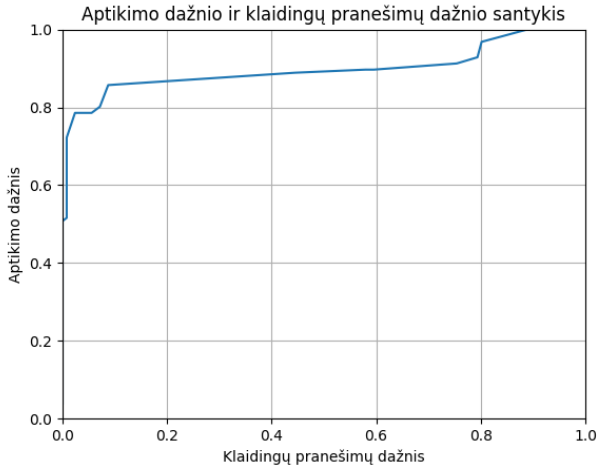
31 pav. LSTM aptikimo dažnio ir klaidingų pranešimų dažnio santykis kai grupuojama po 3 įrašus



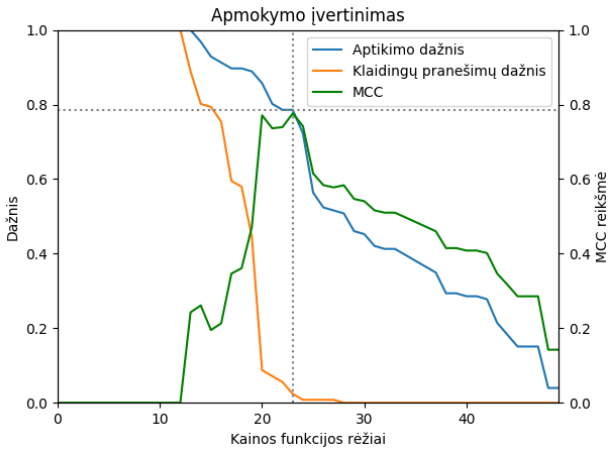
32 pav. LSTM apmokymo įvertinimas kai grupuojama po 3 įrašus

LSTM su 3 įrašų grupėmis vietoj 9 pasirodė prasčiau, tačiau vis dar pasiekė nulinį klaidingų pranešimų dažnį. Šiuo būdu pasiektas 76,19% aptikimo dažnis. Grupuojant įrašus į grupes po 3 gautos 42 įrašų grupės.

Galiausiai LSTM išbandytas su negrupuotais duomenimis, rezultatai pavaizduoti **33 pav.** ir **34 pav.**



33 pav. LSTM aptikimo dažnio ir klaidingų pranešimų dažnio santykis kai įrašai negrupuojami

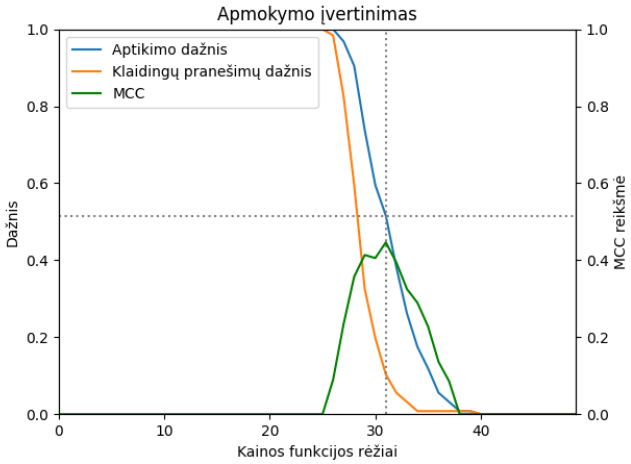


34 pav. LSTM apmokymo įvertinimas kai įrašai negrupuojami

Iš LSTM be grupuotų duomenų gaunami panašūs rezultatai kaip ir su grupuotais duomenimis, tačiau klaidingų pranešimų dažnis nebe nulinis. Geriausiu atveju pasiektas 78,57% aptikimo dažnis su 2,38% klaidingų pranešimų dažniu. Iš grafiko matosi, kad pasiektas ir nulinis klaidingų pranešimų dažnis, tačiau tuo atveju aptikimo dažnis tik apie 60%.

Taip pat norint patikrinti pasirinktą modelį, duomenys išbandyti su kitais giluminio mokymosi algoritmais. Pirmiausia išbandytas variacinis autoenkoderis su panašiais parametrais kaip ir LSTM. Pagrindinis skirtumas yra atsitiktinumo duodantis sluoksnis pačiame tinklo viduryje ir reguliarizacijos funkcija. Taip pat variacinis autoenkoderis naudoja paprastus sluoksnius enkoderyje ir dekoderyje, o ne LSTM. Gauti rezultatai pavaizduoti **35 pav.** ir **36 pav.**

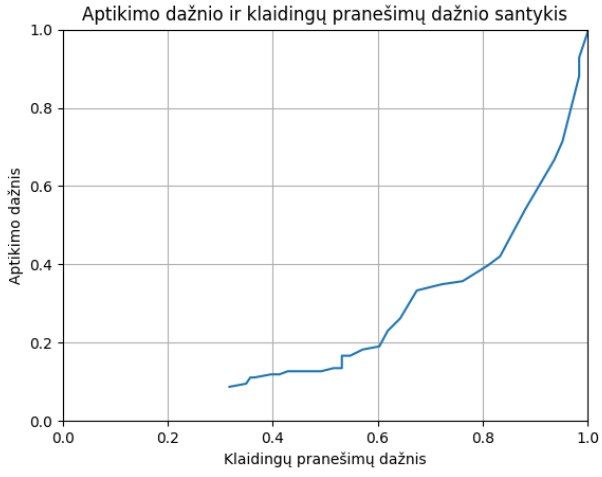
35 pav. VAE aptikimo dažnio ir klaidingų pranešimų dažnio santykis



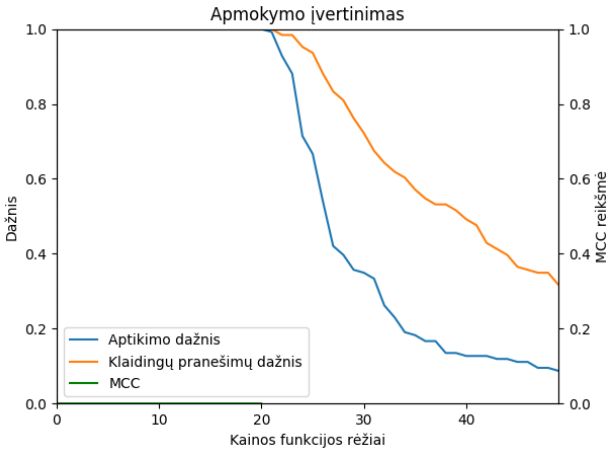
36 pav. VAE apmokymo įvertinimas

Variacinis autoenkoderis pasirodė prasčiau negu LSTM autoenkoderis, geriausiu atveju pasiektas 51,59% aptikimo dažnis ir 10,32% klaidingų pranešimų dažnis. Pažvelgus į grafiką intervalas, kuriame santykis tarp aptikimo ir klaidingų pranešimo dažnis yra ganėtinai mažas, tad praktiškai šį giluminio mokymosi metodą būtų sudėtinga panaudoti. Tačiau VAE turi daugiau parametrų, kuriuos galima modifikuoti ir potencialiai galima būtų pasiekti didesnį aptikimo dažnį. Taip pat galima būtų bandyti LSTM sluoksnius panaudoti VAE.

Galiausiai išbandytas įprastas autoenkoderis, gauti rezultatai pavaizduoti **37 pav.** ir **38 pav.**



37 pav. Paprasto autoenkoderio aptikimo dažnio ir klaidingų pranešimų dažnio santykis

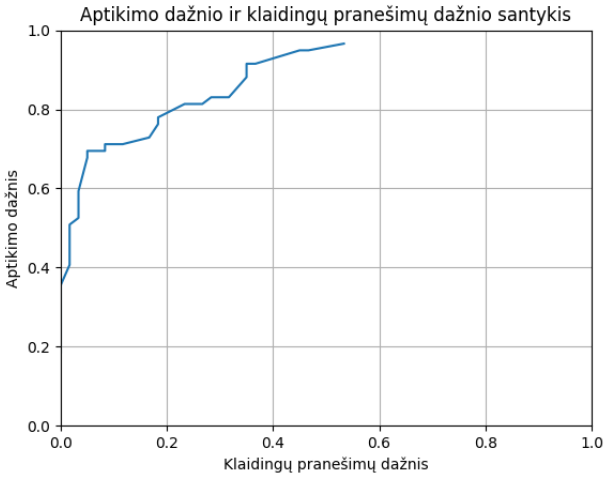


38 pav. Paprasto autoenkoderio apmokymo įvertinimas

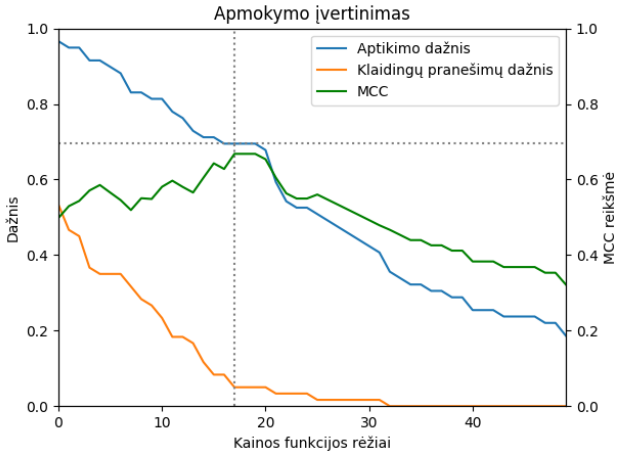
Paprastas autoenkoderis visiškai nesugeba taikliai nustatyti anomalių įrašų. Šio modelio efektyvumas prastesnis nei atsitiktinai įrašų normalumą spėjančios sistemos.

3.3.2. Metodo įvertinimas su surinktais duomenimis

Norint palyginti šiame darbe surinktus duomenis, geriausiai pasirodęs modelis – LSTM autoenkoderis, išbandytas su „ADFA-LD“ duomenų rinkiniu. Su šiuo rinkiniu duomenys nebuvo grupuojami, nes duomenų rinkinys aiškiai suklasifikuotas ir laiko žymių neturi. Gauti rezultatai pavaizduoti **39 pav.** ir **40 pav.**



39 pav. LSTM aptikimo dažnio ir klaidingų pranešimų dažnio santykis su „ADFA-LD“



40 pav. LSTM apmokymo įvertinimas su „ADFA-LD“

Su „ADFA-LD“ pasiektas 69,49 % aptikimo dažnis su 5% klaidingų pranešimų dažniu. Tai prastesnis rezultatas nei gautas su šiame darbe surinktais duomenimis, tačiau modelis atskirai lyginamiems duomenims nebuvo derinamas, todėl surinktų duomenų rezultatai gali būti truputi geresni, nei yra iš tikrųjų. Taip pat „ADFA-LD“ duomenys turi didesnę puolimo scenarijų įvairovę. Tačiau „ADFA-LD“ duomenis gan sudėtinga surinkti. Netrikdant sistemos darbo, foniniame procese be didelių pastangų jų surinkti neišeitų, ypač „Windows“ OS.

3.4. Skyriaus apibendrinimas

Šiame skyriuje aprašyti naudojami duomenų rinkiniai: surinktas atskirai ir viešas „ADFA-LD“. Aprašytas duomenų paruošimas: tekstiniai duomenys apdirbami su vienkrypte „MD5“ kriptografine funkcija, kurios rezultatas apribotas iki pasirinkto rėžio.

Taip pat aprašytas siūlomas anomalijų aptikimo modelis – LSTM autoenkoderis. Autoenkoderis tikrina atkurtą vektorių su originaliu ir lygina nuokrypį. Jei nuokrypis pakankamai didelis, reiškia įrašas yra įtartinas. LSTM autoenkoderis nuo įprasto skiriasi tuo, kad turi atmintį ir geriau tinka duomenims, kurie turi laiko žymes.

Galiausiai atliktas eksperimentas su trimis skirtingais modeliais ir dviem skirtingais duomenų rinkiniais bei pateikti siūlomo modelio rezultatai. Paaiškėjo, kad geriausiai įtartinam elgesiui aptikti tinka naudoti LSTM autoenkoderis kai įrašai sugrupuojami po 9. Geriausiu atveju gautas 78,57% aptikimo dažnis su 0 neteisingai pažymėtų įrašų grupių. Surinkti duomenys pasirodė geriau negu „ADFA-LD“ duomenų rinkinys, tačiau rezultatai gan panašus ir skirtumą galėjo lemtai tai, kad modelis pritaikytas būtent saviems duomenims. Taip pat savi duomenys turi gan primityvius puolimo scenarijus, kuriuos galima būtų aptikti nenaudojant mašininio mokymosi. Bet šis duomenų surinkimo būdas parodė, kad galima surinkti bent dalį reikalingų duomenų neeikvojant daug kompiuterio resursų.

4. Išvados

Atlikus susijusios tyrimo srities mokslinės literatūros analizę nustatyta, kad įsilaužimams aptikti geriausiai tiktų kompiuterį analizuojanti sistema, kuri veikia nuo normos nukrypusio elgesio aptikimo principu.

Apžvelgus mašininio mokymosi algoritmus bei naujausias jų tendencijas paaiškėjo, kad įsilaužimams aptikti geriausi rezultatai turėtų būti gaunami naudojant giluminius neuroninius tinklus.

Atlikus duomenų rinkinių paiešką pastebėta, kad dauguma viešai prieinamų rinkinių yra sugeneruoti simuliuotoje „Linux“ OS aplinkoje. Pasiūlytas „Windows“ OS skirtas duomenų surinkimo būdas. Pagal minėtą duomenų surinkimo būdą suprogramuotas įrankis, surinkti ir susisteminti vartotojų elgesio duomenys, kuriuos sudaro per pasirinktą laiko intervalą įvykusių TCP sujungimų ir veikiančių procesų pavadinimai. Duomenys apdoroti su „MD5“ vienkryptėmis funkcijomis ir gautas intervalas apribotas iki 1024.

Įvertinus surinktus duomenis nustatyta, kad įsilaužimams aptikti reikalingas neprižiūrimas mašininio mokymosi algoritmas. Įsilaužimams aptikti pasiūlytas metodas, kuris mokosi iš įprasto vartotojo elgesio pasitelkiant giluminį neuroninį tinklą – LSTM autoenkoderį. Vėliau išmokytas tinklas stebi naujus elgesio įrašus, įvertina jų nukrypimo nuo normos reikšmę ir reikšmei viršijus nustatytą ribą pažymi įrašą kaip įtartiną.

Atlikus eksperimentą nustatyta, jog pasiūlytas metodas yra pakankamas daliai įtartinų įvykių atrasti. Šis modelis sugeba aptikti įtartinus įvykius, kai tam pasitelkiamas internetas arba nepaslėptos, mažai naudojamos arba iš vis nenaudojamos programos. Didžiausias modelio efektyvumas pasiektas, kai įrašai grupuojami po 9. Geriausiu atveju gautas 78,57 % aptikimo dažnis su 0 neteisingai pažymėtų įrašų grupių.

5. Literatūros sąrašas

8 Best HIDS Tools—Host-Based Intrusion Detection Systems. (2019). Retrieved December 27, 2019, from https://www.dnsstuff.com/host-based-intrusion-detection-systems

Aldairi, M., Karimi, L., & Joshi, J. (2019). A trust aware unsupervised learning approach for insider threat detection. *Proceedings - 2019 IEEE 20th International Conference on Information Reuse and Integration for Data Science, IRI 2019*, 89–98. https://doi.org/10.1109/IRI.2019.00027

Auso, N. (2015). The difference between NIDS and HIDS and how do they help. Retrieved January 8, 2020, from https://www.techyv.com/questions/difference-between-nids-and-hids-and-how-do-they-help/

Balajinath, B., & Raghavan, S. V. (2001). Intrusion detection through learning behavior model. *Computer Communications*, *24*(12), 1202–1212. https://doi.org/10.1016/S0140-3664(00)00364-9

Boubez, T. (2015). *Patent No. US 2015/0341246A1*. USA.

Boukerche, A., & Annoni Notare, M. S. M. (2002). Behavior-based intrusion detection in mobile phone systems. *Journal of Parallel and Distributed Computing*, *62*(9), 1476–1490. https://doi.org/10.1006/jpdc.2002.1857

Brownlee, J. (2019). A Tour of Machine Learning Algorithms. Retrieved December 31, 2019, from https://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/

Chattopadhyay, P., Wang, L., & Tan, Y. P. (2018). Scenario-based insider threat detection from cyber activities. *IEEE Transactions on Computational Social Systems*, *5*(3), 660–675. https://doi.org/10.1109/TCSS.2018.2857473

Cooper, S. (2019). Host-Based Intrusion Detection Systems Explained – 6 Best HIDS Tools Reviewed. Retrieved January 1, 2020, from https://www.comparitech.com/net-admin/hids-tools-software/

Creech, G. (2014). *Developing a high-accuracy cross platform Host-Based Intrusion Detection System capable of reliably detecting zero-day attacks*. 215.

Fuchsberger, A. (2005). Intrusion detection systems and intrusion prevention systems. *Information Security Technical Report*, *10*(3), 134–139. https://doi.org/10.1016/j.istr.2005.08.001

Goyal, A. (2019). Popular Machine Learning Algorithms That You Should Know! Retrieved January 23, 2020, from https://medium.com/eduonix/popular-machine-learning-algorithms-that-you-should-know-f0815e92fc22

Grimmer, M., Röhling, M. M., Kreusel, D., & Ganz, S. (2019). A Modern and Sophisticated Host Based Intrusion Detection Data Set. *16. Deutscher IT-Sicherheitskongress*, 10. Retrieved from https://www.bsi.bund.de/DE/Service/Aktuell/Veranstaltungen/IT-Sicherheitskongress/IT-Sicherheitskongress_node.html

Hashem, Y., Takabi, H., & Dantu, R. (2017). Insider Threat Detection Based on Users' Mouse Movements and Keystrokes Behavior. *Secure Knowledge Management Workshop*.

Hu, T., Niu, W., Zhang, X., Liu, X., Lu, J., & Liu, Y. (2019). An Insider Threat Detection Approach Based on Mouse Dynamics and Deep Learning. *Security and Communication Networks*, *2019*. https://doi.org/10.1155/2019/3898951

Igbe, O., & Saadawi, T. (2018). Insider Threat Detection using an Artificial Immune system Algorithm. *2018 9th IEEE Annual Ubiquitous Computing, Electronics and Mobile Communication Conference, UEMCON 2018*, 297–302. https://doi.org/10.1109/UEMCON.2018.8796583

Maurizio Martellini, A. M. (2017). Cyber Related CBRNe Attacks. In *Cyber and Chemical, Biological, Radiological, Nuclear, Explosives Challenges: Threats and Counter Efforts* (p. 407).

Meng, F., Lou, F., Fu, Y., & Tian, Z. (2018). Deep learning based attribute classification insider threat detection for data security. *Proceedings - 2018 IEEE 3rd International Conference on Data Science in Cyberspace, DSC 2018*, 576–581. https://doi.org/10.1109/DSC.2018.00092

NetMarketShare. (2020). Market Share Statistics for Internet Technologies. Retrieved January 24, 2021, from https://netmarketshare.com/operating-system-market-share.aspx

Olah, C. (2015). Understanding LSTM Networks. Retrieved January 24, 2021, from http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Qdo, R. Q. H. Q., Ri, V. L. V, Wkuhdw, Q., Xvlqj, H., & Rulhqwhg, J. (2017). *HoneynetTrap: Analysis of Insider threat Detection using Agent oriented PN2 simulator*. 433–438.

Rocca, J., & Rocca, B. (2019). Understanding Variational Autoencoders (VAEs). Retrieved May

22, 2021, from https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

Roos, M. (2019). Recurrence in biological and artificial neural networks. Retrieved January 8, 2020, from https://towardsdatascience.com/recurrence-in-biological-and-artificial-neural-networks-e8a6d5639781

Spafford, E., & Zamboni, D. (2000). Data collection mechanisms for intrusion detection systems. *Center for Education and Research in Information Assurance and Security, CERIAS Technical Report*, (April 2001). Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.74.8299&rep=rep1&type=pdf

Subasi, A. (2020). Autoencoders. In *Practical Machine Learning for Data Analysis Using Python* (p. 520).

Templeton, G. (2015). Artificial neural networks are changing the world. What are they? Retrieved January 8, 2020, from https://www.extremetech.com/extreme/215170-artificial-neural-networks-are-changing-the-world-what-are-they

Ussath, M., Jaeger, D., Cheng, F., & Meinel, C. (2017). Identifying Suspicious User Behavior with Neural Networks. *Proceedings - 4th IEEE International Conference on Cyber Security and Cloud Computing, CSCloud 2017 and 3rd IEEE International Conference of Scalable and Smart Cloud, SSC 2017*, 255–263. https://doi.org/10.1109/CSCloud.2017.10

Yang, G., Cai, L., Yu, A., & Meng, D. (2018). A General and Expandable Insider Threat Detection System Using Baseline Anomaly Detection and Scenario-Driven Alarm Filters. *Proceedings - 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications and 12th IEEE International Conference on Big Data Science and Engineering, Trustcom/BigDataSE 2018*, 763–773. https://doi.org/10.1109/TrustCom/BigDataSE.2018.00110

Yeung, D. Y., & Ding, Y. (2003). Host-based intrusion detection using dynamic and static behavioral models. *Pattern Recognition*, *36*(1), 229–243. https://doi.org/10.1016/S0031-3203(02)00026-2

Zhang, J., Chen, Y., Yang, K., Zhao, J., & Yan, X. (2019). Insider threat detection based on adaptive optimization DBN by grid search. *2019 IEEE International Conference on*

Intelligence and Security Informatics, ISI 2019, 173–175. https://doi.org/10.1109/ISI.2019.8823459