Upotreba umjetne inteligencije u lovu na kibernetičke prijetnje

Žugaj, Antonio

Master's thesis / Diplomski rad

2024

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

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:211:554426

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Antonio Žugaj

Upotreba umjetne inteligencije u lovu na kibernetičke prijetnje

DIPLOMSKI RAD

Varaždin, 2024.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Antonio Žugaj

Matični broj: 0016136491

Studij: ***Informacijsko i programsko inženjerstvo***

Upotreba umjetne inteligencije u lovu na kibernetičke prijetnje

DIPLOMSKI RAD

Mentor/Mentorica:

Doc. dr. sc. Igor Tomičić

Varaždin, rujan 2024

Antonio Žugaj

Izjava o izvornosti

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

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

_____

Sažetak

U ovom radu se istražuje primjena umjetne inteligencije za detekciju kibernetičkih prijetnji, s naglaskom na prepoznavanje skeniranja portova putem ECOD modela strojnog učenja. Koristeći CRISP-DM metodologiju, izvršene su sve ključne faze razvoja, od razumijevanja problema i pripreme podataka do modeliranja, evaluacije i implementacije. Poseban je fokus na izazovu prikupljanja i obrade podataka iz stvarnog okruženja, gdje se koristio skup podataka dobiven iz hrvatske organizacije. Model koji je razvijen je postigao izvanrednu točnost od 99,99% u detekciji skeniranja portova. Rezultati rada potvrđuju ovakav pristup kao značajno poboljšanje za sigurnosne sustave, pružajući pouzdanu zaštitu od potencijalnih kibernetičkih napada. Ovaj rad doprinosi području kibernetičke sigurnosti, demonstrirajući učinkovitost strojnog učenja u prepoznavanju i odgovoru na prijetnje u stvarnom vremenu.

Ključne riječi: Umjetna inteligencija, AI, kibernetičke prijetnje, ECOD, Outlier Detection, skeniranje portova

Sadržaj

1. Uvod

U modernom svijetu, koji svakim danom postaje podložniji i ovisniji o računalnoj i informatičkoj infrastrukturi, kibernetički napadi postaju prijetnja za sve veći opseg svakodnevnog života. Od individualnih osoba do multi milijunskih kompanija, svi su podložni opasnosti sigurnosnih propusta i ciljanih kibernetičkih napada. Posljedica tih napada može uzrokovati veliku smetnju i materijalnu štetu za izrazito važne aspekte naše svakodnevice. Iz tog razloga je važno posvetiti što veći fokus na zaštitu od kibernetičkih napada. Pasivna zaštita od napada, koju mnogi koriste i smatraju da je zadovoljavajuća, u većini slučajeva nije dovoljna. Potrebna je aktivna komponenta zaštite od prijetnji. Takav aktivan proces traženja prijetnji se naziva lov na kibernetičke prijetnje. On uključuje velike napore iterativnog analiziranja velike količine podataka i oslanja se na ispravno prosuđivanje i iskustvo samog analitičara. Napore i ispravnost samog procesa možemo olakšati pomoću nedavnog i izrazito rapidnog rasta umjetne inteligencije. Kreiranjem modela umjetne inteligencije specifično namijenjen kao pomagalo analitičarima bi moglo višestruko unaprijediti sigurnost naše svakodnevice.

Ovaj rad obrađuje temu jednog takvog modela. Pošto je cilj lova na kibernetičke prijetnje detektirati prijetnju u što ranijem stadiju napada, fokus rada je detekcija planiranja samog napada, odnosno testiranja sustava na potencijalne vektore napada. Bit će prikazana izrada modela koji će nam omogućiti isticanje sumnje korištenja alata Nmap unutar zapisa firewall-a naše organizacije. Rad sadrži opis prikupljanja, verificiranja i normaliziranja podataka za treniranje modela, te treniranje i verifikaciju samog finalnog modela umjetne inteligencije. Za izradu modela se koristi CRISP-DM metodologija i detaljno su opisane sve njezine faze. Finalni rezultat ovog rada je model umjetne inteligencije koji detektira zapise koji se ponašaju van očekivanog ponašanja zapisa na firewall-ima krajnjim uređajima.

2. Metodologija i alati

Razvoj modela umjetne inteligencije je zahtjevan proces koji zahtjeva pažljiv odabir alata i metodologije kako bi se osigurala preciznost, pouzdanost i učinkovitost. Zbog rasta popularnosti i korištenja umjetne inteligencije, količina dostupnih alata i metodologija nije mala. To otežava izbor najučinkovitije metodologije i alata za temu ovoga rada.

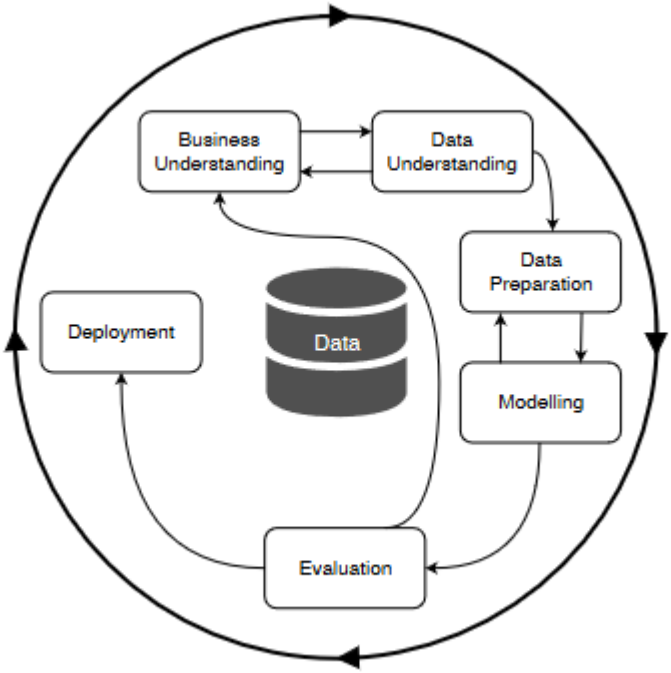
Metodologija koja se koristi unutar ovoga rada je CRISP-DM, odnosno Cross-Industry Standard Process for Dana Mining. CRISP-DM metodologija se sastoji od šest faza koje omogućavaju strukturirani pristup razumijevanja i sustavni razvoj modela umjetne inteligencije.

Alat pomoću kojeg se manipuliraju podacima i obavlja samo treniranje i vrednovanje modela umjetne inteligencije je programski jezik Python. Za implementaciju modela se koristi niz naprednih alata i biblioteka dostupnih u Python programskom jeziku. Najvažnije biblioteke su PyOD, Pandas, NumPy i Scikit-learn.

Kroz naredna poglavlja detaljno će se objasniti svaki od ovih alata i metodologija, te način na koji su korišteni u izradi sustava za lov na kibernetičke prijetnje. Cilj nam je pružiti cjelovit prikaz metodološkog pristupa i tehničkih resursa koji stoje iza uspješne implementacije ovog sustava.

2.1. CRISP-DM

Kao što smo već naveli, u ovome radu se koristiti CRISP-DM metodologijom kako bi se dobio strukturirani razvoj našeg modela. Razlog odabira CRISP-DM za metodologiju je činjenica da je ona de-facto standard i industrijski neovisan procesni model za primjenu projekata rudarenja podataka.[1] Cross-Industry Standard Process for Data Mining pruža okvir koji se sastoji od šest strukturiranih faza. Te faze su razumijevanje domene, razumijevanje podataka, priprema podataka, modeliranje, evaluacija i implementacija.[2] CRISP-DM metodologija je iterativna metodologija. Što znači da se određene faze mogu ponovno izvršiti kako bi se unaprijedio krajnji model i kako bi se ispravili pogreške ili krive pretpostavke oko modela.



Slika 1: CRISP-DM Metodologija[4]

Korištenje CRISP-DM metodologije omogućava strukturirano i efikasno razvijanje modela lova na kibernetičke prijetnje. Svaka faza metodologije pruža jasne smjernice koje osiguravaju našem radu da bude temeljit, dosljedan i usmjeren na postizanje visokokvalitetnih rezultata. Svaku od faza ćemo detaljno opisati u narednim poglavljima.

2.2. Python

Python je ključan alat u razvoju modela za detekciju kibernetičkih prijetnji zahvaljujući svojoj svestranosti, jednostavnosti i bogatoj kolekciji biblioteka za strojno učenje, analizu podataka, te manipulacijom samih podataka. U ovom radu, Python je odabran kao glavni alat, odnosno programski jezik, zbog niza prednosti koje pruža u procesima analize podataka, modeliranja i evaluacije. Drugim riječima, Python se etablirao kao jedan od najpopularnijih jezika za znanstveno računalstvo. Zahvaljujući interaktivnoj prirodi visoke razine i ekosustavu znanstvenih knjižnica, to je privlačan izbor za algoritamski razvoj i istraživačku analizu podataka.[4]

Python sam po sebi ne može izvršiti potrebne korake za realizaciju modela. Moramo uključiti određene javno dostupne biblioteke kako bi upotpunili funkcionalnost izrade modela umjetne inteligencije. Prva od tih biblioteka je Pandas. Pandas je esencijalna biblioteka za manipulaciju i analizu podataka. U ovome radu se koristi ta biblioteka za unos sirovih podataka iz .pickle formata, te kako bi se moglo, nakon učitavanja, transformirati podatke u oblik pogodan za daljnji rad. Format DataFrame, u koji se učitavaju sirovi podaci, je mnogo jednostavniji za rukovati i manipulirati od sirovih podataka.

Druga korištena biblioteka je NumPy. Ona u sebi sadrži numeričke operacije i pruža podršku u radu s matematičkim funkcijama poput rada s velikim i višedimenzionalnim nizovima i matricama. U slučaju ovoga rada nam koriste kako bi određene atribute prilagodili statističkim operacijama.

Scikit-learn je jedna od najpopularnijih biblioteka za strojno učenje u Python-u, stoga se koristi u ovome radu. Ona pruža alate za pred procesiranje podataka, modeliranje i evaluaciju. Najvažnija funkcionalnost, za ovaj rad, je podjela podataka u trening i testne skupove.

Zadnja korištena biblioteka je biblioteka PyOD, odnosno Python Outlier Detection. PyOD je specijalizirana biblioteka za detekciju anomalija u podacima. Biblioteka sadrži veliki broj algoritama koje se koriste za implementaciju i evaluaciju modela za detekciju anomalija i ta je funkcionalnost ona koja je najviše potrebna u sklopu ovoga rada.

3. Razumijevanje domene

Prva faza CRISP-DM metodologije, razumijevanje domene, ključna je za definiranje ciljeva i opsega projekta, odnosno modela za detekciju korištenja alata Nmap unutar zapisa firewall-a krajnjih uređaja. Ova faza fokusira se na razumijevanje problema, identificiranje ključnih izazova i postavljanje jasnih zahtjeva za model koji će se razviti. Kroz ovu fazu osiguravamo da svi sljedeći koraci budu usklađeni s našim potrebama i da konačni model pruži stvarnu vrijednost. Cilj ove faze je stvoriti sliku o ciljevima modela kojeg kreiramo, stvoriti ideju o tipovima i formatu podataka koji su nam potrebni, detaljno razumjeti tematiku problema i stvoriti plan za izradu modela. [2]

U sklopu ove faze će se napraviti pregled dosadašnjih radova na temu ovoga rada i blisko vezane teme. Tijekom pregleda će se napraviti analiza svakog rada, te iz njih izvući korisne informacije i zaključke koje će nam koristiti pri definiranju daljnjeg plana izrade modela detekcije anomalija.

Detaljno će se razjasniti ključni koncepti i postupci kibernetičkog napada i lova na kibernetičke napade. Ti koncepti će koristiti kako bi mogli točnije ustanoviti je li određeni događaj kibernetički napad i kako bi mogli ispravno trenirati model tako da prepoznaje maliciozne akcije.

3.1. Kibernetičke prijetnje

Kibernetički napadi predstavljaju sve veći problem u današnjem digitalno povezanom svijetu. Oni ugrožavaju sigurnost podataka, privatnost pojedinaca i operativnu stabilnost organizacija. Posljedice uspješno izvršenih nada mogu zadati organizacijama veliku materijalnu štetu. Uz samu materijalnu štetu, neuspjeh obrane od napada uzrokuje smanjenjem povjerenja i integriteta same organizacije. To smanjenje ugleda može uzrokovati katastrofalne posljedice za organizaciju ili pojedinca u smislu sadašnjeg ili budućeg poslovanja.

U kontekstu razvoja modela za detekciju kibernetičkih prijetnji korištenjem umjetne inteligencije, ključno je razumjeti različite vrste kibernetičkih napada, njihove metode i motive napadača. Ovo razumijevanje omogućuje razvijanje učinkovitih modela koji mogu prepoznati i odgovoriti na ove prijetnje. Motivacija iza napada može biti financijski dobit, industrijska špijunaža, političke prijetnje ili sobna vedeta.

Postoje razni načina klasificiranja tipova kibernetičkih napada. Mogu se klasificirati prema cilju, legalnoj klasifikaciji, temelju ozbiljnosti zahvaćenosti, opsegu zahvaćenosti i tipu mreže.[5] U svrhu ovoga rada, fokus je na klasifikaciji prema cilju.

Napadi prema cilju se mogu definirati kao napati koji uskraćuju usluge (eng. Denial Of Service), Napadi pristupa (eng. Access Attack) i izviđački napadi (eng. Reconnaissance Attack).[5] Fokus krajnjeg modela je uhvatiti prijetnju u što ranijem stadiju napad. Stoga je procjena da se treba fokusirati na izviđačke napade. Oni su početni dijelovi velikog broja napada i stoga imaju veliki potencijal biti najranija točka detekcije prijetnje. Izviđački napadi se dijele na skeniranje portova (eng. Scanning the port), njuškanje paketa (eng. Packet sniffers), čišćenje pinga (eng. Sweeping the Ping), te upiti vezani uz internetske informacije (eng. Queries Regarding Internet Information).[5]

Daljnji korak razumijevanja domene je pogledati problem kibernetičkih prijetnji iz drugog kuta. Kibernetički napadi se mogu modelirati kako bi imali strukturiranu slijednu formu. Postoje razni modeli koji strukturiraju kibernetičke napade u složene slijedne faze. Razlog zašto je važno poznavati modele napada je taj da oni mapiraju korake koje tipičan napad koristi kako bi ugrozio sustav, odnosno organizaciju. U svrhu ovoga rada je odabran takozvani Kill Chain model. On je jedan od najpoznatijih modela modeliranja napada i jako je primjenjiv u domeni kibernetičkih prijetnji.

Kill chain za nametanje je jedna od tehnika modeliranja napada, koja definira napad kao lanac radnji. To je strukturirani napad, budući da napadač napreduje u napadu u uređenom lancu prema planu.[6] On se definira na sedam faza koje se slijedno izvršavaju. Svaka faza koristi zaključke i podatke prijašnje faze kako bi ispravno izvršili slijedeće koraka i u krajnjem cilju uspješno izvršili napad.

Faze Kill Chain modela su izviđanje (eng. Reconnaissance), oružavanje (eng. Weaponization), dostava (eng. Delivery), eksploatacija (eng. Exploitation), instalacija (eng. Installation), komanda i kontrola (eng. Command and Control), te djelovanje na ciljeve (eng. Actions on Objectives).[7] Kao što je već navedeno u prijašnjem dijelu rada, cilj je uhvatiti, odnosno detektirati, napadača prije nego što je počinio štetu. Ako se analizira Kill Chain model, može se uvoditi da je prva faza u kojoj se može detektirati napadača, faza izviđanja.

Izviđanje znači prikupljanje informacija o potencijalnom cilju. Cilj može biti pojedinac ili organizacijski entitet. Izviđanje se dalje može raščlaniti na identifikaciju cilja, odabir i profiliranje.[8] Faza izviđanja se izvršava kroz pasivno i aktivno izviđanje. Pasivno izviđanje su sve radnje iz kojih dobivamo informacije o ciljanoj meti bez da ta meta može detektirati naše radnje. S druge strane aktivno izviđanje su sve radnje kojima dobivamo informacije o meti a da sigurnosti tim ciljane mete može posumnjati ili detektirati naše radnje.[8]

Fokus je dakle na aktivnom izviđanju. Ako se pogledaju postupci za aktivno izviđanje i usporede s prijašnjom analizom klasifikacije napada, može se primijetiti da se došlo do istih zaključaka. Stoga krajnji model mora moći detektirati napadača koji testira naš sustav za ranjivosti. Daljnjim istraživanjem jedan javno dostupan alat iskače kao potencijalni razlog našeg modela. Taj alat je Nmap. Funkcionalnost toga alata ćemo obraditi u slijedećem poglavlju.

3.2. NMAP

Nmap (Network Mapper) je moćan i popularan alat za mrežnu skeniranje koji se koristi za otkrivanje hostova i usluga na računalnoj mreži analiziranjem paketa podataka. Po mnogima je najbolji port skener i istaknuti dio alata mrežnih odjela diljem svijeta. Zbog svoje popularnosti je široko korišten u zajednici IT sigurnosti radi laganog i efektivnog mrežnog skeniranja. Mrežno skeniranje je proces otkrivanja aktivnih host-ova na mreži i informacije o host-ovima, kao što su operativni sustav, aktivni portovi, usluge i aplikacije.[9]

Ta funkcionalnost ga čini ekstremno korisnim za mrežne i sigurnosne timove organizacija. Alat daje mogućnost laganog mapiranja mrežne arhitekture i pregleda kritičnih točaka. Nažalost za istu svrhu ga mogu koristiti i napadači. Isto kako ga članovi organizacije mogu koristiti za detekciju sigurnosnih propusta koji se trebaju ukloniti, tako i napadač koristi te iste detektirane propuste kako bi napao infrastrukturu organizacije.

Nmap kao alat je razvijen za više različitih operacijskih sustava. Postoji grafičko sučelje iako je najpopularnije klasično sučelje kroz komandnu liniju. Uz otkrivanje host-ova, skeniranje portova, identificiranja verzija usluga i otkrivanje operativnih sustava, Nmap daje mogućnost pisanja skripti kroz NSE (eng. Nmap Scripting Engine). Takve skripte mogu otkriti detaljne informacije o sustavu organizacije. Potencijalno ček i neke ranjivosti koje ni sama organizacije ne zna.

Iz tog razloga je ovaj alat toliko opasan. Pomoću jedne linije koda može prikazati sve otvorene i nezaštićene portove mreže koje napadač može koristiti kao vektor svoga napada. Moderni firewall-ovi su dobili mogućnost detekcije skeniranja portova tako da detektiraju TCP trostruko rukovanje. Nažalost Nmap je našao način i da to zaobiđe. Alat može označiti zastavicu koja mu dopušta potajno skeniranje portova na način da nikad ne dovrši trostruko rukovanje, već samo traži povratnu informaciju od porta.

To je razlog zašto je odabran ovaj alat kao cilj krajnjeg modela. Svako skeniranje alatom Nmap ostaje zapisano u zapisima na krajnjim točkama firewall-a. Analitičari bi analizom mogli otkriti korištenje Nmap alata unatoč činjenici da koristi potajan način rada. Problem je što je količina zapisa pre velika i količina analitičara pre mala. No ako bi koristili model umjetne inteligencije koji je učen da otkriva takve zapise, posao analitičara bi se drastično smanjio.

3.3. Pregled dosadašnjih radova

U ovom poglavlju će se napraviti pregled dosadašnjih radova na temu korištenja umjetne inteligencije kod lova na kibernetičke prijetnje. Poznavanje dosadašnjih radova omogućuje primjenu zaključaka autora na onaj rad. Zaključci mogu ukazivati na uspješnost neke metode modeliranja, na problematiku oko pojedinog aspekta kreiranja modela umjetne inteligencije te na očekivanu uspješnost samog modela. Radovi su pronađeni koristeći se ključnim riječima poput 'AI threat hunting', 'port scanning AI' i 'AI threat detection'. U narednom tekstu će se analizirati odabrani radovi i iz njih izvući zaključke.

Prvi rad koji se analizira je napisan od strane Jin Kim, Nara Shin, Seung Yeon Jo i Sang Hyun Kim. Naslov rada je Metoda otkrivanja upada koristeći Duboku Neuralnu Mrežu (eng. Method of Intrusion Detection using Deep Neural Network). U radu se istražuje ideja IDS-a (Intrusion Detection System) koji se oslanja na DNN (Deep Neural Network). Rad naglašava problematiku sve kompleksnijih mrežnih napada i limitiranosti IDS sustava koji se oslanjaju na analizu događaja temeljena na pravilima i imaju visoku stopu krivih detekcija. Model je postigao preciznost od 99% i uspio je detektirati razne tipove napada koje uključuje i skeniranje portova. [10]

Drugi rad koji se analizira je napisan od strane Mohamed Amine Ferrag, Merouane Debbah i Muna Al-Hawawreh. Naslov rada je Generativni AI za lov na kibernetičke prijetnje u Iot mrežana s omogućenim 6G (eng. Generative AI for Cyber Threat-Hunting in 6G-enabled IoT Networks). Rad obrađuje temu korištenja generativne umjetne inteligencije kao unaprjeđenje u lovu na kibernetičke prijetnje u IoT (Internet of Things) mrežama. Eksperimenti su pokazali uspješnost korištenja takvih modela s 95% točnosti prepoznavanja. Unatoč obećavajućim rezultatima, rad identificira nekoliko izazova, kao što su skalabilnost, decentralizirano treniranje, kvaliteta podataka, energetska učinkovitost, privatnost i tokenizacija, koje je potrebno riješiti kako bi se u potpunosti iskoristio potencijal generativne umjetne inteligencije u ovom kontekstu. Autori na kraju pozivaju na daljnja istraživanja u ovom području.[11]

Treći rad koji se analizira je napisan od strane Hernán Domínguez-Limaico, Willams Nicolalde Quilca, Marcelo Zambrano, Fabián Cuzme-Rodríguez i Edgar Maya-Olalla. Naslov rada je Umjetna neuronska mreža temeljena na sustavu detekcije uljeza za softverski definiranu mrežu (eng. Intruder Detection System Based Artificial Neural Network for Software Defined Network). Rad prezentira implementaciju ANN-a (Artificial Neural Network), pod imenom Snort + RNA, u IDS radi smanjenja rizika od kibernetičke prijetnje. Testiranje je pratilo

PDCA model iz ISO/IEC 27001 standarda i hakerske procese. Model je uspješno detektirao anomalije i generirao alarme. Jedina mana je bila poteškoća čitanja svih paketa kod DoS napada. Unatoč mani se eksperiment pokazao kao uspješan. [12]

Četvrti rad koji se analizira je napisan od strane Rhiannon Weaver. Naslov rada je Identifikacija anomalija mrežnog ponašanja na portovima (eng. Identifying Anomalous Port-Specific Network Behavior). U radu se predstavlja detekcija anomalija na portovima mreže pomoću statističkog modela detekcije anomalija. Odabrani model u ovome radu je MVC (Minimum Covariance Distance). Model pokazuje potencijal prema velikoj uspješnosti. Problemi koji su nastali tokom rada je korištenje resursa kod velike količine zapisa, osjetljivost modela na parametrizaciju koja utječe na točnost modela, te na visoka stopa, od 0.8%, lažno pozitivnih detekcija. Autor ukazuje na veliki potencijal modela detekcije anomalija, ali se moraju kreirati novi modeli koji će rješavati primijećene probleme.[13]

Peti rad koji se analizira je napisan od strane Wassim El-Hajj, Fadi Aloul i Zoueir Trabelsi. Naslov rada je O otkrivanju skeniranja priključaka pomoću Fuzzy sustava za otkrivanje upada (eng. On Detecting Port Scanning using Fuzzy Based Intrusion Detection System). U ovome radu se prikazuje integracija fuzzy logike s Snort sustavom za otkrivanje upada (IDS). Rad je pokazao kako korištenje ovakvog modela povećava učinkovitost IDS-a u smanjenju lažno pozitivnih rezultata. Neke poteškoće s ovakvim modelom su potrebe za konstantnim mijenjanjem i podešavanjem parametara kako bi model mogao detektirati uzorke koje do tad nije vidio. Stoga novi uzorci napada stvaraju opasnost ovome modelu. Također je model imao poteškoća ponekad razaznati običan promet od onog malicioznog. Mogao je detektirati bolje od samog IDS-a, ali ne na dovoljnoj razini.[14]

Šesti rad koji se analizira je napisan od strane Ashton Webster, Margaret Gratian, Ryan Eckenrod, Daven Patel i Michel Cukier. Naslov rada je Poboljšana metoda za otkrivanje skeniranja mreže temeljeno na anomalijama (eng. An Improved Method for Anomaly-Based Network Scan Detection). U radu se kreiralo više različitih modela, te analizirao njihov učinak u detekciji anomalija skeniranja mreže. Korišteni modeli su bili Random Forest, AODE, PRISM, SMO i Decision Table. Najbolji rezultat je ostvario Random Forest s točnošću od 99.94%. Problem kojeg autori naglašavaju je potreba nadziranog učenja kod modela s najvišom točnošću. Prikupljanje podataka za nadzirano učenje je problematično i autor daje preporuku kreiranja modela koji koristi ne nadzirano ili polu-nadzirano učenje.[15]

3.4. Izrada cilja i plana

Izrada cilja i plana je zadnji korak u razumijevanju domene. U ovom poglavlju će se formulirati zahtjevi i ciljevi za krajnji model. Cilj je izrada modela umjetne inteligencije koji će primati podatke određene mreže. Ti podatci će biti generirani zapisi od strane firewall-a krajnjih točaka u organizaciji. Iz tih podataka model će morati analizirati promet i ispravno zaključiti koji zapisi su generirani pri uporabi alata za mrežno skeniranje. Na temelju prijašnjih radova se može postaviti cilj učinkovitosti modela između 99% i 100%. Veliki problem prijašnjih radova je bilo konstantno ili otežano podešavanje parametara modela u nadgledanom učenju. Potreba rada je pronaći model koji ima mogućnost nenadgledanog učenja. Veliki potencijal su pokazali algoritmi detekcije anomalija, ali oni imaju problem korištenja velike količine resursa kod velikog volumena prometa. Velika veličina starijih algoritama se osniva na parametrima koji diktiraju točnost modela. Pronalaskom novijeg algoritma koji rješava ove poteškoće bi unaprijedili iskoristivost ovakvih modela u praksi.

4. Razumijevanje podataka

Druga faza CRISP-DM (Cross-Industry Standard Process for Data Mining) metodologije, poznata kao razumijevanje podataka, ključna je za uspjeh svakog projekta vezanog uz rudarenje podataka. U kontekstu lova na kibernetičke prijetnje, razumijevanje podataka omogućuje nam da prepoznamo obrasce i anomalije koje ukazuju na potencijalne sigurnosne rizike. Ova faza uključuje detaljnu analizu dostupnih podataka, identifikaciju njihovih karakteristika i razumijevanje kako ti podaci mogu doprinijeti otkrivanju kibernetičkih prijetnji.

Koraci izvođenja ove faze su prikupljanje početnih podataka, pregled i istraživanje podataka, analiza i interpretacija, te verifikacija kvalitete podataka.[2] Svaki od ovih koraka su bitni za uspjeh finalnog modela. Ispravno izvršavanje koraka omogućuje sakupljanje ispravnih podataka na kojima će se sam model učiti. U slučaju da je ova faza, a ujedno i njezini koraci, krivo napravljena, može uzrokovati netočnost modela ili da je model ispravan samo za prikupljeni skup podataka.

Korake druge faze CRISP-DM metodologije će se opisati kroz poglavlja prikupljanje podataka i opis podataka. U sklopu poglavlja opisa podataka će se napraviti analiza, interpretacija i verifikacija samih podataka kako bi zadovoljili sve korake same metodologije.

4.1. Prikupljanje podataka

Prikupljanje podataka je možda najvažniji korak izrade samog modela umjetne inteligencije. Ako se model trenira na lošem skupu podataka, izlazni model neće biti primjenjiv. Iz tog razloga je bitno pronaći reprezentativni skup podataka. Skup koji će modelu dati mogućnost biti primjenjiv na slučajeve izvan trening okruženja.

Postoje razni izvori podataka. Jedna opcija je samostalno prikupljanje podataka. Ta opcije povećava kvalitetu podataka i provjerenost valjanosti, ali zahtjeva puno vremena i truda. Druga opcija je pretraživanje dostupnih baza podataka. Pošto je za treniranje model potrebna velika količina podataka, većina odabiru dostupne baze podataka. Za potrebe ovoga rada se nije odlučilo za baze podataka. Razlog tome je što se ne može uvijek verificirati ispravnost svih zapisa unutar baze. Također je za krajnji model potreban specifični skup podataka kojeg je teško pronaći u bazama podataka.

Skup podatak, koji će se koristiti za treniranje modela, je zaprimljen iz jedne velike Hrvatske organizacije. Organizacija zahtijeva ostati anonimna i iz tog razloga se promijenilo

ime skupa podataka i svaki podatak koji upućuje na organizaciju je postavljen na nasumičnu vrijednost. Nasumične vrijednosti ne utječi na rezultat učenja modela.

Skup podataka je prikupljen s firewall-a organizacije. Zapisi s firewall-a prikazuju normalni mrežni promet u zahvaćenom intervalu. Uz normalan mrežni promet se u nasumičnim intervalima izvršavalo skeniranje portova pomoću Nmap alata. Broj nasumičnih skeniranja je 257. Zapisi su spremljeni unutar .pickle formata. Razlog tome je naše korištenje Python programskog jezika. Pickle format nam daje mogućnost laganog čitanja i manipuliranja podacima.

4.2. Opis podataka

Skup podataka, na temelju kojih će se trenirati naš model umjetne inteligencije, se sadrži od 15,668,198 zapisa. Zapisi predstavljaju sav mrežni promet jedne Hrvatske organizacije u jednom vremenskom intervalu. Svaki pojedinačni zapis unutar skupa podataka predstavlja jednu uspostavu komunikacije koju je firewall zabilježio. Jedan takav zapis predstavlja jedan red u tablici.

Svaki pojedinačni zapis unutar tablice je opisan pomoću sedam atributa. Ti atributi su vremenski zapis (timestamp), IP adresa izvora (source_ip), IP adresa destinacije (destination_ip), port destinacije (destination_port), protokol (protocol), akcija (action) i oznaka (label). Svaki zapis unutar tablice ima ispunjene sve atribute i svi atributi su potrebni kako bi ispravno opisali jedan zapis.

Oznaka vremenskog zapisa, označena kao 'timestamp', predstavlja vremenski trenutak u kojem se zapis dogodio, odnosno trenutak u kojem se dogodila jedna instanca komunikacije koja je prošla kroz firewall. Sam vremenski zapis u sebi sadrži podatke o godini, mjesecu, danu, satu, minuti i sekundi svakog zapisa. Format u kojem su ti podaci zapisani je 'yyyy-mm-dd hh:mm:ss'. Vremenski okvir svih zapisa je od 1. veljače 2023. godine do 28. veljače 2023. godine.

IP adresa izvora označava IP adresu s koje se pokušala uspostaviti komunikacija. Izvor te adrese može biti izvan organizacije ili unutar same organizacije. IP adresa je predstavljena u decimalnom zapisu gdje je svaki byte podatak odvojen točkom. To je standardni način bilježenja IP adresa. Primjer jedne adrese je '192.168.0.1'. Svaka IP adresa označava jednu jedinstvenu krajnju točku.

IP adresa destinacije označava IP adresu prema kojoj se pokušava uspostaviti komunikacija. Izvor te adrese, isto kao i adresa izvora, može biti izvan organizacije ili unutar same organizacije. Također isto kao i izvorišna adresa, odredišna IP adresa je predstavljena u decimalnom zapisu gdje je svaki byte podatak odvojen točkom. Svaka IP adresa označava jednu jedinstvenu krajnju točku. Pošto su zapisi zabilježeni na firewall-u, zapisi ne bilježe međusobnu komunikaciju krajnjih točaka unutar same organizacije i komunikaciju vanjskih krajnjih točaka izvan organizacije.

Port destinacije označava port krajnje točke na kojem se pokušava ostvariti komunikacija. Portovi su virtualne točke na računalima koje omogućuju različitim aplikacijama i uslugama da komuniciraju preko mreže. Svaka mrežna komunikacija koristi dva porta. Jedan je izvorni, drugi je destinacijski. U ovome slučaju bilježi se samo destinacijski port. On je izrazito važan jer je on cilj skeniranja napadača. Port se označava kao dekadski broj.

Protokol označava protokol s kojim se pokušava ostvariti komunikacija. Protokol predstavlja skup pravila i standarda koji omogućuju uređajima na mreži da međusobno komuniciraju i razmjenjuju podatke. Samo se jedan protokol se može koristi unutar svake komunikacije. Protokol je zapisan sa svojim skraćenim nazivom pomoću alfanumeričkih znakova.

Akcija označava uspješnost pokušaja komunikacije kroz firewall. Firewall se može smatrati filterom koji može propustiti promet ili blokirati promet. Podatak o uspješnosti propuštanja se zapisuje unutar atributa 'action'. U slučaju da se komunikacija blokira vrijednost atributa za taj zapis će biti 'drop', dok u slučaju da se komunikacija propusti vrijednost atributa 'action' će biti 'accept'.

Zadnji atribut skupa podataka je oznaka, odnosno 'label'. To je binarni atribut koji može poprimiti vrijednost '0' ili '1'. Zapisi u kojima je oznaka postavljena na '0' su normalni uobičajeni zapisi, dok su zapisi koji imaju vrijednost oznake '1' bilježe mrežni promet skeniranja portova pomoću alata Nmap. Unutar skupa podataka postoji 2,327 zapis koji je označen s '1', dok su ostalih 15,665,871 zapisa označeni s '0'.

U svim zapisima su prisutni svi atributi i vrijednosti nisu nasumično generirane. IP adrese su zbog anonimnosti organizacije promijenjene u odnosu na stvarnu vrijednost, ali to ne utječe na funkcionalnost modela. Pošto su dohvaćeni podatci direktno iz organizacije, možemo biti sigurni u vjerodostojnost i kvalitetu samih podataka. Prikaz izgleda zapisa se može vidjeti na slijedećoj slici.

	timestamp	source_ip	destination_ip	destination_port	protocol	action	label
0	2023-02-01 00:00:00	143.55.191.11	143.55.191.132	1947.0	udp	drop	0
1	2023-02-01 00:00:00	143.14.36.100	143.14.39.255	1947.0	udp	drop	0
2	2023-02-01 00:00:00	143.91.187.21	55.107.116.177	2508.0	tcp	drop	0
3	2023-02-01 00:00:00	143.95.220.78	143.95.220.132	1947.0	udp	drop	0
4	2023-02-01 00:00:00	143.219.191.181	143.219.246.69	123.0	udp	accept	0
...	...	...	...	...	...	...	...
15668193	2023-02-28 23:59:58	143.219.191.44	143.219.191.1	99999.0	icmp	drop	0
15668194	2023-02-28 23:59:59	143.91.178.13	143.191.227.17	53.0	udp	accept	0
15668195	2023-02-28 23:59:59	143.176.246.27	143.176.45.22	161.0	udp	accept	0
15668196	2023-02-28 23:59:59	143.2.199.239	143.148.191.20	5555.0	tcp	accept	0
15668197	2023-02-28 23:59:59	143.176.246.27	143.176.45.22	161.0	udp	accept	0

Slika 2: Originalni skup podataka

5. Priprema podataka

Priprema podataka jedna je od ključnih faza u CRISP-DM (Cross-Industry Standard Process for Data Mining) metodologiji, koja ima za cilj osigurati da podaci budu spremni za modeliranje i analizu. Ova faza obuhvaća niz aktivnosti koje uključuju prikupljanje, čišćenje, transformaciju i oblikovanje podataka na način koji omogućava učinkovitu primjenu algoritama strojnog učenja i drugih analitičkih tehnika.[2]

Svrha ove faze je uzeti sirove podatke, koji su opisani u prethodnom poglavlju, i transformirati ih u podatke koji su pogodniji za daljnji rad. Podaci mogu biti krivog formata ili ih treba normalizirati kako ne bi stvarali problem kod daljnjih koraka izrade modela umjetne inteligencije. Kvaliteta podataka uvelike utječe na uspješnost modela, stoga je ova faza od presudne važnosti za cjelokupan proces izrade modela. Čak i ako se koristi najnaprednija analitička tehnologija, loša kvaliteta podataka može dovesti do nepouzdanih rezultata.

Aktivnosti koje uključuje ova faza su odabir podataka, čišćenje podataka, konstrukcija podataka i transformacija podataka.[2] Prilikom izrade modela je potrebno izabrati samo one atribute koji imaju utjecaj na sam model i izbaciti sve nepotrebne koji su nepotrebni. Za te odabrane atribute je potrebno napraviti čišćenje, ako postoji potreba za njim. U ovu fazu ulazi i stvaranje novih atributa iz postojećih kako bi dobili kvalitetnije parametre za model, te je potrebno transformirati sve ostale atribute kako bi bile pogodne za sljedeću fazu izrade modela.

5.1. Transformacija podataka

Analizom podataka se uvodilo kako mali broj atributa može stvoriti problem kod kreiranja modela. Zapravo samo dva, kojima je direktno samo potrebna promjena formata. Ti atributi su IP adresa izvora (source_ip) i IP adresa odredišta (destination_ip). Radi boljeg razumijevanja IP adresa, kreirao se kod koji će pretvoriti IP adrese u njezine dekadske vrijednosti. Programski kod prikazan ispod predstavlja transformaciju podataka.

```
    def ip_to_decimal(ip):
        ip = ip.split('.')
        return
int(ip[0])*256**3+int(ip[1])*256**2+int(ip[2])*256**1+int(ip[3])


    import pandas as pd
    df = pd.read_pickle('data.pickle')
    df['source_ip'] = df['source_ip'].apply(lambda x: ip_to_decimal(x))
    df['destination_ip'] = df['destination_ip'].apply(lambda x:
ip_to_decimal(x))
```

Python programski kod prikazuje funkciju koju smo nazvali ip_to_decimal(). Funkcija prima IP adresu, te ju razdvaja prema znaku '.'. Nakon toga svaki odjeljak pretvara u dekadsku verziju i sumira u jedinstveni broj. Ostali dio koda predstavlja glavni program. U početnim linijama koda se učitavaju svi podatci pomoću pandas biblioteke. Zatim za svaku izvorišnu i odredišnu adresu se poziva prije kreirana funkcija i pretvaraju se IP adrese u broj. Taj broj se sprema u isti atribut u kojem se ranije nalazile IP adrese. Izgled podatak nakon transformacije se može pogledati na slici ispod.

	timestamp	source_ip	destination_ip	destination_port	protocol	action	label
0	2023-02-01 00:00:00	2402795275	2402795396	1947.0	udp	drop	0
1	2023-02-01 00:00:00	2400068708	2400069631	1947.0	udp	drop	0
2	2023-02-01 00:00:00	2405153557	929789105	2508.0	tcp	drop	0
3	2023-02-01 00:00:00	2405424206	2405424260	1947.0	udp	drop	0
4	2023-02-01 00:00:00	2413543349	2413557317	123.0	udp	accept	0
...	...	...	...	...	...	...	...
15668193	2023-02-28 23:59:58	2413543212	2413543169	99999.0	icmp	drop	0
15668194	2023-02-28 23:59:59	2405151245	2411717393	53.0	udp	accept	0
15668195	2023-02-28 23:59:59	2410739227	2410687766	161.0	udp	accept	0
15668196	2023-02-28 23:59:59	2399324143	2408890132	5555.0	tcp	accept	0
15668197	2023-02-28 23:59:59	2410739227	2410687766	161.0	udp	accept	0

15668198 rows × 7 columns

Slika 3: Skup podataka nakon transformacija

5.2. Konstrukcija podataka

U ovom poglavlju će se od postojećih atributa stvoriti novi atributi koji su pogodniji za rad s modelom, odnosno njegovo učenje. Prva transformacija koja se mora napraviti je vremenski zapis. U kasnijim koracima transformacije će se kreirati statistički zapis iz vremena. Za njega nam trenutačni zapisi vremena, u kojima se bilježe minute i sekunde, ne paše. Iz tog razloga moramo prvo mora svesti sve vremenske zapise na sat događaja komunikacije. Nadalje je potrebno grupirati podatke kako bi iz grupiranih podataka mogli izvesti dva nova atributa. Ti atributi su broj komunikacija (communications_count) i broj jedinstvenih portova (unique_ports). Navedene konstrukcije se mogu vidjeti na kodu ispod. Kod se nastavlja na kod prikazan u prijašnjem poglavlju.

```
df['hour'] = df['timestamp'].dt.floor('h')
grouped = df.groupby(['source_ip', 'destination_ip', 'hour', 'label'])
aggregated_data = grouped.agg(
        communications_count=pd.NamedAgg(column='destination_port', aggfunc='size'),
        unique_ports=pd.NamedAgg(column='destination_port', aggfunc=lambda x: x.nunique())
    ).reset_index()
```

Python programski kod prikazuje korištenje funkcije 'floor()' kako bi spustili sve vremenske zapise na njihov sat. Taj zapis se spremilo u novi atribut pod nazivom sat (hour). Nakon toga se grupiraju podatci prema atributima IP adrese izvora, IP adrese destinacije, satu i oznake. Takva grupacija kreira grupe koje označavaju komunikaciju između izvorne i destinacijske adrese unutar određenog sata. Nad novim grupiranim objektu, pod nazivom 'grouped', se odrađuje agregiranje podataka na način da se stvaraju dva nova atributa. Ti atributi su broj komunikacija (communications_count) i broj jedinstvenih portova (unique_ports). Broj komunikacija ima vrijednost veličine destinacijskog porta po grupama. U drugim riječima to znači koliko se različitih komunikacija dogodilo između dvije adrese unutar određenog sata. Broj jedinstvenih portova ima vrijednost veličine destinacijskog porta po grupama koji su jedinstveni. U drugim riječima to znači koliko različitih portova je korišteno u komunikaciji između dvije adrese unutar određenog sata. Ovakvom grupacijom su se izgubili neki atributi poput 'timestamp', 'protocol' i 'action'. Ti atributi nisu potrebni za izradu našeg modela, stoga se nadalje ne pamte. Izgled skupa podataka nakon ovih promjena se može vidjeti na slici ispod.

	source_ip	destination_ip	hour	label	communications_count	unique_ports
0	932934734	932934788	2023-02-01 00:00:00	0	10	2
1	932934734	932934788	2023-02-01 01:00:00	0	9	2
2	932934734	932934788	2023-02-01 02:00:00	0	10	2
3	932934734	932934788	2023-02-01 03:00:00	0	9	2
4	932934734	932934788	2023-02-01 04:00:00	0	10	2
...	...	...	...	...	...	...
432656	2515858588	2515858683	2023-02-28 19:00:00	0	5	1
432657	2515858588	2515858683	2023-02-28 20:00:00	0	5	1
432658	2515858588	2515858683	2023-02-28 21:00:00	0	5	1
432659	2515858588	2515858683	2023-02-28 22:00:00	0	5	1
432660	2515858588	2515858683	2023-02-28 23:00:00	0	5	1

432661 rows × 6 columns

Slika 4: Skup podataka nakon agregacije

Sljedeći korak konstrukcije podataka je kreiranje statističkih podataka o komunikacijama. Veliki broj komunikacija ne mora nužno značiti da se u tom periodu dogodilo skeniranje portova. Veliki broj komunikacija se može dogoditi zbog potrebe za slanjem velike količine poruka. Ono što se treba bilježiti je tko je inicirao komunikacija i tko komunicira sa sumnjivo velikim brojem destinacija. Iz tog razloga se moraju konstruirati ta dva nova atributa. Ta konstrukcija se može vidjeti na kodu ispod. Kod se nastavlja na prijašnje prikazan kod.

```
def source_ip_stats(df):
    ip_stats = pd.DataFrame(columns = ['source_ip', 'as_sip_ip',
'as_sip_hits'])
    s_ip = df['source_ip'].unique()
    as_sip = []
    for ip in s_ip:
        as_sip = df[df['source_ip']==ip]
        as_sip_hits = as_sip['communications_count'].sum()
        as_sip_ip = as_sip['destination_ip'].nunique()
        ip_stats.loc[len(ip_stats.index)] = [ip, as_sip_ip, as_sip_hits]
    df = pd.merge(df, ip_stats, how='inner', left_on='source_ip',
right_on='source_ip')
    return df
aggregated_data = source_ip_stats(aggregated_data)
aggregated_data = aggregated_data.sort_values(by='hour')
aggregated_data = aggregated_data.reset_index(drop=True)
df = aggregated_data
```

U prikazanom Python kodu se može vidjeti deklaraciju funkcije 'source_ip_stats()'. Ta funkcija prima skup podataka iz prethodnih koraka i kreira statistiku za IP adresu izvora. Unutar funkcije se kreira novi pomoćni skup podataka, zvan 's_ip', s atributima IP adresa izvora (source_ip), broj komunikacija s jedinstvenim IP adresama, te broj komunikacija pokrenutih od IP adrese (as_sip_hits). Iz originalnog skupa podatak se izvlače sve jedinstvene IP adrese izvora i pomoću petlje se prolazi kroz sve njene članove. Za svaku adresu se spremaju sve komunikacije u kojima ta adresa ima ulogu izvorne adrese i naknadno iz tog skupa se zbrajaju sve komunikacije. Taj podatak se sprema u atribut 'as_sip_hits'. Na skupu u kojima je odabrana adresa izvorna, se izvadi broj svih jedinstvenih destinacijskih adresa i spremimo u atribut 'as_spi_ip'. Ti novi atributi se nadodaju na skup podataka 's_ip'. Ovaj postupak se napravi za svaku adresu izvora i nakon izlaska iz petlje pomoću 'merge()' funkcije se spoji stari skup podataka s novim skupom koji sadrži statistiku. Funkcija vraća novi skup podataka.

U glavnom programu se primjenjuje novokreiranu funkciju na agregiranim podacima iz prethodnih koraka. vraćeni podatci se sortiraju prema atributu sata, te se resetira indeks. Na slici ispod se može vidjeti izgled skupa podatak nakon konstrukcije statističkih podataka.

	source_ip	destination_ip	hour	label	communications_count	unique_ports	as_sip_ip	as_sip_hits
0	932934734	932934788	2023-02-01 00:00:00	0	10	2	1	5473
1	2409498033	2411692339	2023-02-01 00:00:00	0	19	1	3	27153
2	2399324047	2410148874	2023-02-01 00:00:00	0	103	1	12	756738
3	2400068718	2400068863	2023-02-01 00:00:00	0	33	2	1	14448
4	2413557364	2413543192	2023-02-01 00:00:00	0	4	1	67	289472
...	...	...	...	...	...	...	...	...
432656	2400631242	2400631295	2023-02-28 23:00:00	0	5	1	1	2969
432657	2400631245	2400631295	2023-02-28 23:00:00	0	5	1	2	2858
432658	2413556770	2413556737	2023-02-28 23:00:00	0	19	1	2	10907
432659	2413557251	2413520745	2023-02-28 23:00:00	0	22	1	50	7048
432660	2515858588	2515858683	2023-02-28 23:00:00	0	5	1	1	2853

432661 rows × 8 columns

Slika 5: Skup podataka nakon statistike adrese

Nakon statističkih podataka, potrebna je još jedna konstrukcija atributa vezana uz vrijeme. Pošto je vrijeme cikličko i određena vrsta prometa se tipično događa u istom periodu ciklusa skoro svaki dan. Potrebno je stvoriti atribute koji prikazuju vrijeme kao funkciju sinusa i kosinusa. Na taj način se može lakše naučiti naš budući model o kružnom svojstvu sata i dana u tjednu. Slijedeći kod prikazuje zadnje korake u fazi konstrukcije podataka, a ujedno i pripreme podataka.

```
import numpy as np
df['hour_sin'] = np.sin(2 * np.pi * df['hour'].dt.hour / 24)
df['hour_cos'] = np.cos(2 * np.pi * df['hour'].dt.hour / 24)
df['day_sin'] = np.sin(2 * np.pi * df['hour'].dt.day / 7)
df['day_cos'] = np.cos(2 * np.pi * df['hour'].dt.day / 7)
df = df.drop('hour', axis = 1)
df = df.drop('label', axis = 1)
```

U gornjem Python kodu se može vidjeti pretvorbu atributa sata u atribute kosinusa i sinusa sata, te atribute kosinusa i sinusa dana. Pri tim transformacijama se koristila 'numpy' biblioteka. Nakon spremanja novih podataka u svoje vlastite atribute, koristi se funkciju 'drop()' kako bi se izbrisao atribut sata. Taj atribut se mijenja novim atributima i stoga ga se neće više koristiti. Također se isto napravi s atributom oznakom. Oznaka označava koje su komunikacije generirane Nmap alatom, stoga se taj atribut mora maknuti prije učenja modela. Prije micanja iz modela se spremi atribut u posebnu varijablu, u ovom slučaju 'y', kako bi u kasnijim koracima mogli usporediti rezultat modela s pravim vrijednostima. Finalni izgled skupa podataka se može vidjeti na slici ispod.

	source_ip	destination_ip	communications_count	unique_ports	as_sip_ip	as_sip_hits	hour_sin	hour_cos	day_sin	day_cos
0	932934734	932934788	10	2	1	5473	0.000000	1.000000	7.818315e-01	0.62349
1	2409498033	2411692339	19	1	3	27153	0.000000	1.000000	7.818315e-01	0.62349
2	2399324047	2410148874	103	1	12	756738	0.000000	1.000000	7.818315e-01	0.62349
3	2400068718	2400068863	33	2	1	14448	0.000000	1.000000	7.818315e-01	0.62349
4	2413557364	2413543192	4	1	67	289472	0.000000	1.000000	7.818315e-01	0.62349
...	...	...	...	...	...	...	...	...	...	...
432656	2400631242	2400631295	5	1	1	2969	-0.258819	0.965926	-9.797174e-16	1.00000
432657	2400631245	2400631295	5	1	2	2858	-0.258819	0.965926	-9.797174e-16	1.00000
432658	2413556770	2413556737	19	1	2	10907	-0.258819	0.965926	-9.797174e-16	1.00000
432659	2413557251	2413520745	22	1	50	7048	-0.258819	0.965926	-9.797174e-16	1.00000
432660	2515858588	2515858683	5	1	1	2853	-0.258819	0.965926	-9.797174e-16	1.00000

432661 rows × 10 columns

Slika 6: Skup podataka nakon konstrukcija

6. Modeliranje

Faza modeliranja u CRISP-DM (Cross-Industry Standard Process for Data Mining) metodologiji je ključni korak gdje se podaci, pripremljeni u prethodnim fazama, koriste za stvaranje modela umjetne inteligencije. Cilj ove faze je izraditi matematičke i statističke modele koji mogu identificirati obrasce, predvidjeti buduće događaje ili donositi odluke na temelju dostupnih podataka. Koraci u ovoj fazi metodologije su odabir tehnike modeliranja, podjela podataka na testni i trening skup, izrada modela, te podešavanje parametara.[2]

Modeliranje je iterativni proces koji traje dok se ne ostvare najpoželjnije izlazne vrijednosti modela. Mnoge korake možemo izvršavati više puta. Svakim izvršavanjem bilježiti efektivnost modela, te nakon raznih iteracija odabrati onu najbolju.

U sljedećim poglavljima će se objasniti izbor tehnike modeliranja i razjasniti način na koji funkcionira. Nakon što razumijemo metodu modeliranja, pokazat će se način na koji se izvodi modeliranje pomoću Python programskog jezika. Prije samog učenja modela se naravno podijeli pripremljeni skup podataka na trening i testni skup. Usporedbom uspješnosti modela i promjenama nekih parametara se dobio model koji zadovoljava naše potrebe.

6.1. Odabir tehnike modeliranja

Odabir tehnike modeliranja je izrazito važan korak. Različite tehnike modeliranja su pogodnije za različite problematike, odnosno modele. Proučavanjem tehnika koje bi se mogle primijeniti na problem rada, jedna se skupina tehnika isticala. Pošto je problem, u najjednostavnijim riječima, problem prepoznavanja anomalija u velikom skupu podataka, tehnike koje su se isticale su bile tehnike otkrivanja odstupanja u podacima (eng. Outlier Detectin). U pregledu dosadašnjih radova se zaključilo da takve tehnike pokazuju veliki potencijal, ali imaju i negativne aspekte koje model ovoga rada želi izbjeći.

Postoje razne metode kojima se postiže otkrivanje odstupanja u podacima. Prvi faktor kojeg moramo razmotriti je da model mora imati nenadzirani algoritam učenja. Problem mnogih prijašnje analiziranih modela je bila činjenica da moraju koristiti nadzirano učenje. Takvo učenje otežava pronalazak skupa podataka i čini krajnji rezultat usko povezan s treniranim skupom podataka. Taj problem nije samo kod tehnika otkrivanja odstupanja, već i kod drugih tehnika koje smo analizirali.

Mnogi nenadzirani algoritmi otkrivanja odstupanja imaju određena ograničenja. Mnogi algoritmi pate od takozvanog prokletstva dimenzionalnosti, odnosno točnost i brzina rada opadaju kako se povećava broj podataka. Razlog tome je ciklična procjena gustoće i izračuni udaljenosti između svih podataka. Taj problem, kao i problem nadziranog učenja, ima tehnika MCD iz prethodno analiziranog rada. Prilikom velikog skupa podataka za analizu, model zahtjeva preveliki broj resursa kako bi bio primjenjiv u produkciji. Autor predlaže korištenje novijih algoritama koji ne pate od tih problema.

Drugi problem je taj da većima metoda zahtijevaju podešavanje hiperparametra, što je teško u okruženju bez nadziranja.[16] Takvi parametri se ne nalaze u skupu podataka, već se moraju ručno postavljati i, ovisno o metodu učenja modela, ne funkcioniraju bez njih. Problem nastaje kod učenja modela iz različitih skupova podataka. Za svaki skup podataka bi se trebali podešavati hiperparametri kako bi se izvuklo najviše iz modela. Takvi parametri se najviše koriste kod nadgledanih jer stvaraju poteškoća kod nenadgledanih na način da teško zaključujemo kako parametar utječe na kvalitetu modela. Iz tog razloga je metoda koja se koristi za model rada tehnika ECOD.

ECOD (Empirical Cumulative Distribution Function Outlier Detection) je metoda za otkrivanje odstupanja u podacima, koja se temelji na empirijskoj kumulativnoj distribucijskoj funkciji (ECDF). Metoda koristi empirijsku kumulativnu distribucijsku funkciju kako bi odredila vjerojatnost da neka vrijednost pripada skupu podataka. Na temelju ovih vjerojatnosti, ECOD

identificira podatke koji imaju vrlo nisku vjerojatnost pojavljivanja, odnosno one koji odstupaju od većine podataka.[16]

ECOD rješava prethodno navedene probleme. On je model otkrivanja odstupanja koji su pokazali visoku točnost. U usporedbi s drugim sličnim modelima poput MCD-a, ECOD rješava ključne poteškoće. Problemi tih modela su zahtjevni izračuni koji zahtijevaju puno resursa i brzina obrade velikog skupa podataka. ECOD model omogućuje bolju i manje zahtjevniju obradu podataka koji daje mogućnost realne primjene modela. Ta činjenica ga čini bržim u usporedbi s drugim modelima.[16] Također problem prijašnje analiziranih modela je činjenica da koriste nadzirano učenje i da su jako ovisne o zadanim parametrima. ECOD je model koji ne koristi nadzirano učenje. To mu omogućuje učenja na puno većem skupu podataka. Podaci ne moraju biti označeni kao maliciozni što drastično smanjuje podešavanje skupa za učenje. ECOD model se uči s skupom podataka koji predstavlja normalno ponašanje mrežnog prometa. Algoritam naknadno može usporediti nove testne zapise i učinkovito detektirati odstupanje. Problem kod takvog načina je da moramo biti u potpunosti sigurni na podaci s kojima se uči model ne sadrže maliciozne zapise. Autor MCD modela, iz prethodno analiziranog rada, zaziva za novijim algoritmima koji će unaprijediti iskoristivost tehnika otkrivanja odstupanja. ECOD je jedan od tih novijih algoritama i iz tog razloga ga se koristi u ovome radu.

6.2. Izrada modela

U ovom poglavlju će se prikazati izrađivanje modela. Prvi korak izrade modela je podijeliti pripremljeni skup podataka na skupove koji će biti korišteni za treniranje i za testiranje. Unutar skupa za treniranje je normalan promet, dok se u skupu za testiranje nalaze instance skeniranja portova. Nakon podjele podataka se može započeti treniranje samog modela umjetne inteligencije. Nakon treniranja mora se kreirat prag za anomalije. Svaki zapis koji prijelazi taj prag će se tretirati kao anomalija, odnosno skeniranje porta. Nakon kreiranja praga za anomalije, primijeni se model na testni skup podataka. Nakon ispitivanja se mogu pregledati rezultate. Python koji prikazuje gore navedene korake se može vidjeti ispod.

```
from sklearn.model_selection import train_test_split
train_df, test_df = train_test_split(df, test_size = 0.2, shuffle = False)
from pyod.models import ecod
ecod = ecod.ECOD(n_jobs = -1)
ecod.fit(train_df)
anomaly_scores = ecod.decision_scores_
threshold = np.average(anomaly_scores) + N * np.std(anomaly_scores)
ecod.threshold_ = threshold
anomalies = ecod.predict(test_df)
anomalies.sum()
test_df.iloc[anomalies == 1]
```

U kodu prvo se uključuje biblioteka 'sklearn' biblioteku i pomoću njezine funkcije 'train_test_split()' se kreira testni i trening skup. U skup za testiranje ulazi 20% podataka. Podaci nisu premještani kako se skeniranje portova ne bi stavilo u skup za treniranje. Nakon toga se uključuje biblioteka 'PyOD' i preko njezine funkcije se kreira ECOD model. Modelu se da skup za učenje preko 'fit()' funkcije. Slijedeće linije koda izračunavaju prag anomalija. Pri izračunu praga anomalija se koristi varijabla N. Ta varijabla je parametar koji se može mijenjati kako bi se dobio što precizniji model. Isprobat će se nekoliko vrijednosti, ali one na koje ćemo se fokusirati su 4, 8 i 7. Nakon izračuna praga, koji se nalazi u varijabli 'threshold', testira se model na način da mu se da skup podataka za testiranje. Nakon što model detektira anomalije, ispiše se sumu svih anomalija i detaljno ih se pregleda u testnom skupu. Evaluaciju modela s različitim pragom anomalija izvršavamo u slijedećem poglavlju.

7. Evaluacija

Faza evaluacije u CRISP-DM (Cross-Industry Standard Process for Data Mining) metodologiji predstavlja ključni trenutak u kojem se ispituje uspješnost modela razvijenog tijekom prethodne faze modeliranja. Cilj ove faze je utvrditi koliko dobro model ispunjava zadane ciljeve. Evaluacija je presudna jer omogućava donošenje odluka o tome hoće li se model dalje koristiti, trebaju li se napraviti dodatne prilagodbe ili je potrebno razviti novi model.[2]

U prethodnoj fazi modeliranja su se kreirala nekolicina modela koji će se evaluirali. Model koji bude imao najveću točnost će biti onaj model kojeg će se smatrati rješenjem. Detaljno će se evaluirati tri modela. Modeli se razlikuju prema vrijednosti parametra 'N' u izračunu praga anomalija. Vrijednosti koje 'N' poprima su 4, 8 i 7. U daljnjem tekstu će se nazivati model 1 onaj kojem je 'N' jednak 4, modelu 2 je 'N' jednak 8, te je modelu 3 'N' jednak 7.

Način na koji će se evaluirati modeli je testiranjem točnosti identificiranja prometa koji je generiran skeniranjem portova. To znači da će model morati točno odrediti koji zapis je skeniranje portova i također mora zanemariti normalan ne maliciozan promet. Ovaj uvjet je također i cilj koji smo definirali na početku izrade modela. Ciljana točnost nam je između 99% i 100%. Za izračun točnosti ćemo koristiti formulu:

$$Točnost = \frac{TP + TN}{TP + TN + FP + FN}$$

Jednadžba koriti četiri varijable. TP označava True Positive, TN označava True Negative, FP označava False Positive i FN označava False Negative rezultate. Drugim riječima, točnost je postotak koji se dobije dijeljenjem broja zapisa koji su točno prepoznati s svim zapisima.

Evaluacijom modela 1 se dobije točnost od 99.35% točnosti. Od 257 skeniranja portova, model je detektirao 821 skeniranje s 564 false positive-a. Evaluacijom modela 2 se dobije točnost od 99.92%. Ovaj model je detektirao samo 193 skeniranja što znači da imamo 64 false negative-a. Evaluacijom modela 3 se dobije točnost od 100%. Naravno nisu se testirali svi mogući slučajevi, stoga se kaže da je uspješnost 99.99%. Model 3 je uspješno prepoznao i označio sve zapise u testnom skupu. Iz tog razloga se koristi model 3 kao konačni model.

8. Implementacija

Faza implementacije u CRISP-DM metodologiji predstavlja završni korak u kojem se rezultati prethodnih faza pretvaraju u stvarna rješenja. Glavni cilj faze implementacije je prenijeti model iz okvira eksperimentalnog rada u realno okruženje gdje može ostvarivati svoje ciljeve, bilo kroz automatizaciju procesa, pružanje korisnih predikcija, ili podršku u donošenju odluka.[2]

Ciljevi koji su se postavili u prvoj fazi izrade modela umjetne inteligencije su se ostvarili i dokazali pomoću testnog seta. Daljnja implementacija model bi bila kreiranje sustava koji bi izvlačio zapise svih firewall-ova u određenoj organizaciji, analizirao ih te obavijestio sigurnosne analitičare o anomalijama. Zapisi bi se periodički trebali dostavljati sustavu. Nakon zaprimanja, podaci bi se trebali agregirati u skup, te transformirati u oblik kojeg model može pročitati. Krajem transformacije model bi pokušao pronaći anomalije u skupu podataka, baš kao što se pokazuje u radu. Nakon otkrivanja anomalija, sustav bi trebao obavijestiti analitičara sigurnosti o incidentu preko grafičkog sučelja ili preko obavijesti spajanjem na SMTP server. Druga opcija bi bila implementirati samu provjeru i obavještavanje od strane modela u sam pojedinačni firewall. Ovakva opcija bi bila prikladnije jer bi riješila problem skalabilnosti i zagušenja mreže. S druge strane stvara problem integriranja modela s firewall-om.

Za implementaciju i izradu ovakvog sustava potrebna su dodatna istraživanje i eksperimenti u kojima se dokazuje uspješnost modela u simulaciji mreže. Također bi se trebali testirati modeli trenirani na drugim skupovima podataka koristeći isti ili slične tehnike učenja. Također se treba raspravljati o potrebi ovakvog rješenja s obzirom na komercijalna SIEM rješenjima i moderne firewall-ove koji imaju neke mehanizme detekcije skeniranja portova. Potrebno je usporediti efektivnost modela kreiranih na način koji je opisan u ovome radu s komercijalnim rješenjima kako bi se u daljnjoj budućnosti kreirali sustavi koji su napredniji od sadašnjih.

9. Zaključak

U ovom radu se uspješno razvio model umjetne inteligencije za detekciju skeniranja portova. Završni model je postigao točnost od 99.99% na testnim podacima. Ovaj rezultat potvrđuje učinkovitost korištenja umjetne inteligencije u lovu na kibernetičke prijetnje i na činjenicu da je umjetna inteligencija ključna za povećanje sigurnosti sustava u budućnosti.

Visoka točnost ukazuje na kvalitetu odabrane tehnike učenja modela, te ukazuje na mogućnost daljnjeg razvoja i primjene. Ovaj rad pridonosi širem području kibernetičke sigurnosti pružajući okvir za razvoj sličnih modela koji se mogu koristiti u stvarnim poslovnim okruženjima. Iako su postignuti rezultati iznimno dobri, postoji prostor za daljnja istraživanja. Budući radovi mogu uključivati analizu performansa modela u različitim mrežnim okruženjima i različitim načinima korištenja skeniranja portova. Također u daljnjim radovima možemo proširiti model kako bi imao mogućnost detektiranja većeg broja sigurnosnih prijetnji.

Izrađen je model koji otklanja određene krucijalne mane modela iz prijašnjih istraživanja. Mogućnost nenadgledanog učenja modela daje priliku prikupljanja velikog skupa podataka na kojem bi se modeli učili. Također se eliminira osjetljivost krajnjih rezultata o parametrima i samom uzorku skupa podataka. Model je efikasan i ima mogućnost brzog obrađivanja podataka što ga za razliku od prijašnjih modela i čini iskoristivim u realnoj mrežnoj okolini.

Zaključno, ovaj rad predstavlja značajan korak naprijed u primjeni umjetne inteligencije u kibernetičkoj sigurnost, demonstrirajući kako se suvremene tehnologije mogu koristiti za učinkovitu zaštitu mrežnih resursa.

Popis literature

[1] C. Schröer , F. Kruse, J. M. Gómez, „*A Systematic Literature Review on Applying CRISP-DM Process Model*“, 2020, CENTERIS

[2] N. Hotz, „*What is CRISP DM?*“, 2024, Dana Science Process Alliance, Dostupno: https://www.datascience-pm.com/crisp-dm-2/

[3] Martínez-Plumed, F., Contreras-Ochando, L., Ferri, C., Hernandez-Orallo, J., Kull, M., Lachiche, N. J. A. H., Ramírez-Quintana, M. J., Flach, P. A., „CRISP-DM Twenty Years Later: From Dana Mining Processes to Data Science Trajectories“, 2019, Dostupno: https://research-information.bris.ac.uk/ws/portalfiles/portal/220614618/TKDE_Data_Science_Trajectories_PF.pdf

[4] N. Chinacky, M. J. Schillaci, „Python: Batteries Included“, 2007, Dostupno: https://csc.ucdavis.edu/~cmg/Group/readings/pythonissue_1of4.pdf

[5] M. Uma, G. Padmavathi, „A Survey on Various Cyber Attacks and Their Classification“, 2011, Dostupno: http://ijns.jalaxy.com.tw/contents/ijns-v15-n5/ijns-2013-v15-n5-p390-396.pdf

[6] E. M. Hutchins, Michael J. Cloppert, R. M. Amin, „Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains“, Dostupno: https://www.lockheedmartin.com/content/dam/lockheed-martin/rms/documents/cyber/LM-White-Paper-Intel-Driven-Defense.pdf

[7] Al-Mohannadi, Hamad; Mirza, Qublai K.A.; Namanya, Anitta P. Awan, Irfan U.; Cullen, Andrea J.; Pagna Disso, Jules F, „Cyber-Attack Modeling Analysis Techniques: An Overview“, 2016,Dostupno:https://bradscholars.brad.ac.uk/bitstream/handle/10454/10703/hamad_ICI.pdf?sequence=2&isAllowed=y

[8] T. Yadav, R. A. Mallari, „Technical Aspects of Cyber Kill Chain“, 2016, Dostupno: https://arxiv.org/pdf/1606.03184

[9] A. Orebaugh, B. Pinkard, „Nmap In the Enterprise Your Guise to Network Scanning„, Dostupno:https://gidemy.com/Downloads/Books/University%20and%20College%20Level/Computer%20Science/Networking/Nmap-in-the-Enterprise-Your-Guide-to-Network-Scanningtqw_darksiderg.pdf

[10] J. Kim, N. Shin, S. Y. Jo, S. H. Kim, „Method of Intrusion Detection using Deep Neural Network“, 2019, Dostupno: https://www.researchgate.net/profile/Jin-Kim-43/publication/315469263_Method_of_intrusion_detection_using_deep_neural_network/links/606fb24b299bf1c911ba1c61/Method-of-intrusion-detection-using-deep-neural-network.pdf

[11] M. A. Ferrag, M. Debbah, M. Al-Hawawreh, „Generative AI for Cyber Threat-Hunting in 6G-enabled IoT Networks“, 2023, Dostupno: https://arxiv.org/pdf/2303.11751

[12] H. Domínguez-Limaico, W. N. Quilca, M. Zambrano, F. Cuzme-Rodríguez , E. Maya-Olalla, „Intruder Detection System Based Artificial Neural Network for Software Defined Network“, 2022, Dostupno: https://link.springer.com/chapter/10.1007/978-3-031-11295-9_23

[13] R. Weaver, „Identifying Anomalous Port-SpecificNetwork Behavior“, Software Engineering Institute,2010,Dostupno:https://insights.sei.cmu.edu/documents/829/2010_005_001_15224.pdf

[14] W. El-Hajj, F. Aloul, Z. Trabelsi, „On Detecting Port Scanning using Fuzzy Based Intrusion Detection System“, 2008, Dostupno: https://ieeexplore.ieee.org/document/4599918

[15]A. Webster, M. Gratian, R. Eckenrod, D. Patel, M. Cukier, „An Improved Method for Anomaly-Based Network Scan Detection“, 2015, Dostupno: https://link.springer.com/chapter/10.1007/978-3-319-28865-9_21

[16] Zheng Li, Yue Zhao, Xiyang Hu, Nicola Botta, Cezar Ionescu, and H. George Chen. „Ecod: unsupervised outlier detection using empirical cumulative distribution functions“, 2022, IEEE Transactions on Knowledge and Data Engineering

Popis slika