Politechnika Warszawska
Wydział Matematyki i Nauk Informacyjnych

Inżynieria Cyberbezpieczeństwa
Projekt

Model uczenia maszynowego do wykrywania wybranych anomalii sieciowych

Adrian Stańdo, Mateusz Stączek, Kinga Ułasik

Kierunek: Inżynieria i Analiza Danych
15 czerwca 2022

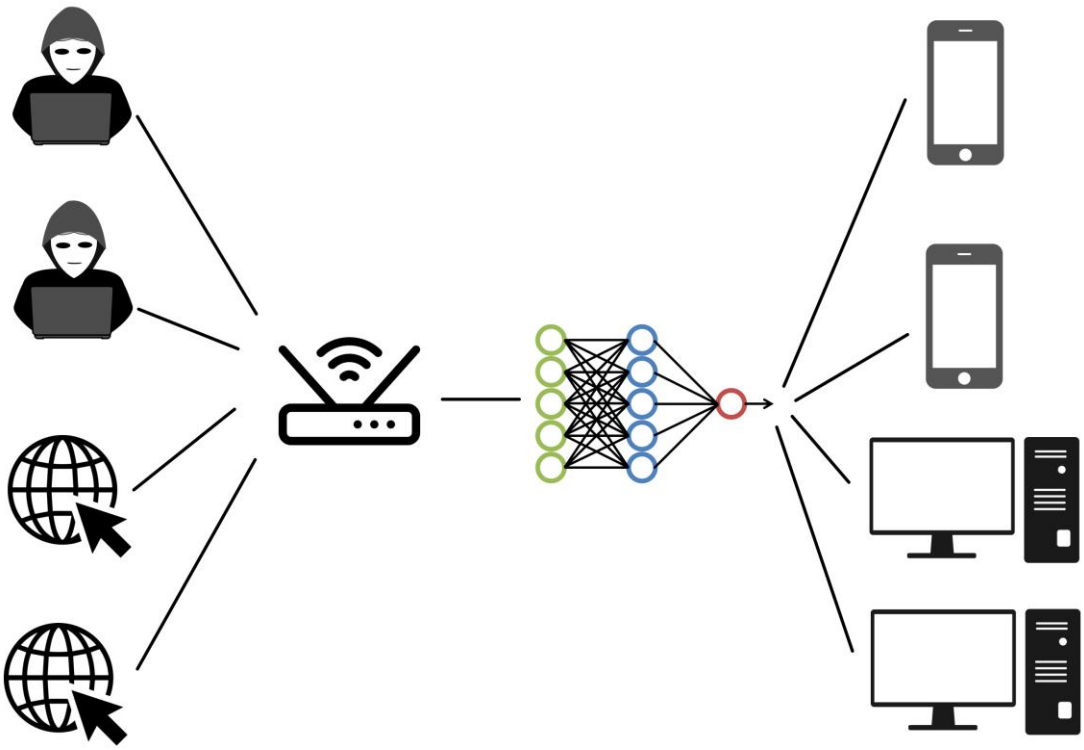
Historia zmian			
Wersja	Data	Kto	Opis
01	2022-03-09	Adrian Stańdo	Wersja wstępna
02	2022-03-13	Adrian Stańdo	Dodanie wstępu i opisu problemu, stworzenie rysunku opisującego przedstawiony problem
03	2022-03-21	Kinga Ułasik	Dodanie analizy problemu, schemat tworzenia rozwiązania
04	2022-03-26	Mateusz Stączek	Opisanie istniejących rozwiązań
05	2022-04-06	Kinga Ułasik	Dodanie planu na rozwiązanie, opisanie propozycji wykorzystanych modeli ML, stworzenie schematu działania drzewa decyzyjnego
06	2022-04-13	Mateusz Stączek	Opis dataset'u
07	2022-06-05	Mateusz Stączek	Omówienie problemów implementacyjnych, testowania aplikacji oraz dodanie wniosków
08	2022-06-07	Kinga Ułasik	Uzupełnienie historii zmian, podziału zadań i wprowadzenie ostatnich poprawek

Podział zadań	
Imię i nazwisko	Zadanie
Adrian Stańdo	Opis problemu, znalezienie zbioru danych, wstępne modelowanie, backend aplikacji, testowanie aplikacji, praca nad prezentacją końcową
Mateusz Stączek	Opis istniejących rozwiązań, backend aplikacji, zarządzanie zespołem, praca nad prezentacją końcową
Kinga Ułasik	Analiza problemu, opis preferowanego rozwiązania, GUI aplikacji, wprowadzenie ostatecznych poprawek dokumentacji, praca nad prezentacją końcową

Spis Treści

1. Opis problemu

Cyberataki budzą coraz większe obawy w Internecie, w szczególności w Internecie Rzeczy (IoT). Wraz z szybko rosnącą liczbą urządzeń i ich możliwościami, rosnąca liczba ataków naraża sieci i systemy informatyczne na niebezpieczeństwa. Przykładowe typy ataków to: Malware, DDoS, MITM, Scan. Wykrywanie takich ataków jest trudnym zadaniem, dlatego w nowoczesnych rozwiązaniach wykorzystuje się techniki uczenia maszynowego.



Rysunek 1. Schemat przedstawiający opisywany problem.

2. Analiza problemu

Wraz z rosnącym zapotrzebowaniem na korzystanie z Internetu wzrasta również ilość danych przepływających między połączonymi urządzeniami, a bezpieczeństwo tych danych jest sprawą wysokiej wagi. Każdy atak związany z bezpieczeństwem sieciowym urządzeń wpływa na zaufanie i poczucie bezpieczeństwa użytkowników i programistów.

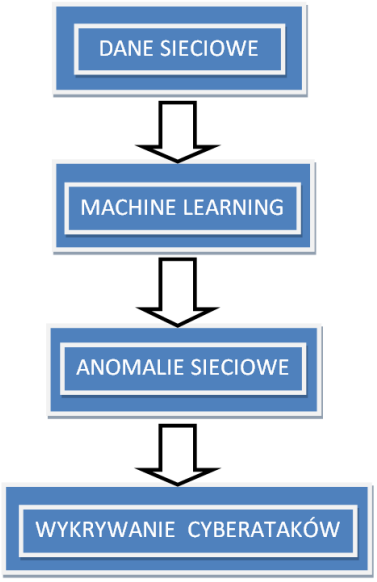
Ataki na sieci mogą również zaburzyć płynne funkcjonowanie państwa. Ataki na infrastrukturę krytyczną, taką jak banki, elektrownie czy systemy łączności wojskowej, mogą zagrozić stabilności państwa lub doprowadzić do katastrofy ekologicznej (np. awarie elektrowni atomowych).

Niewątpliwie, cyberataki są związane z anomaliami w ruchu sieciowym, czyli odstępstwami od normy w ruchu sieciowym. Dane takie jak logowanie użytkowników, próby logowania, przepustowość aplikacji czy wykorzystanie sieci mogą okazać się niezwykle cenne w perspektywie wykrywania ataków. Poprzez pojęcie śledzenie anomalii sieciowych rozumiane jest monitorowanie i śledzenie wszystkich źródeł danych, przewidywanie i identyfikowanie zagrożeń oraz analizowanie informacji od hostów, agentów, użytkowników.

Rozwój dziedzin uczenia maszynowego i uczenia głębokiego w ostatnich latach doprowadziły do powstania rozwiązań wykrywania anomalii sieciowych opartych na tych technikach. Predykcje takich modeli są dokładne, szybkie oraz, przede wszystkim, dużo bardziej efektywne niż gdyby wykonywał je człowiek. Modele takie mogą również znajdować powiązania i korelacje, których ludzie dotąd nie znali, doprowadzając do dalszego rozwoju technik cyberbezpieczeństwa. Z tych wszystkich powodów, modele ML są coraz częściej spotykane w obronnych systemach sieciowych.

Warto również zauważyć, że jeśli specjaliści od cyberbezpieczeństwa zbadają i nauczą się przeciwdziałać danemu typowi ataków, hakerzy na pewno znajdą nowy. Stąd też wynika skuteczność i popularność modeli uczenia maszynowego – ataki nie muszą być dobrze zbadane, wystarczy, że dane z ruchu sieciowego będą tylko poprawnie zaetykietowane jako atak, a taki model można szybko dotrenować w razie nowych zagrożeń. Ewentualnie, można również użyć technik nienadzorowanego uczenia maszynowego, aby wykryć ewentualne nieprawidłowości.

Kroki, jakie są podejmowane przy opracowywaniu kompletnego rozwiązania opartego na modelach uczenia maszynowego przedstawia poniższy rysunek.



Rysunek 2. Schemat przedstawiający kolejne etapy tworzenia rozwiązania postawionego problemu.

Istnieje wiele typów i technik ataków na sieci oraz wciąż powstają nowe. Obecnie, te najczęściej występujące to:

- DDoS (Denial of service) - jest to typ ataku, kiedy przez serwer/router przepływa nietypowa (zbyt duża) ilość ruchu sieciowego. Taki atak powoduje problemy wydajnościowe i w efekcie niedostępność sieci/serwera.
- Malware (Malicious software) - jest to typ ataku, w którym atakujący może uzyskać dostęp do danych na urządzeniu oraz może przejąć kontrolę nad urządzeniami.
- MITM (Man-in-the-middle) - polega na przechwytywaniu/podsłuchiwaniu ruchu sieciowego oraz zmianie pakietów, które do sieci wpływają i wypływają.
- Scan – zbiera informacje o urządzeniach i otwartych portach w celu znalezienia słabych punktów sieci przed poważniejszym atakiem.

3. Istniejące rozwiązania

Skoro jest ryzyko ataku na sieć bądź urządzenie, powstały również narzędzia służące do minimalizacji szkód oraz przeciwdziałania takim atakom. Część dostępnych rozwiązań jest dostępna publicznie, a część jako oprogramowanie lub usługi komercyjne.

Zdecydowanie najpowszechniejszą formą ochrony jest właściwe zaprojektowanie sieci. Liczy się przy tym, w szczególności, opracowanie dobrych reguł, m.in. które pakiety do sieci mogą przychodzić, a które mają zostać odrzucone. Służy do tego firewall. Jego zaletą jest prostota działania - wystarczy "powiedzieć", jakie pakiety mają zostać odrzucone i tak się będzie działo. Nie jest wymagana żadna sztuczna inteligencja do podejmowania decyzji, ponieważ firewall bazuje na do tej pory poznanych już wzorcach.

Jest to równocześnie jego wadą - może okazać się nieprzygotowany na pewne charakterystyczne rodzaje ataków, które są albo nieznane, albo nie zwracają jego uwagi. Przykładowo, jeśli "atak" polegałby na nieznacznej, lecz charakterystycznej zmianie przesyłanych pakietów poprzez dodanie nowych pakietów, należy firewall może się okazać bezradny i skorzystać z innego rozwiązania.

Sprawdzić się być może wykorzystanie oprogramowania służącego do obserwowania całego ruchu sieciowego. Przykładowo, program może "zapamiętać" obecny ruch sieciowy jako "stan normalny" (ang. baseline), by w przyszłości porównywać stan do zapamiętanego i ewentualnie zgłaszać potencjalne anomalie.

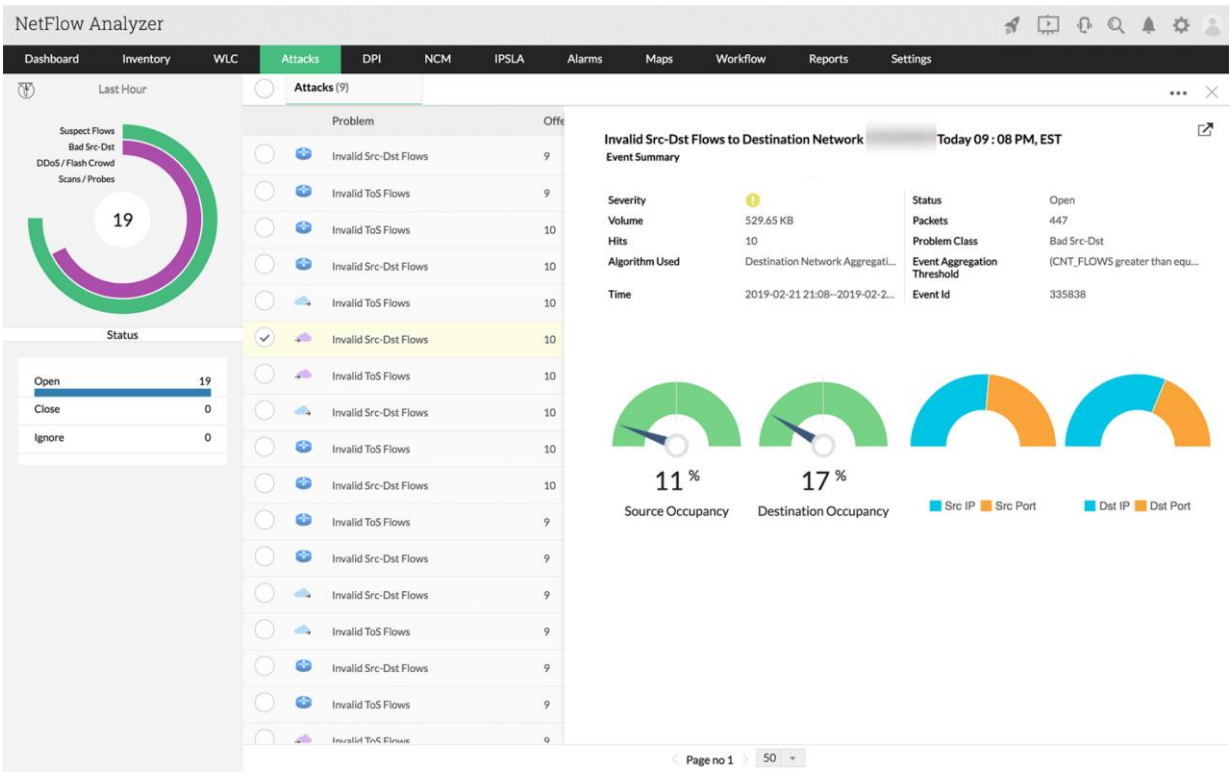
Zarówno z pomocą Cisco NetFlow (funkcjonalność dostępna wg Wikipedii od 1996 r.) jak i innych programów (przykłady za chwilę), możliwe jest obserwowanie obecnych w sieci pakietów i szybkie zauważanie anomalii. Jest powszechnie dostępne na odpowiednich urządzeniach, aczkolwiek podobno nie jest aż tak często używane, jak by się po jego przydatności mogło wydawać. Być może jest to spowodowane tym, że ciągłe monitorowanie całego ruchu sieciowego jest trudne z uwagi na jego objętość.

Zaletą narzędzi przeznaczonych do ciągłego monitorowania ruchu w poszukiwaniu anomalii jest pewna gotowa agregacja i forma przedstawienia wyników. Dla porównania program opensource jakim jest Wireshark potrafi on, owszem, monitorować ruch sieci w czasie rzeczywistym, jak również podglądać wszelkie szczegóły pakietów, jednak nie przedstawi zarejestrowanych danych w formie ułatwiającej zauważenie niebezpieczeństwa bądź nie wyśle powiadomienia.

Przykładowe programy open source służące do ciągłego monitorowania ruchu, to między innymi ElastiFlow oraz NFSen (wraz z NFDump).

Rysunek 3. Interfejs graficzny narzędzia open source służącego do ciągłego monitorowania sieci w poszukiwaniu anomalii. Program: NFSen

Przykładowe programy closed source, dodatkowo występujące w wersjach płatnych, to np. NetFlow Analyzer od ManageEngine. Główna zaleta płatnych rozwiązań to bardziej dopracowany (czasem niestety kosztem funkcjonalności i prezentowanych informacji) interfejs użytkownika, jak również możliwość wsparcia technicznego w przypadku pytań lub problemów.



Rysunek 4. Interfejs graficzny narzędzia closed source służącego do ciągłego monitorowania sieci, między innymi w poszukiwaniu anomalii. Program: NetFlow Analyzer od ManageEngine.

Bardziej zaawansowane narzędzia wykorzystują sztuczną inteligencję i modele specjalnie wytrenowane do tego, aby wśród napływających danych w ilościach z kategorii BigData wyłapać pakiety pochodzące z ataków różnymi sposobami.

Przykładem może być Flowmon ADS od Progress, który wg. dostępnych na stronie informacji używa wielu modeli uczenia maszynowego oraz algorytmów, aby zagrożenia znajdować wcześnie, w czasie rzeczywistym oraz całkowicie automatycznie.

Jak widać, zgodnie z istniejącymi zagrożeniami, istnieje wiele dostępnych narzędzi mających im zapobiegać na różne sposoby. Warto mieć świadomość ich istnienia oraz stosować je nim będzie za późno.

4. Opis preferowanego rozwiązania

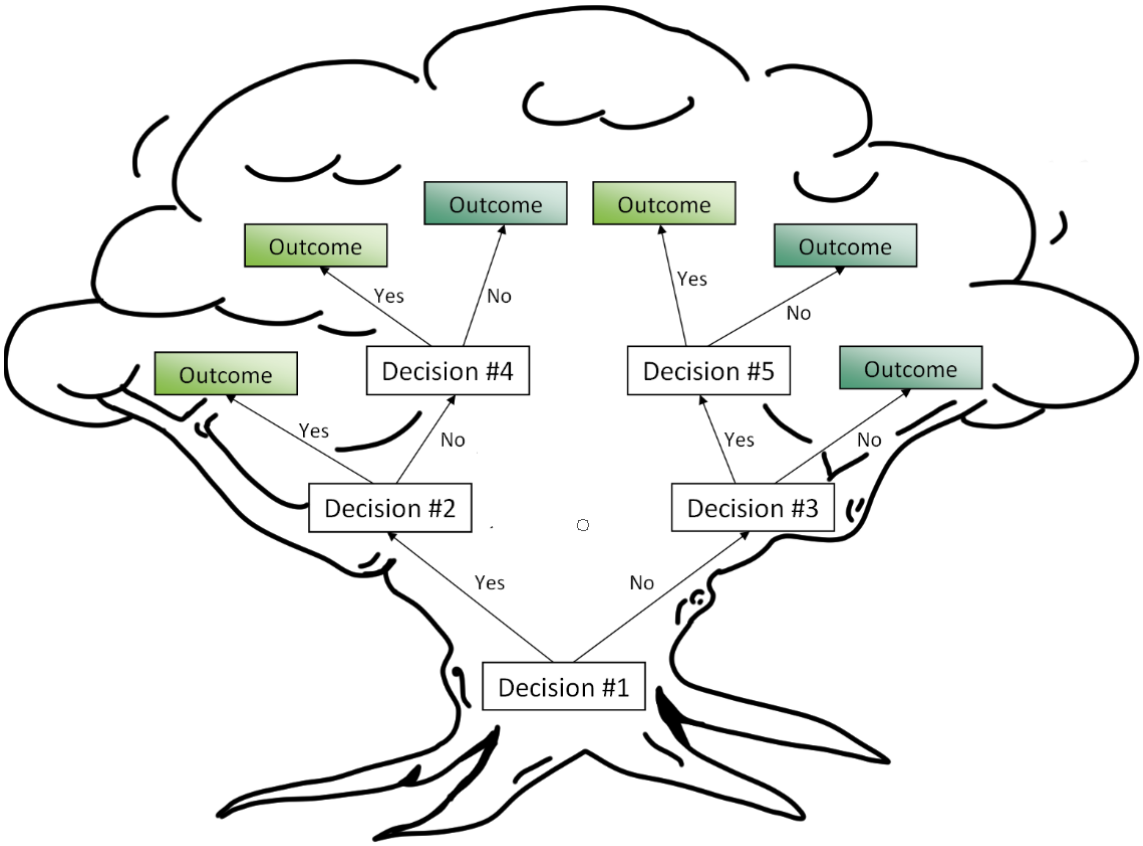
W celu przeciwdziałania cyberatakom wskazanym jest wybranie rozwiązania wykorzystującego bogaty wachlarz możliwości, który rozpościera przed nami Machine Learning. Rozwój tej dziedziny informatyki na przestrzeni ostatnich lat, pozwala na przewidywanie dowolnych wartości na podstawie odpowiednich danych i przy użyciu umiejętności i wiedzy.

Korzystając z nabytego przez nas doświadczenia stworzymy model wykrywający anomalie w ruchu sieciowym trenując modele drzew decyzyjnych wykorzystując algorytmy: Decision Tree, Random Forest oraz Gradient Boosting Machine. Dodatkowo planujemy wypróbować jeden z nowszych modeli wykorzystujący architekturę ludzkiego mózgu, czyli sieci neuronowe.

Decision Tree to popularna kategoria algorytmów uczenia maszynowego powszechnie stosowana w uczeniu nadzorowanym, w którym każda próbka ma zestaw cech i wynik klasyfikacji. Jej struktura przypomina kształt drzewa, w którym każdy węzeł wewnętrzny reprezentuje ocenę atrybutu, każda gałąź reprezentuje wynik wyniku oceny, a każdy węzeł liścia reprezentuje wynik klasyfikacji.

Random Forest to kolejny algorytm nadzorowanej klasyfikacji, tworzy on las drzew decyzyjnych. Model dokonuje predykcji poprzez uśrednienie przewidywanych wyników każdego drzewa składowego, głównym powodem częstego stosowania tego algorytmu jest szybkość jego wykonywania. Jest on bardziej przydatny przy analizowaniu dużych ilości danych.

Gradient boosting machine to algorytm dla zarówno zadań regresji jak i klasyfikacji. Konstruuje on model predykcyjnych na podstawie zespołu drzew decyzyjnych. Główną ideą jest wykorzystywanie gradientu opadania funkcji straty, która opisuje, jak zawodny jest model. Im większa wartość funkcji straty, tym większe prawdopodobieństwo, że model popełni błędy. Jeśli zmiany w modelu sprawiają, że funkcja straty będzie spadać, to oznacza, że model się poprawia.



Rysunek 5. Schemat działania drzewa decyzyjnego

Sieci neuronowe stanowią podzbiór uczenia maszynowego i są sercem algorytmów głębokiego uczenia. Ich nazwa i struktura są inspirowane ludzkim mózgiem, naśladując sposób, w jaki neurony biologiczne przekazują sobie wzajemnie sygnały. Sieci składają się z warstw węzłów, zawierających warstwę wejściową, jedną lub więcej warstw ukrytych oraz warstwę wyjściową. Sieci neuronowe potrafią modelować relacje między zależnościami, które są nieliniowe i złożone, ujawniać ukryte relacje, wzorce i przewidywania, modelować wysoce niestabilne dane oraz wariancje potrzebne do przewidywania rzadkich zdarzeń, właśnie takich jak wykrywanie anomalii.
Po przetestowaniu przedstawionych powyżej algorytmów uczenia maszynowego nastąpi decyzja i wybranie otrzymującego najlepszego wyniki. Następnie należy dostroić hiperparametry oraz przeprowadzić ostateczne testy w celu sprawdzenia, jak końcowo radzi sobie model. Zwieńczeniem pracy nad projektem wykrywania anomalii w ruchu sieciowym będzie stworzenie intefejsu pozwalającego na wygodne i intuicyjne korzystanie ze stworzonego rozwiązania.

Dane, które będą wykorzystane do stworzenia modelu, pochodzą z artykułu [10] i będą przetworzone analogicznie do opisanych tam kroków. Konkretniej, dane oryginalnie pochodzą z plików z formatu *pcap* wygenerowanych przez program Wireshark, na podstawie sztucznie zaprogramowanego ruchu sieciowego. Po przetworzeniu plików programem CICFlowMeter

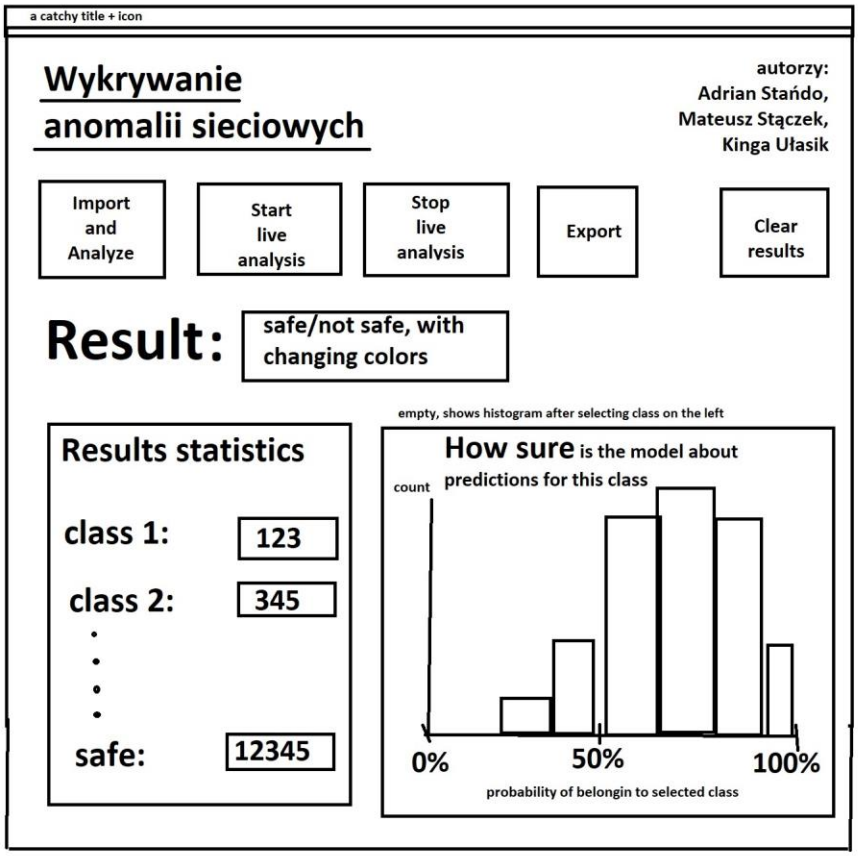
(dostępnym w publicznym repozytorium), otrzymany został zbiór tabularyczny w formacie *csv* zawierający 80 cech, niestety bez wyjaśnienia wszystkich kodowanych liczbowo wartości (dokumentacja programu CICFlowMeter jest niespójna oraz w artykule nie została wskazana wersja tego programu). Zbiór danych był następnie sprawdzony w celu znalezienia i usunięcia zmiennych mocno skorelowanych, a także zmiennych nie zawierających przydanych informacji (kolumny mające jedną unikalną wartość). Na dodatek, ze zbioru zostały usunięte kolumny odpowiadające za adresy IP odbiorców i nadawców, ponieważ te zmienne mogą się zmieniać i model nie powinien ich brać pod uwagę.

5. Wybrane problemy implementacyjne

Całość aplikacji została napisana w języku Python, a w implementacji zostało wykorzystanych kilka popularnych bibliotek. Aplikacja została przetestowana na systemie operacyjnym Ubuntu. Wykorzystane narzędzia i biblioteki to:

1. CICflowmeter - narzędzie wykorzystane do przetworzenia plików z historią pakietów z formatu pcap do formatu csv,
2. pygubu i tkinter oraz PIL – narzędzie graficzne do projektowania interfejsu użytkownika oraz biblioteka do poprawek wcześniej zaprojektowanego interfesju,
3. sklearn – biblioteka, która zawiera wiele gotowych implementacji różnych modeli uczenia maszynowego oraz z której wykorzystaliśmy model Random Forest,
4. pickle, pandas, numpy – biblioteki do zapisu/odczytu wytrenowanego modelu oraz przetwarzania danych i obliczeń.

Projekt graficzny interfejsu został przedstawiony na rysunku poniżej (rys. 6).



Rysunek 6. Wstępny schemat interfejsu aplikacji.

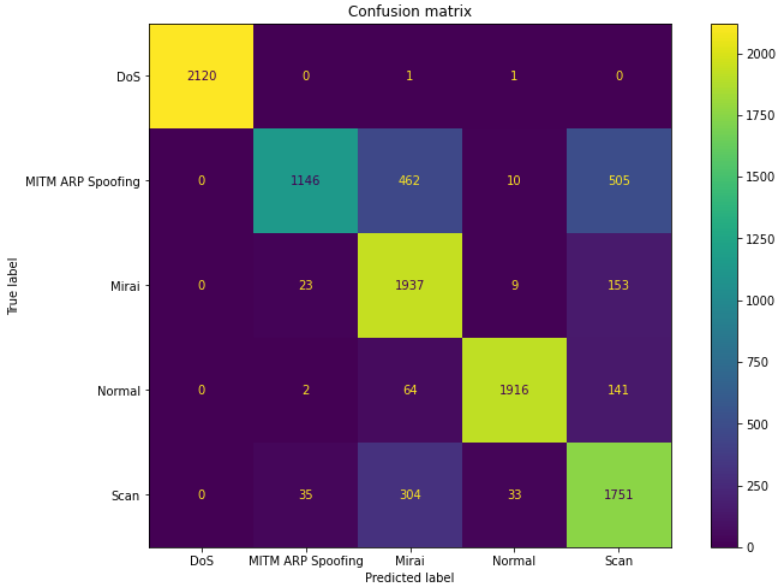
Ostateczna wersja interfejsu użytkownika nie zawiera opcji analizy ruchu sieciowego na żywo - pozwala ona tylko analizować pakiety zebrane wcześniej przez inny program. Niemniej jednak, ta funkcjonalność mogłaby zostać dodana w przyszłych wersjach aplikacji. Nie została ona

jednak zaimplementowana w tej wersji z powodu komplikacji związanych z uprawnieniami aplikacji do dostępu do danych ruchu sieciowego w komputerze.
Model uczenia maszynowego, którym był Random Forest z funkcją podziału węzłów gini oraz 100 drzewami bez limitu głębokości, był trenowany na danych ze sztucznej i małej sieci opisanej w artykule [5] i osiągnął na niej wyniki:

Metryka	Zbiór treningowy	Zbiór testowy
Accuracy	0.8406	0.8358
F1 score weighted	0.8385	0.8332
ROC AUC weighted	0.9783	0.9782

Rysunek 7. Metryki wytrenowanego modelu Random Forest

Macierz konfuzji przedstawia się na danych pochodzących z tamtej sieci dość dobrze:



Rysunek 7. Macierz pomyłek wytrenowanego modelu.

Taki model został przez nas uznany za bardzo dobry, cechujący się poprawnością predykcji na poziomie ponad 80% na zbiorze testowym. Pozostałe metryki wraz z macierzą pomyłek wskazują, że model poprawnie jest w stanie wskazać elementy należące do każdej z 5 klas obecnych w danych. Klasy odpowiadają za 4 typy ataków oraz ruch normalny, nie stanowiący zagrożenia.
Warto tutaj zaznaczyć, że zbiór danych z [5] był zbalansowany, czyli stosunek liczby obserwacji każdego typu ataku był taki sam.

6. Testowanie aplikacji

Końcowa wersja aplikacji jest podobna do zaprojektowanej i ma następujące funkcjonalności:

1. Wczytaj plik csv
2. Wczytaj plik pcap
3. Wykonaj predykcje modelu na wczytanym pliku (plik pcap jest przetwarzany przy użyciu CICflowMaker na csv)
4. Wyświetl predykcje modelu w 2 miejscach:
    a. Duży napis zmieniający kolor i mówiący, czy jest bezpiecznie
    b. Lista, ile pakietów było jakiego typu atakiem. Dostępne typy to:
        i. DDOS (Distributed Denial of Service)
        ii. Mirai (botnet)
        iii. Scan (skanowanie portów)
        iv. MITM (Man in the Middle)

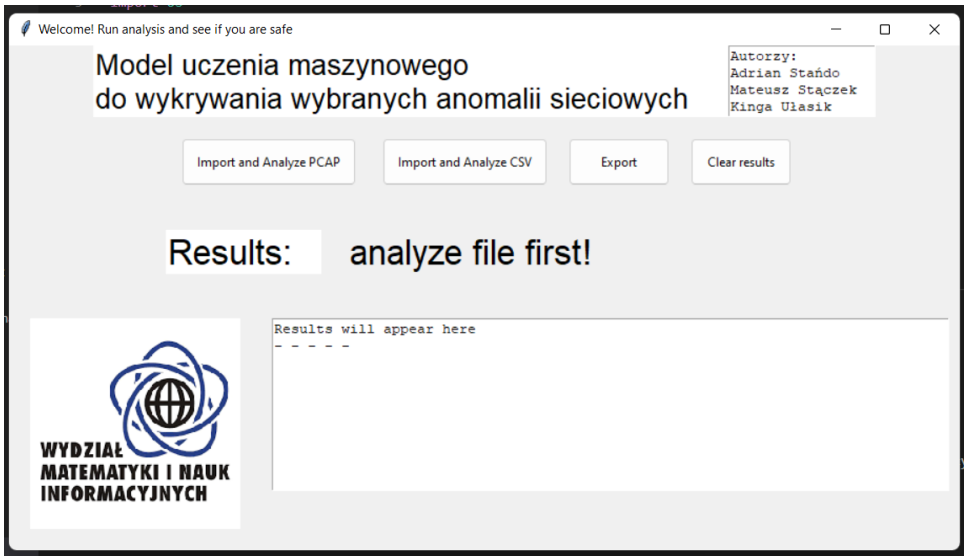
Działanie aplikacji zostało przetestowane na danych pochodzących z artykułu i sztuczniej sieci komputerowej, a także na rzeczywistych danych zebranych na naszych komputerach.

Na danych z artykułu model był trenowany i testowany, gdzie też wykazał się wysokimi wynikami. Dokładnie takie same wyniki otrzymano przy testowaniu modelu poprzez aplikację na tych samym plikach.

Dane z rzeczywistego ruchu zostały zebrane w ciągu kilkudziesięciu sekund, kiedy na komputerze otwierano kilka stron internetowych, czekano aż się załadują, a następnie kończono zbieranie danych. Zbieranie odbywało się przy pomocy programu Wireshark lub przy użyciu wiersza poleceń i polecenia *sudo tcpdump -i any -w example.pcap*.

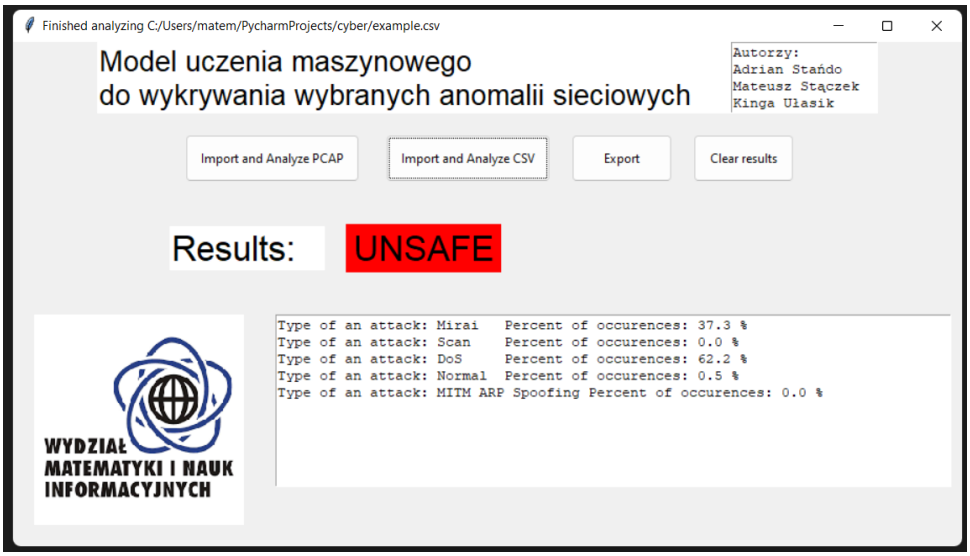
Program, po otrzymaniu danych z ruchu rzeczywistego wykazywał niepokojąco dużo ataków, głównie DDOS. Wszystko zostało powtórzone wielokrotnie i uznaliśmy, że żadnego niebezpieczeństwa ze strony sieci i aktualnie trwających ataków nie ma (tzn. ataków nie ma).
Przyczyną, dlaczego model na danych testowych podobnych do danych ze zbioru treningowego sobie dobrze radził i znajdował wszystkie rodzaje ataków poprawnie oraz dlaczego na danych ze świata rzeczywistego ciągle znajdował niebezpieczeństwa, może być coś o nazwie Data Drift. Pojęcie to oznacza, że dane ze świata rzeczywistego zaczynają znacząco odbiegać od tych, na których model był trenowany, co skutkuje właśnie takimi rezultatami jak błędne predykcje modelu.

Aplikacja po uruchomieniu wygląda następująco:



Rysunek 8. Zaimplementowany schemat interfejsu aplikacji wraz z wynikami analizy przykładowego pliku

Aplikacja po wykonaniu predykcji wygląda następująco:



Rysunek 9. Zaimplementowany schemat interfejsu aplikacji wraz z wynikami analizy przykładowego pliku

7. Wnioski

Otaczająca nas rzeczywistość wymaga przystosowania i zmian, w celu bezpiecznego korzystania z nowych technologii. W szczególności, obecne w Internecie zagrożenia mogą przedostać się do sieci lokalnych niepostrzeżenie oraz pozostać tam przez długi czas niezauważone.

Z tego powodu wykonaliśmy powyższy projekt, który został zakończony powstaniem aplikacji z interfejsem graficznym, która wykorzystuje model uczenia maszynowego, dokładniej Random Forest z biblioteki sklearn trenowany na danych z artykułu [5], do znajdowania, czy analizowany plik z historią ruchu sieciowego zawiera ślady ataków typu DDOS, Mirai, Scan lub MITM.

Jest to przykład jednej z wielu takich aplikacji, które pomagają zwiększać bezpieczeństwo korzystania z sieci w dzisiejszych czasach. Wyzwaniem przy ich tworzeniu jest takie ich zaprojektowanie, aby sprawdzały się w nowych warunkach, przy różnym ruchu sieciowym oraz z różnych urządzeń.

Nasza aplikacja zwraca uwagę na problemy, które mogą się przejawić, gdy dane przekazywane modelowi uczenia maszynowego są znacząco różne od tych wykorzystywanych w trakcie procesu jego uczenia. Problem ten ma nazwę Data Drift i objawił się u nas tym, że aplikacja informowała nas o atakach DDOS oraz Mirai oraz braku ruchu normalnego, gdy jedyne co zrobiliśmy, to otworzyliśmy stronę Google i USOSa.

W przyszłości warto zastanowić się, czy nie wykorzystać w takich modelach czegoś o nazwie online learning, czyli dostrajania modelu w trakcie bieżącej jego pracy w nowych warunkach. W ten sposób, zmniejszone zostanie zagrożenie wystąpienia problemu Data Drift, a model by się był w stanie przystosować do nowych warunków.

8. Literatura

1. ElastiFlow – Oprogramowanie do ciągłego monitorowania ruchu sieciowego: https://docs.elastiflow.com/docs/ (dostęp 26.03.2022) oraz obrazek z prezentacji: https://nsrc.org/workshops/2019/ubuntunet-nren-noc/netmgmt/en/netflow/nfsen.pdf
2. Nfsen – Oprogramowanie do ciągłego monitorowania ruchu sieciowego: http://nfsen.sourceforge.net/ (dostęp 26.03.2022)
3. NetFlow Analyzer od ManageEngine – Closed source oprogramowanie do ciągłego monitorowania ruchu sieciowego: https://www.manageengine.com/products/netflow/ (dostęp 26.03.2022) wraz z obrazkiem z jednej z podstron.
4. Cisco NetFlow: https://en.wikipedia.org/wiki/NetFlow (dostęp 26.03.2022)
5. Ullah, Imtiaz & Mahmoud, Qusay. (2020). A Scheme for Generating a Dataset for Anomalous Activity Detection in IoT Networks. 508-520. 10.1007/978-3-030-47358-7_52.