dr inż. Rafał Kozik
Wydział Telekomunikacji Informatyki i Elektrotechniki
Uniwersytet Technologiczno-Przyrodniczy w Bydgoszczy

Autoreferat

1. Imię i nazwisko: Rafał Kozik

2. Posiadane dyplomy, stopnie naukowe/artystyczne – z podaniem nazwy, miejsca i roku ich uzyskania oraz tytułu rozprawy doktorskiej.

Rok Uzyskania	Tytuł/Stopień	Szczegóły
2013	Stopień naukowy doktora	Wydział Telekomunikacji Informatyki i Elektrotechniki Uniwersytet Technologiczno-Przyrodniczy w Bydgoszczy Tytuł rozprawy: „Złożone algorytmy komputerowej wizji dla celów wspomagania osób niewidomych” Promotor: prof. dr. hab. Ryszard S. Choraś
2008	Tytuł magistra inżyniera	Wydział Telekomunikacji Informatyki i Elektrotechniki Uniwersytet Technologiczno-Przyrodniczy w Bydgoszczy

3. Informacje o dotychczasowym zatrudnieniu w jednostkach naukowych/artystycznych

Rok	Stanowisko	Miejsce
2013-obecnie	Adiunkt	Wydział Telekomunikacji Informatyki i Elektrotechniki Uniwersytet Technologiczno-Przyrodniczy w Bydgoszczy
2008-2013	Asystent	Wydział Telekomunikacji Informatyki i Elektrotechniki Uniwersytet Technologiczno-Przyrodniczy w Bydgoszczy

4. Wskazanie osiągnięcia wynikającego z art. 16 ust. 2 ustawy z dnia 14 marca 2003 r. o stopniach naukowych i tytule naukowym oraz o stopniach i tytule w zakresie sztuki (Dz.U. nr 65, poz. 595 ze zm.), będącego podstawą złożenia wniosku

A. Tytuł osiągnięcia naukowego

Techniki analizy i klasyfikacji danych dla celów zwiększenia bezpieczeństwa aplikacji i sieci teleinformatycznych

B. Osiągnięcie naukowe – jednotematyczny cykl publikacji naukowych

Liczba publikacji wchodzących w skład osiągnięcia	**11**
Liczba publikacji z listy JCR	**7**
Punktacja ministerstwa (z uwzględnieniem procentowego udziału)	**151**
Sumaryczna punktacja ministerstwa	**220**
Sumaryczny IF osiągnięcia	**7,223**

1. **[SCN17] Rafał Kozik [80%],** Choraś M.[20%], Pattern Extraction Algorithm for NetFlow-Based Botnet Activities Detection, Security and Communication Networks, vol. 2017, Article ID 6047053, 10 pages, https://doi.org/10.1155/2017/6047053, 2017 ***IF=0.904***

   Mój wkład:
   - *Zaproponowanie podejścia omówionego w artykule, a w szczególności zaproponowanie architektury systemu, opracowanie algorytmu analizy przepływów sieciowych w oknach czasowych oraz metody uczenia klasyfikatora Random Forest na podstawie próby niezbilansowanej (ang. data imbalance)*
   - *Zaplanowanie i wykonanie opisanych eksperymentów, a w szczególności dobór metryk oceny skuteczności proponowanego rozwiązania*
   - *Analiza uzyskanych wyników*
   - *Kluczowy udział w pisaniu manuskryptu*

   Mój udział procentowy oceniam na: 80%

2. **[PRL18] Kozik Rafał [100%],** Distributing extreme learning machines with Apache Spark for NetFlow-based malware activity detection, Pattern Recognition Letters, Volume 101, 14-20, 2018 ***IF=1.952***

   Praca samodzielna

3. **[JPDC18] Rafał Kozik [50%],** Choraś M.[20%], Massimo F.[20%], Francesco P.[10%], A scalable distributed machine learning approach for attack detection in edge computing environments, Journal of Parallel and Distributed Computing, Volume 119, 18-26, 2018 **IF=1.815**.

Mój wkład:

- *Zdefiniowanie metody omówionej w artykule oraz jego implementacja, a w szczególności opracowanie schematu uczenia klasyfikatora ELM w oparciu o zasoby chmurowe dla zastosowań w obliczeniach brzegowych (ang. edge computing)*
- *Udział w opracowaniu eksperymentów, a w szczególności dobór scenariuszy związanych z IoT i obliczeniami brzegowymi*
- *Analiza uzyskanych wyników*
- *Redagowanie finalnej wersji manuskryptu oraz koordynacja procesu publikacyjnego*

Mój udział procentowy oceniam na: 50%

4. **[CISIM14] Kozik Rafał [65%],** Choraś M. [15%], Renk R.[10%], Hołubowicz W.[10%], A Proposal of Algorithm for Web Applications Cyber Attacks Detection , in Saeed K. and Snasel V. (Eds.): Computer Information Systems and Industrial Management, CISIM 2014, November, Vietnam, Lecture Notes in Computer Science, vol. 8838, 680-687, Springer, 2014 **(publikacja indeksowana w WoS)**

   Mój wkład:

   - *Zaproponowanie podejścia omówionego w artykule oraz jego implementacja, a w szczególności opracowanie kodowania żądań HTTP w oparciu o algorytm kompresji LZW*
   - *Opracowanie i wykonanie eksperymentów, a w szczególności dokonanie porównania proponowanego rozwiązania z innymi metodami*
   - *Analiza uzyskanych wyników*
   - *Koordynacja procesu publikacyjnego*

   Mój udział procentowy oceniam na: 65%

5. **[CISIS15] Kozik Rafał [60%],** Choraś M. [20%], Renk R. [10%], Hołubowicz W. [10%], Patterns Extraction Method for Anomaly Detection in HTTP Traffic, in: Herrero A., Baruque B., Sedano J., Quintan H., Corchado E. (Eds), International Joint Conference CISIS'15 and ICEUTE'15, Advances in Intelligent Systems and Computing, 227-236, 2015 **(publikacja indeksowana w WoS)**

   Mój wkład:

   - *Zaproponowanie podejścia omówionego w artykule oraz jego implementacja, a w szczególności opracowanie algorytmu ekstrakcji struktury żądania dla celów poprawy skuteczności detekcji ataków w warstwie aplikacji.*
   - *Opracowanie i wykonanie eksperymentów, a w szczególności porównanie różnych technik ekstrakcji cech*
   - *Analiza uzyskanych wyników*
   - *Koordynacja procesu publikacyjnego*

   Mój udział procentowy oceniam na: 60%

6. **[IGPL15]** Choraś Michał [50%], **Kozik Rafał [50%],** Machine learning techniques applied to detect cyber attacks on web applications, Logic Journal of the IGPL, vol. 23(1): 45-56, 2015 (published 2014) ***IF=0.461***

   Mój wkład:

   - *Kluczowy udział w opracowaniu podejścia omówionego w artykule oraz jego implementacji, a w szczególności zaproponowanie podejścia grafowego do segmentacji żądań*

- *Udział w opracowaniu scenariuszy badawczych*
- *Redagowanie publikacji*

Mój udział procentowy oceniam na: 50%

7. **[3PGCIC15] Kozik Rafał [70%],** Choraś M.[30%], Adapting an Ensemble of One-Class Classifiers for a Web-Layer Anomaly Detection System , in Proc. Of 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, 3PGCIC, November, IEEE CPS, Cracow, 724-729, 2015 **(publikacja indeksowana w WoS)**

   Mój wkład:
   - Zaproponowanie podejścia omówionego w artykule oraz jego implementacja, a w szczególności opracowanie metody ekstrakcji struktury żądania oraz techniki jego połączenia z algorytmami klasyfikacji
   - Opracowanie i wykonanie scenariuszy badawczych
   - Opracowanie analizy wyników

   Mój udział procentowy oceniam na: 70%

8. **[HAIS16] Kozik Rafał [80%],** Choraś M.[20%], Solution to Data Imbalance Problem in Application Layer Anomaly Detection systems, in Martinez-Alvarez F., Troncoso A., Quintian H., Corchado E. (Eds.): Hybrid Artificial Intelligent Systems, HAIS 2016, LNAI, Springer, vol. 9648, 441-450, 2016 **(publikacja indeksowana w WoS)**

   Mój wkład:
   - Zaproponowanie podejścia omówionego w artykule oraz jego implementacja, a w szczególności zaproponowanie wykorzystania metod uczenia na podstawie próby niezbilansowanej (ang. data imbalance) oraz techniki ekstrakcji struktury żądania z wykorzystanie algorytmu genetycznego
   - Opracowanie i wykonanie scenariuszy badawczych
   - Koordynowanie procesu publikacyjnego

   Mój udział procentowy oceniam na: 80%

9. **[SCN16] Kozik Rafał [80%],** Choraś M.[10%], Hołubowicz W.[10%], Evolutionary-based packets classification for anomaly detection in web layer, Security and Communication Networks, Wiley, vol. 9, Issue 15, 2901-2910, 2016 ***IF=1.067***

   Mój wkład:
   - Zaproponowanie podejścia omówionego w artykule oraz jego implementacja
   - Opracowanie i wykonanie scenariuszy badawczych a w szczególności porównanie różnych metod ekstrakcji struktury żądania oraz technik klasyfikacji
   - Udział w opracowaniu analizy wyników

   Mój udział procentowy oceniam na: 80%

10. **[IGPL17] Kozik Rafał [70%]**, Choraś M. [20%] and Hołubowicz W.[10%], Packets tokenization methods for web layer cyber security, Logic Journal of the IGPL - 2017, 25, 1, 103-113, 2017 (published: 2016) ***IF=0. 575***

   Mój wkład:
   - Zaproponowanie podejścia omówionego w artykule oraz jego implementacja, a w szczególności opracowanie algorytmu grupowanie żądań oraz ekstrakcji ich struktury

- Udział w opracowaniu scenariuszy badawczych, a w szczególności dokonanie porównania różnych metod detekcji ataków i anomalii w warstwie aplikacji
- Udział w opracowaniu analizy wyników

Mój udział procentowy oceniam na: 70%

11. **[IGPL18] Kozik Rafał [75%]**, Choraś M. [25%], Protecting the application layer in the public domain with machine learning methods, Logic Journal of the IGPL, 2018, DOI: https://doi.org/10.1093/jigpal/jzy029 ***IF=0. 449***

    Mój wkład:
    - Zaproponowanie podejścia omówionego w artykule oraz jego implementacja, a w szczególności opracowanie algorytmu ekstrakcji struktury żądania w oparciu o tablice sufiksowe, a także połączenie niniejszej metody z klasyfikatorem Random Forest
    - Udział w opracowaniu scenariuszy badawczych, a w szczególności dokonanie porównania różnych algorytmów
    - Udział w opracowaniu analizy wyników

    Mój udział procentowy oceniam na: 75%

C. Omówienie celu naukowego wyżej wymienionych prac i osiągniętych wyników wraz z omówieniem ich ewentualnego wykorzystania

Wprowadzenie

Tematyka cyberbezpieczeństwa obejmuje szerokie spektrum zagadnień, od problemów czysto technicznych (np. opracowywanie nowych narzędzi do zwalczania cyberzagrożeń), organizacyjnych (np. podnoszenie kompetencji kadry w zakresie cyberbezpieczeństwa), po strategiczne (np. strategia cyberbezpieczeństwa).
W swoich badaniach skupiam się jednak nad dwoma kluczowymi problemami ochrony sieci teleinformatycznej przed cyberatakami:

- **P1.** Analiza zagregowanego ruchu sieciowego w celu wykrycia anomalii i zainfekowanych hostów.
- **P2.** Analiza żądań w warstwie aplikacji w celu ochrony aplikacji przed cyberatakami.

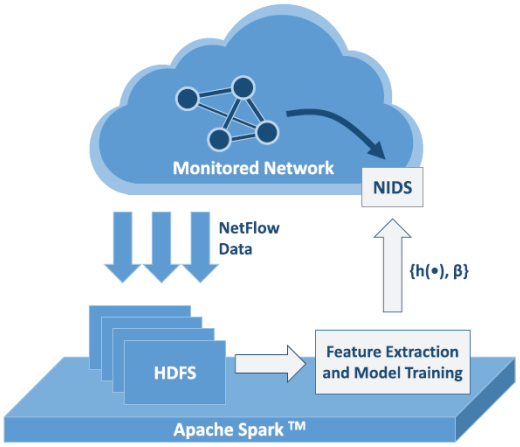
Jeżeli chodzi problematykę analizy zagregowanego ruchu (**P1**), moje badania skupione są głównie wokół analizy zachowania maszyn wchodzących w skład sieci teleinformatycznych. Na podstawie gromadzonego ruchu sieciowego, jaki generują maszyny (hosty), możliwe jest zbudowanie mechanizmów detekcji anomalii, cyberataków oraz symptomów wskazujących, że dany host został zainfekowany złośliwym oprogramowaniem (np. Ransomware, SPAM-bot, etc.).
Jeżeli chodzi o warstwę aplikacji (**P2**), istotność tego tematu potwierdzona jest licznymi raportami[1] oraz doniesieniami z prasy[2]. Aktualnie podatności w systemach webowych są łatwiejsze do wykrycia przez cyberprzestępców ze względu na szereg dostępnych narzędzi (np.

[1] Internet Security Threat Report https://www.websecurity.symantec.com/security-topics/istr-2017-infographic
[2] Włamanie na serwery polskiej komisji wyborczej http://www.locos.pl/index.php/incydenty-epidemie/377-wlamanie-na-serwery-polskiej-komisji-wyborczej

skanery podatności), otwartość i dostępność kodu aplikacji (lub jej części, np. szkieletu do zarządzania treścią) oraz łatwy dostęp do podatnej aplikacji.
W obu obszarach zastosowań (analiza ruchu sieciowego oraz ochrona warstwy aplikacji) moje badania w dużej mierze skupiają się na opracowaniu technik ekstrakcji cech (na podstawie analizowanych danych) i ich klasyfikacji dla celów wykrywania zagrożeń sieciowych.
Przedstawiony cykl publikacji naukowych stanowi osiągnięcie, będące podstawą niniejszego wniosku habilitacyjnego i zawiera kluczowe aspekty moich badań naukowych. Przedstawione one zostały tematycznie z podziałem na wspomniane wcześniej obszary zastosowań.

C.1. Analiza zagregowanego ruchu sieciowego w celu wykrywania anomalii i zainfekowanych hostów

Analiza zagregowanego ruchu sieciowego może być przeprowadzona w oparciu o przepływy sieciowe. Przepływ sieciowy jest definiowany jako strumień pakietów płynący w jednym kierunku pomiędzy zadanym źródłem a punktem przeznaczenia. Przepływy są opisywane poprzez szereg parametrów takich jak adres IP, numer portu (usługi) nadawcy oraz odbiorcy, czas trwania połączenia, liczbę przesłanych bajtów, czy typ protokołu. Analiza przepływów sieciowych jest jedną z popularnych technik przy audytowaniu i monitorowaniu sieci teleinformatycznych. Jednak dane zebrane dla niewielkich rozmiarów sieci informatycznych w kilkugodzinnym okresie czasu charakteryzują się dużym wolumenem. Ponadto, pojedynczy przepływ zazwyczaj nie zawiera wystarczającej ilości informacji do wykrycia anomalnego zachowania maszyn budujących sieć informatyczną. Zazwyczaj wymagany jest dodatkowy element analizy, który pozwoli spojrzeć całościowo (i po różnym kątem) na zebrane dane. Dlatego, w obszarze tym, poszukiwane są rozwiązania wspomagające analizę dużych zbiorów danych (ang. Big Data). W szczególności, w zastosowaniach cyberbezpieczeństwa dają one możliwość efektywniejszego wykrywania złośliwego oprogramowania. Jednocześnie poszerzają one wachlarz narzędzi administratora do walki z cyberzagrożeniami.
W ramach moich badań **[SCN17, PRL18]**, opracowany został nowatorski system analizy ruchu sieciowego dla celów wykrywania anomalii sieciowych. W celu uzyskania dużej skalowalności procesu gromadzenia, analizy i klasyfikacji danych, koncepcja i architektura systemu oparta została of system Apache Spark. Jest to darmowe i wysoce skalowalne rozwiązanie wspomagające tworzenie elastycznego klastra obliczeniowego. Stworzony system i opracowane algorytmy analizują ruch sieciowy gromadzony w postaci przepływów sieciowych (ang. NetFlow). Wyznaczony (w oparciu o klaster obliczeniowy) model detekcji przekazany zostaje bezpośrednio do NIDS (ang. Network Intrusion Detection System). Ogólny schemat ideowy przedstawiony został na Rys. 1.

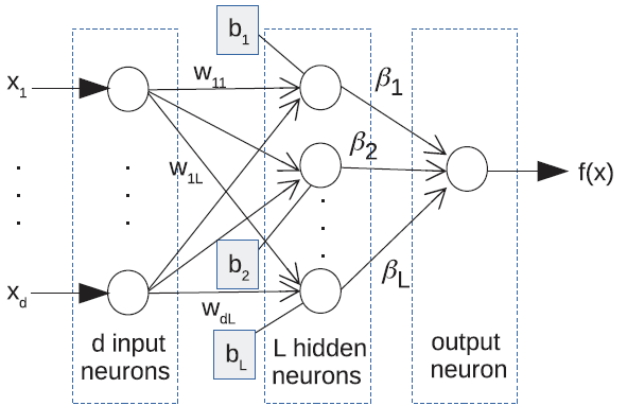


Rys. 1 Ogólny zarys ideowy proponowanego systemu opartego o platformę Apache Spark [JDPC18].

Badania przedstawione w **[SCN17]** w szczególności skupiały się na opracowaniu skalowalnej i innowacyjnej metody ekstrakcji cech oraz analizy ruchu. Aby w pełni wykorzystać zasoby klastra obliczeniowego zaproponowano wykorzystanie rozproszonego systemu plików HDFS[3] (ang. Hadoop Distributed File System) oraz skalowalny wzorzec przetwarzania danych o nazwie MapReduce[4]. Podejście to pozwoliło równolegle przeanalizować zachowanie każdej maszyny (wchodzącej w skład sieci informatycznej) na podstawie zebranych przepływów sieciowych. W celu zbudowania modelu zachowania pojedynczej maszyny opracowano procedurę wyznaczania wektorów cech.
W pierwszej kolejności wszystkie przepływy sieciowe gromadzone są w tzw. oknach czasowych o stałej długości. W następnym kroku, dane grupowane są na podstawie klucza, który stanowi adres IP nadawcy oraz numer okna. Dla każdej grupy przepływów wyznaczone zostają cechy statystyczne takie jak liczba przypływów, sumaryczna ilość przesłanych bajtów, liczba unikatowych docelowych adresów IP (oraz portów) oraz średni czas trwania przepływów. Wyznaczone w ten sposób wektory cech posłużyły do zbudowania modelu zachowania sieci informatycznej. W tym celu wykorzystano klasyfikator Random Forest (RF). Aby ponownie wykorzystać zasoby klastra obliczeniowego zaproponowano użycie rozproszonej implementacji klasyfikatora RF w oparciu o wzorzec przetwarzania danych o nazwie MapReduce. W badaniach poruszono także problem niezbalansowania (ang. Data Imbalance) danych i zaproponowano sposób jego rozwiązania. W celu zweryfikowania proponowanego rozwiązania wykorzystano testowy zbiór danych CTU[5]. Eksperymenty pokazały, iż proponowana metoda posiada większą skuteczność niż inne znane metody porównane na tym zbiorze danych (np. BotHuner, CCD, BClus).
W badaniach przedstawionych w **[PRL18]** dokonano modyfikacji metody przedstawionej w [SCN17] w oparciu o nowatorskie podejście, które wykorzystuje (zaimplementowany we wspomnianej wcześniej architekturze Apache Spark) klasyfikator ELM (ang. Extreme Learning Machines)**.** W proponowanym podejściu oryginalne wektory cech poddane są przekształceniu z wykorzystaniem pojedynczej warstwy neuronów. W kolejnym kroku odpowiedź tej warstwy wykorzystana jest do uczenia dyskryminacyjnego klasyfikatora linowego. Podobnie jak w [SCN17], algorytm ekstrakcji cech bazuje na cechach statystycznych wyznaczanych dla przepływów sieciowych gromadzonych w tak zwanych oknach czasowych. Wektory cech rozproszone zostają pomiędzy maszyny budujące klaster obliczeniowy. Pozwala

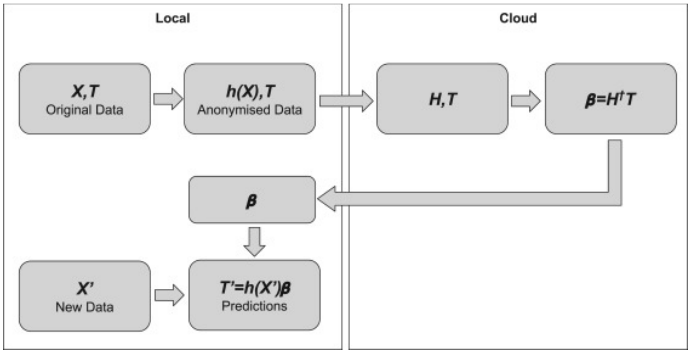
[3] HDFS https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html
[4] MapReduce https://hadoop.apache.org/docs/r1.2.1/mapred_tutorial.html
[5] CTU Dataset https://mcfp.weebly.com/

to zrównoleglić proces uczenia. Klasyfikator ELM wykorzystuje architekturę sztucznej sieci neuronowej z jedną warstwą ukrytą (Rys. 2). W odróżnieniu do tradycyjnej sieci neuronowej procesowi uczenia podlegają wyłącznie wagi w ostatniej warstwie. Dodatkowo, proces uczenia nie ma formy iteracyjnej, a wartości wag wyznaczane są w oparciu o klasyczne narzędzia algebry liniowej.



Rys. 2 Schemat wykorzystanego klasyfikatora ELM.

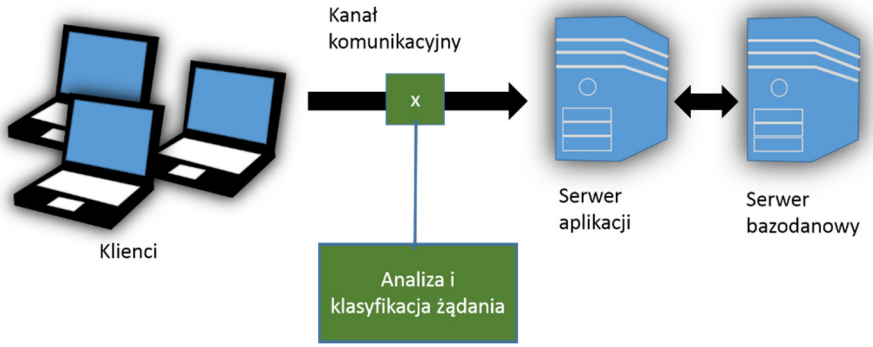
W moich badaniach wykorzystany został fakt, iż dane uczące pozyskane z przepływów sieciowych budują macierz o dużej liczbie wierszy (dużą liczbą pomiarów) i małej liczbie kolumn (12 atrybutów opisujących przepływ). Pozwala to zrównoleglić proces wyznaczania wartości wag ostatniej warstwy. Rezultaty otrzymane na testowej bazie CTU pokazały, iż proponowana metoda pozwala osiągnąć dużą skuteczność wykrywania anomalnego zachowania hostów przy stosunkowo niskim współczynniku fałszywych alarmów.
W **[JPDC18]** zaproponowano wykorzystanie (opisanego wcześniej) klasyfikatora ELM do wykrywania ataków sieciowych w środowisku IoT (ang. Internet of Things). W niniejszym rozwiązaniu zaproponowano wykorzystanie innowacyjnego modelu Security-as-a-Service, gdzie złożone obliczeniowo operacje przetwarzane są w chmurze. W proponowanym podejściu model detekcji, który docelowo wykorzystany zostaje w środowisku IoT, zbudowany zostaje w chmurze (z wykorzystaniem platformy Apache Spark) na podstawie danych uczących. Ideowy schemat przedstawiający algorytm uczenia klasyfikatora pokazany został na Rys. 3.



Rys. 3 Uczenie klasyfikatora ELM w oparciu o przetwarzanie w chmurze.

C.2. Analiza zagregowanego ruchu sieciowego w celu wykrywania anomalii i zainfekowanych hostów

Ochrona aplikacji webowych/sieciowych przed hakerami jest aktualnie jednym z kluczowych wyzwań dla cyberbezpieczeństwa. Szacuje się, iż aktualnie dużo łatwiej jest znaleźć i wykorzystać podatność aplikacji webowej niż dokonać ataku na inny komponent w sieci (np. na system operacyjny serwera). Dodatkowo utrudnieniem dla mechanizmów bezpieczeństwa jest różnorodność funkcjonalna i technologiczna aplikacji webowych[6]. Jest to dużym utrudnieniem przy tworzeniu reguł zapory sieciowej, która ma za zadanie odfiltrować złośliwe żądania. Ponadto, każda z aplikacji może mieć specyficzny protokół komunikacji między klientem a serwerem (np. HTTP, JSON-RPC, RESTFul API, SOAP). Dlatego w tym obszarze istnieje wiele wyzwań związanych z ekstrakcją cech a także budową odpowiedniego mechanizmu detekcji.
W typowej aplikacji webowej dwa kluczowe elementy jej architektury to część kliencka stanowiąca interfejs dostępu (ang. front-end) oraz część serwerowa (ang. back-end) dostarczająca usługi. Zazwyczaj poprzez interfejs dostępowy klient wysyła żądanie do zdalnego serwera i w odpowiedzi otrzymuje wynik. Idea ochrony warstwy aplikacji w takiej architekturze polega na przechwyceniu żądań przesyłanych przez klientów i odpowiednio odfiltrowanie tych złośliwych (np. zawierających umyślnie wstrzyknięty złośliwy kod). Ogólny zarys ideowy pokazano na Rys. 4.



Rys. 4 Schemat ideowy analizy żądań przesyłanych od klienta do serwera.

W tym kontekście w badaniach przedstawionych w **[CISIM14]** zaproponowano nowatorską technikę ekstrakcji cech bazującą na metodzie kompresji LZW[7]. W pierwszej kolejności gromadzone są dane uczące, które zawierają żądania poprawne jak i złośliwe. W przypadku systemu webowego żądania poprawne można pozyskać w trakcie jego użytkowania (np. w trakcie testowania funkcjonalnego). Natomiast żądania złośliwe poprzez szereg narzędzi automatyzujących testy penetracyjne. Niemniej, w opisywanym rozwiązaniu, do porównania wyników, zaproponowałem wykorzystanie ogólnodostępnej bazy testowej CSIC[8]. W pierwszej kolejności, dla zbioru uczącego wyznaczony zostaje słownik w postaci:

$$D: słowo \rightarrow \{i: i \in N\}$$

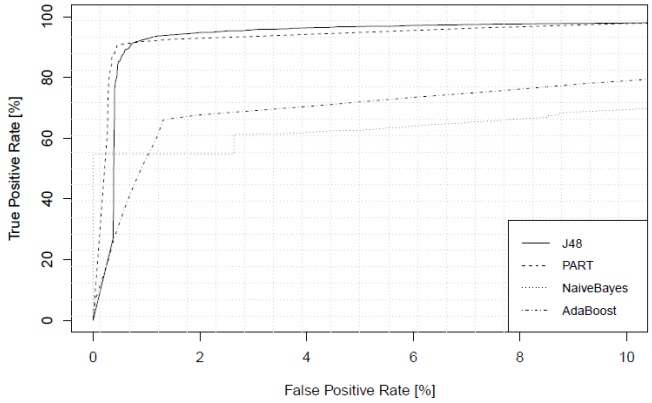
Przekształca on zadany ciąg znaków na reprezentacje liczbową. W opisywanym rozwiązaniu ciąg znaków stanowi całe ciało żądania HTTP, które zostaje przesłane z interfejsu klienta do aplikacji webowej.

[6] OWASP Top 10 https://www.owasp.org
[7] LZW Lempel-Ziv-Welch https://www2.cs.duke.edu/csed/curious/compression/lzw.html
[8] CSIC'10 dataset http://www.isi.csic.es/dataset/

```
Data: Set of HTTP payloads S
Result: Dictionary D
s = empty string
while there is still data to be read in S do
    ch ← read a character
    if (s + ch) ∈ D then
        s ← s+ch;
    else
        D ← D ∪ (s + ch);
        s ← ch;
    end
end
```

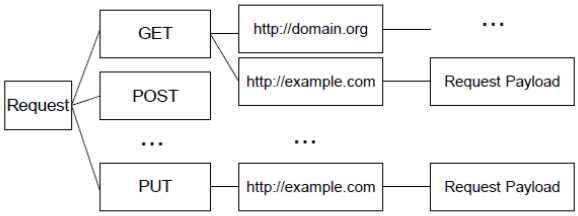
Rys. 5 Algorytm sekwencyjnego budowania słownika.

Wyznaczone z użyciem słownika wektory cech dodatkowo kodowane są z użyciem histogramu. Uzyskany w ten sposób zbiór wektorów o stałej długości, wykorzystany zostaje do uczenia zbioru klasyfikatorów. Proponowana metoda pozwoliła wykryć 96% anomalii, generując przy tym około 3,5% fałszywych alarmów. Krzywą ROC dla różnych metod klasyfikacji pokazano na Rys. 6.



Rys. 6 Krzywa ROC dla wybranych klasyfikatorów.

Badania przedstawione w **[CISIS15]** są kontynuacją metody opisanej w [CISIM14]. W niniejszych badaniach postawiono hipotezę, iż wydzielenie z żądań struktury pozwala na uzyskanie mniejszego końcowego błędu klasyfikacji. W niniejszych badaniach skupiono się na protokole HTTP, który jest jednym z najbardziej popularnych protokołów w warstwie aplikacji. Sam algorytm ekstrakcji struktury na podstawie zawartości żądania jest dwuetapowy. W pierwszej kolejności szereg żądań zostaje pogrupowany według metody żądania (np. GET, POST, PUT, DELETE, itd.) oraz adresu URL zasobu, na którym ma być wykonane dane żądanie. Schemat ideowy tego procesu został pokazany na Rys. 7.



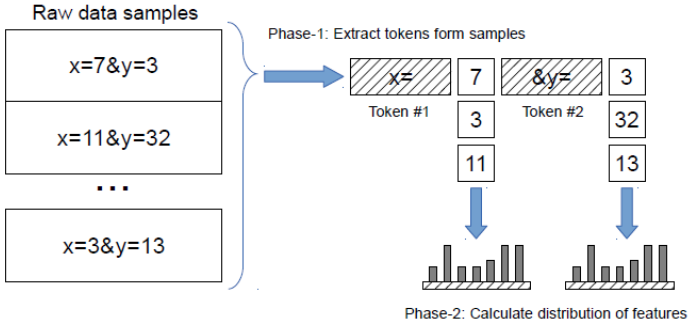
Rys. 7 Schemat ideowy wstępnej segmentacji struktury żądania.

Każde z żądań, które zostaje przydzielone do odpowiedniej grupy, na podstawie metody żądania i adresu URL, posiada ciało żądania (ang. Request Payload), które zawiera dodatkowe parametry. Przykłady parametrów żądania dla metody POST zostały pokazane na Rys. 8.

```
POST /test/demo_form.php HTTP/1.1
Host: w3schools.com
cht=p&chs=500x250&chdl=
first+legend%7Csecond+legend%
7Cthird+legend&chl=first+label%
7Csecond+label%7Cthird+label&
chco=FF0000|00FFFF|00FF00,6699CC|
CC33FF|CCCC33&chp=0.436326388889&
chtt=My+Google+Chart&chts=000000,
24&chd=t:5,10,50|25,35,45
```

Rys. 8 Przykładowa zawartość żądania (wytłuszczony tekst) – metoda POST.

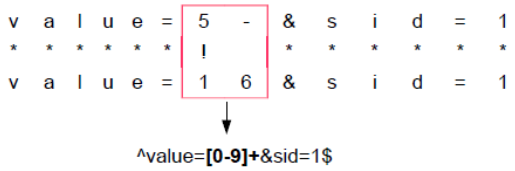
Charakterystyczną cechą ciała żądania jest to, iż jego parametry mogą być zapisane w różnych formatach (np. pola klucz-wartość połączone znakiem '&' lub szereg wartości połączonych znakiem '|'). Typowym formatem zapisu bardziej złożonych obiektów dla współczesnych aplikacji webowych jest notacja JSON (ang. JavaScript Object Notation). Jednak bez względu na sposób zapisu, żądania przesyłane do zadanego adresu URL będą miały podobną strukturę (np. nazwy i wartości parametrów, kolejność ich występowania, itd.). Dlatego w niniejszych badaniach skupiono się na zaproponowaniu algorytmu do jej ekstrakcji.



Rys. 9 Wyznaczenie struktury żądań z wykorzystaniem tokenów.

W proponowanym podejściu struktura żądania zostaje opisana z wykorzystaniem tzw. tokenów. Token definiowany jest jako ciąg znaków, który pojawia się wielokrotnie w zbiorze żądań wysyłanych do zadanego adresu URL. Przykład opisania struktury szeregu żądań HTTP z wykorzystaniem tokenów pokazany został na Rys. 9. Tokeny pozwalają podzielić zawartość żądania HTTP na mniejsze elementy. Ciągi znaków, które znajdują się pomiędzy tokenami opisane zostają z wykorzystaniem histogramu znaków. W ten sposób pojedyncze żądanie posiada (w zależności od struktury) kilka wektorów cech.
W niniejszej pracy, w celu zidentyfikowania potencjalnej listy tokenów, wykorzystana została metoda LZW. W tym przypadku listę tokenów stanowi zbiór ciągów zapisanych w słowniku LZW. Lista ta zostaje w kolejnych krokach przetworzona. W pierwszej kolejności, usunięte są te tokeny, które nie pojawiają się we wszystkich żądaniach do zadanego adresu URL. W drugim kroku usuwane są także te tokeny, które są częścią innych tokenów. Eksperymenty na bazie testowej pozwoliły zweryfikować poprawność hipotezy, iż wydzielenie z żądań struktury pozwala polepszyć proces klasyfikacji.

W **[IGPL15]** opracowano odmienny i nowatorski algorytm ekstrakcji tokenów z żądań HTTP. Proponowana metoda wykorzystuje technikę iteracyjnego grupowania ciągu znaków w coraz to bardziej ogólne komponenty opisane wyrażeniami regularnymi. Przykład złączenia dwóch ciągów „value=5&sid=1” oraz „value=16&sid=1” w jeden komponent opisany wyrażeniem regularnym „^value=[0-9]+&sid=1$” pokazano na Rys. 10.



Rys. 10 Przykład wyniku grupowania dwóch ciągów znakowych.

W proponowanym podejściu zbiór żądań przedstawiony jest jako graf $G = (V, E)$, gdzie V to zbiór wierzchołków (każdy wierzchołek przedstawia pojedyncze żądanie HTTP) a E to zbiór krawędzi przedstawiający podobieństwo miedzy wierzchołkami. Do opisania dystansu między dwoma wierzchołkami (różnicy między dwoma żądaniami HTTP), wykorzystano dystans Levenshteina[9]. W każdej iteracji algorytmu sprawdzona zostaje pojedyncza krawędź grafu. Wierzchołki, dla których dystans jest mniejszy niż ustalony próg zostają złączone i przedstawione jako wyrażenie regularne. Do wygenerowania wyrażenia regularnego na podstawie dwóch ciągów znaków wykorzystany został algorytm Needlemana-Wunscha[10], który znajduje dopasowanie między dwoma ciągami poprzez wprowadzanie modyfikacji (wstawienie pustego znaku, albo usunięcie znaku) do obu z nich. Dla załączonego przykładu pokazanego na Rys. 10, w pierwszym ciągu zostanie usunięta cyfra 5 (znak „!” pod liczbą) i wstawiony jeden znak pusty (symbol „-”). W drugim ciągu zostanie usunięta cyfra 1. Znaki usunięte bądź pokrywające się z przesunięciem (pustym znakiem) zostają opisane z użyciem poniższych wyrażeń regularnych:

- [0-9]+ gdy stanowią one cyfry,
- [a-z]+ gdy stanowią one małe litery,
- [A-Z]+ gdy stanowią one duże litery,
- [znak_specjalny]+ gdy stanowią one znaki specjalne (np. !,*,?, itd.).

Jeżeli wybrane znaki mogą stanowić połącznie kilku z powyższych wzorców (np. małe litery i liczby) opisane zostaną jako [0-9a-z]+. Proponowana metoda pozwoliła uzyskać zbliżone do [CISIM14] rezultaty (94% wykrytych anomalii przy 4% fałszywych alarmów). Jednak wartością dodana w przypadku tej metody jest możliwość bezpośredniego generowania wyrażeń regularnych reprezentujących grupę żądań HTTP. Wyrażenia te mogą być potencjalnie wykorzystane jako sygnatury w zaporze sieciowej albo jako filtry w przy walidacji danych przesyłanych z formularzy do serwera.
W **[3PGCIC15]** zaproponowano dwie modyfikacje w stosunku do [IGPL15] oraz [CISIS15]. Pierwsza z nich dotyczy sposobu ekstrakcji struktury żądania HTTP a druga sposobu detekcji w oparciu o uczenie maszynowe. W odróżnieniu od poprzedniego podejścia, lista N kandydujących tokenów $T = \{t_1, \dots, t_N\}$ wyznaczona zostaje w oparciu o drzewo sufiksowe. Pozwala ono na wyznaczenie zestawu wspólnych podciągów (także najdłuższego wspólnego podciągu), który stanowi listę potencjalnych tokenów. Z listy potencjalnych tokenów należy

[9] Dystans Levenshteina http://www.levenshtein.net/
[10] Needleman, Saul B. & Wunsch, Christian D. (1970). "A general method applicable to the search for similarities in the amino acid sequence of two proteins". Journal of Molecular Biology. 48 (3): 443–53. doi:10.1016/0022-2836(70)90057-4

wybrać podzbiór *S*, gdyż nie wszystkie tokeny z listy *T* wchodzą w skład analizowanych żądań (*P*). Ponadto tokeny te mogą występować w konkretnej kolejności.
Dlatego, w kolejnych iteracjach algorytmu aktualizowana jest sekwencja wybranych tokenów *S* oraz macierz $m_{i,j}$ która utrzymuje informacje na temat wystąpienia *i*-tego tokenu w *j*-tym żądaniu HTTP. Cała procedura opisana jest następujących algorytmem:

```
1  S ← {0}
2  WHILE ∀ p ∈ P, p ∉ ∅: //dopóki żadna z sekwencji nie jest pusta
3    d ← {0}
4    FOR t_i ∈ T:
6      // znajdź najdalszą pozycję tokenu t_i
7      d ← d ∪ max_j m_{i,j}
8
9    END
10   // znajdź token znajdujący się najbliżej początku sekwencji
11   k = argmin_i d_i
12   S ← S ∪ t_k
13   Ze każdej sekwencji p ∈ P usuń znaki poprzedzające i zawierające t_k
14   Zaktualizuj macierz m
15 END
```

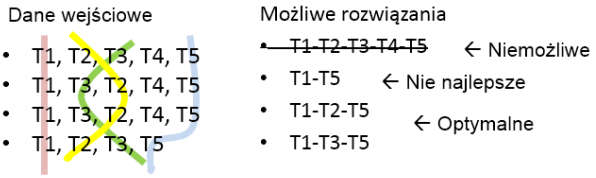
Cały algorytm powtarzany jest dopóki żadna ze sekwencji żądania HTTP nie jest pusta. W pierwszym kroku każdej iteracji przeglądana jest lista tokenów i dla każdego tokenu zapisana jest (w wektorze *d*) możliwie najdalsza pozycja w sekwencjach HTTP. Następnie na podstawie listy *d*, wybierany jest ten token, który znajduje się najbliżej początku analizowanych sekwencji HTTP. Wybrany token zapisujemy w liście rozwiązań *S*. Dodatkowo ze wszystkich sekwencji *p* usuwane są znaki poprzedzające i zwierające token t_k. Ostatnim krokiem każdej iteracji jest zaktualizowanie macierzy *m*.
W dalszym etapie, podobnie jak we wcześniejszych rozwiązaniach, ciągi znaków znajdujących się pomiędzy tokenami zostają opisane z użyciem histogramu. W niniejszych badaniach dokonano również oceny możliwości dalszego polepszenia skuteczności detekcji w oparciu o techniki hybrydyzacji klasyfikatorów (boosting i bagging). W badaniach rozważono dwa klasyfikatory bazowe o nazwach Decision Stump (drzewo decyzyjne z pojedynczym węzłem) oraz Reduced Error Pruning Tree (RepTree).
W **[HAIS16]** zaproponowano nowatorskie podejście wykorzystujące algorytm genetyczny do wyznaczenia struktury na podstawie szeregu żądań HTTP. W proponowanym rozwiązaniu, podobnie jak w [3PGCIC15] lista *N* kandydujących tokenów $T = \{t_1, \dots, t_N\}$ wybrana zostaje w oparciu o drzewo sufiksowe. Jednak stosunkowo odmienny jest sposób ich dalszego doboru celem ekstrakcji struktury żądania. W niniejszej pracy zaproponowano algorytm selekcji tokenów, który został o poparty o globalną funkcję kosztu:

$$C(x) = \sum_{i}^{n} v_i x_i$$

gdzie v_i oznacza wartość tokenu (liczba znaków wchodzących w skład tokenu – faworyzowane są dłuższe tokeny) a $x_i \in \{0,1\}$ fakt iż *i*-ty token (z *n* dostępnych) został wybrany. Optymalizacja funkcji C(x) odbywa się przy warunkach ograniczających zdefiniowanych jako:

$$\sum_{i}^{n} w_i x_i \leq W$$

gdzie w_i oznacza koszt dobru danego tokenu. Koszt ten powiązany jest z pozycją, na jakiej występuje dany token w szeregu żądań. Przykładowo, jeżeli token występuje na końcowej pozycji i zostaje wybrany jako pierwszy, wówczas blokuje on możliwość wyboru tokenów znajdujących się przed nim (pokazano to na Rys. 11).



Rys. 11 Różne kombinacje doboru tokenów (możliwe rozwiązania) dla przykładowych sekwencji danych wejściowych.

W takiej sytuacji w_i otrzymuje dużą wartość. Z kolei wartość W jest wyznaczone eksperymentalnie. Pozwala ona unikać sytuacji, gdzie tylko niewielka liczba tokenów zostaje wybrana jako rozwiązanie.
W dalszym etapie, podobnie jak we wcześniejszych rozwiązaniach, ciągi znaków znajdujących się pomiędzy tokenami zostają opisana z użyciem histogramu. Wektory te użyte zostają do trenowania zrównoważonego klasyfikatora Cost-sensitive AdaBoost. Proponowana metoda pozwala uzyskać duży współczynnik wykrywalności ataków (91.5%) przy niskim stosunku fałszywych alarmów (0.7%).
W **[SCN16]** przedstawiono kontynuacje badań nad ewolucyjną metodą wyznaczania tokenów. W niniejszej pracy rozszerzono znacznie sekcje dotyczącą eksperymentów. W szczególności porównano ze sobą metody sygnaturowe (PHPIDS, ApacheMod) oraz rozwiązania bazujące na detekcji anomalii. Uwzględniono takie podejścia jak:

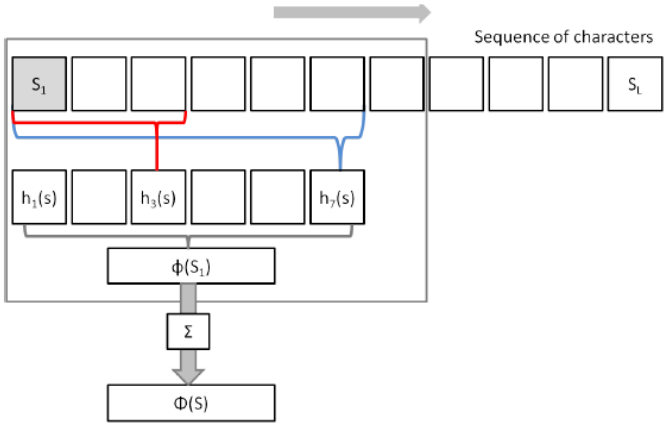
- ICD (Idealised Character Distribution), która wykorzystuje histogram znaków (wyznaczony dla całego żądania HTTP) oraz metrykę Chi-kwadrat.
- ICDSeg, która analizuje argumenty (parametry w postaci *url?paramer1=wartość1&parametr2=wartość2*) żądania HTTP i dla każdej wartości wykorzystuje metodę ICD.
- Kompresyjną[11], która analizują parametry statystyczne żądań po kompresji.
- LVSD, która wyznacza nieparametryczny model klasyfikacji żądań na podstawie dystansu Levenshteina.
- ADS{Chi, RT}, które stanowią połączenie proponowanej metody ekstrakcji struktury z metryką Chi-kwadrat (Chi) oraz klasyfikatorem „Reduced Error Pruning Tree" (RT).

Tab. 1 Porównanie wybranych metod z proponowanym rozwiązaniami (ADSChi oraz ADSRT)

Type	Method	CSIC'10+ True Positive Rate	CSIC'10+ False Positive Rate	CSIC'10+ Precision
Signature-based	PHPIDS	0.2040	0.0125	0.9071
	ApacheMod	0.2630	0.0034	0.9786
Anomaly-based	ICD	0.3320	0.0010	0.9890
	Compression	0.4300	0.0000	1.0000
	LVSD	0.6230	0.0010	0.9970
	ICDSeg	0.8340	0.0110	0.9650
	ADSChi	0.9110	0.0070	0.9810
	ADSRT	0.9190	0.0070	0.9800

[11] Gordon Rueff, Lyle Roybal, Denis Vollmer, SCADA Protocol Anomaly Detection Utilizing Compression (SPADUC), The INL is a U.S. Department of Energy National Laboratory operated by Battelle Energy Alliance, scientific report, 2013

Wyniki porównania zestawione zostały w Tab. 1. Eksperymenty pozwoliły potwierdzić, iż proponowana metoda ekstrakcji struktury pozwala uzyskać lepsze wyniki detekcji (True Positive Rate) przy stosunkowo niskim poziomie fałszywych alarmów (False Positive Rate).
W **[IGPL17]** zaproponowano dodatkowy algorytm grupowania żądań HTTP. Został on zaproponowany w celu uniezależnienia algorytmu ekstrakcji tokenów od protokołu HTTP. W poprzednich rozwiązaniach struktura żądania była analizowana niezależnie dla każdego unikatowego adresu URL. Ogólny schemat przetwarzania żądań HTTP pokazany został na Rys. 12.



Rys. 12 Ogólny schemat przetwarzania ciągów znakowych żądania HTTP (S) do wektora cech ($\Phi(S)$) w oparciu o okno przesuwne i zbiór 7 funkcji haszujących (h), które podlegają konkatenacji (ϕ) i operacji sumowania ($\sum$) dla każdej pozycji okna przesuwnego.

W niniejszym podejściu każde żądanie kodowane jest z wykorzystaniem okna przesuwnego i zbioru 32-bitowych funkcji haszujących w postaci:

$$h: A \rightarrow \{0,1\}^d$$

gdzie *A* to dowolnej długości ciąg znaków alfanumerycznych a *d* jest stałą równą 32. Każda funkcja haszująca generuje 32-bitowy ciąg dla zadanej części analizowanej sekwencji. W kolejnym etapie ciągi bitowe zostają złączone razem w procesie konkatenacji. Taki ciąg bitowy sumowany jest z aktualnym stanem akumulatora (akumulowana jest liczba jedynek na poszczególnych bitach). Następnie okno analizy przesunięte zostaje na kolejną pozycję w sekwencji żądania a cała procedura zostaje wykonana ponownie. Cały algorytm zostaje zakończony, gdy cały ciąg żądania HTTP zostanie przeanalizowany.
Wyznaczone w ten sposób wektory cech opisujące żądanie HTTP, zostają pogrupowane z wykorzystaniem algorytmu k-means. Dla każdego klastra uruchamiana zostaje procedura wyznaczania (i opisu) struktury żądań wchodzących w jego skład.
W niniejszych badaniach (w odróżnieniu do [SCN16]), problem ekstrakcji struktury sformalizowany został, jako problem k-LCS (Multiple Sequences Longest Common Subsequence). Problem LCS dla k=2 zdefiniowany jest jako:

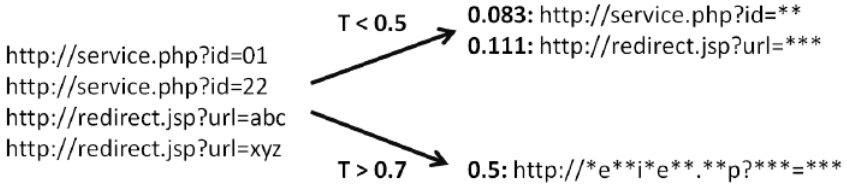
$$LCS(A_i, B_j) = \begin{cases} LCS(A_{i-1}, B_{j-1}) + a_i & a_i = b_j \\ longest(LCS(A_i, B_{j-1}), LCS(A_{i-1}, B_j)) & a_i \neq b_j \\ \emptyset & otherwise \end{cases}$$

Złożoność obliczeniowa tak zdefiniowanego problemu wynosi $O(n^k)$, gdzie *n* to długość najdłuższej sekwencji znaków a *k* to liczba sekwencji. Dlatego w niniejszej pracy wykorzystano algorytm heurystyczny, który w sposób iteracyjny dokonuje dopasowania wybranych par

ciągów znakowych. Innymi słowy wielokrotnie rozwiązuje się problem LCS dla dwóch ciągów znakowych. Procedura przebiega w następujących krokach:

1. W pierwszej kolejności wyznacza się macierz wzajemnego podobieństwa wszystkich analizowanych ciągów znakowych żądań HTTP.
2. Tak zwane drzewo przewodnie jest budowane na podstawie wyznaczonej wcześnie macierzy. Pozwala ono kontrolować przebieg procesu dopasowania poszczególnych par.
3. W pierwszej kolejności dopasowane są pary, które mają największy współczynnik podobieństwa. Dopasowane pary zostają ze sobą złączone.
4. W kolejnych krokach dopasowywane są pary coraz bardziej odległe od siebie.

Jakość końcowego wyniku może być kontrolowana przez globalną wartość progu (wartość *T* pokazana na Rys. 13) na podstawie, którego następuje dopasowanie par.



Rys. 13 Wynik algorytmu analizy sekwencji znaków dla różnych wartości progu T.

Porównanie proponowanej metody z innymi rozwiązaniami pokazany został w Tab. 2. Uzyskano porównywalne wyniki w stosunki do [SCN16]. Niemniej, zaproponowana procedura wstępnego grupowania z wykorzystaniem algorytmu k-means, pozwala uniezależnić proces ekstrakcji struktury od adresu URL żądania HTTP.

Tab. 2 Porównanie proponowanej metody (StrEXT+RF) z innymi algorytmami.

Type	Method	Performance Characteristics	
		True Positive Rate	False Positive Rate
Signature-based	PHPIDS	0.204	0.013
	ApacheMod	0.263	0.003
Anomaly-based	StrExt	0.771	0.01
	ICD	0.697	0.064
	RF	0.849	0.056
	n-grams	0.809	0.142
	StrExt + RF	**0.927**	**0.064**

W [IGPL18] przedstawiono kontynuację badań pokazanych w [SCN16]. W szczególności do ekstrakcji listy kandydujących tokenów wykorzystano odmienną technikę, która wykorzystuje tablice sufiksowe. Dodatkowo, w procesie klasyfikacji wykorzystano modyfikację klasyfikatora Random Forest. Modyfikacja pozwala przypisać odpowiednie wagi za popełniane przez klasyfikator błędy klasyfikacji (np. fałszywe alarmy). Pozwoliło to uzyskać współczynnik fałszywych alarmów na poziomie 5% przy wartości współczynnika detekcji ataków na poziomie 93%.

Rezultaty moich prac, zaproponowane metody i algorytmy mogą znaleźć zastosowanie jako dodatkowe narzędzia polepszające bezpieczeństwo systemów teleinformatycznych. Do tych narzędzi zalicza się:

- Systemy monitoringu ruchu sieciowego dla celów detekcji anomalii w zachowaniu maszyn budujących sieć teleinformatyczną.
- Modele detekcji zagrożeń na podstawie przepływów sieciowych.
- Zapory ogniowe oraz system detekcji anomalii w warstwie aplikacji.
- Systemy filtrowania i walidacji żądań przesyłanych przez klienta do serwera

Do głównych osiągnieć naukowych habilitanta, będących wkładem w dziedzinę informatyki, zalicza się:

1. Opracowanie innowacyjnych metod analizy zagregowanego ruchu sieciowego w oparciu o uczenie maszynowe i narzędzia typu Big Data.
2. Implementacja mechanizmu uczenia klasyfikatora ELM (Extreme Learning Machine) w oparciu o zasoby osadzone w chmurze obliczeniowej.
3. Opracowanie szeregu technik detekcji ataków w warstwie aplikacji w oparciu o analizę zawartości żądań.
4. Opracowanie i zaimplementowanie innowacyjnych metod wyznaczania struktury żądania na podstawie gromadzonych danych w celu poprawy skuteczności wykrywania anomalii i cyberataków w warstwie aplikacji.

5. Omówienie pozostałych osiągnięć naukowo-badawczych

5.1. Przed doktoratem (2008-2013)

Przed uzyskaniem stopnia doktora obszar tematyczny moich badań był odmienny od tych, które prowadziłem po doktoracie.
Przede wszystkim moje badania przed uzyskaniem stopnia doktora skupiały się na metodach przetwarzania obrazów i analizy ich zawartości w różnych rozwiązaniach aplikacyjnych takich jak biometria i wspomaganie osób niewidomych w poruszaniu się. Sumaryczne zestawienie wskaźników bibliometrycznych pokazane zostało w poniższej tabeli.

Sumaryczny IF	0.814
Publikacje z listy JCR	1
Publikacje WoS	21
Cytowania WoS	15
Cytowania WoS bez uwzględnienia samocytowań	13
h-index WoS	2

5.2. Po doktoracie (2013-2018)

Po uzyskaniu stopnia doktora obszar tematyczny moich badań skupił się na zagadnieniach cyberbezpieczeństwa, ochrony infrastruktury krytycznej oraz na zastosowaniach metod uczenia

maszynowego i analizy danych do poprawy jakości procesu wytwarzania oprogramowania. Sumaryczne zestawienie wskaźników bibliometrycznych pokazane zostało w poniższej tabeli.

Sumaryczny IF	14.945
Publikacje z listy JCR	14
Publikacje WoS	58
Cytowania WoS	131
Cytowania WoS bez uwzględnienia samocytowań	86
h-index WoS	6

Poza 7 publikacjami z listy JCR, który zostały wykazane w osiągnięciu habilitacyjnym, jestem także współautorem 7 innych publikacji z listy JCR.
Tematyka moich badań skupia się wokół trzech głównych grup tematycznych, które w wraz z wybranymi publikacjami, przedstawione zostały poniżej.

Ochrona infrastruktury krytycznej i zarządzenie kryzysowe

Główne zagadnienia związane z ochroną infrastruktury krytycznej i zarządzaniem kryzysowym powiązane są z europejski projektami badawczymi, w których uczestniczyłem tj. FP7 CIPRNet, FP7 Inspire oraz FP7 Intersection. W swoich badaniach poruszam takie problemy jak:

- modelowanie kluczowych aspektów cyberbezpieczeństwa [1,2] dla potrzeb symulacji zachowania systemów teleinformatycznych,
- definiowanie dedykowanych usług teleinformatycznych dla potrzeb zarządzania kryzysowego [3],
- możliwości zastosowania algorytmów uczenia maszynowego do modelowaniu zachowania systemów i wzorców ataków [4],

1. Kozik Rafał, Choraś M., Hołubowicz W., Renk R., Increasing Protection and Resilience of Critical Infrastructures - Current challenges and approaches, Journal of the Polish Safety & Reliability Association, vol. 6, number 3, 79-84, 2015
2. Ficco M., Choraś M., Kozik Rafał, Simulation Platform for Cyber-Security and Vulnerability Analysis of Critical Infrastructures, Journal of Computational Science, vol. 22, pp.179-186, 2017 **IF=1.748**
3. Kozik Rafał, Choraś M., Flizikowski A., Theocharidou M., Rosato V., Rome E., Advanced services for critical infrastructures protection, Journal of Ambient Intelligence and Humanized Computing, vol. 6(6), 783-795, Springer, 2015 **IF=0.835**
4. Choraś M., Rafał Kozik, Machine Learning Techniques for Threat Modelling and Detection, Security and Resilience in Intelligent Data-Centric Systems and Communication Networks / Massimo Ficco, Francesco Palmieri, Elsevier, pp.179-192, 2017

Analiza jakości procesu wytarzania oprogramowania

Kolejny obszar moich zainteresowań badawczych wynika bezpośrednio z założenia, iż problemy z występowaniem luk bezpieczeństwa w oprogramowaniu są ściśle powiązane z jakością procesu jego wytwarzania. W swoich pracach [5,6,7,8] skupiam się na:

- aspektach pomiaru jakości wytwarzanego kodu,
- sposobach pomiaru procesu zarządzającego wytwarzaniem oprogramowania,
- metodach oraz algorytmach wczesnym wykrywaniem problemów projektowych, które mogą negatywnie rzutować na jakość kodu.

5. Kozik Rafał, Choraś M., Puchalski D., Renk R., Data Analysis Tool Supporting Software Development Process, In proceedings of 14th International Scientific Conference INFORMATICS, ISBN 978-1-5386-0888-3, IEEE catalog number CFP17E80-PRT, Poprad, pp.179-184, 2018
6. Kozik Rafał, Choraś M., Puchalski D., Renk R. (2019) Platform for Software Quality and Dependability Data Analysis. In: Zamojski W., Mazurkiewicz J., Sugier J., Walkowiak T., Kacprzyk J. (eds) Contemporary Complex Systems and Their Dependability. DepCoS-RELCOMEX 2018. Advances in Intelligent Systems and Computing, vol 761. Springer, Cham
7. Kozik Rafał, Choraś M., Damian P., Rafał R., Q-Radpis Framework for Advanced Data Analysis to Improve Rapid Software Development. Journal of Ambient Intelligence and Humanized Computing https://doi.org/10.1007/s12652-018-0784-5 **IF=1.423**
8. Choraś M., Kozik Rafał, Puchalski D. et al. Increasing product owners' cognition and decision-making capabilities by data analysis approach. Journal of Cognition, Technology & Work. https://doi.org/10.1007/s10111-018-0494-y **IF=1.26**

Przetwarzanie rozproszone

Kolejnym ważnym aspektem moich badań jest wykorzystanie technik i narzędzi przetwarzania rozproszonego do analizy, wizualizacji i detekcji cyberzagrożeń [9]. Ponadto w obszarze moich zainteresowań są także nowe techniki udostępniania usług i aplikacji w środowiskach chmurowych. Rozwiązania te mają na celu pozwolić użytkownikowi na zredukowanie czasu dostępu do usługi [10].

9. Rafał Kozik, "Distributed System for Botnet Traffic Analysis and Anomaly Detection," 2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Exeter, 2017, pp. 330-335.
10. Łaskawiec S., Choraś M., Kozik Rafał. New Solutions for exposing Clustered Applications deployed in the cloud. Cluster Computing Journal, DOI: 10.1007/s10586-018-2443-1 **IF=1.601** (w produkcji)

Aplikacyjne zastosowania metod analizy danych i uczenia maszynowego

Kolejną grupę tematyczną moich zainteresowań i badań stanowią różnego rodzaju zastosowania aplikacyjne technik uczenia maszynowego i analizy danych. Przykładem jest mój udział w badaniach związanych z technikami wykrywaniem anomalii w funkcjonowaniu sieci

teleinformatycznych [11,12] oraz metodami ekstrakcji cech z obrazu w oparciu o uproszczony model kory wizyjnej ssaków [13].

11. Saganowski L., Andrysiak T., Kozik Rafał and Choraś M., DWT-based anomaly detection method for cyber security of wireless sensor networks , Security and Communication Networks, vol. 9, Issue 15, 2911-2922, Wiley, 2016 **IF=1.067**

12. Andrysiak T., Saganowski Ł., Choraś M., and Kozik Rafał. Proposal and comparison of network anomaly detection based on long-memory statistical models, Logic Journal of IGPL, vol. 25, no. 6, 944-956, 2016 **IF=0.575**

13. Rafał Kozik, A simplified visual cortex model for efficient image codding and object recognition, Image Processing and Communications Challenges 5 / Ed. Ryszard S. Choraś, Berlin, Heidelberg : Springer-Verlag, 2014