

POLITECHNIKA WARSZAWSKA

DYSCYPLINA NAUKOWA
INFORMATYKA TECHNICZNA I TELEKOMUNIKACJA
DZIEDZINA NAUK INŻYNIERYJNO-TECHNICZNYCH

Rozprawa doktorska

mgr inż. Jędrzej Tadeusz Bieniasz

Rozproszone metody ukrywania informacji w sieciach

Promotor
dr hab. inż. Krzysztof Szczypiorski, prof. Uczelni

WARSZAWA 2022

Streszczenie

Niniejsza rozprawa doktorska przedstawia rezultaty badań teoretycznych i praktycznych dotyczących różnych aspektów rozproszonych metod ukrywania informacji w sieciach. Steganografia, w tym steganografia sieciowa i rozproszona, w ciągu ostatnich 20 lat była intensywnie badana pod kątem możliwości jej stosowania w działaniach ofensywnych w cyberprzestrzeni. Wzrost popularności jej zastosowania przez atakujących widoczny jest w treści raportów opisujących incydenty bezpieczeństwa komputerowego. Z drugiej strony techniki ukrywania informacji wpisujące się w szeroko pojętą dziedzinę metod *cyber deception* mogą być wykorzystane w zadaniach cyberobrony systemów teleinformatycznych oraz ich użytkowników.

W ramach prac badawczych podjęto zagadnienie rozproszonych metod ukrywania informacji w sieciach w czterech komplementarnych kierunkach:

1. zbadanie charakteru i możliwości stosowania metod rozproszonej steganografii w ofensywnych operacjach w cyberprzestrzeni,
2. rozszerzenie metodyk modelowania cyberzagrożeń poprzez uwzględnienie technik ukrywania informacji w sieciach,
3. realizacja kompletnej ścieżki badań na danych na potrzeby detekcji metod steganografii rozproszonej – od stworzenia symulacji metod steganografii, przez implementację prototypu systemu detekcji, po referencyjne zastosowanie metod *data science*,
4. zastosowanie technik ukrywania informacji jako mechanizmu protekcji systemów i użytkowników w podejściu *cyberfog*.

Przedstawione osiągnięcia rozwijają stan wiedzy w dyscyplinie informatyka techniczna i telekomunikacja oraz stanowią wkład do rozwoju nauk o cyberbezpieczeństwie (ang. *science of cybersecurity*) – od poznawania metod cyberatakujących, przez budowanie rozwiązań cyberobrony opartych na tej wiedzy i danych, po proponowanie nowych mechanizmów bezpieczeństwa sieci.

Słowa kluczowe: cyberbezpieczeństwo, steganografia sieciowa, ukrywanie informacji w sieciach, cyberobrona, nauka o danych, bezpieczeństwo sieci, systemy wieloagentowe, wykrywanie cyberzagrożeń, wykrywanie włamań do sieci

Abstract

This thesis presents the results of theoretical and practical research on various aspects of distributed methods of hiding information in networks. In the last 20 years steganography, including network and distributed steganography, has been intensively researched for its applicability for both offensive and defensive activities in cyberspace. The growing popularity of its use by attackers is clearly shown in the computer security incident reports. On the other hand, steganography as a part of a broader domain of *cyber deception* techniques could provide several capabilities for cyber defense.

In this work the distributed methods of information hiding in networks were examined in four complementary directions:

- research on the nature and applicability of distributed methods of steganography: to offensive operations in cyberspace,
- extension of modeling cyber threats with information hiding techniques in networks applied,
- realization the end-to-end research process for the detection of distributed steganography methods – from creating simulation testbed of steganography methods, through the implementation of a prototype of the multi-agent detection system to the reference applications of *data science* algorithms,
- application of information hiding techniques as a cyber protection solution for systems and users aligned with cyberfog concept.

The presented research contributes to expanding the state of knowledge in the discipline of *Information and Communication Technology*. The results of the work fall within the scope of *science of cybersecurity* – from learning of cyber offensive methods, through building cyber defense solutions based on this knowledge and data, to proposing new network security mechanisms.

Keywords: cybersecurity, network steganography, information hiding, cyber defense, data science, network security, multiagent systems, cyber threat detection, network intrusion detection

Spis treści

1. Wstęp	9
1.1. Cel rozprawy	9
1.2. Motywacja	10
1.3. Zakres rozprawy – cykl publikacji	10
2. Badania nad realizacją rozproszonych metod ukrywania informacji w sieciach	15
2.1. Podsumowanie artykułów [A1] i [A2]	15
2.2. Osiągnięcia badawcze autora	18
3. Badania nad wykrywaniem rozproszonych metod ukrywania informacji w sieciach	20
3.1. Cyberobrona oparta na wiedzy i danych – od modelowania do wykrywania cyberzagrożeń	20
3.1.1. Podsumowanie artykułu [A3]	20
3.1.2. Osiągnięcia badawcze autora	22
3.2. Monitorowanie i detekcja kanałów steganograficznych w podejściu rozproszonym	23
3.2.1. Podsumowanie artykułów [A4] i [A5]	23
3.2.2. Osiągnięcia badawcze autora	29
4. Badania nad defensywnymi zastosowaniami rozproszonych metod ukrywania informacji w sieciach	31
4.1. Podsumowanie artykułów [A6] i [A7]	31
4.2. Osiągnięcia badawcze autora	33
5. Podsumowanie	35
6. Dorobek naukowy autora	39
6.1. Publikacje	39
6.2. Wystąpienia konferencyjne	42
6.2.1. Konferencje naukowe	42
6.2.2. Konferencje branżowe	42
6.3. Projekty	43
6.4. Nagrody i wyróżnienia	43
A. Artykuły	51
A.1. Artykuł [A1]	52

A.2. Artykuł [A2]	57
A.3. Artykuł [A3]	81
A.4. Artykuł [A4]	108
A.5. Artykuł [A5]	130
A.6. Artykuł [A6]	152
A.7. Artykuł [A7]	157
B. Oświadczenia współautorów	174

1. Wstęp

1.1. Cel rozprawy

Celem niniejszej rozprawy doktorskiej jest opracowanie i ocena skuteczności metod podnoszenia cyberbezpieczeństwa w kontekście dwóch obszarów zastosowania rozproszonych metod ukrywania informacji w sieciach:

1. inteligentnego wykrywania i reagowania na cyberzagrożenia wykorzystujące rozproszone metody ukrywania informacji,
2. mechanizmów ochrony sieci opartych na rozproszonych metodach ukrywania informacji.

Cel ten osiągnięto poprzez:

1. Rozszerzenie koncepcji steganograficznego systemu plików SocialStegDisc opracowanego na podstawie metody StegHash, implementacja tego systemu oraz weryfikacja jego aspektów działania wraz z oceną bezpieczeństwa. Dodatkowo wyniki badań mają wskazać możliwość zastosowania metod StegHash i SocialStegDisc w rozwiązaniach bezpieczeństwa sieci zgodnych z koncepcją *cyberfog*.
2. Zaproponowanie nowego sposobu modelowania komunikacji steganograficznej złośliwego oprogramowania za pomocą kanałów Command & Control (C2). Dotychczas istniejące metodyki modelowania cyberzagrożeń, takie jak Cyber Kill Chain i MITRE ATT&CK, nie uwzględniały wprost steganografii wśród technik ofensywnych.
3. Zaprojektowanie, zaimplementowanie i ocena nowego rozwiązania w zakresie detekcji metod ukrywania informacji przy zastosowaniu koncepcji systemów wieloagentowych. Na potrzeby tego zadania zrealizowanego ścieżkę badań danych od stworzenia symulacji metod steganografii przez implementację prototypu systemu detekcji po referencyjne zastosowanie metod analizy danych (ang. *data science*). Istotnym aspektem opracowanego rozwiązania jest zdolność do wykrywania węzłów sieciowych, stanowiących źródło ataków.
4. Opracowanie i zaimplementowanie nowego systemu komunikacyjnego, wykorzystującego rozproszoną steganografię opartą na wcześniejszych metodach badanych przez autora – StegHash i SocialStegDisc oraz dopasowaną do podejścia *cyberfog*. Nie zaprezentowano dotychczas praktycznej realizacji takiego systemu zgodnego z koncepcją *cyberfog*, podnoszącą dzięki temu odporność sieci teleinformatycznych na cyberataki.

1.2. Motywacja

Zaproponowane w rozprawie rozwiązania oparte są na założeniu, że efektywna cyberobrona we współczesnym środowisku teleinformatycznym polega na zdolności do budowy **mechanizmów cyberbezpieczeństwa adekwatnych do konkretnych działań**, które atakujący mogą podjąć w celu **zaatakowania tego środowiska**. W takim podejściu do cyberobrony istotne jest stosowanie metod ilościowych i jakościowych do oceny wydajności czy skuteczności detekcji oraz reagowania na cyberzagrożenia. Na tej podstawie możliwe jest podejmowanie strategicznych decyzji, które prowadzą do uzupełnienia luk w zdolnościach do cyberobrony i podnoszenia całościowego poziomu cyberodporności. **Opieranie decyzji strategicznych i ocen efektywności dotyczących cyberbezpieczeństwa na dogłębnym zrozumieniu technik rozproszonego ukrywania informacji stanowi podstawę nowoczesnego podejścia do inteligentnej cyberobrony**. Wskazane zagadnienia odniesione do literatury przedmiotu dotyczą takich pojęć, jak: *intelligence-driven cyber defense* [1], *intelligence-driven incident response* [2], *data-driven network intrusion detection* [3], *cybersecurity data science* [4] oraz *big data analytics for information security* [5].

1.3. Zakres rozprawy – cykl publikacji

Osiągnięciu celu rozprawy służą zrealizowane przez autora badania teoretyczne i praktyczne różnych aspektów rozproszonego ukrywania informacji w sieciach. Rezultaty tych badań przedstawia seria 7 artykułów, na której oparta jest niniejsza rozprawa doktorska:

- [A1] Bieniasz, J., & Szczypiorski, K. (2017). **SocialStegDisc: Application of steganography in social networks to create a file system**. 2017 3rd International Conference on Frontiers of Signal Processing (ICFSP), 2017, pp. 76-80. DOI: <https://doi.org/10.1109/ICFSP.2017.8097145>. (► str. 53)

Rodzaj publikacji: **materiały pokonferencyjne** (IF: –; liczba cytowań: **7**)

Punkty ministerialne (1.12.2021): **20**

Wkład autora rozprawy: **75%**

- [A2] **Bieniasz, J., & Szczypiorski, K. (2019). Methods for Information Hiding in Open Social Networks.** Journal of Universal Computer Science, 25(2), 74-97. DOI: <https://doi.org/10.3217/jucs-025-02-0074>. (► str. 58)

Rodzaj publikacji: **artykuł w czasopiśmie** (IF: **1.056ui** ; liczba cytowań: **2**)

Punkty ministerialne (1.12.2021): **40**

Wkład autora rozprawy: **65%**

- [A3] **Bieniasz, J., & Szczypiorski, K. (2019). Steganography Techniques for Command and Control (C2) Channels.** In Botnets. Architectures, Countermeasures, and Challenges. (pp. 189-216). CRC Press. DOI: <https://doi.org/10.1201/9780429329913-5>. (► str. 82)

Rodzaj publikacji: **rozdział w monografii** (IF: –; liczba cytowań: **0**)

Punkty ministerialne (1.12.2021): **50**

Wkład autora rozprawy: **75%**

- [A4] **Bieniasz, J., Stępkowska, M., Janicki, A., & Szczypiorski, K. (2019). Mobile Agents for Detecting Network Attacks Using Timing Covert Channels.** Journal of Universal Computer Science, 25(9), 1109-1130. DOI: <https://doi.org/10.3217/jucs-025-09-1109> (► str. 109)

Rodzaj publikacji: **artykuł w czasopiśmie** (IF: **1.056**; liczba cytowań: **7**)

Punkty ministerialne (1.12.2021): **40**

Wkład autora rozprawy: **65%**

- [A5] **Bieniasz, J., & Szczypiorski, K. (2021). Dataset Generation for Development of Multi-Node Cyber Threat Detection Systems.** Electronics. 2021; 10(21):2711. DOI: <https://doi.org/10.3390/electronics10212711>. (► str. 131)

Rodzaj publikacji: **artykuł w czasopiśmie** (IF: **2.690**; liczba cytowań: **1**)

Punkty ministerialne (1.12.2021): **100**

Wkład autora rozprawy: **75%**

- [A6] Bieniasz, J., & Szczypiorski, K. (2018). **Towards Empowering Cyber Attack Resiliency Using Steganography**. 2018 4th International Conference on Frontiers of Signal Processing (ICFSP), 2018, pp. 24-28. DOI: <https://doi.org/10.1109/ICFSP.2018.8552068>. (► str. 153)

Rodzaj publikacji: **materiały pokonferencyjne** (IF: –; liczba cytowań: 1)

Punkty ministerialne (1.12.2021): **20**

Wkład autora rozprawy: **75%**

- [A7] Bieniasz, J., Bąk, P., & Szczypiorski, K. (2022). **StegFog: Distributed Steganography Applied To Cyber Resiliency In Multi Node Environments**. IEEE Access, 2022. DOI: <https://doi.org/10.1109/ACCESS.2022.3199749>. (► str. 158)

Rodzaj publikacji: **artykuł w czasopiśmie** (IF: **3.476**; liczba cytowań: 0)

Punkty ministerialne (1.12.2021): **100**

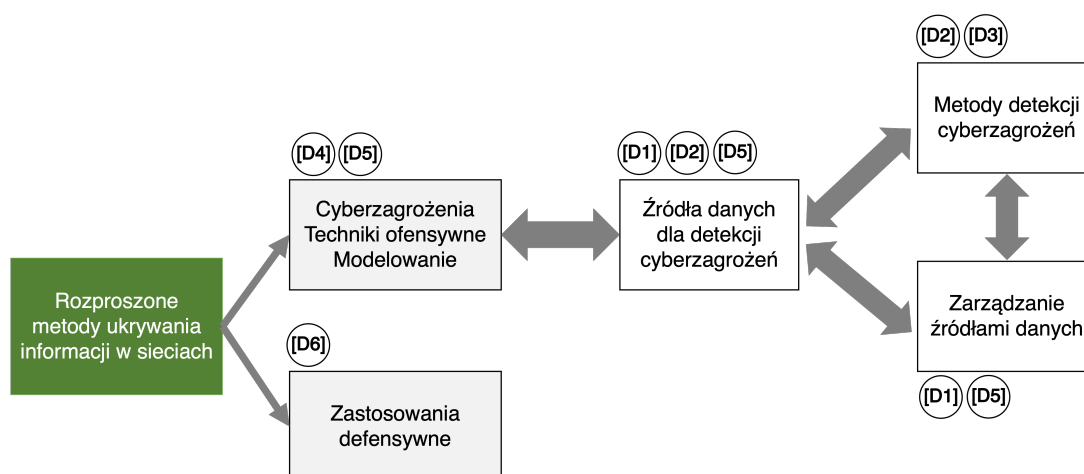
Wkład autora rozprawy: **60%**

Steganografia, w tym steganografia sieciowa i rozproszona, klasyfikowana w domenie cyberbezpieczeństwa w obszarze technik *cyber deception* [6] [7], w ciągu ostatnich 20 lat była intensywnie badana pod kątem możliwości stosowania w działaniach ofensywnych w cyberprzestrzeni, co podsumowuje m.in. publikacja [8]. Rosnącą skalę jej wykorzystania przez atakujących potwierdzają analizy raportów z incydentów bezpieczeństwa komputerowego, które podsumowuje artykuł [A3]. Ponadto w artykule tym przedstawiono sposób zastosowania metodyk modelowania cyberzagrożeń do metod steganografii. Samą naturę rozproszonych metod steganografii badano w ramach prac, których wyniki przedstawiono w artykułach [A1] i [A2]. W artykułach [A4] i [A5] opisano realizację *end-to-end* procesu badań danych na potrzeby detekcji metod steganografii rozproszonej – od stworzenia symulacji metod steganografii, przez implementację prototypu systemu detekcji, po referencyjne zastosowanie metod *data science*. W artykule [A6] zwrócono uwagę na drugą naturę steganografii, która może stanowić z powodzeniem mechanizm bezpieczeństwa systemów i użytkowników, np. w zakresie zapewniania prywatności czy bezpieczeństwa danych. Artykuł [A6] formułuje teoretyczne założenia nowego protokołu dla rozproszonego systemu steganograficznego poprzez zastosowanie koncepcji zaprezentowanych w artykułach [A1] i [A2]. Artykuł [A7] rozwija koncepcję z [A6] poprzez implementację prototypu nowego systemu steganograficznego StegFog, przeprowadzenie badań w zakresie parametrów

operacyjnych tego systemu oraz dokonanie oceny jego poziomu bezpieczeństwa. Dla osiągnięcia celu rozprawy zrealizowano następujące działania badawcze:

- [D1] Stworzenie środowiska symulacji rozproszonych metod steganografii na potrzeby zbierania zestawów danych do ich analizy.
- [D2] Opracowanie architektury rozwiązania i implementacja prototypu systemu wieloagentowego do monitorowania i wykrywania cyberzagrożeń wykorzystujących metody ukrywania informacji.
- [D3] Zbadanie efektywności metod z obszaru *data science* do analizy i wykrywania symulowanych metod steganografii oraz zaproponowanie nowego algorytmu lub architektury przetwarzania danych do wykrywania steganografii.
- [D4] Opracowanie koncepcji nowej metody steganograficznej o charakterze rozproszonym na potrzeby badania jej stosowalności, efektywności i cech charakterystycznych.
- [D5] Usystematyzowanie wiedzy o cyberzagrożeniach związanych z rozproszonymi metodami steganografii, w tym poprzez zastosowanie metod modelowania cyberzagrożeń do budowania cyberobrony opartej na wiedzy i danych.
- [D6] Zbadanie możliwości stosowania metod steganografii rozproszonej w kontekście obronnym - np. jako mechanizmu zapewniania bezpieczeństwa danych czy prywatności.

Rysunek 1 przedstawia, jak każde z działań realizowanych w ramach prac badawczych wnosi wkład do budowy cyberobrony w zakresie detekcji cyberzagrożeń, opierając się na wiedzy i danych. Tabela 1 podsumowuje wkład każdego z artykułów [A1]–[A7] do realizacji poszczególnych działań badawczych [D1]–[D6].



Rysunek 1: Działania badawcze [D1]–[D6] w odniesieniu do procesu budowy cyberobrony w zakresie detekcji cyberzagrożeń na podstawie wiedzy i danych

Tabela 1: Realizacja działań badawczych [D1]–[D6] prowadzących do osiągnięcia celów rozprawy w artykułach [A1]–[A7]

Artykuł	Działania	Rozdział rozprawy
[A1]	[D4]	Rozdział 2
[A2]	[D4] [D5]	
[A3]	[D5]	Rozdział 3, pkt 3.1.
[A4]	[D1] [D2] [D3] [D5]	Rozdział 3, pkt 3.2.
[A5]	[D1] [D2] [D3] [D5]	
[A6]	[D6]	Rozdział 4
[A7]	[D6]	
Podsumowanie rozprawy		Rozdział 5

2. Badania nad realizacją rozproszonych metod ukrywania informacji w sieciach

2.1. Podsumowanie artykułów [A1] i [A2]

W rozdziale 2 omówiono badania autora rozprawy w zakresie opracowania, implementacji i testów metody steganograficznej, wykorzystującej jako nośnik ukrytych informacji otwarte sieci społecznościowe (OSN). Oryginalna idea została przedstawiona przez K. Szczypiorskiego w [9], a następnie była rozwijana w ramach wspólnych badań z autorem rozprawy.

Jako rozwinięcie metody StegHash autor rozprawy zaproponował metodę SocialStegDisc w artykule [A1] (tekst artykułu jest załączony na str. 53). Pełne rozwinięcie koncepcji, badania i ogólna synteza rozważań nad rozproszonym ukrywaniem informacji w otwartych sieciach społecznościowych została zaprezentowana w artykule [A2] (tekst artykułu jest załączony na str. 58).

Wraz z pojawieniem się otwartych sieci społecznościowych badane są ich możliwości zastosowania w systemach steganograficznych. Celem tych rozważań jest możliwość wyprzedzającego opracowywania metod cyberobrony, gdyż w ostatnich latach zauważalny jest wzrost wykorzystania takich metod do komunikacji cyberprzestępców czy jako mechanizmów operacyjnych w złośliwym oprogramowaniu, w szczególności w tym używanym w zaawansowanych atakach wieloetapowych (komunikacja Command & Control, wycieki informacji, szpiegostwo). Ta kwestia jest szerzej poruszona przez autora w kolejnych badaniach przedstawionych w rozdziale 3 w pkt. 3.1.

W publikacji [10] autorzy definiują dwa podstawowe rodzaje komunikacji kanałami ukrytymi: o wysokiej oraz niskiej entropii. W komunikacji ukrytej o wysokiej entropii nośnikiem danych są obiekty multimedialne: obrazy, filmy, muzyka itp. W ten sposób definiowany jest klasyczny rodzaj steganografii – steganografia multimedialna. W steganografii multimedialnej pojedynczy obiekt przechowuje ukryte informacje w swojej strukturze, a metoda ukrywania jest charakterystyczna dla danego rodzaju nośnika. Ukryte kanały oparte na steganografii multimedialnej cechują się wysoką przepływnością, ale za to są łatwe do wykrycia i podatne na zaburzenia, wynikające z przetwarzania obrazów w systemach komputerowych. Model komunikacji ukrytej o niskiej entropii polega na zastosowaniu tekstu jako nośnika ukrytych informacji. Założeniem w tym modelu jest istnienie wcześniej znanego i uwspólnionego sekretu, który jest niezbędny do odkodowywania informacji z takiego kanału. W porównaniu ze steganografią multimedialną steganografia oparta na tekście charakteryzuje się niską przepływnością, przy jednoczesnym dobrym poziomie bezpieczeństwa. Jedną

z propozycji jest wykorzystanie takiego kanału do sterowania lokalizacją steganogramów, które mogą być rozmieszczone w środowisku rozproszonym, np. w serwisach internetowych, takich jak sieci społecznościowe. Propozycje metod steganograficznych o niskiej entropii przedstawiono w [11], a wśród nich:

- wykorzystanie pierwotnych nazw plików umieszczanych w otwartych sieciach społecznościowych - przy założeniu, że sieć przechowuje w metadanych tę nazwę. Proponowane jest wykorzystanie standardowych schematów nazewnictwa plików zdjęć nadawanych automatycznie przez aparaty fotograficzne. Kanał ukryty może zostać oparty na numerach sekwencyjnych, będących częścią składową nazw tych plików;
- wykorzystanie mechanizmu tagów umieszczanych przy publikowanych zdjęciach przez użytkowników. Tagi to rodzaj słów kluczowych, które opisują zawartość zdjęcia w taki sposób, aby inni użytkownicy mieli łatwość w ich odnalezieniu. Utworzenie ukrytego kanału jest możliwe przez kombinację wielu obrazów z wieloma różnymi tagami, która stanowi odpowiednio przygotowany sposób kodowania.

Przykłady innych podejść do ukrytych kanałów wykorzystujących różne cechy języków naturalnych przedstawiono w [12]. W pracy [13] zaproponowano model ukrytego kanału oparty na steganogramach przenoszonych jako bitmapy kodowane za pomocą języka naturalnego w wiadomościach publikowanych w serwisie Twitter. Powstały kanał jest bardzo bezpieczny, ale istnieje duże prawdopodobieństwo jego zaburzenia.

Wszystkie przytoczone metody wykorzystują klasyczną steganografię multimedialną, łatwą w detekcji lub bardziej wyspecjalizowaną, o ograniczonej przepływności steganograficznej. Proponowane metody zakładają wykorzystanie wielu kont użytkowników w otwartych sieciach społecznościowych oraz równoczesnego dostępu do wielu materiałów, co może być rozpoznane jako działanie niepożądane po stronie dostawców tych sieci i w konsekwencji efekty działań wskazanych metod mogą zostać zablokowane.

Jednym z zastosowań operacyjnych metod steganografii jest jej wykorzystanie do opracowywania systemów plików oferujących możliwość ukrywania informacji – np. bezpośrednio samych danych czy też wykonywanych operacji na danych. Przełomową pracą w rozwoju steganograficznych systemów plików jest artykuł [14]. Autorzy zaproponowali w nim metodę na wzór działania niewidzialnego atramentu. W metodzie tej użytkownik posiadający hasło może odnaleźć i odcodować ukryte informacje. System StegFS zaprezentowany w artykule [15] stanowi praktyczną implementację koncepcji [14] jako rozszerzenie standardowego systemu plików Linux ext2fs.

Kolejnym przykładem rozwoju koncepcji steganograficznego systemu plików jest wykorzystanie, np. w [16], wielu rozproszonych plików klastra do ukrywania informacji w kodach pozycji tych plików względem siebie w klastrach.

Wśród ostatnich propozycji warto wyróżnić pracę [17], w której przedstawiono sposób ukrywania informacji w znacznikach czasowych plików zapisywanych w klasycznych systemach plików. Odkryto, że informacje mogą zostać skutecznie ukryte w 24 bitach (3B) kodujących część nanosekundową znacznika w systemie plików NTFS. Dla każdego pliku możliwe jest wykorzystanie dwóch rodzajów znaczników: utworzenia oraz ostatniego dostępu. W ten sposób jeden plik tworzy 6B przestrzeni na ukryte dane.

Badania przedstawione w [A1] i [A2] udowadniają, że pierwotna koncepcja metody StegHash [9], jako nowego podejścia do łączenia steganografii tekstu z innymi nośnikami, takimi jak obrazy, filmy i piosenki, jest zasadna i ma praktyczne zastosowania, m.in. w realizacji steganograficznego systemu plików SocialStegDisc. W ramach takiego systemu każdy obiekt multimedialny jest indeksowany za pomocą unikatowej permutacji zestawu n hashtagów. Pomiedzy obiektami tworzona jest logiczna struktura łańcucha poprzez wybraną funkcję hashującą. Na podstawie indeksu wejściowego (permutacji zbioru hashtagów) funkcja ta generuje indeks kolejnego obiektu (kolejna pseudolosowa permutacja zbioru hashtagów). Dla n hashtagów, m -bajtowych wiadomości i 100% dokładności działania systemu jego pojemność wynosi $n! * m$ bajtów. Przykładowo: dla $n = 12$ i $m = 10$ bajtów byłoby to 4,46 TB.

Steganograficzny system plików SocialStegDisc umożliwia wykonywanie klasycznych operacji na kolejnych blokach ukrytej przestrzeni danych – odczytywanie, zapisywanie lub modyfikowanie, w tym usuwanie. SocialStegDisc stanowi zatem ulepszenie oryginalnego sposobu działania metody StegHash, oferując kompromis między wymaganiami pamięci a czasem obliczeń. Podczas testów sprawdzono, że SocialStegDisc działa analogicznie do StegHash, ale nakład operacji jest równomierniej rozłożony, oferując przy tym mechanizmy klasycznego systemu plików w zakresie operacji CRUD (Create, Read, Update, Delete). Niewykrywalność systemu SocialStegDisc może być oceniona według następujących kryteriów:

- poziom niewykrywalności oferowany przez wykorzystanie otwartych sieci społecznościowych,
- poziom niewykrywalności wynikający z metody StegHash,
- poziom bezpieczeństwa serwisów społecznościowych w zakresie wykrywania anomalii w zachowaniu użytkowników sieci. System SocialStegDisc może wielokrotnie komunikować się z tą samą usługą w krótkich odstępach czasu.

Z perspektywy takiej usługi częste i automatyczne wysyłanie zapytań będzie traktowane jako zagrożenie, np. atak typu *Denial-of-Service*,

- poziom niewykrywalności steganografii multimedialnej przez otwarte sieci społecznościowe. Problem ten był badany m.in. dla Facebooka. W [18] autorzy przetestowali algorytmy odkażające steganografię po stronie serwerów Facebooka. Z badań wynika, że algorytmy Facebooka są skuteczne w wykrywaniu znanej do tej pory steganografii, w szczególności ukrytych informacji o większym rozmiarze.

W [19] zaproponowano sposób oceny technik steganografii sieciowej w trzech kategoriach niewykrywalności: **dobry**, **zły** i **brzydki**. Ten sposób klasyfikacji może być analogicznie zastosowany do innych metod steganograficznych. Według tej klasyfikacji SocialStegDisc można zaklasyfikować jako **dobrą** metodę steganograficzną, podobnie jak StegHash. To oznacza, że obserwator nie jest w stanie odkryć zachodzącej komunikacji gdziekolwiek w sieci, nawet u odbiorcy danych ukrytych.

2.2. Osiągnięcia badawcze autora

Główne osiągnięcia badawcze zaprezentowane w artykułach [A1] (str. 53) i [A2] (str. 58) to:

- Opracowanie koncepcji steganograficznego systemu plików o nazwie SocialStegDisc, w którym mechanizm indeksacji danych oparty na StegHash[9] został powiązany z klasyczną konstrukcją systemu plików - listą połączeniową i wskaźnikami na kolejne bloki pamięci. Koncepcja obejmuje dwie możliwości ich utworzenia.
- Zdefiniowanie steganograficznych realizacji typowych operacji klasycznego systemu plików - tworzenie / odczyt / usuwanie.
- Implementacja *proof-of-concept* systemu SocialStegDisc.
- Przeprowadzenie analizy i testów systemu SocialStegDisc w zakresie możliwości skalowania rozmiarów dysku oraz plików, sposobu tworzenia sekretów, wydajności działania w realnym środowisku otwartej sieci społecznościowej.
- Dokonanie ewaluacji aspektów niewykrywalności i niezawodności systemu.
- Wskazanie na problemy szczególne, np. nierównomierność przestrzeni adresowej.
- Przeprowadzenie oceny wpływu wykorzystania takiej koncepcji w cyberprzestrzeni, w szczególności pod kątem jej wykrycia czy też przeanalizowania w ramach procesów kryminalistyki śledczej.

- Powiązanie koncepcji rozproszonych systemów steganograficznych z możliwością ich wykorzystania jako mechanizmu cyberodporności systemów – w odniesieniu do koncepcji zaproponowanej przez Kotta w [20]. Temat ten został rozwinięty przez autora w rozdziale 4.

3. Badania nad wykrywaniem rozproszonych metod ukrywania informacji w sieciach

3.1. Cyberobrona oparta na wiedzy i danych – od modelowania do wykrywania cyberzagrożeń

3.1.1. Podsumowanie artykułu [A3]

W rozdziale 3 w pkt. 3.1. omówiono badania teoretyczne autora rozprawy w zakresie wykorzystania modelowania cyberzagrożeń w procesie projektowania rozwiązań ich detekcji. W tym kontekście autor dokonał przeglądu stanu wiedzy dotyczącej zastosowania steganografii do realizacji komunikacji Command & Control (C2). Ukryta komunikacja jest coraz częściej wykorzystywana przez złośliwe oprogramowanie i botnety oraz w ramach celowanych ataków typu APT (ang. *Advanced Persistent Threats*).

Rezultaty badań teoretycznych autora ujęto w artykule [A3] (tekst artykułu jest załączony na str. 82). Na podstawie przeglądu raportów dotyczących rzeczywistych cyberataków i wykrytych botnetów zaproponowano taksonomię podstawowych modeli komunikacji steganograficznych kanałów C2. Przegląd ten objął zarówno analizę złośliwego oprogramowania w tradycyjnych środowiskach komputerowych, jak i platformach mobilnych, dla których istotność zagadnień cyberbezpieczeństwa w ostatnich latach gwałtownie rośnie. Zgodnie z proponowaną strategią cyberobrony opartej na proaktywnym modelowaniu cyberzagrożeń rezultaty artykułu [A3] mogą służyć jako *know-how* niezbędne do analizy, wykrywania i przełamania kanałów C2, gdy są one zabezpieczone steganografią.

W ostatnich latach można znaleźć coraz więcej przykładów zastosowanie metod steganografii do ukrytego przechowywania danych i ukrytej komunikacji. Raporty różnych organizacji zajmujących się obsługą incydentów bezpieczeństwa komputerowego oraz tworzeniem rozwiązań dla cyberbezpieczeństwa, np. Kaspersky [21], McAfee [22] czy Fortinet [23], ostrzegały, że techniki ukrywania informacji są coraz częściej stosowane przez cyberatakujących w złośliwym oprogramowaniu.

Zastosowanie steganografii do wprowadzania i aktywowania złośliwego oprogramowania umożliwia omijanie typowych mechanizmów bezpieczeństwa, takich jak: aplikacje Endpoint Detection & Response (EDR), antywirusy/antymalware, systemy wykrywania włamań/zapobiegania włamaniom, zapory sieciowe itp. Wynika to z tego, że ruch sieciowy czy pliki multimedialne z ukrytymi danymi są rozpoznawane przez te mechanizmy jako bezpieczne, normalne i niepodważane. W ogólności steganografia ma za zadanie pomóc złośliwemu oprogramowaniu uniknąć wykrycia lub znacząco utrudnić jego analizę wsteczną w procesach kryminalistyki cyfrowej.

Współcześnie cyberataki są analizowane w sposób procesowy, gdzie badany jest pełny przebieg wyrządzania szkód przez cyberatakujących. Wykonanie złośliwego oprogramowania na systemach ofiary czy komunikacja typu C2 są w tym ujęciu wyróżniane jako jeden z etapów, który również może się wielokrotnie powtórzyć, np. w ramach realizacji przejmowania kolejnych systemów danego środowiska teleinformatycznego. Przyjętym pojęciem określającym współczesne długotrwałe i wieloetapowe cyberataki jest APT [24]. APT opisuje cyberataki, które łączą w sobie następujące cechy:

- wielowarstwowe i wieloetapowe kampanie włamań do systemów komputerowych i innych, np. systemów przemysłowych,
- prowadzone są w długim czasie przez dobrze przygotowane i wyszkolone grupy atakujących, którzy mają dostęp do potrzebnych zasobów,
- cele atakujących koncentrują się na zagadnieniach strategicznych, przede wszystkim w obszarze pozyskiwania wysoce istotnych informacji, takich jak własność intelektualna czy tych zastrzeżonych dotyczących bezpieczeństwa narodowego. Celem są infrastruktury teleinformatyczne podmiotów o znaczeniu strategicznym – zarówno w obszarze państwa (np. operatorzy infrastruktury krytycznej i usług kluczowych, administracja rządowa czy agendy badawcze), jak i podmioty prywatne (np. banki i operatorzy usług finansowych, firmy technologiczne czy farmaceutyczne).

Techniki ukrywania informacji należy uznać za jedną z możliwości, które cyberatakujący mogą wykorzystać do osiągnięcia realizacji celów założonych kampanii ofensywnych w cyberprzestrzeni. Ewolucja złośliwego oprogramowania i ofensywnych operacji w cyberprzestrzeni przyczyniła się do rozwoju nowych podejść w zakresie cyberobrony i cyberodporności infrastruktur teleinformatycznych. Jednym z kierunków rozwoju jest szerokie pozyskiwanie każdej możliwej wiedzy o cyberzagrożeniach na różne sposoby, np. na podstawie historii analiz powłamaniowych, poprzez symulacje technik cyberataków, emulację znanych grup cyberatakujących, wymianę wiedzy między różnymi podmiotami czy badania hipotetycznych scenariuszy.

Istotnym podsumowaniem tego podejścia jest artykuł [1], który zawiera syntezę dwóch fundamentalnych zagadnień:

- **Jak modelować cyberatak?** Odpowiedzią jest 7-etapowy, łańcuchowy model cyberataku – Cyber Kill Chain, który ma służyć do opisywania kolejnych etapów włamań. Koncepcja ta stanowi fundament wszelkich działań w obszarze modelowania cyberzagrożeń i stanowi standard wymiany wiedzy o cyberzagrożeniach w domenie cyberbezpieczeństwa.

- **Jak wykorzystać wiedzę o cyberatakach do budowy cyberobrony i cyberodporności?** Odpowiedzią jest cyberobrona oparta na wiedzy i danych, osadzona w iteracyjnym procesie gromadzenia i wymiany wiedzy o przeciwnikach w cyberprzestrzeni i ich technikach. Modele cyberataków Cyber Kill Chain na każdym etapie mogą być powiązane z właściwymi dla nich wskaźnikami działania (ang. *Indicator of Compromise*) oraz rozwiązaniami cyberbezpieczeństwa w zakresie wykrywania czy blokowania. Całościowo takie modele cyberataków mogą służyć do identyfikowania wzorców łączących poszczególne włamania i incydenty w szerzej prowadzone kampanie. Ostatecznie powstaje przy tym pętla informacji zwrotnej, która umożliwia zmniejszenie prawdopodobieństwa sukcesu atakującego przy każdej kolejnej próbie skompromitowania środowiska.

3.1.2. Osiągnięcia badawcze autora

Główne osiągnięcia badań teoretycznych autora zaprezentowane w artykule [A3] (str. 82) to:

- Przeprowadzenie syntezy stanu wiedzy o modelowym stosowaniu metod steganografii do komunikacji Command & Control (C2). Punktem odniesienia jest taksonomia stosowanych metod steganografii do ukrywania informacji w sieciach, w szczególności metody steganografii multimedialnej, sieciowej i hybrydowej, realizowanej jako rozproszona (np. routing steganograficzny).
- Opracowanie teoretycznych modeli operacyjnych kanałów C2 wykorzystujących metody steganografii.
- Odniesienie wykorzystania steganografii do realizacji kanałów C2 w ramach metodyk modelowania cyberzagrożeń:
 - opis wykorzystania steganografii w ramach etapu C2 w modelu Cyber Kill Chain [1],
 - wykonanie selekcji technik z matrycy MITRE ATT&CK[25] i zmapowanie ich jako metody steganograficzne (według stanu matrycy z 2019 roku nie istniała wyróżniona technika reprezentująca steganografię).
- Przegląd możliwych metod wykrywania i przeciwdziałania kanałom C2 zabezpieczonym steganografią.
- Dokonanie podsumowania cyberataków z lat 2010–2018 odkrytych w środowiskach komputerowych i mobilnych, w których zastosowano kanały C2 wykorzystujące

metody steganografii. Każdy przytoczony przykład został zaklasyfikowany według autorskiej taksonomii modeli ukrytych kanałów C2 oraz katalogu technik MITRE ATT&CK [25].

Przegląd przedstawiony w artykule [A3] obejmował przykłady cyberataków wykorzystujących steganografię do zabezpieczenia kanałów C2 z lat 2010–2018. Po 2018 roku pojawiły się kolejne ataki, które dodatkowo podkreślają istotność poruszanego zagadnienia wykorzystania różnych metod steganografii rozproszonej do zabezpieczenia kanałów C2 – patrz np. [26], [27] czy [28]. Jak zostało wskazane w pkt. 3.1.1. Wprowadzenia, w momencie pisania artykułu [A3] matryce MITRE ATT&CK [25] nie zawierały technik steganografii. Dlatego w artykule [A3] autor dokonał powiązania analizowanych przykładów kanałów C2 wykorzystujących metody steganografii do technik zawartych w matrycach MITRE ATT&CK. W 2020 roku dokonano oficjalnej aktualizacji matrycy, dodając steganografię jako podtechnikę T1001.002 w ramach techniki Obfuskacja danych [T1001] [29]. Pokrywa ona jedynie metody steganografii obrazkowej, zatem mapowanie metod steganografii sieciowej i hybrydowej na matryce MITRE ATT&CK dokonane w artykule [A3] stanowi nadal istotny wkład w modelowanie cyberzagrożeń, wykorzystujących te rodzaje metod ukrywania informacji.

3.2. Monitorowanie i detekcja kanałów steganograficznych w podejściu rozproszonym

3.2.1. Podsumowanie artykułów [A4] i [A5]

W rozdziale 3 w pkt. 3.2. omówiono badania autora w zakresie metod monitorowania i detekcji kanałów steganograficznych w rozproszonych środowiskach teleinformatycznych z wykorzystaniem systemów wieloagentowych. Rezultaty tych badań ujęto w artykułach [A4] (tekst artykułu jest załączony na str. 109) oraz [A5] (tekst artykułu jest załączony na str. 131).

Artykuł [A4] definiuje koncepcję teorii obserwacji zmiany. Na jej podstawie proponuje algorytm wykrywania anomalii jako niezbędnego kroku w procesie śledzenia ścieżki realizacji ataku, prowadząc w kierunku węzłów źródłowych. Reprezentacją ataków poddawanych monitorowaniu i wykrywaniu są metody steganograficzne oparte na kodowaniu informacji w zależnościach czasowych. W ramach stosowania metody zaprezentowanej w artykule można np. wykryć i śledzić kradzież danych w ramach komunikacji C2. Wykrycie węzłów źródłowych takiej operacji umożliwia efektywne reagowanie na nie, np. przez blokadę ruchu sieciowego.

Realizacją operacyjną tej koncepcji było zastosowanie modelu systemów wieloagentowych (ang. *multi-agent systems*). Systemy cyberobrony oparte na systemach wieloagentowych są analizowane od dawna. Jednym z kamieni milowych rozważań nad tą koncepcją jest metoda przedstawiona w [30]. Kolejne podsumowania prac badawczych w tym obszarze przedstawiono w publikacjach [31], [32] oraz [33]. Ze względu na rozwój środowisk teleinformatycznych i mocy obliczeniowych w ostatniej dekadzie widoczne jest coraz szybsze przyjmowanie się tych koncepcji w rozwiązaniach praktycznych, a zainteresowanie tymi rozwiązaniami w środowiskach akademickich, przemysłu, organów ścigania, a nawet wojska szybko rośnie. Systemy wykrywania cyberzagrożeń oparte na podejściu wieloagentowym są uznawane za skuteczne, a przede wszystkim jako takie, które mogą być jedynymi adekwatnymi i skutecznymi rozwiązaniami w obliczu ewolucji cyberataków, złośliwego oprogramowania i ofensywnych operacji w cyberprzestrzeni.

W [34] autor przekonuje, że inteligentne agenty autonomiczne będą standardem na polu bitwy przyszłości, przenikającej się z cyberprzestrzenią. Ponadto przedstawia kilka nowatorskich pomysłów i podsumowuje zagadnienie w zunifikowanej, referencyjnej architekturze takiego wieloagentowego systemu do cyberobrony.

W artykule [A4] opracowano koncepcję systemu wykorzystującego agenty monitorujące w węzłach sieciowych (*routerach programowalnych*), które co określony czas przesyłają stan obserwacji do centralnej jednostki obliczeniowej. Modelem reprezentacji danych przetwarzanych przez agenty i jednostkę centralną są histogramy. W metodzie opisanej w artykule [A4] do wykrywania anomalii wykorzystywana jest reprezentacja histogramowa czasu nadejścia pakietów (ang. *Inter Arrival Time* – IAT) [35]. To samo podejście może być z powodzeniem stosowane do obserwacji różnych parametrów, dla których – poza zbieraniem jej atomowej wartości – konieczne jest przechowywanie informacji o stanie histogramów do detekcji cyberzagrożeń, co pokazano m.in. w [36].

Proponowaną w artykule [A4] architekturę systemu rozszerza artykuł [A5], gdzie zaprezentowano referencyjną architekturę fizyczną i logiczną, obejmującą jako fundamentalne elementy – jednostkę centralną, agenta, router programowalny oraz platformę wieloagentową. Zaimplementowano prototyp platformy wieloagentowej do zarządzania agentami programowymi instalowanymi na routerach z otwartym systemem operacyjnym. Celem działania wskazanych agentów jest przechwytywanie ruchu sieciowego z interfejsów routera, kolekcjonowanie danych do celów wykrywania anomalii oraz śledzenie źródeł ataków sieciowych. Równolegle do prototypu systemu rozwijano środowisko testowe, które w wersji zaprezentowanej w artykule [A5] umożliwia:

- szybkie prototypowanie nowych przypadków użycia,
- automatyczne generowanie nowych topologii sieci, tj. decyzji dotyczących:
 - liczby routerów i hostów,
 - zakresu adresów IP i routing,
 - zapewnienia dostępu do Internetu,
 - dołączenia firewalla,
 - umieszczenia jednostki centralnej całego systemu.

Na podstawie tych parametrów opracowane narzędzie generuje losową topologię sieci teleinformatycznej i następnie przesyła ją do silnika emulacji sieci za pośrednictwem interfejsu API lub pliku konfiguracyjnego. Przygotowanie takiej automatyzacji ma na celu zminimalizowanie wpływu topologii sieci na skuteczność nowo opracowanych algorytmów wykrywania cyberzagrożeń poprzez ich trenowanie na danych pozyskanych z różnych topologii. Należy zauważyć, że kluczowym aspektem przygotowanego rozwiązania było zautomatyzowanie losowości mechanizmów:

- generowania łączy pomiędzy zestawem węzłów,
- wyboru nadawców i odbiorców dla każdego profilu strumienia danych sieciowych (normalny lub złośliwy).

Takie podejście pozwala na generowanie zestawów danych z wielu scenariuszy sieciowych i konfiguracji. Może również zapewnić możliwość zminimalizowania wpływu czynników związanych z konfiguracjami w szerokim spektrum badań dotyczących wykrywania cyberzagrożeń na podstawie danych. Zbiory danych zostały zebrane w określonych scenariuszach sieciowych z niewielkim stopniem zmienności w konfiguracji danych sieci nadawca-odbiorca (normalnej lub złośliwej). Potrzeba prac nad takim rozwiązaniem wynika z założenia, że do stosowania badań opartych na metodach analizy danych (ang. *data science*) niezbędne jest posiadanie dobrze przygotowanych zbiorów danych.

Istnieją trzy główne podejścia do pozyskiwania danych do prowadzenia badań nad metodami analizy danych do wykrywania cyberzagrożeń:

- zbieranie danych z rzeczywistych sieci produkcyjnych oraz w ramach analizy powłamaniowej (historyczne próbki ataków),
- budowanie modeli sieci produkcyjnych i symulowanie komunikacji sieciowej (złośliwej i normalnej),

- zastosowanie algorytmów deterministycznych, statystycznych, uczenia maszynowego i innych do generowania danych.

Pierwsze podejście jest wysoce pożądane, ponieważ praca na rzeczywistych danych powinna gwarantować małą liczbę błędów wyuczonych algorytmów detekcji w sytuacji cyberataków. Główne wyzwanie związane z tym podejściem polega na tym, że niewiele organizacji mogłoby wykorzystać takie dane do badań. Cyberataki są bardzo rzadkie w odniesieniu do całkowitego czasu obserwacji niezbędnego do pozyskania reprezentatywnego zbioru danych. Oznacza to, że zebranie wystarczającej liczby przykładów, aby wytrenować algorytmy detekcji, zajęłoby bardzo dużo czasu.

Kolejnym wyzwaniem związanym z takimi danymi jest kwestia prywatności i zgodności z prawem pod względem przechowywania takich danych w dłuższym okresie. W związku z tym najczęściej wykorzystuje się podejście symulacyjne do badań nad systemami wykrywania cyberzagrożeń.

Jedną z referencyjnych procedur do generowania takich danych przedstawia [37]. Autorzy tego artykułu zaproponowali parametryczną konfigurację symulowanych wzorców komunikacji sieciowej, zwanych profilami, dzięki czemu jakość otrzymanych zbiorów danych poprawiła się. Inne systematyczne podejście zostało przedstawione w [38]. W artykule zaproponowano nową koncepcję symulowanych zestawów danych dla systemów wykrywania cyberzagrożeń, obejmującą:

- zbieranie danych z sensorów rozproszonych w węzłach sieci,
- umożliwienie ciągłej komunikacji i koordynacji pomiędzy sensorami,
- wykorzystanie jednostki centralnej w celu usprawnienia podejmowania decyzji o detekcjach,
- automatyzację scenariuszy sieciowych, w których zebrano dane,
- wykorzystanie metod analizy danych w zakresie algorytmów *nadpróbkowania* najmniej reprezentatywnych próbek, tj. tych reprezentujących ruch złośliwy.

Trzecie podejście stosuje modelowanie i analizę danych oraz metody uczenia maszynowego do generowania danych syntetycznych. Ogólnie takie algorytmy mogłyby wesprzeć drugie podejście poprzez zwiększanie podobieństwa generowanych danych do rzeczywistych lub wyeliminowanie niedociągnięć w symulacjach (komplementarność między podejściami).

Przykład zastosowania uczenia maszynowego w celu poprawy metryk wykrywalności i poprawienia reprezentacji danych dla niewielkiej liczby złośliwych próbek przedstawiono w [39]. W celu wygenerowania syntetycznych próbek zastosowano metody

przeciwstawnego uczenia maszynowego (ang. *adversial machine learning*), tj. generatywne sieci przeciwstawne (ang. *Generative Adversial Networks* – GAN). Na tak uzyskanych danych przeprowadzono trenowanie algorytmów detekcji. Ta sama metoda rozwiązuje również problemy związane z niezrównoważeniem próbek w reprezentacji danych lub brakiem danych danego typu na wejściu. Takie podejście znacznie poprawia wydajność algorytmów detekcji. Głównym wyzwaniem w stosowaniu uczenia maszynowego do wykrywania cyberzagrożeń jest możliwość wyjaśnienia i przejrzystości takich algorytmów.

Artykuł [A5] podsumowuje publicznie dostępne zbiory danych, które mogą być wykorzystane do badań nad algorytmami detekcji cyberzagrożeń: [37], [38], [39], [40], [41], [42], [43], [44], [45], [46], [47].

Metodologia badań zaprezentowana w artykułach [A4] i [A5] opiera się na drugim podejściu do pozyskiwania danych na potrzeby badań metod detekcji, ponieważ wśród przedstawionych publicznych zestawów danych żaden nie oferuje przykładów metod steganografii sieciowej. W artykule [A4] przedstawiono wyniki symulacji metody steganografii kodującej informacje w zależnościach między czasami nadejścia pakietów.

Artykuł [A5] rozszerzył bazę symulacji o metodę opartą na celowym opóźnieniu i traceniu pakietów. Ponadto obok symulacji metod steganografii możliwe było uruchamianie scenariuszy ruchu sieciowego typowego dla sieci LAN generowanego przy przeglądaniu stron WWW (protokoły HTTP/HTTPS), komunikacji VoIP (SIP), oglądaniu materiałów wideo (RTP), przesyłaniu plików (FTP/SFTP) czy zdalnym dostępie do serwerów (SSH). Dzięki pozyskaniu właściwych danych możliwe było przeprowadzenie badań nad algorytmami detekcji cyberzagrożeń z wykorzystaniem metod uczenia maszynowego do wykrywania cyberzagrożeń [48]. Typowy proces analizy polega na uczeniu klasyfikatora opartego na uczeniu maszynowym przy użyciu danych, zwykle oznaczonych, uzyskanych ze środowisk produkcyjnych lub emulowanych. Następnie klasyfikator jest testowany przy użyciu danych ewaluacyjnych. Naukowcy zaproponowali wiele różnych algorytmów uczenia maszynowego, począwszy od podstawowych, takich jak k-Nearest Neighbors (k-NN), a skończywszy na algorytmach genetycznych lub różnych typach sieci neuronowych, takich jak perceptrony wielowarstwowe (MLP), mapy Kohonena [49] lub techniki głębokiego uczenia [50].

Badania zaprezentowane w artykule [A4] skupiają się na wykorzystaniu metod uczenia maszynowego i mierzonych poziomów klasyfikacji ruchu sieciowego jako złośliwy (klasyfikacje prawdziwie dodatnie (ang. *true positive* – TP) i fałszywie dodatnie (ang. *false positive* – FP) oraz do obliczania metryki anomalii d w zadanym oknie czasowym. Dane pozyskiwane z rozproszonych węzłów są na bieżąco agregowane w rozpatrywanym

oknie czasowym i dla każdej z porcji danych ustalana jest wartość metryki d . Przy przekroczeniu określonych progów wartości metryki d oraz w odniesieniu do wartości tej metryki z innych węzłów, algorytm decyzyjny wirtualnie podąża od węzła do węzła w kierunku rosnącej wartości d , osiągając w ten sposób potencjalne źródło ataku.

Badania w artykule [A4] w kilku eksperymentach potwierdziły skuteczność proponowanej metody, oferując efektywny sposób na odnajdywanie źródeł cyberataków wykorzystujących steganografię w danej sieci. Najlepszymi algorytmami uczenia w zastosowaniu do wyznaczania metryki d okazały się algorytm lasu losowego (ang. *Random Forest*) oraz klasyczna sztuczna sieć neuronowa (ang. *Multilayer Perceptron*).

Artykuł [A5] uzupełnia profil badań z artykułu [A4] poprzez:

- przeprowadzenie badań wykrywania cyberzagrożeń na podstawie popularnej reprezentacji danych - modelu strumieniowym, który zawierał 79 różnych metryk statystycznych generowanych przez narzędzie CICFlowMeter [51];
- rozbudowanie procesu analizy danych o:
 - wyrównanie reprezentacji danych próbek złośliwych i normalnych;
 - selekcję istotnych parametrów z wykorzystaniem korelacji;
- przeprowadzenie badań nad metodami uczenia maszynowego w zakresie poprawy jakości klasyfikacji wynikającej z dodania wyżej wymienionych etapów;
- przeprowadzenie badań skuteczności wykorzystania metod uczenia maszynowego w problemie klasyfikacji wieloklasowej, tj. dyskryminacja ruchu normalnego lub jednej z dwóch metod steganografii;
- zbadanie efektywności klasyfikacji cyberataków z wykorzystaniem złożonej architektury sieci neuronowej MLP.

Oryginalnym aspektem badań zaprezentowanych w artykułach [A4] i [A5] było wykazanie możliwości stosowania technik ukrywania informacji należących do szerokiej gałęzi cyberbezpieczeństwa, określanej jako *cyber deception*. Najważniejszym warunkiem do prowadzenia badań nad wykrywaniem cyberzagrożeń jest dostępność odpowiednich danych. Wszystkie znane zestawy danych, które są dostępne, skupiają się na powszechnie znanych typach cyberataków i nie ma wśród nich przykładów metod steganograficznych. Istotnym krokiem do osiągnięcia rezultatów badawczych było opracowanie środowiska sieciowego zintegrowanego z narzędziami do generowania scenariuszy ataków i następnie zbierania zestawów danych. Osiągnięto to dzięki zaproponowaniu i zaimplementowaniu prototypu systemu wieloagentowego jako wielowęzłowej platformy detekcji anomalii sieciowych. Na podstawie zebranych danych

możliwe było przeprowadzenie badań nad algorytmami detekcji wykorzystujących metody reprezentacji danych i klasyfikacji szkodliwych przepływów sieciowych.

Ostateczne wyniki obu artykułów potwierdzają przydatność zbadanych metod detekcji technik ukrywania informacji. Cyberatakujący coraz częściej wykorzystują różne metody z obszaru *cyber deception* w swoich operacjach i w budowaniu wykorzystywanych narzędzi, czyli złośliwego oprogramowania. Wspomniane wcześniej zaawansowane cyberataki APT mogą łączyć wiele technik z różnych warstw systemów teleinformatycznych w każdym z kroków cyberataku modelowanego jako Cyber Kill Chain [1]. W szczególności zastosowanie metod w warstwie aplikacji jest uważane za istotne zagrożenie, co autor rozprawy badał bezpośrednio się w ramach artykułów [A1] i [A2] (rozdział 2) oraz rozważał teoretycznie w artykule [A3] (rozdział 3, pkt 3.1.). Zatem w nadchodzących latach należy zwiększyć intensywność badań zmierzających do poprawy zdolności w zakresie cyberobrony przed technikami *cyber deception*.

3.2.2. Osiągnięcia badawcze autora

Główne osiągnięcia autora rozprawy w zakresie badań przedstawionych w artykule [A4] (str. 109) to:

- synteza stanu wiedzy,
- opracowanie scenariuszy testowych w zakresie realizacji symulacji metod steganografii we wskazanym środowisku wraz ze sformułowaniem hipotez badawczych dla każdego z nich,
- koordynacja realizacji scenariuszy testowych wraz z bieżącą oceną uzyskiwanych rezultatów,
- przygotowanie pełnej ścieżki analizy danych,
- zaimplementowanie i przeprowadzenie badań zastosowań metod uczenia maszynowego na danych zebranych podczas realizacji scenariuszy symulacji ataków,
- dokonanie ewaluacji uzyskanych rezultatów oraz sformułowanie wniosków końcowych, w szczególności uprawdopodobnienie zaproponowanej koncepcji algorytmicznej wykrywania anomalii do poszukiwania węzłów źródłowych cyberataków w środowisku rozproszonym.

Główne osiągnięcia autora rozprawy w zakresie badań przedstawionych w artykule [A5] (str. 131) to:

- synteza stanu wiedzy,
- opracowanie scenariuszy testowych w zakresie realizacji symulacji metod steganografii we wskazanym środowisku wraz ze sformułowaniem hipotez badawczych dla każdego z nich,
- koordynacja realizacji scenariuszy testowych wraz z bieżącą oceną uzyskiwanych rezultatów,
- implementacja części prototypu platformy detekcji i monitorowania w rozproszonej sieci teleinformatycznej wielu węzłów,
- przygotowanie pełnej ścieżki analizy danych jako rozszerzenie procesu z artykułu [A4] poprzez zastosowanie metod rozszerzenia zbioru próbek i selekcji metryk istotnych,
- zaimplementowanie i przeprowadzenie badań zastosowań metod uczenia maszynowego na danych zebranych podczas realizacji scenariuszy symulacji ataków,
- przeprowadzenie badań nad wyselekcjonowanymi w [A4] metodami uczenia maszynowego - lasem losowym i siecią neuronową MLP w rozszerzonych konfiguracjach architektury,
- dokonanie ewaluacji uzyskanych rezultatów oraz sformułowanie wniosków końcowych.

4. Badania nad defensywnymi zastosowaniami rozproszonych metod ukrywania informacji w sieciach

4.1. Podsumowanie artykułów [A6] i [A7]

W rozdziale 4 omówiono badania autora w zakresie koncepcji rozproszonego ukrywania informacji w sieciach z punktu widzenia zapewniania cyberodporności i aspektów prywatności. Wpisuje się to w ogólny trend wykorzystania technik z obszaru *cyber deception* w szerszym zakresie rozwiązań w cyberbezpieczeństwie. Należy mieć na uwadze stosowanie tych technik do realizacji operacji ofensywnych, gdzie atakujący podnoszą dzięki nim swoją skuteczność (trudniejsze wykrycie, omijanie zabezpieczeń czy brak zostawiania oczywistych śladów działania). Techniki *cyber deception* rozważane są od wielu lat także jako rozwiązania defensywne dla cyberobrony, jako np. *honeypoty* [52]. Taksonomię technik *cyber deception* stosowanych w cyberobronie przedstawiają m.in. publikacje [6] oraz [7]. Rezultaty badań autora w zakresie defensywnych zastosowań rozproszonych metod ukrywania zaprezentowano w artykułach [A6] (tekst artykułu jest załączony na str. 153) oraz [A7] (tekst artykułu jest załączony na str. 158).

Cloud computing jest ostatnio szeroko rozwijany i stosowany jako scentralizowany model obliczeń. Równolegle rozwijają się sieci teleinformatyczne Internetu Rzeczy (IoT), obejmujące różne inteligentne i połączone urządzenia, np. telefony komórkowe, urządzenia do noszenia tzw. *wearables*, czujniki, urządzenia sterowania sieciami energetycznymi itp. Liczba takich urządzeń miała przekroczyć 50 miliardów do 2020 r. [53].

Pomiędzy tymi trendami rozwija się kolejny paradygmat przetwarzania danych w podejściu rozproszonym, określany jako mgła obliczeniowa (ang. *fog computing*) [54]. W [20] autorzy traktują koncepcję mgły obliczeniowej jako mechanizm bezpieczeństwa sieci i danych, określając ją jako *cyberfog*. Podejście to zakłada, że system cechujący się większą niepewnością z punktu widzenia cyberatakujących zapewnia większą odporność na ataki. Jego bezpośrednią realizacją jest dzielenie danych na wiele fragmentów i rozproszenie ich na wielu urządzeniach użytkowników końcowych. Nawet jeśli system mógłby zostać częściowo skompromitowany, przechwycone informacje byłyby bezużyteczne dla atakującego, a jednocześnie byłyby nadal przydatne dla użytkowników systemu.

Autorzy dostrzegli wiele zalet tego podejścia, wskazując przy tym na wyzwania w zakresie złożoności zarządzania danymi i siecią, przepustowości, pamięci masowej, zapotrzebowania na energię baterii, opóźnień w ponownym odtwarzaniu danych i przerywanej łączności. Ostateczne rozwiązania wymagają właściwego kompromisu

między dostępnością a poufnością w zależności od zadań i okoliczności operacyjnych.

Opisana koncepcja cybermgły została podjęta w badaniach zaprezentowanych w artykule [A6] jako rozwinięcie zastosowań metod opisanych w artykułach [A1] i [A2] (rozdział 2). Efektem tych prac jest koncepcja rozproszonego systemu steganograficznego dla urządzeń IoT, która stosuje założenia mechanizmów StegHash i SocialStegDisc. Koncepcja ta obejmuje architekturę warstwową systemu w zakresie: warstwy aplikacyjnej, systemu plików, sektorów pamięci urządzenia oraz zarządzania pamięcią fizyczną. Każda z tych warstw zawiera, obok komponentu realizacji zadań podstawowych, komplementarny komponent steganograficzny właściwy dla danej warstwy. Całościowo system ma realizować protokół rozproszonej komunikacji kanałami steganograficznymi o charakterze hybrydowym, tj. łączącym steganografię komunikacyjną i steganografię przechowywania danych.

Motywacją badań przedstawionych w artykule [A7] było zaproponowanie nowego rozproszonego systemu steganograficznego do ochrony sieci zgodnych z podejściem *cyberfog*, jednocześnie rozwiązującego niektóre ze zdefiniowanych wyzwań dla takiego systemu, co zostało opisane w [20], a także w artykule [A6]. Docelowy system musi być możliwy do wdrożenia na urządzeniach końcowych użytkowników i oferować akceptowalny kompromis między dostępnością a poufnością danych.

Artykuł [A7] formułuje nowy system steganografii rozproszonej, określony jako StegFog. StegFog wykorzystuje schemat indeksowania wprowadzony przez StegHash i zapewnia podstawowe operacje, takie jak w SocialStegDisc (obie metody steganograficzne podsumowują artykuły [A1] i [A2]). Konstrukcja systemu wykorzystuje pamięć urządzeń końcowych użytkowników jako nośnik dla obiektów z ukrytymi danymi. Proponowane są również metody routingu fragmentów danych i znajdowania kolejnych fragmentów w sieci P2P (ang. *Peer-to-Peer*). Analiza stanu wiedzy przeprowadzona w artykule [A7] wskazuje na rosnące potrzeby projektowania, wdrażania i walidacji nowych mechanizmów bezpieczeństwa w podejściu *fog computing*. Artykuł [A7] uzupełnia stan wiedzy w zakresie badań nad rozproszoną steganografią o kompleksowe opracowanie nowego systemu steganograficznego StegFog o charakterze hybrydowym, zastosowanego w połączeniu z podejściem *fog computing*. StegFog łączy trzy typy klasycznych podejść do steganografii w jeden system rozproszonej steganografii:

- steganografię opartą na plikach, np. obrazach (steganografia obrazkowa), do przechowywania danych,
- steganografię tekstową do zarządzania wskaźnikami do ukrytych fragmentów danych,
- steganografię sieciową na potrzeby ukrytej komunikacji w systemie.

W ramach artykułu [A7] zaimplementowano działający prototyp systemu StegFog, aby następnie poddać go badaniom w zakresie bezpieczeństwa i wydajności. Ważnym aspektem badania bezpieczeństwa była analiza mechanizmu adresacji StegFog, który opiera się na generowaniu i kodowaniu ukrytych informacji przy pomocy łańcuchów znaków alfanumerycznych. Do ewaluacji wykorzystano entropię względną jako popularną metrykę detekcji dla takich metod w powiązaniu z algorytmicznym przetwarzaniem łańcuchów tekstowych oraz języków. W przypadku StegFog wykazano wysoką odporność tego mechanizmu. Następnie dokonano oceny parametrów operacyjnych zgodnie z wyzwaniem zdefiniowanymi przez [20], w szczególności pod względem dystrybucji fragmentów danych, złożoności zarządzania siecią, szybkością przesyłania danych, zajętości dysku urządzenia końcowego i wykorzystania jego mocy obliczeniowej. Bezpieczeństwo StegFog zapewnia przede wszystkim charakter logicznego połączenia pomiędzy rozproszonymi częściami łańcucha danych. System mógłby zostać właściwie skompromitowany tylko wtedy, gdyby atakujący przejął moduł funkcji generowania sekretów. Bez niego przechwycone części, np. poprzez skompromitowanie jednego lub kilku urządzeń, byłyby bezużyteczne. Zaproponowane rozwiązanie spełnia tym samym podstawowe wymaganie dotyczące podejścia *cyberfog*. Implementację prototypu StegFog można uznać za jeden z pierwszych działających systemów zgodnych z podejściem *cyberfog*, zdefiniowanym przez [20].

4.2. Osiągnięcia badawcze autora

Główne osiągnięcia autora rozprawy w zakresie badań przedstawionych w artykule [A6] (str. 153) to:

- zaproponowanie podstawowych mechanizmów steganograficznych dla komunikacji w paradygmacie *cyberfog* wraz z propozycją ich realizacji:
 - mechanizmu indeksacji opartego na metodzie StegHash (rozdział 2),
 - mechanizmu dyspersji opartego na metodzie SocialStegDisc (rozdział 2) i przy wykorzystaniu pamięci urządzeń IoT,
 - rozszerzeniu zbioru nośników ukrytych informacji poza materiały multimedialne,
 - zastosowaniu istniejącej platformy komunikacji steganograficznej takiej jak np. platforma TrustMAS [55];
- zaprojektowanie wstępnej architektury rozwiązania w przekroju warstw: aplikacyjnej, systemu plików, sektorów pamięci urządzenia, zarządzania pamięcią fizyczną;

- opracowanie założeń dla protokołu komunikacji i aspektów operacyjnych systemu końcowego;
- ewaluacja teoretyczna aspektów: bezpieczeństwa, niezawodności, wykorzystania pamięci i czasu działania dla proponowanego systemu w odniesieniu do każdej z jego warstw.

Główne osiągnięcia autora rozprawy w zakresie badań przedstawionych w artykule [A7] (str. 158) to:

- rozwinięcie koncepcji teoretycznej przedstawionej w artykule [A6] w nowy system steganograficzny StegFog wraz z zaprojektowaniem rozszerzonej i uogólnionej architektury tego systemu;
- zdefiniowanie podstawowych mechanizmów i protokołów systemu StegFog w zakresie:
 - zapisywania i odczytywania danych ukrytych w systemie,
 - ukrytej notyfikacji o dostępności danych,
 - ukrytego routingu do kolejnych fragmentów danych,
 - mechanizmu adresacji kolejnych fragmentów danych;
- implementacja prototypu systemu StegFog, który można uznać za jeden z pierwszych działających systemów zgodnych z podejściem *cyberfog*, zdefiniowanym przez [20];
- ewaluacja teoretyczna aspektów systemu StegFog na podstawie projektu architektury oraz zdefiniowanych protokołów: bezpieczeństwa, niezawodności, wykorzystania pamięci i czasu działania;
- ewaluacja praktyczna bezpieczeństwa mechanizmu adresacji StegFog z wykorzystaniem znanych metod detekcji opartych na teorii kodowania (entropia względna) i algorytmicznym przetwarzaniu łańcuchów tekstowych oraz języków;
- ewaluacja praktyczna parametrów operacyjnych systemu StegFog, takich jak czas działania, zajętość pamięci czy przepływność steganograficzna.

5. Podsumowanie

W rozprawie zaprezentowano wyniki badań nad różnymi aspektami rozproszonych metod ukrywania informacji w sieciach. Proces badawczy realizowano zgodnie z nowoczesnym podejściem do budowania cyberobrony opartej na wiedzy i danych, w szczególności w obszarze detekcji cyberzagrożeń. Rozprawa udowadnia, jak cyberbezpieczeństwo w swoim wymiarze praktycznym łączy się z koniecznością prowadzenia badań naukowych. Potrzebę powiązania nauki i cyberbezpieczeństwa podsumowuje raport *Science of Cyber-Security* [56] z 2010 roku.

W artykule [57] autor podejmuje dyskusję, czy nauka o cyberbezpieczeństwie (ang. *science of cybersecurity*) to już nowa dyscyplina, czy jeszcze obszar badawczy na styku innych nauk. Jako wkład do tej dyskusji zaproponował opis praktycznych problemów cyberbezpieczeństwa za pomocą klas opisanych notacją matematyczną i funkcyjną. Tabela 2 przedstawia zastosowanie tej notacji do opisu działań badawczych [D1]–[D6], zrealizowanych w artykułach [A1]–[A7].

Tabela 2: Zastosowanie notacji z [57] do opisu działań badawczych [D1]–[D6] zrealizowanych w artykułach [A1]–[A7]

Artykuł	Działania badawcze (rozdział 1, pkt 1.3.)	Opis problemu na podstawie [57]
[A1]	[D4]	$T_d, N_d, i \rightarrow T_a$
[A2]	[D4] [D5]	$T_d, N_d, i \rightarrow T_a$
[A3]	[D5]	$T_d, N_d, i \rightarrow T_a$
[A4]	[D1] [D2] [D3] [D5]	$T_d, N_d, i \rightarrow T_a$ $N_d, T_a, i \rightarrow T_d$
[A5]	[D1] [D2] [D3] [D5]	$T_d, N_d, i \rightarrow T_a$ $N_d, T_a, i \rightarrow T_d$
[A6]	[D6]	$T_d, T_a, i \rightarrow N_d$
[A7]	[D6]	$T_d, T_a, i \rightarrow N_d$

T_d	oznaczenie dla technik i narzędzi cyberobrony
N_d	oznaczenie dla środowiska teleinformatycznego podlegającego ochronie
T_a	oznaczenie dla technik i narzędzi cyberatakujących
I	oznaczenie dla incydentów bezpieczeństwa w cyberprzestrzeni

Główny cel budowania cyberobrony opartej na wiedzy i danych formalnie może być opisany przekształceniem:

$$T_d, N_d, T_a \rightarrow I(t)$$

co oznacza, że dobre algorytmy cyberobrony (T_d), wymagania dotyczące bezpiecznych

architektur sieci i ich ogólna znajomość (N_d), a także poznawanie natury możliwych działań ofensywnych w cyberprzestrzeni (T_a) prowadzą do osiągnięcia zdolności do obrony przed przyszłymi cyberzagrożeniami ($I(t), t > t_{now}$) oraz w przypadku konieczności reakcji na trwające incydenty ($I(t), t < t_{now}$). Łącząc to z podsumowaniem w tabeli 2, widać, że rezultaty prac przedstawionych w artykułach [A1]–[A7] dostarczają rozwiązań, których wyniki mogą być opisane jako T_d , T_a oraz N_d . Wyniki te spełniają cele rozprawy sformułowane w rozdziale 1 w pkt. 1.1.

Podsumowanie osiągnięć niniejszej rozprawy doktorskiej i jej wkładu do stanu wiedzy:

- [O1] Opracowanie koncepcji nowej metody steganograficznej SocialStegDisc i zaimplementowanie jej prototypu na potrzeby badania jej stosowalności, efektywności i cech charakterystycznych, w szczególności w odniesieniu do możliwości jej wykorzystania w cyberatakach.

Opis prac przedstawiono w rozdziale 2 na podstawie artykułów [A1] (str. 53) i [A2] (str. 58).

- [O2] Przeprowadzenie analizy teoretycznej stanu wiedzy o zastosowaniu steganografii w kanałach Command & Control (C2), opracowanie autorskiej taksonomii modeli operacyjnych takich kanałów oraz rozszerzenie metodyk modelowania cyberzagrożeń w zakresie stosowania w nich technik steganografii.

Opis prac przedstawiono w rozdziale 3 w pkt. 3.1. na podstawie artykułu [A3] (str. 82).

- [O3] Zaprojektowanie i zaimplementowanie autorskiego środowiska do symulacji rozproszonych metod steganografii na potrzeby zbierania zestawów danych do ich analizy.

Opis prac przedstawiono w rozdziale 3 w pkt. 3.2. na podstawie artykułów [A4] (str. 109) i [A5] (str. 131).

- [O4] Opracowanie architektury oraz implementacja prototypu systemu wieloagentowego do monitorowania i wykrywania cyberzagrożeń wykorzystujących metody ukrywania informacji.

Opis prac przedstawiono w rozdziale 3 w pkt. 3.2. na podstawie artykułów [A4] (str. 109) i [A5] (str. 131).

- [O5] Zaproponowanie, implementacja i przeprowadzenie badań weryfikujących słuszność koncepcji nowego algorytmu do wykrywania steganografii w podejściu rozproszonym za pomocą systemu wieloagentowego.

Opis prac przedstawiono w rozdziale 3 w pkt. 3.2. na podstawie artykułów [A4] (str. 109) i [A5] (str. 131).

- [O6] Implementacja i przeprowadzenie badań weryfikujących zastosowanie znanych algorytmów uczenia maszynowego (*Random Forest*, *Multilayer Perceptron*) do wykrywania steganografii w podejściu rozproszonym za pomocą systemu wieloagentowego.

Opis prac przedstawiono w rozdziale 3 w pkt. 3.2. na podstawie artykułów [A4] (str. 109) i [A5] (str. 131).

- [O7] Opracowanie sposobu zastosowania metod StegHash i SocialStegDisc jako mechanizmów architektury bezpieczeństwa danych w koncepcji *fog computing* poprzez zaprojektowanie stosu protokołów do realizacji operacyjnej koncepcji oraz wykonanie analizy aspektów jej bezpieczeństwa.

Opis prac przedstawiono w rozdziale 4 na podstawie artykułu [A6] (str. 153).

- [O8] Implementacja, badania empiryczne oraz ocena bezpieczeństwa prototypu systemu StegFog, który powstał poprzez rozwinięcie założeń przedstawionych w artykule [A6].

Opis prac przedstawiono w rozdziale 4 na podstawie artykułu [A7] (str. 158).

Większość prac badawczych przedstawiona w rozprawie były realizowana w ramach projektów badawczo-rozwojowych dofinansowanych w programie CyberSecIdent Narodowego Centrum Badań i Rozwoju (NCBiR):

- Artykuły [A2] oraz [A7] - projekt pt. *Zaawansowane Laboratorium Kryminalistyki Śledczej*, 2017–2019. Numer projektu CYBERSECIDENT/369234/I/NCBR/2017.
- Artykuły [A3], [A4] oraz [A5] - projekt pt. *Platforma detekcji anomalii sieciowych (PDAS)*, 2017–2019. Numer projektu CYBERSECIDENT/369532/I/NCBR/2017.

Założeniem projektów realizowanych we wskazanym programie było, aby kończyły się one wdrożeniem wyników prac i komercjalizacją przez jednego z konsorcjantów. Oznaczało to, że prace badawcze w tych projektach musiały być ukierunkowane na osiągnięcie zarówno satysfakcjonujących wyników naukowych, np. ulepszenie czy wymyślenie nowej metody, jak i możliwości ich praktycznego zastosowania w jakimś produkcie czy usłudze. Biorąc ten aspekt pod uwagę, zaprezentowane rezultaty stanowią istotne osiągnięcie jako połączenie metod naukowych z praktyczną inżynierią teleinformatyczną w celu rozwiązywania kluczowych wyzwań podnoszenia bezpieczeństwa w cyberprzestrzeni.

6. Dorobek naukowy autora

Niniejszy rozdział prezentuje najważniejszy dorobek naukowy autora, osiągnięty w okresie 2017–2022. Według stanu z 5 września 2022 roku w serwisie Google Scholar (https://scholar.google.com/citations?user=YFcDJ_QAAAAJ&hl=pl oraz Repo PW (<https://repo.pw.edu.pl/info.seam?tab=&enttype=author&id=WUTf5885ace40cd47ce83945896632616f3&lang=pl>) autor osiągnął w tym okresie:

- h-indeks **3**,
- sumaryczny poziom cytowań wynoszący **37**,
- całkowitą punktację MEiN równą **459** na podstawie **17** publikacji.

6.1. Publikacje

Artykuły

- **Bieniasz, J., Bąk P., & Szczypiorski, K. (2022). StegFog: Distributed Steganography Applied To Cyber Resiliency In Multi Node Environments. IEEE Access. 2022.**

IF: **3.476**; liczba cytowań: **0**

Punkty ministerialne (1.12.2021): **100**

- **Bieniasz, J., & Szczypiorski, K. (2021). Dataset Generation for Development of Multi-Node Cyber Threat Detection Systems. Electronics. 2021; 10(21):2711.**

IF: **2.690**; liczba cytowań: **1**

Punkty ministerialne (1.12.2021): **100**

- **Bieniasz, J., & Szczypiorski, K. (2019). Steganography Techniques for Command and Control (C2) Channels. In Botnets. Architectures, Countermeasures, and Challenges. (pp. 189-216). CRC Press.**

IF: —; liczba cytowań: **0**

Punkty ministerialne (1.12.2021): **50**

- **Bieniasz, J.,** Stępkowska, M., Janicki, A., & Szczypiorski, K. (2019). **Mobile Agents for Detecting Network Attacks Using Timing Covert Channels.** Journal of Universal Computer Science, 25(9), 1109-1130.

IF: **1.056**; liczba cytowań: **7**

Punkty ministerialne (1.12.2021): **40**

- **Bieniasz, J., & Szczypiorski, K.** (2019). **Methods for Information Hiding in Open Social Networks.** Journal of Universal Computer Science, 25(2), 74-97.

IF: **1.056**; liczba cytowań: **2**

Punkty ministerialne (1.12.2021): **40**

Materiały konferencyjne

- Kukliński, S., Czerwiński, M., **Bieniasz, J.,** Kamińska, K.H, Paczesny, D., & Szczypiorski, K. (2020) **Evaluation of Privacy and Security of GSM Using Low-Cost Methods.** 2020 World Conference on Computing and Communication Technologies (WCCCT), 2020, pp. 96-105.

IF: –; liczba cytowań: **0**

Punkty ministerialne (1.12.2021): **20**

- Brodzki, A. M., & **Bieniasz, J..** (2019). **Yet Another Network Steganography Technique Based on TCP Retransmissions.** 2019 5th International Conference on Frontiers of Signal Processing (ICFSP), 2019, pp. 35-39.

IF: –; liczba cytowań: **1**

Punkty ministerialne (1.12.2021): **20**

- **Bieniasz, J., & Szczypiorski, K. (2018). New method for information hiding in open social networks.** Proc. SPIE 10808, Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments 2018, 108082Z.

IF: –; liczba cytowań: **0**

Punkty ministerialne (1.12.2021): **20**

- Bąk, P., **Bieniasz, J.**, Krzemiński, M., & Szczypiorski, K. (2018). **Application of perfectly undetectable network steganography method for malware hidden communication.** 2018 4th International Conference on Frontiers of Signal Processing (ICFSP), 2018, pp. 34-38.

IF: –; liczba cytowań: **13**

Punkty ministerialne (1.12.2021): **20**

- **Bieniasz, J., & Szczypiorski, K. (2018). Towards Empowering Cyber Attack Resiliency Using Steganography.** 2018 4th International Conference on Frontiers of Signal Processing (ICFSP), 2018, pp. 24-28.

IF: –; liczba cytowań: **1**

Punkty ministerialne (1.12.2021): **20**

- **Bieniasz, J., & Szczypiorski, K. (2017). SocialStegDisc: Application of steganography in social networks to create a file system.** 2017 3rd International Conference on Frontiers of Signal Processing (ICFSP), 2017, pp. 76-80.

IF: –; liczba cytowań: **7**

Punkty ministerialne (1.12.2021): **20**

6.2. Wystąpienia konferencyjne

6.2.1. Konferencje naukowe

Prezentacje publikacji

- **Evaluation of Privacy and Security of GSM Using Low-Cost Methods.** 2020 World Conference on Computing and Communication Technologies (WCCCT).
- **Yet Another Network Steganography Technique Based on TCP Retransmissions.** 2019 5th International Conference on Frontiers of Signal Processing (ICFSP).
- **New method for information hiding in open social networks.** XLII-nd IEEE-SPIE Joint Symposium Wilga 2018.
- **Application of perfectly undetectable network steganography method for malware hidden communication.** 2018 4th International Conference on Frontiers of Signal Processing (ICFSP).
- **Towards Empowering Cyber Attack Resiliency Using Steganography.** 2018 4th International Conference on Frontiers of Signal Processing (ICFSP).
- **SocialStegDisc: Application of steganography in social networks to create a file system.** 2017 3rd International Conference on Frontiers of Signal Processing (ICFSP).

Prezentacje otwierające (keynotes)

- **Recent advances in information hiding techniques research at WUT.** 2019 5th International Conference on Frontiers of Signal Processing (ICFSP).
- **Can your OSN profile be a part of the biggest data leakage in history?.** 2017 3rd International Conference on Frontiers of Signal Processing (ICFSP).

6.2.2. Konferencje branżowe

- **Dane kontrATT&Ckują – cyberobrona oparta na danych.** Udział, wygłoszenie prelekcji (nagranie) oraz moderator dyskusji podczas konferencji TheHackSummit 2020.
- **Szukanie igły w stogu siana, czyli obrona przed zagrożeniami w cyberprzestrzeni oparta na danych i metodach ich modelowania.** Wygłoszenie prelekcji podczas zdalnego spotkania środowiska SecOps Polska (nagrywane na YouTube).

6.3. Projekty

- *Grafowa reprezentacja zdarzeń i strumieni komunikacji na potrzeby detekcji cyberzagrożeń.* (2021–2022). Kierownik: mgr inż. Jędrzej Bieniasz. Projekt dofinansowany w ramach wewnętrznego konkursu Centrum POB Cyberbezpieczeństwo i Analiza Danych Inicjatywa Doskonałości–Uczelnia Badawcza.
- *Budowa i rozwój zintegrowanego środowiska do testów cyberbezpieczeństwa oraz odkrywania złożonych podatności i błędów.* (2021–2022). Kierownik: mgr inż. Jędrzej Bieniasz. Projekt dofinansowany w ramach Grantu Rektorskiego PW.
- *Laboratorium badania podatności stacjonarnych i mobilnych urządzeń informatycznych oraz algorytmów i oprogramowania.* (2021–2024). Kierownik w PW: dr hab. inż. Krzysztof Szczypiorski, prof. Uczelni. Projekt dofinansowany przez Narodowe Centrum Badań i Rozwoju w ramach programu CyberSecIdent.
- *Platforma detekcji anomalii sieciowych.* (2017–2019). Kierownik w PW: dr hab. inż. Krzysztof Szczypiorski, prof. Uczelni. Projekt dofinansowany przez Narodowe Centrum Badań i Rozwoju w ramach programu CyberSecIdent.
- *Zaawansowane Laboratorium Kryminalistyki Śledczej.* (2017–2019). Kierownik w PW: dr hab. inż. Krzysztof Szczypiorski, prof. Uczelni. Projekt dofinansowany przez Narodowe Centrum Badań i Rozwoju w ramach programu CyberSecIdent.

6.4. Nagrody i wyróżnienia

- Zespołowa Nagroda Ministra w 2021 roku za znaczące osiągnięcia w zakresie działalności dydaktycznej (za stworzenie studiów I stopnia na kierunku cyberbezpieczeństwo).
- Nagroda Rady Dyscypliny Naukowej *Informatyka Techniczna i Telekomunikacja* w 2021 roku za pozyskanie grantu badawczego dofinansowanego przez Narodowe Centrum Badań i rozwoju pt. *Laboratorium badania podatności stacjonarnych i mobilnych urządzeń informatycznych oraz algorytmów i oprogramowania.*
- Stypendium na dofinansowanie działalności polegającej na prowadzeniu badań naukowych lub prac rozwojowych dla uczestnika studiów doktoranckich na Wydziale Elektroniki i Technik Informacyjnych PW (2019).
- Nagroda *Best Service Award* w 2019 roku za organizację konferencji IFIP Networking 2019 na Politechnice Warszawskiej.

- Udział w programie Doktorat Wdrożeniowy 2017–2018.
- Stypendium doktoranckie 2018–2021.
- Zwiększone stypendium z dotacji projakościowej 2017–2021.
- Stypendium dla najlepszych doktorantów PW 2017–2021.

Literatura

- [1] E. M. Hutchins, M. J. Cloppert, and R. M. Amin, “Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains,” *Leading Issues in Information Warfare and Security Research*, vol. 1, 01 2011.
- [2] S. J. Roberts and R. Brown, *Intelligence-Driven Incident Response: Outwitting the Adversary*. O’Reilly Media, Inc., 1st ed., 2017.
- [3] D. Chou and M. Jiang, “A Survey on Data-Driven Network Intrusion Detection,” *ACM Comput. Surv.*, vol. 54, oct 2021.
- [4] I. H. Sarker, A. S. M. Kayes, S. Badsha, H. Alqahtani, P. A. Watters, and A. Ng, “Cybersecurity data science: an overview from machine learning perspective,” *Journal of Big Data*, vol. 7, pp. 1–29, 2020.
- [5] K. Szczypiorski, L. Wang, X. Luo, and D. Ye, “Big Data Analytics for Information Security,” *Secur. Commun. Networks*, vol. 2018, pp. 7657891:1–7657891:2, 2018.
- [6] N. C. Rowe, “A Taxonomy of Deception in Cyberspace,” 2006-03. <http://hdl.handle.net/10945/35976>.
- [7] M. H. Almeshekah and E. H. Spafford, *Cyber Security Deception*, pp. 23–50. Cham: Springer International Publishing, 2016.
- [8] W. Mazurczyk, S. Wendzel, S. Zander, A. Houmansadr, and K. Szczypiorski, *Information Hiding in Communication Networks: Fundamentals, Mechanisms, Applications, and Countermeasures*. Wiley-IEEE Press, 1st ed., 2016.
- [9] K. Szczypiorski, “StegHash: New Method for Information Hiding in Open Social Networks,” *IJET International Journal of Electronics and Telecommunications*, vol. 62, no. 4, pp. 347–352, 2016.
- [10] F. Beato, E. D. Cristofaro, and K. B. Rasmussen, “Undetectable communication: The Online Social Networks case,” in *2014 Twelfth Annual International Conference on Privacy, Security and Trust*, pp. 19–26, July 2014.
- [11] A. Castiglione, B. D’Alessio, and A. D. Santis, “Steganography and Secure Communication on Online Social Networks and Online Photo Sharing,” in *2011 International Conference on Broadband and Wireless Computing, Communication and Applications*, pp. 363–368, Oct 2011.

- [12] M. Chapman, G. I. Davida, and M. Rennhard, “A Practical and Effective Approach to Large-Scale Automated Linguistic Steganography,” in *Proceedings of the 4th International Conference on Information Security, ISC '01*, (Berlin, Heidelberg), pp. 156–165, Springer-Verlag, 2001.
- [13] A. Wilson, P. Blunsom, and A. D. Ker, “Linguistic steganography on Twitter: hierarchical language modeling with manual interaction,” in *Media Watermarking, Security, and Forensics 2014, San Francisco, CA, USA, February 2, 2014, Proceedings*, p. 902803, 2014.
- [14] R. Anderson, R. Needham, and A. Shamir, “The Steganographic File System,” in *Information Hiding*, Lecture Notes in Computer Science, pp. 73–82, Springer, 1998.
- [15] A. D. McDonald and M. G. Kuhn, *StegFS: A Steganographic File System for Linux*, pp. 463–477. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000.
- [16] A. Venckauskas, N. Morkevicius, G. Petraitis, and J. Ceponis, “Covert Channel for Cluster-based File Systems Using Multiple Cover Files,” *ITC*, vol. 42, no. 3, pp. 260–267, 2013.
- [17] S. Neuner, A. G. Voyiatzis, M. Schmiedecker, S. Brunthaler, S. Katzenbeisser, and E. Weippl, “Time is on my side: Steganography in filesystem metadata,” *Digital Investigation*, vol. 18, Supplement, pp. 76–86, 2016.
- [18] J. Hiney, T. Dakve, K. Szczypiorski, and K. Gaj, “Using Facebook for Image Steganography,” *2015 10th International Conference on Availability, Reliability and Security*, pp. 442–447, 2015.
- [19] K. Szczypiorski, A. Janicki, and S. Wendzel, “The Good, The Bad and The Ugly: Evaluation of Wi-Fi Steganography,” *JCM*, vol. 10, no. 10, pp. 747–752, 2015.
- [20] A. Kott, A. Swami, and B. West, “The fog of war in cyberspace,” tech. rep., US Army Research Laboratory, 2016.
- [21] Kaspersky Lab, “Kaspersky Lab Identifies Worrying Trend in Hackers Using Steganography.” https://usa.kaspersky.com/about/press-releases/2017_kaspersky-lab-identifies-worrying-trend-in-hackers-using-steganography, 2017. [Online; Dostęp: 10.04.2022].
- [22] McAfee Labs, “McAfee Labs Threats Report.” <https://www.mcafee.com/enterprise/en-us/assets/reports/rp-quarterly-threats-jun-2017.pdf>, 2017. [Online; Dostęp: 10.04.2022].

- [23] J. Jarvis, “Steganography: Combatting Threats Hiding in Plain Sight.” <https://www.fortinet.com/blog/threat-research/steganography--combatting-threats-hiding-in-plain-sight.html>, 2018. [Online; Dostęp: 10.04.2022].
- [24] A. Alshamrani, S. Myneni, A. Chowdhary, and D. Huang, “A Survey on Advanced Persistent Threats: Techniques, Solutions, Challenges, and Research Opportunities,” *IEEE Communications Surveys Tutorials*, pp. 1–1, 2019.
- [25] “MITRE ATT&CK: globally-accessible knowledge base of adversary tactics and techniques based on real-world observations..” <https://attack.mitre.org>. [Online; Dostęp: 10.04.2022].
- [26] Threatpost, “MoleRats APT Returns with Espionage Play Using Facebook, Dropbox,” 2020. <https://threatpost.com/molerats-apt-espionage-facebook-dropbox/162162/>, [Online; Dostęp: 17.04.2022].
- [27] Security Affairs, “njRAT RAT operators leverage Pastebin C2 tunnels to avoid detection,” 2020. <https://securityaffairs.co/wordpress/112147/cyber-crime/njrat-rat-pastebin-c2.html>, [Online; Dostęp: 17.04.2022].
- [28] Cisco Talos Intelligence Group, “North Korean attackers use malicious blogs to deliver malware to high-profile South Korean targets,” 2021. <https://blog.talosintelligence.com/2021/11/kimsuky-abuses-blogs-delivers-malware.html>, [Online; Dostęp: 17.04.2022].
- [29] “MITRE ATT&CK: globally-accessible knowledge base of adversary tactics and techniques based on real-world observations. Data Obfuscation: Steganography (T1001.002)..” <https://attack.mitre.org/techniques/T1001/002/>. [Online; Dostęp: 10.04.2022].
- [30] J. Balasubramanian, J. Garcia-Fernandez, D. Isacoff, E. Spafford, and D. Zamboni, “An architecture for intrusion detection using autonomous agents,” in *Proceedings 14th Annual Computer Security Applications Conference (Cat. No.98EX217)*, pp. 13–24, 1998.
- [31] A. Herrero and E. Corchado, “Multiagent Systems for Network Intrusion Detection: A Review,” in *Advances in Intelligent and Soft Computing*, vol. 63, pp. 143–154, 01 2009.
- [32] M. Docking, A. V. Uzunov, C. Fiddymment, R. Brain, S. Hewett, and L. Blucher, “UNISON: Towards a Middleware Architecture for Autonomous Cyber Defence,” in *2015 24th Australasian Software Engineering Conference*, pp. 203–212, 2015.

- [33] I. A. Saeed, A. Selamat, M. F. Rohani, O. Krejcar, and J. A. Chaudhry, “A Systematic State-of-the-Art Analysis of Multi-Agent Intrusion Detection,” *IEEE Access*, vol. 8, pp. 180184–180209, 2020.
- [34] A. Kott, “Intelligent Autonomous Agents are Key to Cyber Defense of the Future Army Networks,” *The Cyber Defense Review*, vol. 3, no. 3, pp. 57–70, 2018.
- [35] A. Kind, M. P. Stoecklin, and X. Dimitropoulos, “Histogram-based traffic anomaly detection,” *IEEE Transactions on Network and Service Management*, vol. 6, no. 2, pp. 110–121, 2009.
- [36] Z. Miller, W. Deitrick, and W. Hu, “Anomalous network packet detection using data stream mining,” *Journal of Information Security*, vol. 2, no. 04, p. 158, 2011.
- [37] I. Sharafaldin, A. H. Lashkari, and A. Ghorbani, “Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization,” in *ICISSP*, 2018.
- [38] M. Ring, S. Wunderlich, D. Grödl, D. Landes, and A. Hotho, “Creation of Flow-Based Data Sets for Intrusion Detection,” *Journal of Information Warfare*, vol. 16, no. 4, pp. 41–54, 2017.
- [39] M. H. Shahriar, N. I. Haque, M. A. Rahman, and M. Alonso, “G-IDS: Generative Adversarial Networks Assisted Intrusion Detection System,” in *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*, pp. 376–385, 2020.
- [40] Canadian Institute for Cybersecurity, “Intrusion detection evaluation dataset (ISCXIDS2012).” <https://www.unb.ca/cic/datasets/ids.html>, [Online; Dostęp: 11.04.2022].
- [41] Canadian Institute for Cybersecurity, “Intrusion Detection Evaluation Dataset (CICIDS2017).” <https://www.unb.ca/cic/datasets/ids-2017.html>, [Online; Dostęp: 11.04.2022].
- [42] Canadian Institute for Cybersecurity, “A Realistic Cyber Defense Dataset (CSE-CIC-IDS2018).” <https://www.unb.ca/cic/datasets/ids-2018.html>, [Online; Dostęp: 11.04.2022].
- [43] Canadian Institute for Cybersecurity, “NSL-KDD dataset.” <https://www.unb.ca/cic/datasets/ns1.html>, [Online; Dostęp: 11.04.2022].

- [44] N. Moustafa and J. Slay, “UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set),” in *2015 Military Communications and Information Systems Conference (MilCIS)*, pp. 1–6, 2015.
- [45] G. Maciá-Fernández, J. Camacho, R. Magán-Carrión, P. García-Teodoro, and R. Therón, “UGR’16: A new dataset for the evaluation of cyclostationarity-based network IDSs,” *Comput. Secur.*, vol. 73, pp. 411–424, 2018.
- [46] Center for Applied Internet Data Analysis, “CAIDA datasets.” <https://www.caida.org/catalog/datasets/overview/>, [Online; Dostęp: 11.04.2022].
- [47] Information Marketplace For Policy and Analysis of Cyber Risk & Trust. <https://www.impactcybertrust.org>, [Online; Dostęp: 11.04.2022].
- [48] A. L. Buczak and E. Guven, “A survey of data mining and machine learning methods for cyber security intrusion detection,” *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016.
- [49] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, and W.-Y. Lin, “Intrusion detection by machine learning: A review,” *Expert Systems with Applications*, vol. 36, no. 10, pp. 11994–12000, 2009.
- [50] D. Kwon, H. Kim, J. Kim, S. C. Suh, I. Kim, and K. J. Kim, “A survey of deep learning-based network anomaly detection,” *Cluster Computing*, Sep 2017.
- [51] “CICFlowMeter: Ethernet traffic Bi-flow generator and analyzer,” 2018. <https://github.com/ahlashkari/CICFlowMeter>, [Online; Dostęp: 05.02.2020 (wersja aplikacji)].
- [52] L. Spitzner, “The HoneyNet Project: trapping the hackers,” *IEEE Security & Privacy*, vol. 1, no. 2, pp. 15–23, 2003.
- [53] D. Evans, “The internet of things: How the next evolution of the internet is changing everything,” *CISCO white paper*, vol. 1, no. 2011, pp. 1–11, 2011.
- [54] M. Chiang and T. Zhang, “Fog and IoT: An Overview of Research Opportunities,” *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854–864, 2016.
- [55] K. Szczypiorski, I. Margasinski, W. Mazurczyk, K. Cabaj, and P. Radziszewski, “TrustMAS: Trusted Communication Platform for Multi-Agent Systems,” in *OTM Conferences*, 2008.

- [56] JASON. MITRE Corporation., *Science of Cyber-security*. JASON, MITRE Corporation, 2010.
- [57] A. Kott, *Towards Fundamental Science of Cyber Security*, pp. 1–13. New York, NY: Springer New York, 2014.

A. Artykuły

W rozdziale podano artykuły, które stanowią cykl publikacji w ramach rozprawy doktorskiej.

- Artykuł [A1]: **SocialStegDisc: Application of Steganography in Social Networks to Create a File System** - str. 53
- Artykuł [A2]: **Methods for Information Hiding in Open Social Networks** - str. 58
- Artykuł [A3]: **Steganography techniques for Command and Control (C2) Channels** - str. 82
- Artykuł [A4]: **Mobile Agents for Detecting Network Attacks Using Timing Covert Channels** - str. 109
- Artykuł [A5]: **Dataset Generation for Development of Multi-Node Cyber Threat Detection Systems** - str. 131
- Artykuł [A6]: **Towards Empowering Cyber Attack Resiliency Using Steganography** - str. 153
- Artykuł [A7]: **StegFog: Distributed Steganography Applied To Cyber Resiliency In Multi Node Environments** - str. 158

A.1. Artykuł [A1]

2017 3rd International Conference on Frontiers of Signal Processing

SocialStegDisc: Application of Steganography in Social Networks to Create a File System

Jędrzej Bieniasz

Institute of Telecommunications
Warsaw University of Technology
Warsaw, Poland
e-mail: J.Bieniasz@stud.elka.pw.edu.pl

Krzysztof Szczypiorski

Institute of Telecommunications
Warsaw University of Technology
Warsaw, Poland
e-mail: ksz@tele.pw.edu.pl

Abstract—In this paper, the concept named SocialStegDisc was introduced. This concept is as an application of the original idea of StegHash method which uses multimedia files as a carrier of hidden information and connects them logically with a mechanism of hashtags. SocialStegDisc is a new kind of mass-storage characterized by unlimited space and provides the set of operations on files: creation, reading, deletion and modification. Furthermore, the attempts to improve the operation of StegHash by trade-off between memory requirements and computation time are presented: a) an application of linked list structure; b) introduction of a dynamic verification of free addresses to the multimedia objects. The features, limitations and opportunities of the design were discussed.

Keywords—filesystem; hashtag; information hiding; open social net-works; steganography; StegHash

I. INTRODUCTION

In an increasing number of scenarios of digital economy, security mechanisms in storage, modification or sharing of sensitive data require to be considered and applied. Most IT systems use external mass storage devices (not embedded directly in the processor) to store data of various sizes. Access to and management of their physical form is ensured by a filesystem.

An external storage device, referred to as a drive, is deemed to be a safe drive when physical and software implementation of a given drive performs security services: confidentiality, authenticity and integrity of data. The fourth desired feature of safe drives is hiding performance of basic operations in a filesystem. Steganography may be a way of hiding communication with drives or performance of drive operations. The purpose of steganography is to hide information in a carrier of this information in a way which is unnoticeable to the observer.

Throughout the years, application of steganography in mass storage has evolved into a separate field of research referred to as File System Steganography. A milestone in this research area was the work of Anderson, Needham and Shamir [1]. They proposed a steganographic filesystem allowing to deny the existence of entire file directories. In general, steganographic filesystems are an application of steganography, therefore their development is often preceded by appearance of new steganographic techniques. This work is an example of such coupling. In 2016 a new type of a

steganographic technique called StegHash was proposed in [2]. In this system, multimedia files placed in open social networks (OSNs) are a carrier of hidden information. The essence of the system is the existence of a logical connection between them by means of a mechanism of hashtags, that is text markers in the form of #<tag>, commonly applied in OSNs. In addition, the author of [2] proposes an option of applying the StegHash technique to create a steganographic filesystem analogous to the existing classic filesystems such as FAT (File Allocation Table) or NTFS (New Technology File System). This work focuses on extending this application of the StegHash method by proposing a new type of technique called SocialStegDisc. The paper is structured as follows: first we reviewed the literature on steganographic filesystems and use of OSNs in steganography in Section II. Section III presents the concept of the SocialStegDisc method. In Section IV features, limitations and opportunities of the new system were discussed. In Section V we present details of the implementation. The paper concludes with a brief summary in Section VI.

II. REVIEW OF LITERATURE

A. Steganographic Filesystems

As mentioned above, a breakthrough in the development of steganographic filesystems is an article written by Anderson, Needham and Shamir [1]. The authors proposed a method utilizing the way in which invisible ink works. A user with a password is able to find and decode hidden information. The StegFS system presented in [3] is a practical implementation of the concept presented in [1] as an extension of a standard filesystem, Linux Ext2fs.

In recent years, distributed filesystems used in clusters of various kinds have been gaining popularity. For such systems, proposals of applying steganography to hide information in them also appear, for instance in [4]. The main idea behind the method presented in [4] consists in using multiple files to encode hidden information by means of their relative positions in clusters.

Among recent proposals, work presented in [5] is worth highlighting, in which Neuner et al. presented a way of hiding information in timestamps of files saved in classic filesystems. They discovered that information may be effectively hidden in 24 bits (3B) coding the nanosecond part of the timestamp in the NTFS filesystem. For each file, it is

possible to use two types of timestamps: of creation and of last access. In this way, one file creates 6B of space for hidden data.

B. Use of OSNs in Steganography

As soon as open social networks appeared, steganography researchers began searching for options to apply them in steganographic systems.

In [6] the authors define two types of communication: with high and low entropy. In a high entropy model, multimedia files such as images, movies or music are data carriers. This is a classic example of multimedia steganography, where a single object carries hidden information. A covert channel of this type is characterized by high throughput but is easy to detect. A low entropy model consists in using text data to carry hidden information. In addition, information from such a hidden information carrier is decoded by means of a previously known method (secret). A hidden channel of this type is characterized by a low throughput but a good level of security. The authors also propose using such a channel to establish the location of the appropriate steganogram, and they indicate Internet services, for instance social networks, as a carrier. Proposals of low entropy steganographic methods are presented in [7]. The first one uses file names as a hidden information carrier and requires OSNs not to change them. For this method, the authors propose using standard file naming schemes by photographic devices. Sequence numbers in these names are a hidden data carrier. The second method assumes using tags in pictures. Numerous images with numerous users' tags creates a hidden channel. On the basis of a prepared set of pictures and a set of tags, it is possible to develop a matrix encoding hidden information. Other approaches based on language features are presented in [8] and [9]. Steganograms are transferred as bitmaps coded by means of language permutation. The created channel is very safe but there is a high likelihood of its distortion.

All these methods use a classic multimedia steganography, easy to detect, or a more sophisticated one, with a limited steganographic throughput. The proposed methods assume using multiple user accounts in OSNs and a parallel access to an abundance of files, which may be an insecurity indicator on the servers' side and an operation blockage.

III. CONCEPT OF THE SYSTEM

There is a basic limitation to applying StegHash in SocialStegDisc: memory occupancy. Along with the increase of the number of n hashtags, the volume of the dictionary increases proportionately to $n!$. Therefore, keeping the entire dictionary in the memory places a heavy burden on the system, and generation of the dictionary and restoring the set of used addresses will consume a large amount of time. The main goal behind keeping these dictionaries is ensuring uniqueness and reliability of addressing. Therefore, for a higher n it is proposed to introduce a dynamic verification of uniqueness of the generated addresses. Two space-time tradeoff-type modifications will be formulated on its basis along with an application of the linked list mechanism.

A. Dynamic Verification of Uniqueness of Addresses

The first attempt to increase the efficiency of the StegHash method is to eliminate storing results from generation of the utilized address space when creating a chain and later restoring it. Such a set was stored because it was necessary to verify the uniqueness of the generated addresses. In this proposal, uniqueness is verified by checking in the OSNs whether there is a file with a given address.

B. Space-time Tradeoff Modifications and Linked List Mechanism

From the perspective of a correct system operation, the system of addressing, that is indexing SocialStegDisc segments with hashtags, is of key importance. A characteristic feature of such addressing is non-linearity. Subsequent addresses are generated by means of a specific function on the basis of the initial address. The issue of addressing in SocialStegDisc is an analogy to the problem of memory space fragmentation in classic filesystems. Until now, a strong assumption was a prohibition of hiding addresses of the system due to security reasons. This implies that it is not possible to apply mechanisms servicing memory fragmentation, for instance allocation tables or storing the index to another memory block together with the data. As a consequence, it is not possible to make available a function of deleting single files because an occurrence of any gap in addressing, if it is not known that the gap exists, would mean that the files located after the gap would be lost. Under this approach, once a file is added to SocialStegDisc, it will remain there. In this article, a solution is proposed which would allow to introduce structures for efficient management of resources (addresses), known from classic filesystems, to SocialStegDisc. To this end, the assumption of non-storing the addresses will be weakened by introducing such a mechanism which will simultaneously not decrease the security level.

Direct addresses to objects may be presented by means of an address to the object of reference (beginning of the SocialStegDisc instance) and a counter, whose value will serve to control obtaining the address of the subsequent object. In every multimedia object, space for this counter is added to hidden space. Due to this, the SocialStegDisc system assumes the structure of a linked list which allows to supplement the system with file deletion. An example of how this operates is presented in the figure below.

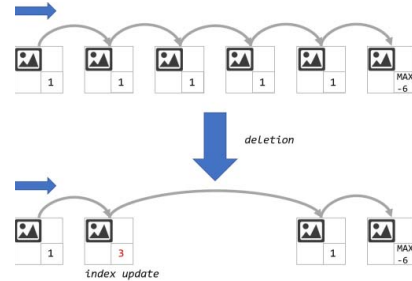
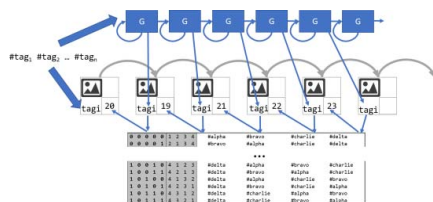


Figure 1. Mechanism of a linked list in SocialStegDisc

When StegHash with a modification eliminating the dictionary of used addresses is applied, when a chain of objects is created, a code of the address to the next object is placed in the additional space, in line with the coding from the dictionary of permutations. The scheme of this procedure is presented in Figure 2.



During reading, the address code is retrieved from the hidden space and then converted into an address in the form of a set of hashtags. Thus, access to the next object is possible. To maintain the continuity of the address space, when deleting a file in the multimedia object preceding the beginning of the given file, the counter should be updated to the value retrieved from the last object storing the file to be deleted. The set of the used addresses should also be updated. The released addresses are not lost but they go back to the pool and may be generated in the subsequent iterations of pseudo-random generation function.

Another proposed modification of space-time tradeoff type is a total elimination of original StegHash dictionaries and using the linked list mechanism to store relevant counters allowing to restore the chain. To this end, first of all, the algorithm of dynamic generation of addresses should be modified so that the function, apart from the new address, returns the current status of the sampling counter in the course of generation of subsequent addresses. With the starting point known, the obtained counter allows to retrieve the address returned in the course of generation at the counter status. As these states of the counter are recorded in the hidden space defined for the linked list mechanism, they allow to traverse the chain objects from a known initial

File deletion which entails creation of free address space, in this case updates the index in the object preceding the space to the value of the index from the last deleted file.

The issue of addressing in SocialStegDisc is an analogy to the problem of memory space fragmentation in classic filesystems, which has been highlighted when developing the concept. The SocialStegDisc concept introduces a mechanism of storing indexes to the next object in the system, due to which the drive assumes the structure of a linked list. Thanks to the application of a linked list mechanism, it is possible to efficiently use the space and propose a file deletion mechanism. The greatest problem is the calculation overhead and algorithms verifying the correctness of the operation scheme.

The system is characterized by non-linearity in filling in the byte space. This effect occurs when the last used object hides a smaller number of bytes than possible. On the basis of the number of bytes, it is possible to determine the length of the chain of multimedia files created in line with the StegHash technique essence. This size amounts to:

$$L = \left[\frac{M}{m} \right] \quad (1)$$

where M is the number of bytes on which the operation is performed, and m is the size of a single logical block. When $M \bmod m \neq 0$ in the last multimedia file in the chain, fewer bytes are placed than the size of space allocated in the multimedia file (m).

The SocialStegDisc undetectability level depends on:

- 1) The level of undetectability by open social networks of applied methods of multimedia steganography, serving to hide data;
- 2) The level of undetectability of the StegHash steganographic method which is a system of indexing multimedia objects and building the SocialStegDisc drive space from single blocks;
- 3) The level of security of social networks with respect to detection of anomalies in network users' behaviour. The SocialStegDisc system may communicate with the same service multiple times at short intervals. From the perspective of such a service, frequent and automated sending of queries will be treated as a threat, for instance as an attack of a Denial-of-Service type;
- 4) The level of detectability of multimedia steganography. This problem was under research

for Facebook, among others. In [10], the authors tested algorithms sanitizing steganography on the side of Facebook servers. The research shows that Facebook algorithms are efficient in the detection of steganography known so far, in particular for hidden information of greater size.

In [11], a way of evaluation of network steganography techniques was proposed in three categories of undetectability: good, bad and ugly. The presented classification manner may be applied to other steganographic methods. StegHash may be considered a good steganographic method, which transitively means that the proposed SocialStegDisc technique is also characterized by this feature. The observer is unable to detect the ongoing communication anywhere in the network, even at the receiver of hidden data.

V. IMPLEMENTATION

A. Environment

The system implementation proposal presented in this article is a proof-of-concept of the SocialStegDisc system. In Figure 4 a scheme of operation of the developed testing environment is presented.

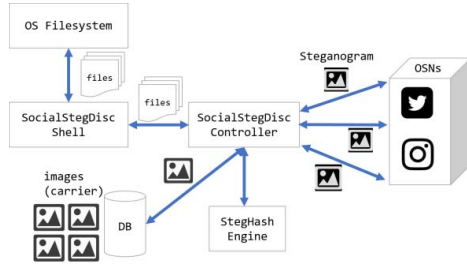


Figure 4. Scheme of operation of SocialStegDisc

The main client application is the SocialStegDisc Shell terminal, which translates the client queries to appropriate operations of the file directory and operations on files, using the Controller module. In addition, two modules known from StegHash [2] have been used:

- 1) StegHash Engine – a module responsible for the creation of appropriate addressing of multimedia files with the use of hashtags.
- 2) Database – contains a set of multimedia files used as a carrier of hidden data.

B. Generation of Addresses

The basic operation of the system is generation of addresses. This index plays two roles:

- 1) Unambiguously identifies the object,
- 2) Identifies the social network in which it is embedded.

In the StegHash system [2], addresses were generated by means of any type of permutation generation algorithm, and uniqueness was ensured by rejecting the already drawn addresses. For SocialStegDisc, generation of addresses in a chain must be characterized by restorability with the use of

appropriate initial conditions. To this end, an approach of Rivest [12] was used, who presented a reference code for a program generating a random sample of s size from set S . Assuming that $s = S$, the sample is a searched permutation of set S . In Figure 5, an exemplary code in Python of a chain generation method with no dictionaries stored is presented.

```
def go_to_perm(seed, counter):
    input = ""
    for num in seed:
        input += str(num)
    i = 0
    output_list = []
    while(i < counter):
        if(len(output_list) == len(seed)):
            for num in output_list:
                input = ""
                input = str(num)
                output_list = []
            i = i + 1
            hash_input = input + str(i)
            hash = hashlib.sha256(hash_input.encode('utf-8'))
            output = int(hash.hexdigest(),16)
            pick = int(output % len(seed))
            if not pick in output_list:
                output_list.append(pick)
            return output_list
```

Figure 5. Method of address generation based on reference address

The main part of the method is using the hash function for pseudorandom generation of the next value which is a candidate to become an element of an output permutation. In the example of the code, the function SHA-256 was used [13], but it is possible to apply other hash functions, for instance SHA-512 [13]. Using a hash function with a greater output size reduces the number of sampling iterations when completing the output table. The output value from the hash function is converted to a hexadecimal number, based on which a decimal value is generated, which is a pseudorandom number generated in a given loop iteration. This number undergoes the operation of obtaining remainder from division by the set size. At the end, it is verified whether the sampled value already exists in the output set from a given iteration of permutation generation. If not, it should be added to the list. Generation ends when the table of numbers has been obtained of the size of a set undergoing permutation. The returned table of numbers is mapped to relevant hashtags.

C. Hosting in Social Media

When selecting a way of multimedia hosting, attention should be paid to multimedia files processing on the side of social network services and the impact this processing has on non-breach of hidden data. This problem was researched for, among others, Facebook. In [10], the authors verified various approaches to eliminate the compression effect and algorithms sanitizing steganography on the side of Facebook servers. Research shows that Facebook algorithms are efficient in detecting the steganography known so far, in particular for hidden information of greater size. Therefore, it should be verified first whether the social networks selected

for the needs of practical implementation of SocialStegDisc ensure non-interference with the byte content of files.

Problems occurring with sanitization of steganography on the side of social networks or with its deformation may be circumvented by using an intermediary layer in the form of file upload services. An advantage of such approach is that such service has no impact on the file content, and thus – there is a certainty of non-breach of steganographic content. Then, in the social service using a hashtag mechanism only a link and a set of hashtags addressing this file are shared.

VI. SUMMARY

In this work, the authors' own concept of a new steganographic filesystem called SocialStegDisc was formulated. This system works in analogy to classic filesystems such as FAT or NTFS. A carrier of hidden information are multimedia files embedded in various open social networks. Every multimedia object is indexed with a unique permutation of a set of n hashtags. Between the objects there is a logical structure created by a set hashing function. On the basis of an input index (permutation of a set of hashtags), this function generates an index of the next object (permutation of a set of hashtags). Thus, subsequent blocks of hidden space may be read, recorded or modified in sequences, which is an analogy to servicing classic filesystems.

Researching such issues as the SocialStegDisc technique focuses on their usefulness in real scenarios, especially with respect to information leak. On one hand, it may seem that development of such concepts may serve only terrorists or other criminals in their attempts to hide communication supporting their actions. But above all, the purpose of research on this method was to demonstrate its application options and to determine on this basis the vulnerability of the systems used in it. Such vulnerability may be taken into account at the following iterations of the system development and be entirely eliminated.

REFERENCES

- [1] Anderson, R., Needham R., Shamir A. 1998. "The Steganographic File System". *Proceedings of the Second International Workshop on Information Hiding*, 73–82.
- [2] Szczypiorski, K. 2016. "StegHash: New Method for Information Hiding in Open Social Networks". *IJET International Journal of Electronics and Telecommunication*, 62 (4) : 347–352.
- [3] McDonald, A., Kuhn, M. 2000. "StegFS: A Steganographic File System for Linux". *Proceedings of the Third International Workshop on Information Hiding*, 463–477.
- [4] Venckauskas, A., Morkevicius, N., Petraitis, G., Ceponis, J. 2013. "Covert Channel for Cluster-based File Systems Using Multiple Cover Files". *ITC 42* (3) : 260–267.
- [5] Neuner, S., Voyiatzis, A., Schmiedecker, M., Brunthaler, S., Katzenbeisser, S., Weippl, E. "Time is on my side: Steganography in filesystem metadata". 2016. *Digital Investigation* 18 : 76–86.
- [6] Beato, F., De Cristofaro, E., Rasmussen, K. 2014. "Undetectable communication: The Online Social Networks case". *Twelfth Annual International Conference on Privacy, Security and Trust*, 19–26.
- [7] Castiglione, A., D'Alessio, B., De Santis, A. 2011. "Steganography and Secure Communication on Online Social Networks and Online Photo Sharing". *International Conference on Broadband and Wireless Computing, Communication and Applications*, 363–368.
- [8] Chapman, M., Davida, G., Rennhard, M. 2001. "A Practical and Effective Approach to Large-Scale Automated Linguistic Steganography". *Proceedings of Information Security: 4th International Conference*, 156–165.
- [9] Wilson, A., Blunson, P., Ker, A. 2014. "Linguistic steganography on Twitter: hierarchical language modeling with manual interaction". *Proceedings of Media Watermarking, Security, and Forensics*, 902803.
- [10] Hiney, J., Dakve, T., Szczypiorski, K., Gaj, K. 2015. „Using Facebook for Image Steganography". *Proceedings of the 2015 10th International Conference on Availability, Reliability and Security*, 442–447.
- [11] Szczypiorski, K., Janicki, A., Wendzel, S. 2015. „The Good, The Bad and The Ugly: Evaluation of Wi-Fi Steganography". *Journal of Communications (JCM)*, 10 (10) : 747–752.
- [12] Rivest, R. "Reference implementation code for pseudo-random sampler for election audits or other purposes". <https://people.csail.mit.edu/rivest/sampler.py> [access: 25.02.2017]
- [13] FIPS 180-4: Secure Hash Standard (SHS). <http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf> [access: 25.02.2017]

A.2. Artykuł [A2]

*Journal of Universal Computer Science, vol. 25, no. 2 (2019), 74-97
submitted: 14/8/18, accepted: 27/2/19, appeared: 28/2/19 © J.UCS*

Methods for Information Hiding in Open Social Networks

Jędrzej Bieniasz, Krzysztof Szczypiorski

(Warsaw University of Technology
Warsaw, Poland)

J.Bieniasz@tele.pw.edu.pl, ksz@tele.pw.edu.pl)

Abstract: This paper summarizes research on methods for information hiding in Open Social Networks. The first contribution is the idea of StegHash, which is based on the use of hashtags in various open social networks to connect multimedia files (such as images, movies, songs) with embedded hidden data. The proof of concept was implemented and tested using a few social media services. The experiments confirmed the initial idea. Next, SocialStegDisc was designed as an application of the StegHash method by combining it with the theory of filesystems. SocialStegDisc provides the basic set of operations for files, such as creation, reading or deletion, by implementing the mechanism of a linked list. It establishes a new kind of mass-storage characterized by unlimited data space, but limited address space where the limitation is the number of the hashtags' unique permutations. The operations of the original StegHash method were optimized by trade-offs between the memory requirements and computation time. Features and limitations were identified and discussed. The proposed system broadens research on a completely new area of threats in social networks.

Keywords: information hiding, open social networks, hashtag, StegHash, SocialStegDisc, filesystem

Categories: C.2.0, C.2.4, D.4.3, D.4.6, K.6.5

1 Introduction

Identifying new multimedia [Szczypiorski, 2016] [Fridrich, 2009] and network [Mazurczyk, 2016] steganography methods and their countermeasures are the main research contributions to steganography in the last few years. Less attention has been paid to text steganography [Chapman, 2001], and we have revisited this attractive subject in combination with social networks to discover probably a new area of threats in internet services. We utilized the popularity of using specific labels called hashtags across social networks and other internet services. With almost no limits for the construction of hashtags, due to the thousands of languages worldwide with dozens (or even hundreds) of alphabets, the infinite set of indexes could be explored. In our work we abstract from the linguistic level and forget the exact meaning of the hashtags as understood by humans. The method of StegHash is the main idea proposed by Szczypiorski [Szczypiorski, 2016a]. It is based on the use of hashtags on various social networks to create an invisible chain of multimedia objects, like images, movies or songs, with embedded hidden messages. These objects are indexed by permutations of preliminarily chosen hashtags. For every set of hashtags containing n elements there is the factorial of n permutations, which could be used as individual indexes of each message.

One of the original ideas of applying the StegHash technique was to establish an index system like in existing classic filesystems, such as FAT (File Allocation Table) or NTFS (New Technology File System). It would result in the creation of a new type of steganographic filesystem beyond previous efforts in this area, where the main ideas were to deny the filesystem operations or to deny the existence of the stored data. A new type of technique called SocialStegDisc [Bieniasz, 2017] is the proof of concept of the application of the StegHash [Szczypiorski, 2016a] method for new steganographic filesystem. The original environment of StegHash was modified to introduce the basic concepts of classic filesystems, such as Create-Read-Update-Delete operations or a defragmentation process. Furthermore, time-memory trade-offs were proposed in the design. The concept was tested to obtain operational results and proof of correctness. The results and the design were analyzed to discover as many features of the method as possible.

As it was mentioned, this paper summarizes results from [Szczypiorski, 2016a] and [Bieniasz, 2017]. The contributions beyond them included in this paper are:

- Section 2 – considerations about state-of-the-art techniques of preserving hidden data in multimedia files after processing by OSNs;
- Section 6 – full section with implementation of SocialStegDisc *proof-of-concept* system with testing;
- Section 7 – full section with discussion about the method;
- Section 8 – full section with malware and digital media forensics perspective on detection and analysis of such techniques like StegHash or SocialStegDisc;
- Section 9 – consideration of StegHash and SocialStegDisc application for cyberfog security approach;

This paper is structured as follows: Section 2 briefly presents the state of the art in social network steganography, including a background to text steganography. Section 3 contains a presentation of the idea of the StegHash method and a typical scenario for the preparation of the steganograms. Section 4 introduces the idea of SocialStegDisc as an application of StegHash for the steganographic filesystem. Section 5 presents the implementation of SocialStegDisc *proof-of-concept*. Section 6 reports experiments on SocialStegDisc. In Section 7 the results and evaluation of SocialStegDisc's operation are presented. Section 8 includes a discussion on the possibility of the detection of the proposed system. Section 9 finally concludes our efforts and suggests future work.

2 Related work

In this section, we present a review of the literature on the three main aspects that are relevant for a full comprehension of our work. The first aspect is applying OSNs to the operations of steganographic techniques. We investigated how OSNs could be used in hidden communication scenarios. Secondly, we focused on how OSNs could impact the preservation of hidden data embedded in multimedia files during their upload and storage. This knowledge is crucial to consider a particular OSN for use in the operation of StegHash. Finally, we look at steganography techniques used beyond

filesystems. This establishes the context to determine how the design of SocialStegDisc broadens the research in this domain.

2.1 Use of OSNs for steganographic techniques

In [Beato, 2014], Beato et al. presented two models of communication: high-entropy and low-entropy. The high-entropy model utilizes multimedia objects, such as images, video and music, etc., to embed hidden messages. This is recognized as classic multimedia steganography. This type of communication is characterized by high steganographic throughput, but the channel is easily detectable. The second model uses a null cipher approach, where the text data (e.g. status updates) carry secret information. The mechanism to decode the steganogram location and the hidden message need a pre-shared code. This communication features lower steganographic throughput, so it is proposed that this method is applied to covert signaling channels, while the main steganogram can be uploaded to another online service.

The efforts of Castiglione et al. [Castiglione, 2011] could be identified as expanding low-entropy steganographic methods. The first method utilizes filenames to carry hidden messages, so it could be used in OSNs that preserve the original filenames. The authors utilized the default naming schemes of popular digital camera producers, where a photo sequence number was chosen as the carrier of the hidden data. This method has a low steganographic throughput but detection is hard, so the scheme is generally safe. The second method takes advantage of inserting tags in images. The proposed covert communication channel requires the uploading of multiple images with the tagging of multiple users. Based on a predefined image and user sequence, a binary matrix can be determined and used to decode hidden messages. This method also has a relatively low steganographic throughput, but it is hard to detect.

Chapman et al. [Chapman, 2001] and Wilson et al. [Wilson, 2014] presented linguistic approaches to hide information in Twitter. Steganograms are carried by a bitmap determined by a permutation of language. The channel is considered to be very secure, although it requires a human processing of the tweets. It has a very low steganographic throughput.

All of the state-of-the-art methods operate with a single OSN, except the signaling channels designed in [Beato, 2014]. They utilize either a classic image steganography approach, which can be detected easily, or more sophisticated methods, for which the steganographic throughput is relatively low, but higher undetectability is introduced. It could be summarized as the classical trade-off aspect of steganography, where the ease of the method is traded off for a higher undetectability rate. The other disadvantage in the proposed methods is the fact that a steganogram sender is linked to the various user accounts that are required for the method. Such behavior could be recognized as suspicions by OSN providers.

2.2 Preserving hidden data in OSNs

To investigate how to preserve hidden data in multimedia files during uploading to OSNs we analyzed the results on image watermarking techniques [Potdar, 2005]. Image watermarking differs from steganography in terms of hiding the author's mark instead of the secret message. Furthermore, watermarks can be explicitly visible. The aim of this is to protect copyrights and to authenticate an originator [Naskar,

2014]. In [Chomphoosang, 2011], the researchers investigated watermarking techniques to be used in OSNs, resulting in attacks and possible solutions. In [Hiney, 2015], the authors verified various approaches to eliminate the compression effect and algorithms sanitizing steganography on the side of the Facebook servers. Research shows that Facebook algorithms are efficient for detecting steganography, in particular for hidden information of greater size. Applying wavelet decomposition for watermarking is presented in [Banos, 2015]. The authors showed the effectiveness of the method by simulating attacks on OSNs. The most far-reaching proposals included redesigning the uploading services of OSNs, like in [Zigomitros, 2012] where a dual watermarking scheme was constructed.

Image watermarking techniques are classified into two types [Potdar, 2005]:

1. Spatial domain watermarking,
2. Transform domain watermarking.

Spatial domain watermarking is about modifying particular pixels of images to embed data. The simplest method of this type is the Least Significant Bit [Bamatraf, 2010], where the bit with no impact on the image's look is modified. The simplest and most popular spatial domain methods are generally vulnerable for image processing and transformations [Mishra, 2014], so more complicated schemes are under consideration. One of the improved LSB methods is Intermediate Significant Bit (ISB) [Zeki, 2009], where the watermarked section is found by the range of each bit-plane. Another advanced method is Patchwork [Bender, 1996], where the watermark is applied by changing the brightness of the pairs of randomly selected pixels.

Transform domain watermarking is about exploiting the coefficients of the transformed image to embed a mark. To transform the domain, four main techniques are considered:

1. Discrete Cosine Transform (DCT),
2. Discrete Wavelet Transform (DWT),
3. Discrete Fourier Transform (DFT),
4. Singular Value Decomposition (SVD).

In DCT algorithms, the image is divided into blocks and the selected set of coefficients are modified to embedding a watermark. DCT is a reversible [Cox, 1997]. DWT decomposes the image into three spatial directions and the watermark is embedded in the wavelet coefficients [Kundur, 1998]. DFT decomposes the image for phase and magnitude, and a watermark is embedded into the magnitude part. Fourier Transformation is robust against geometric attacks [O'Ruanaidh, 1997]. In SVD, the singular matrices of the frequency domain and spatial domain offer space for embedding a watermark. The loss of information in such a technique is very low. SVD could be combined with other techniques to exploit the widest range of properties, for example: SVD-DFT [Zhang, 2006], SVD-DWT [Lai, 2010] or DFT-DWT-SVD [Ansari, 2012]. In general, transform domain watermarking provides high robustness [Olanrewaju, 2011].

2.3 Steganographic filesystems

A breakthrough in the development of steganographic filesystems was the design by Anderson, Needham and Shamir [Anderson, 1998]. The authors proposed a method similar to that of invisible ink. A user with a password is able to find and decode hidden information. The StegFS system presented in [McDonald, 2000] is a practical

implementation of the concept presented in [Anderson, 1998] as an extension of a standard filesystem, Linux Ext2fs.

In recent years, distributed filesystems used in clusters of various kinds have been gaining popularity. For such systems, proposals for applying steganography to hide information in them also appear, for instance in [Venckauskas, 2013]. The main idea behind the method presented in [McDonald, 2000] consists of using multiple files to encode hidden information by means of their relative positions in clusters.

Among recent proposals, the work presented in [Neuner, 2016] is worth highlighting, in which Neuner et al. presented a way of hiding information in the timestamps of files saved in classic filesystems. They discovered that information may be effectively hidden in the 24 bits (3B) coding the nanosecond part of the timestamp in the NTFS filesystem. For each file, it is possible to use two types of timestamps: of creation and of last access. In this way, one file creates 6B of space for hidden data.

3 Idea of StegHash

The method of StegHash [Szczypiński, 2016a] is based on the use of hashtags in the OSNs to connect multimedia files (like images, movies, songs) with embedded hidden messages. The initial set of hashtags is the base for constructing the indexes, which are unique pointers to these files. For a set of hashtags containing n elements there is the factorial of n permutations, and every single instance produces an individual and unique index for a given part of the data. The system consists of a secret initial set of hashtags (a password) and a secret transition generator. Using the transition generator starting from the initial set of hashtags, the link between indexes could be established and then hidden data could be explored as a chain from one to another (Fig. 1). The set of hashtags is logically independent from the OSN technology and could also be used on other Internet web services. The key issue is how to determine the placement of the next multimedia object with hidden data having generated a new index. A search engine dedicated to OSNs should be utilized, due to its capacity to search the hashtags as a primary way of marking messages in social media. The built in search option of the given OSN could also be used.

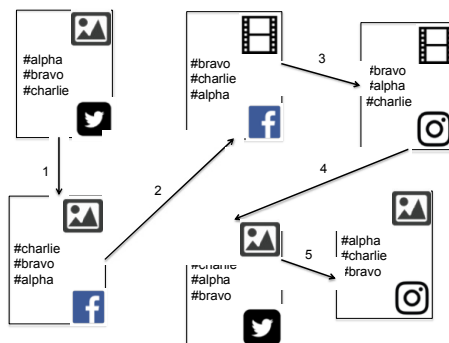


Figure 1: Example of StegHash method.

Let l be the length of an address in bits for creating the index for the group containing n hashtags:

$$l = \lceil \log_2 n! \rceil, \quad n > 1 \quad (1)$$

The length of the address and percent of wasted addresses as a function of n is shown in Figure 2. For $n \in \{5, 10, 12, 22, 28, 29\}$ the number of wasted addresses is below 20%.

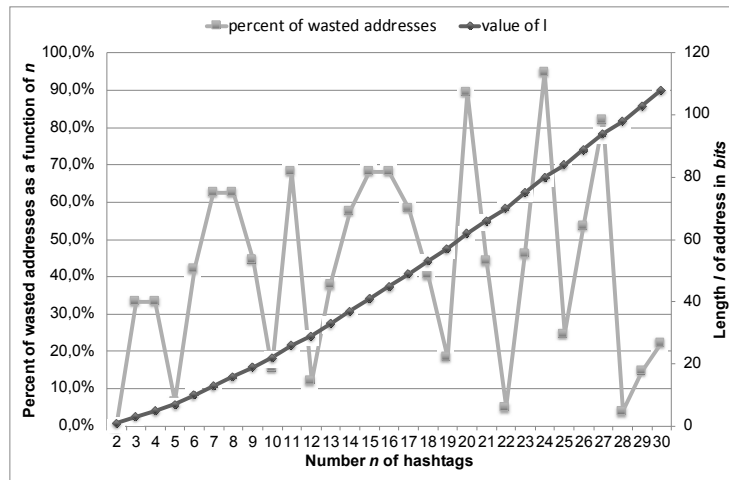


Figure 2: Length of address and percent of wasted addresses as a function of n .

0	1	2		#alpha	#bravo
1	2	1		#bravo	#alpha

0	0	0	1	2	3	#alpha	#bravo	#charlie
0	0	1	2	1	3			
0	1	0	1	3	2	#bravo	#alpha	#charlie
0	1	1	2	3	1	#alpha	#charlie	#bravo
1	0	0	3	1	2	#bravo	#charlie	#alpha
1	0	1	3	2	1	#charlie	#alpha	#bravo
1	0	1	3	2	1	#charlie	#bravo	#alpha
1	1	0	x	x	x			
1	1	1	x	x	x			

Figure 3: Examples for $n \in \{2, 3\}$.

For example, for two hashtags there are two bits for addressing with no wasted space and two permutations (Figure 3); for three hashtags there are three bits (two addresses wasted) and six permutations (Figure 3). To start to use Steghash, there are four main aspects to consider:

1. An algorithm to create a dictionary – dependent only on n .
2. A set of hashtags to create a dictionary – secret initial set.
3. The mapping of the indexes into a dictionary.
4. A secret transition generator to create the link between the addresses (a chain).

The choice of algorithm has no impact on the security if the secret transition generator (point 4) is pseudorandom. For point 2, the balance between the popularity of some hashtags and the number of possible search results should be found. Typically one or two unpopular hashtags are enough to have a unique index and to have reasonable search outputs. Choosing only popular hashtags means that each multimedia object needs reviewing to find the hidden content. A secret transition generator initiated with a secret password, as used in StegHash, produces indexes in a chain to go step by step. The first address is the start, and if we used all the space it would be similar to a circular linked list for the data structure. A secret transition generator needs to be a function based on a pseudorandom code generator or a hash function.

18 #delta	#alpha	#bravo	#charlie	Step	From	To	Addr
20 #delta	#alpha	#charlie	#bravo	1	Facebook	→ Instagram	18
19 #delta	#bravo	#alpha	#charlie	2	Instagram	→ Twitter	20
21 #delta	#bravo	#charlie	#alpha	3	Twitter	→ Instagram	19
22 #delta	#charlie	#alpha	#bravo	4	Instagram	→ Google Plus	21
23 #delta	#charlie	#bravo	#alpha	5	Google Plus	→ Twitter	22
6 #alpha	#delta	#bravo	#charlie	6	Twitter	→ Google Plus	23
10 #alpha	#delta	#charlie	#bravo	7	Google Plus	→ Instagram	6
7 #alpha	#bravo	#delta	#charlie	8	Instagram	→ Twitter	10
0 #alpha	#bravo	#charlie	#delta	9	Twitter	→ Instagram	7
11 #alpha	#charlie	#delta	#bravo	10	Instagram	→ Facebook	0
2 #alpha	#charlie	#bravo	#delta	11	Facebook	→ Twitter	11
8 #bravo	#delta	#alpha	#charlie	12	Twitter	→ Facebook	2
12 #bravo	#delta	#charlie	#alpha	13	Facebook	→ Instagram	8
9 #bravo	#alpha	#delta	#charlie	14	Instagram	→ Google Plus	12
1 #bravo	#alpha	#charlie	#delta	15	Google Plus	→ Instagram	9
13 #bravo	#charlie	#delta	#alpha	16	Instagram	→ Facebook	1
3 #bravo	#charlie	#alpha	#delta	17	Facebook	→ Google Plus	13
14 #charlie	#delta	#alpha	#bravo	18	Google Plus	→ Facebook	3
16 #charlie	#delta	#bravo	#alpha	19	Facebook	→ Twitter	14
15 #charlie	#alpha	#delta	#bravo	20	Twitter	→ Google Plus	16
4 #charlie	#alpha	#bravo	#delta	21	Google Plus	→ Twitter	15
17 #charlie	#bravo	#delta	#alpha	22	Twitter	→ Facebook	4
5 #charlie	#bravo	#alpha	#delta	23	Facebook	→ Google Plus	17
				24	Google Plus	→ Facebook	5
for	#delta	go to	Facebook				
	#alpha		Google Plus				
	#bravo		Twitter				
	#charlie		Instagram				

Figure 4: Example for $n = 4$ with addressing, pointers to social media networks and transition graph.

As stated previously, a search engine designed for the OSNs or the interior search mechanism of the given OSN should be used to find the next element in the chain. There are no effective search engines for some OSNs. To increase the performance of the system, one hashtag or more are utilized as the pointer to the next OSN. This may introduce a vulnerability into the StegHash method, because the prediction of this type of subaddressing could be linked with a given OSN. Figure 4 contains an example with four hashtags. The addressing scheme was generated and then a SHA-512 [NIST, 2015] based function was used to produce a chain. The last hashtag in the index represents the placement of the next message (for X go to Y). Figure 4 also shows a transition graph, which explains how a chain among the messages is built.

4 Idea of SocialStegDisc

There is a basic limitation to applying StegHash [Szczypiorski, 2016a] in SocialStegDisc [Bieniasz, 2017]: memory occupancy. Along with the increase in the number of n hashtags, the volume of the dictionary increases proportionately to $n!$. Therefore, keeping the entire dictionary in the memory places a heavy burden on the system. Generating the dictionary and restoring the set of used addresses will consume a large amount of time. The main goal behind keeping these dictionaries is ensuring uniqueness and reliability of addressing. Therefore, for a higher n it is proposed to introduce a dynamic verification of uniqueness of the generated addresses. Two space-time tradeoff-type modifications will be formulated on its basis along with an application of the linked list mechanism.

4.1 Dynamic verification of uniqueness of addresses

The first attempt to increase the efficiency of the StegHash method is to eliminate the storing of the results from the generation of the utilized address space when creating a chain and later restoring it. Such a set was stored because it was necessary to verify the uniqueness of the generated addresses. In this proposal, the uniqueness is verified by determining whether in the OSNs there is a file with a given address.

4.2 Space-time tradeoff modifications and linked list mechanism

From the perspective of a correct system operation, the system of addressing, which is indexing SocialStegDisc segments with hashtags, is of key importance. A characteristic feature of such addressing is non-linearity. Subsequent addresses are generated by means of a specific function on the basis of the initial address. The issue of addressing in SocialStegDisc is an analogy to the problem of memory space fragmentation in classic filesystems. In StegHash it was assumed that deleting files from such system is not a case, as the address space can be enormous with almost unlimited storage around the Internet. In case of SocialStegDisc the main consideration of design was to offer a full-fledged filesystem. *Delete* procedure is a required basic filesystem operation among *Read*, *Write* and *Update*.

Until now, a strong assumption was a prohibition of hiding the addresses for the system due to security reasons. This implies that it is not possible to apply mechanisms servicing memory fragmentation, for instance allocation tables or storing the index to another memory block together with the data. As a consequence, it is not possible to make available a function for deleting single files due to the resulting gap in the addressing, which if it is not known that the gap exists, would mean that the files located after the gap would be lost. Under this approach, once a file is added to SocialStegDisc, it will remain there. In this article, a solution is proposed that would allow the introduction of structures for the efficient management of resources (addresses), known from classic filesystems, to SocialStegDisc. To this end, the assumption of non-storing the addresses will be weakened by introducing such a mechanism that will simultaneously not decrease the security level.

Direct addresses to objects may be presented by means of an address to the object of reference (beginning of the SocialStegDisc instance) and a counter, whose value will serve to control the obtaining of the address of the subsequent object. In every

multimedia object, a space for this counter is added to a hidden space. Due to this, the SocialStegDisc system assumes the structure of a linked list, which allows to supplement the system with file deletion. An example of how this operates is presented in Figure 5.

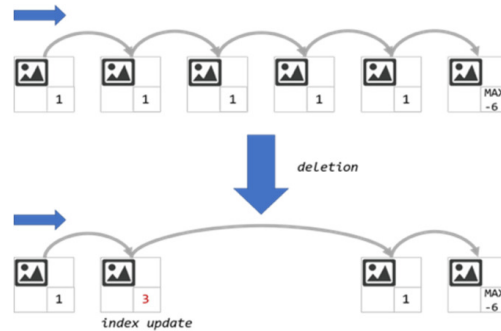


Figure 5: Mechanism of a linked list in SocialStegDisc. It allows to realize delete procedure.

Apart from hiding m bytes of proper data, it is necessary to place a minimum of p bits coding the numerical index in every multimedia object. The selection of the volume p determines the maximum length of the chain, for which the problem of information loss does not occur when the mechanism of file deletion is applied, and amounts to:

$$2^p - 1 \# (2)$$

The hidden counter will be used in the proposed modifications of the space-time tradeoff type, by using it to control the functions establishing the next address. This counter may store:

- The address code for the application of the StegHash method without the dictionary of the used addresses.
- The counter value that specifies how many times the procedure of generation of a subsequent address should be performed to actually obtain it – in the case of the elimination of both original dictionaries of the StegHash method.

When StegHash with the modification eliminating the dictionary of used addresses is applied, and when a chain of objects is created, a code for the address to the next object is placed in the additional space, in line with the coding from the dictionary of permutations. The scheme of this procedure is presented in Figure 6.

During reading, the address code is retrieved from the hidden space and then converted into an address in the form of a set of hashtags. Thus, access to the next object is possible. To maintain the continuity of the address space, when deleting a file in the multimedia object preceding the beginning of the given file, the counter should be updated to the value retrieved from the last object storing the file to be deleted. The set of the used addresses should also be updated. The released addresses are not lost but they go back to the pool and may be generated in subsequent iterations of the pseudo-random generation function. The schemes for the read method is presented in Figure 7.

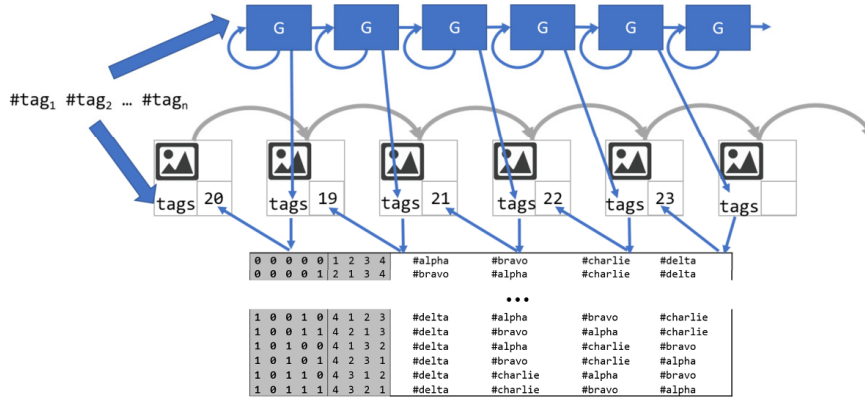


Figure 6: Creation of a chain by means of combined StegHash method and linked list mechanism.

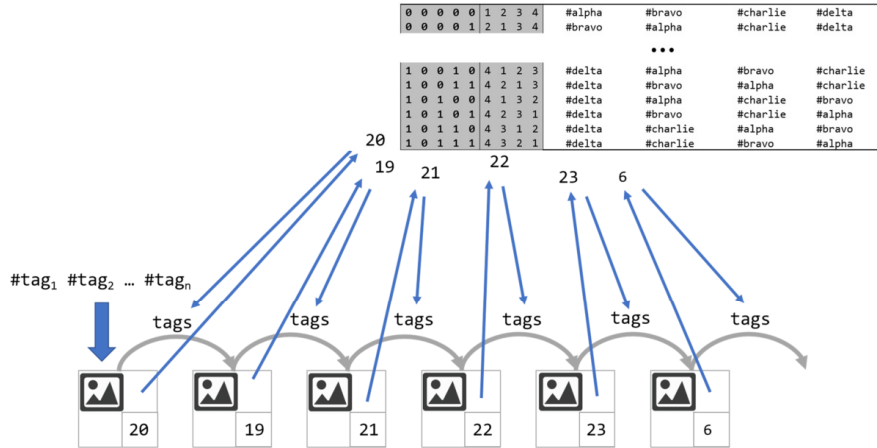


Figure 7: Read of a chain by means of combined StegHash method and linked list mechanism.

Another proposed modification of the space-time tradeoff type is a total elimination of the original StegHash dictionaries and the use of the linked list mechanism to store relevant counters allowing the restoration of the chain. To this end, first of all, the algorithm of dynamic generation of addresses should be modified so that the function, apart from the new address, returns the current status of the sampling counter in the course of the generation of the subsequent addresses. In Figure 8, a scheme of a chain creation is presented. With the starting point known, the obtained counter allows the retrieval of the address returned in the course of the generation at the counter status. As these states of the counter are recorded in the hidden space defined for the linked list mechanism, they allow the chain objects to be traverse from a known initial

address of the SocialStegDisc instance. In Figure 9, a scheme of a chain retrieval is shown.

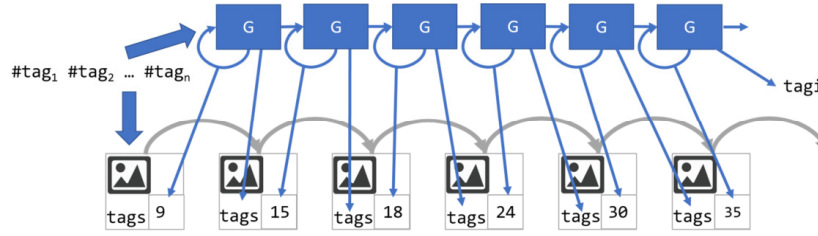


Figure 8: Chain creation with no dictionaries and a linked list mechanism.

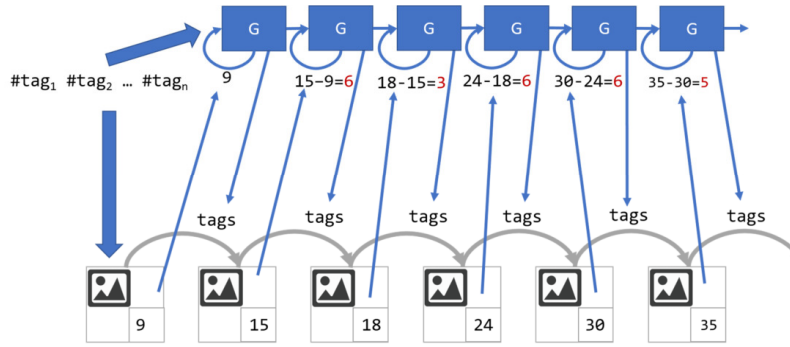


Figure 9: Chain retrieval with no dictionaries and a linked list mechanism.

In both cases, a file deletion that entails the creation of free address space, in this case updates the index in the object preceding the space to the value of the index from the last deleted file.

5 Implementation of SocialStegDisc

5.1 Environment

The system implementation proposal presented in this article is a proof-of-concept of the SocialStegDisc system. The main client application is the SocialStegDisc Shell terminal, which translates the client queries to appropriate operations of the file directory and operations on files, using the Controller module. In addition, two modules from StegHash were used:

- StegHash Engine – a module responsible for the creation of appropriate addressing of multimedia files with the use of hashtags.
- Database – contains a set of multimedia files used as a carrier of hidden data.

5.2 Generation of addresses

The basic operation of the system is the generation of addresses. This index plays two roles:

1. Unambiguously identifies the object.
2. Identifies the social network in which it is embedded.

In StegHash [Szczypiorski, 2016a], addresses were generated by means of any type of permutation generation algorithm, and the uniqueness was ensured by rejecting the already drawn addresses. For SocialStegDisc, the generation of addresses in a chain must be characterized by restorability with the use of appropriate initial conditions. To this end, the approach of Rivest [Neuner, 2016] was used, who presented a reference code for a program generating a random sample of s size from set S . Assuming that $s = S$, the sample is a searched permutation of set S . In Figure 10, an exemplary code in Python of a chain generation method with no dictionaries stored, is presented.

```
def go_to_perm(seed, counter):
    input = ""
    for num in seed:
        input += str(num)
    i = 0
    output_list = []
    while(i < counter):
        if(len(output_list) == len(seed)):
            for num in output_list:
                input = ""
                input = str(num)
                output_list = []
            i = i + 1
            hash_input = input + str(i)
            hash = hashlib.sha256(hash_input.encode('utf-8'))
            output = int(hash.hexdigest(), 16)
            pick = int(output % len(seed))
            if not pick in output_list:
                output_list.append(pick)
    return output_list
```

Figure 10: Method of address generation based on reference address.

The main part of the method is used for the hash function for the pseudorandom generation of the next value that is a candidate to become an element of an output permutation. In the example of the code, the function SHA-256 was used [NIST, 2015], but it is possible to apply other hash functions, for instance SHA-512 [NIST, 2015]. Using a hash function with a greater output size reduces the number of sampling iterations when completing the output table. The output value from the hash function is converted to a hexadecimal number, based on which a decimal value is generated, which is a pseudorandom number generated in a given loop iteration. This number undergoes the operation of obtaining the remainder from division by the set size. Finally, it is verified whether the sampled value already exists in the output set

from a given iteration of the permutation generation. If not, it should be added to the list. Generation ends when a table of numbers has been obtained for the size of the set undergoing permutation. The returned table of numbers is mapped to relevant hashtags.

5.3 Hosting in social media

When selecting a way of multimedia hosting, attention should be paid to multimedia files processing on the side of the social network services and the impact this processing has on non-breaching of the hidden data. This problem was investigated in Section 2.2. Therefore, it should be verified first whether the social networks selected for the needs of practical implementation of SocialStegDisc to ensure non-interference with the byte content of the files. Problems occurring with the sanitization of steganography on the side of social networks or with its deformation may be circumvented using an intermediary layer in the form of the file upload services. An advantage of such an approach is that such a service has no impact on the file content, and thus – there is a certainty of non-breach of the steganographic content. Then, in the social service using a hashtag mechanism only a link and a set of hashtags addressing this file are shared.

6 Practical evaluation of SocialStegDisc

6.1 Size of disk and files

The structure of the classical filesystem is an array of blocks, where a single block is the smallest unit of the disk operations. For SocialStegDisc, this unit is the amount of hidden data in a single multimedia file (images, movies, audiofiles). For example, StegHash uses 10 bytes per multimedia files.

In Table 1 the calculation of the required space for the index in bytes is presented as the function of the number of hashtags. This space is calculated by the formula:

$$p_B = \left\lceil \frac{\log_2 n!}{8} \right\rceil, \quad n > 1 \#(2)$$

p	$[b]$	p_B	$[B]$	max. value	p	$[b]$	p_B	$[B]$	max. value	p	$[b]$	p_B	$[B]$	max. value
2	1		3		31	4		2147483647		60	8		1,15292E+18	
3	1		7		32	4		4294967295		61	8		2,30584E+18	
4	1		15		33	5		8589934591		62	8		4,61169E+18	
5	1		31		34	5		17179869183		63	8		9,22337E+18	
6	1		63		35	5		34359738367		64	8		1,84467E+19	
7	1		127		36	5		68719476735		65	9		3,68935E+19	
8	1		255		37	5		1,37439E+11		66	9		7,3787E+19	
9	2		511		38	5		2,74878E+11		67	9		1,47574E+20	
10	2		1023		39	5		5,49756E+11		68	9		2,95148E+20	
11	2		2047		40	5		1,09951E+12		69	9		5,90296E+20	
12	2		4095		41	6		2,19902E+12		70	9		1,18059E+21	
13	2		8191		42	6		4,39805E+12		71	9		2,36118E+21	
14	2		16383		43	6		8,79609E+12		72	9		4,72237E+21	
15	2		32767		44	6		1,75922E+13		73	10		9,44473E+21	
16	2		65535		45	6		3,51844E+13		74	10		1,88895E+22	
17	3		131071		46	6		7,03687E+13		75	10		3,77789E+22	
18	3		262143		47	6		1,40737E+14		76	10		7,55579E+22	
19	3		524287		48	6		2,81475E+14		77	10		1,51116E+23	
20	3		1048575		49	7		5,6295E+14		78	10		3,02231E+23	
21	3		2097151		50	7		1,1259E+15		79	10		6,04463E+23	
22	3		4194303		51	7		2,2518E+15		80	10		1,20893E+24	
23	3		8388607		52	7		4,5036E+15		81	11		2,41785E+24	
24	3		16777215		53	7		9,0072E+15		82	11		4,8357E+24	
25	4		33554431		54	7		1,80144E+16		83	11		9,67141E+24	
26	4		67108863		55	7		3,60288E+16		84	11		1,93428E+25	
27	4		134217727		56	7		7,20576E+16		85	11		3,86856E+25	
28	4		268435455		57	8		1,44115E+17		86	11		7,73713E+25	
29	4		536870911		58	8		2,8823E+17		87	11		1,54743E+26	
30	4		1073741823		59	8		5,76461E+17		88	11		3,09485E+26	

Table 1: Maximal value of index taken as p bits and p_B bytes needed to encode them.

n	$n!$	p_B [B]
2	2	1
3	6	1
4	24	1
5	120	1
6	720	2
7	5040	2
8	40320	2
9	362880	3
10	3628800	3

Table 2: Number of possible addresses for n hashtags and needed bytes to encode their binary representation.

The maximum size of SocialStegDisc is calculated from:

$$S_D = n! \cdot (m + p_B) \#(3)$$

Table 3 summarizes how the size for files and total size (data and indexes) are related to the number of hashtags by assuming $m = 5, 10, 15$ as the number of bytes available for users' files per single multimedia file. These values are determined by formulas 4 and 5 analogically:

$$data [MB] = \frac{n! \cdot m}{1000} \#(4)$$

$$total disk size [MB] = \frac{n! \cdot (m + p_B)}{1000} \#(5)$$

<i>n</i>	<i>n!</i>	<i>p_s</i> [B]	<i>m</i> = 5		<i>m</i> = 10		<i>m</i> = 15	
			<i>dane</i> [MB]	<i>dane & Licznik</i> [MB]	<i>dane</i> [MB]	<i>dane & Licznik</i> [MB]	<i>dane</i> [MB]	<i>dane & Licznik</i> [MB]
2	2	1	0,01	0,012	0,02	0,022	0,03	0,032
3	6	1	0,03	0,036	0,06	0,066	0,09	0,096
4	24	1	0,12	0,144	0,24	0,264	0,36	0,384
5	120	1	0,6	0,72	1,2	1,32	1,8	1,92
6	720	2	3,6	5,04	7,2	8,64	10,8	12,24
7	5040	2	25,2	35,28	50,4	60,48	75,6	85,68
8	40320	2	201,6	282,24	403,2	483,84	604,8	685,44
9	362880	3	1814,4	2903,04	3628,8	4717,44	5443,2	6531,84
10	3628800	3	18144	29030,4	36288	47174,4	54432	65318,4

Table 3: Relation between size for files, total size (data and indexes) and number of hashtags.

6.2 Secrets

In StegHash, a user uses three secrets:

1. Initial set of hashtags for generation of dictionary.
2. Pointer for the first multimedia file with hidden data which is an initial set for the transition dictionary.
3. Generation function.

SocialStegDisc merges secrets 1. and 2. into one. The user generates pointers for the next files in the chain from the initial set and using the index value hidden in the file with the data together. Furthermore, as an empirical result, it is proposed to have knowledge of the last used address (set of hashtags) in the system as another secret. This does not violate the level of security of the system, but broadens the operations of SocialStegDisc with two functionalities:

1. Knowledge of iteration limit of the system.
2. From access to the last used object in the system, the size of the space left for the user is known.

6.3 Testing social networks' operations

For this work, two social networks with hashtag mechanisms were used: Twitter and Flickr. For every social network two tests were conducted:

1. With original images with well-established steganography methods.
2. With images uploaded to social networks, downloaded from them and next used with well-established steganography methods.

The second case was determined by the assumption that it could help avoid the distortion of steganography as the image already processed by the social network's algorithms could be classified as benign.

The tests for Twitter were negative for both cases, whereas for Flickr all tests passed. It is proposed to avoid such a distortion by treating social networks as a layer for sharing URLs to a multimedia file stored on the service that do not violate the original structure. For testing this alternative design, Twitter was used to share Dropbox URLs to files. Tests were conducted successfully.

This approach forms the basis for the consideration of creating a generic design of such systems:

1. Pointers to objects with a hidden space: a secret coding of keys. In a generic design it could be any set of alphanumeric strings.

2. Storage for files: any Internet service where stored files are searchable by their description consists of pointers introduced at Point 1.
3. Transition generation function: as introduced for StegHash [Szczypiorski, 2016a], a secret transition generator based on a pseudorandom code generator or a hash function.

6.4 Experiments

We tested the system with different sizes of initial data to send: 50B, 100B, 1000B, 5000B, 10000B. The size of the space for the user's data was 10B, so 5, 10, 100, 500 and 1000 images were needed accordingly. One image had size of 9kB. For this setting we used minimal number of hashtags for each case: 3, 4, 5, 6 and 7 accordingly. The system was tested for two novel configurations introduced in Section 4. Experiments confirmed the initial results. For a higher demand for the number of permutations of hashtags the reliability decreases from 100% to 70%, but the mechanism to verify the upload could be introduced to trigger the retransmission. What is more noticeable is the time of operation. The transition generation function based on a hash function with the design from Figure 10 takes much more time to generate the next unique permutation of the hashtags as the number of them available decreases. Future research could place effort on developing transition functions to appease this aspect by needing less tries to find a new unique permutation of the hashtags. Obviously, the lower number of hashtags for the same number of objects to upload, the more it is observed in this kind of hang of the system.

Number of images	Size of hidden data	Minimum number of hashtags	Time of simple uploading pictures [s]	Time of generating all permutations [s]	Number of needed iterations to generate all permutations	Time of generating needed permutation [s]	Number of needed iterations to generate needed permutations
5	50	3	17.5	0.0005	133	0.0002	53
10	100	4	35	0.001	459	0.0004	96
100	1000	5	350	0.04	7334	0.015	2512
500	5000	6	1750	0.53	65962	0.077	12453
1000	10000	7	3500	18.39	888693	0.22	19887

Table 4: Performance of SocialStegDisc

Table 4 presents results of performance metrics from executing SocialStegDisc as the whole system and their main modules. This strategy of testing resulted from main assumptions:

- Time of uploading or checking existence of one picture is mainly a constant value, at 3.5 s; Reading of a single picture lasts 2.5 s;

- Time of generating permutations is non-linear function of needed number them, with strong rise with larger arguments;
- Extracting hidden data from a single steganographic carrier lasts 1s;

		Create [s]		Read [s]	
Number of images	Minimum number of hashtags	SocialSteg Disc ver. 1	SocialSteg Disc ver 2.	SocialSteg Disc ver. 1	SocialSteg Disc ver 2.
5	3	17.5	35	12,5	25
10	4	35	70	25	50
100	5	350	700	250	500
500	6	1750	3500	1250	2500
1000	7	3500	7000	2500	5000

Table 5: Performance of Create and Read operation for both versions of SocialStegDisc

Table 5 includes performance results of *Create* and *Read* operation for both versions of SocialStegDisc. For presented test cases, the system works predictably and with the accepted time. It should be considered that this time might be altered, for example if the method needs to be less detectable or to cover in the background of normal traffic by pretending that operations of SocialStegDisc are not statistically different from it.

7 Discussion

7.1 Addressing scheme and hidden space

The issue of addressing in SocialStegDisc is an analogy to the problem of memory space fragmentation in classic filesystems, which has been highlighted when developing the concept. The SocialStegDisc concept introduces a mechanism of storing indexes to the next object in the system, due to which the drive assumes the structure of a linked list. Thanks to the application of a linked list mechanism, it is possible to efficiently use the space and propose a file deletion mechanism. The greatest problem is the calculation overhead and algorithms to verify the correctness of the operation scheme.

The system is characterized by non-linearity in filling in the byte space. This effect occurs when the last used object hides a smaller number of bytes than possible. On the basis of the number of bytes, it is possible to determine the length of the chain of multimedia files created in line with the StegHash technique essence. This size amounts to:

$$L = \left\lceil \frac{M}{m} \right\rceil \#(6)$$

where M is the number of bytes on which the operation is the performer, and m is the size of a single logical block. When $M \bmod m \neq 0$ in the last multimedia file in the chain, fewer bytes are placed than the size of space allocated in the multimedia file (m).

7.2 Undetectability

The SocialStegDisc undetectability level depends on:

1. The level of undetectability in the open social networks of the applied methods of the multimedia steganography, serving to hide data.
2. The level of undetectability of the StegHash steganographic method, which is a system of indexing multimedia objects and building the SocialStegDisc drive space from single blocks.
3. The level of security of social networks with respect to the detection of anomalies in the network users' behavior. The SocialStegDisc system may communicate with the same service multiple times at short intervals. From the perspective of such a service, frequent and automated sending of queries will be treated as a threat, for instance a Denial-of-Service type attack.
4. The level of detectability of multimedia steganography. This problem was under research for Facebook, among others. In [Ning, 2014], the authors tested algorithms sanitizing steganography on the side of the Facebook servers. The research shows that Facebook algorithms are efficient for the detection of steganography known so far, in particular for hidden information of greater size.

In [Szczypiorski, 2015], a way of evaluation of network steganography techniques was proposed in three categories of undetectability: good, bad and ugly. The presented classification manner may be applied to other steganographic methods. StegHash may be considered a good steganographic method, which transitively means that the proposed SocialStegDisc technique is also characterized by this feature. The observer is unable to detect the ongoing communication anywhere in the network, even at the receiver of the hidden data.

7.3 Reliability

The reliability of the SocialStegDisc system should be assessed by the reliability of individual system operations, these operations are divided into dependent on steganography and independent of it.

Reliability due to multimedia steganography includes:

1. Degree of avoiding steganography cleaning algorithms on social networking sites. This problem was addressed in the assessment of undetectability in the Section 7.2 and it is closely related to the level of system reliability.
2. The influence of image processing algorithms, e.g. compression, on the bit content of a multimedia object. This issue was addressed in the study [Hiney, 2015] where a method was found to overcome this problem. In practical implementation, SocialStegDisc must be included.

It may be necessary to introduce a mechanism to confirm the placement of data in all upload operations. It means download of the loaded object and bit comparison with object before uploading. If the objects are identical, uploading a segment of the

hidden data carried by this object is considered successful. When loading multiple objects, this mechanism becomes a large overhead in the operation of the entire system.

All interaction with websites is done using the TCP / IP stack protocols – IP, TCP and HTTP. The use of TCP in transport layer means the use of mechanisms of windows, confirmations, sequential numbers and retransmissions. The transport speed is adjusted to the conditions in the network, data can be read in the order of transmission and after detection of data loss, data transmission is repeated. Thanks to this, the transport of SocialStegDisc requests has a high level of reliability. On the other hand, if the execution of the request fails, the attempts can be made repeatedly until the positive result.

As the second key element of the system's operation, independent of steganography, the operation of the social network service server should be indicated. The level of reliability, in particular of the largest websites, is very high. These services are scalable and well-protected, for example, by resource overload attacks. However, the user SocialStegDisc may encounter, when interaction with the server is blocked due to:

1. Classifying the interaction SocialStegDisc with the service as a kind of dangerous anomaly, e.g. symptom of the attack Denial-of-Service;
2. Applying the limitation mechanism of access tokens - when exhausted, requests are discarded until the pool is renewed.
3. Changing the interface, blocking searching methods and other modifications introduced by operators of OSNs;

These aspects could limit the use of the method, especially regarding the duration of such communication or completely stop it. Regarding to [Fridrich, 1999], every steganography method is designed basing on *the magic triangle* which means trade-off between steganographic bandwidth, undetectability level and robustness of the method. These features of SocialStegDisc can be flexibly adjusted to the goals to be achieved. Obviously, real cyber threat actors could tolerate long time of working (low steganographic bandwidth) whether it could imply reliable cover for their actions. Higher steganographic bandwidth would mean that their network traffic could be easier distinguished as abnormal activity and thus easily blocked. In that case, we do not much care if OSN could alter their operations. We would like the OSN operators to pay attention to the way their systems operate and this paper may constitute a collection of knowledge for them regarding new vectors of abuse of their systems.

8 Impact on digital forensics methods

The StegHash [Szczypiorski, 2016a] and SocialStegDisc [Bieniasz, 2017] methods have been established as a proof of concept for use during digital crimes, such as information leaks or communication between bad actors mainly observed as benign activities of users of social networks and other Internet services. It can be broadened to research on misusing public Internet services by a combination of steganographic methods to hide data and operations in the logic of using such services. It goes far beyond classical steganalysis methods to find the existence of that activity, for which methods and tools of digital forensics are needed. Unfortunately, they are currently not sufficient and this generates a need for a new approach to deal with such

activities. First of all, finding the way to collect pieces of evidence is a must. Initially, it could be like looking for a needle in a haystack, because there are three main questions to answer:

- What to observe?
- How to observe?
- How to establish verifiable and formal evidence of a digital crime from observations and methods for observing?

All of them are related together, so an approach to one aspect impacts another and vice versa. The natural answer for establishing a class of problem is the big data approach for behavioral analysis. Collecting users' activity, for example:

- Upload and download multimedia files from open social networks;
- Existence of correlation between using an open social network and the next upload or download of a multimedia file from an Internet service;
- Timing of activity;
- Distribution of activity;

appears to be promising as a source for big data analysis to find indicators of the use of StegHash and SocialStegDisc. The next aspect is to choose the observation methods and architectures. We see a need for a comprehensive survey on such methods, architectures and designs, which is what we plan to do in the future. The main categories of algorithms to investigate at first can be identified easily: community detection in graphs, distinguishing automatic from human activity and finding patterns of behavior in large datasets. These algorithms could be used for data collected by observing the central network with aggregated data or in more distributed designs where the particular actors could be detected more accurately. The interesting idea is to combine the proposed algorithms with the design of a moving observer [Szczypiorski, 2015]. The combination could show the strength in increasing confidence and efficiency of detection of the bad actor by analyzing datasets generated at the following points and getting closer to the source of malicious activity. Another consideration is to find methods and algorithms that can be introduced on the open social network side, where simple quotas or blocking anomalous activity are not enough nowadays. The contribution of the authors in this paper is a foundation for their further research on defining a set of indicators for using methods such as StegHash or SocialStegDisc, finding algorithms and the best architectures for the execution of them to establish verifiable and formal evidence of digital crime in an optimal way. Furthermore, the authors see an opportunity to invent some methods that can be implemented in the restricted view of an open social network operator.

9 Conclusions and future work

In the initial experiment we proved that the concept of the StegHash method, as a new approach for combining text steganography with other carriers, like pictures, movies and songs, was correct. Every multimedia object is indexed with a unique permutation of a set of n hashtags. Between the objects there is a logical structure created by a set hashing function. On the basis of an input index (permutation of a set of hashtags), this function generates an index of the next object (permutation of a set of hashtags).

Please note that for n hashtags, m byte messages and 100% accuracy, we have the receiving capacity of $n! \cdot m$ bytes for storage, i.e., for $n = 12$ and $m = 10$ bytes, this would be 4.46 TBytes. Furthermore, a new steganographic filesystem called SocialStegDisc was formulated. Thus, subsequent blocks of hidden space may be read, recorded or modified in sequence, which is an analogy to servicing classic filesystems. The authors proposed the idea of SocialStegDisc as an improvement on the initial scheme of StegHash via a trade-off between memory requirements and computation time. During tests it was validated that SocialStegDisc works analogically to StegHash, but the effort of operations is more widely distributed. Furthermore, SocialStegDisc introduces a classical set of CRUD (Create, Read, Update, Delete) operations on files, which was not possible in the original StegHash method.

We recognized that StegHash and SocialStegDisc could support the development of a “cyberfog” security approach, where solutions to eliminate centralizing crucial, sensitive and critical information are constantly sought. Following [Kott, 2016], a “cyberfog” security approach assumes splitting the data into numerous fragments and dispersing them across multiple end-user devices. Data dispersion presents adversaries with uncertainty as to where to find relevant information and how to reconstruct it from captured parts. We see the possibility for the application of the StegHash indexing scheme and SocialStegDisc filesystems operations directly into such a “cyberfog”. We addressed the construction of such a system in [Bieniasz, 2018]. The design seems to offer a real and functional application. Moreover, we are planning to use big data analytics to find the context in systems that are similar to StegHash and to introduce new methods for the detection of such systems. Finally, in the future we will also pay attention to the theoretical aspect of building relations among hashtags by analyzing other mathematical functions and algorithms that output permutations.

Research into such issues as the SocialStegDisc technique focuses on their usefulness in real scenarios, especially with respect to information leak. It seems that the development of such concepts may serve only terrorists or other criminals in their attempts to hide communication supporting their actions. However, the purpose of the research into this method was to demonstrate its application options and to determine on this basis the vulnerability of the systems used in it. Such vulnerability may be taken into account during the following iterations of the system development and be entirely eliminated.

Acknowledgement

This work has been supported by the National Centre for Research and Development (agreement No. CYBERSECIDENT/369234/I/NCBR/2017) under the CyberSecIdent Programme.

References

[Anderson, 1998] Anderson, R., Needham, R., Shamir, A. 1998. “The Steganographic File System”. Proceedings of the Second International Workshop on Information Hiding, 73–82.

- [Ansari, 2012] Ansari, R., Devanalamath, M. M., Manikantan, K. and Ramachandran, S. "Robust digital image watermarking algorithm in DWT-DFT-SVD domain for color images," in *Communication, Information & Computing Technology (ICCICT)*, 2012 International Conference on. IEEE, 2012, pp. 1–6.
- [Bamatraf, 2010] Bamatraf, A., Ibrahim, R., and Salleh, M. "Digital watermarking algorithm using lsb," in *Computer Applications and Industrial Electronics (ICCAIE)*, 2010 International Conference, 2010, pp. 155–159.
- [Banos, 2015] Banos, O., Lee, S., Yoon, Y., Le-Tien, T. et al., "A novel watermarking scheme for image authentication in social networks," in *Proceedings of the 9th International Conference on Ubiquitous Information Management and Communication*. ACM, 2015, p. 48.
- [Beato, 2014] Beato, F., De Cristofaro, E., and Rasmussen, K. B. "Undetectable communication: The Online Social Networks case," *Privacy, Security and Trust (PST)*, 2014 Twelfth Annual International Conference on, Toronto, ON, 2014, pp. 19-26.
- [Bender, 1996] Bender, W., Gruhl, D., Morimoto, N., and Lu, A. "Techniques for data hiding," *IBM systems journal*, vol. 35, no. 3.4, pp. 313–336, 1996.
- [Bieniasz, 2017] Bieniasz, J., Szczypiorski, K.: "SocialStegDisc: Application of steganography in social networks to create a file system", In *Proc. of 3rd International Conference on Frontiers of Signal Processing (ICFSP 2017)*, Paris, France, 6-8 September 2017
- [Bieniasz, 2018] Bieniasz, J., Szczypiorski, K.: „Towards Empowering Cyber Attack Resiliency Using Steganography”, In *Proc. of 4th International Conference Frontiers of Signal Processing (ICFSP 2018)*, Poitiers, France, 24-26 September 2018
- [Castiglione, 2011] Castiglione, A., D'Alessio, B., and De Santis, A. "Steganography and Secure Communication on Online Social Networks and Online Photo Sharing," *Broadband and Wireless Computing, Communication and Applications (BWCCA)*, 2011 International Conference on, Barcelona, 2011, pp. 363-368.
- [Chapman, 2001] Chapman, M., Davida, G., and Rennhard, M., "A Practical and Effective Approach to Large-Scale Automated Linguistic Steganography", *Proceedings of the Information Security Conference*, October 2001, pp. 156-165.
- [Chomphoosang, 2011] Chomphoosang, P., Zhang, P., Duresi, A., and Barolli, L., "Survey of trust based communications in social networks," in *2011 14th International Conference on Network-Based Information Systems*, 2011.
- [Cox, 1997] Cox, I.J., Kilian, J., Leighton, F. T. and Shamoon, T., "Secure spread spectrum watermarking for multimedia," *IEEE Transactions on Image Processing*, vol. 6, no. 12, pp. 1673–1687, 1997.
- [Fridrich, 1999] Fridrich, J. Applications of data hiding in digital images. In *ISSPA'99 Proceedings of the Fifth International Symposium on Signal Processing and its Applications (IEEE Cat. No. 99EX359)*, volume 1, page 9.
- [Fridrich, 2009] Fridrich, J. "Steganography in Digital Media: Principles, Algorithms, and Applications", Cambridge University Press; 1 edition, December 2009.
- [Ning, 2014] Ning, J., Singh, I., Madhyastha, H., Krishnamurthy, S., Cao, G., Mohapatra, P. "Secret message sharing using online social media", *Proceedings of the Communications and Network Security (CNS)*, 2014 IEEE Conference on, 319-327.
- [Hiney, 2015] Hiney, J., Dakve, T., Szczypiorski, K., and Gaj, K. "Using Facebook for Image Steganography", *Proceedings of the Availability, Reliability and Security (ARES)*, 2015 10th International Conference on, Toulouse, 2015, pp. 442-447.

- [Kott, 2016] Kott, A., Swami, A. and West B. J., "The Fog of War in Cyberspace," in *Computer*, vol. 49, no. 11, pp. 84-87, Nov. 2016.
- [Kundur, 1998] Kundur, D. and Hatzinakos, D. "Digital watermarking using multiresolution wavelet decomposition," in *Acoustics, Speech and Signal Processing*, 1998. Proceedings of the 1998 IEEE International Conference on, vol. 5. IEEE, 1998, pp. 2969–2972.
- [Lai, 2010] Lai, C.-C. and Tsai, C.-C. "Digital image watermarking using discrete wavelet transform and singular value decomposition," *IEEE Transactions on instrumentation and measurement*, vol. 59, no. 11, pp. 3060–3063, 2010.
- [Mazurczyk, 2016] Mazurczyk, W., Wendzel, S., Zander, S., Houmansadr, A., Szczypiorski, K. "Information Hiding in Communication Networks: Fundamentals, Mechanisms, Applications, and Countermeasures", Wiley-IEEE Press; 1 edition, February 2016.
- [McDonald, 2000] McDonald, A., Kuhn, M. 2000. "StegFS: A Steganographic File System for Linux". Proceedings of the Third International Workshop on Information Hiding, 463–477.
- [Mishra, 2014] Mishra, A., Agarwal, C., Sharma, A. and Bedi, P. "Optimized gray-scale image watermarking using DWT-SVD and Firefly algorithm," *Expert Systems with Applications*, vol. 41, no. 17, pp. 7858–7867, 2014.
- [Naskar, 2014] Naskar, R. and Chakraborty, R. S. "Reversible digital watermarking: Theory and practices," *Synthesis Lectures on Information Security, Privacy, & Trust*, vol. 5, no. 1, pp. 1–130, 2014.
- [Neuner, 2016] Neuner, S., Voyiatzis, A., Schmiedecker, M., Brunthaler, S., Katzenbeisser, S., Weippl, E. "Time is on my side: Steganography in filesystem metadata". 2016. *Digital Investigation* 18 : 76–86.
- [NIST, 2015] FIPS 180-4: Secure Hash Standard (SHS). 2015. URL: <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf> [access: 25.02.2018]
- [O'Ruanaidh, 1997] J. J. O'Ruanaidh, J. J., and Pun, T. "Rotation, scale and translation invariant digital image watermarking," 1997.
- [Olanrewaju, 2011] Olanrewaju, R. "Development of intelligent digital watermarking via safe region," PHD, Electrical and Computer Engineering, International Islamic University Malaysia, Kulliyyah of Engineering, 2011.
- [Potdar, 2005] Potdar, V. M., Han, S. and Chang, E. "A survey of digital image watermarking techniques," in *INDIN'05. 2005 3rd IEEE International Conference on Industrial Informatics*, 2005. IEEE, 2005, pp. 709–716.
- [Rivest, 2011] Rivest, R. "Reference implementation code for pseudo-random sampler for election audits or other purposes". 2011. URL: <https://people.csail.mit.edu/rivest/sampler.py> [access: 25.02.2018]
- [Szczypiorski, 2015] Szczypiorski, K., Janicki, A., Wendzel, S.: "The Good, The Bad And The Ugly: Evaluation of Wi-Fi Steganography", *Journal of Communications (JCM)*, Vol. 10(10), pp. 747-752, 2015.
- [Szczypiorski, 2016] Szczypiorski, K. "StegIbiza: a New Method for Information Hiding in Club Music", Proceedings of the 2nd International Conference on Frontiers of Signal Processing (ICFSP 2016), Warsaw, Poland, 15-17 October 2016, pp. 20-24.
- [Szczypiorski, 2016a] Szczypiorski, K. 2016. "StegHash: New Method for Information Hiding in Open Social Networks". *IJET International Journal of Electronics and Telecommunication*, 62 (4) : 347–352.

[Venckauskas, 2013] Venckauskas, A., Morkevicius, N., Petraitis, G., Ceponis, J. 2013. "Covert Channel for Cluster-based File Systems Using Multiple Cover Files". ITC 42 (3) : 260–267.

[Wilson, 2014] Wilson, A., Blunsom, P., Ker, A.: "Linguistic Steganography on Twitter: Hierarchical Language Modelling with Manual Interaction", Proc. SPIE 9028, Media Watermarking, Security, and Forensics 2014, 902803 (February 19, 2014).

[Zeki, 2009] Zeki, A. M. and Abdul Manaf, A. "A novel digital watermarking technique based on isb (intermediate significant bit)," World Academy of Science, Engineering and Technology, vol. 50, pp. 989–996, 2009.

[Zhang, 2006] Zhang, X. and Yang, Y. "A geometric distortion resilient image watermark algorithm based on dft-svd." Jisuanji Gongcheng/ Computer Engineering, vol. 32, no. 18, pp. 120–121, 2006.

[Zigomitros, 2012] Zigomitros, A., Papageorgiou, A., and Patsakis, C. "Social network content management through watermarking," in 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications. IEEE, 2012, pp. 1381–1386.

Chapter 5

Steganography Techniques for Command and Control (C2) Channels

Jedrzej Bieniasz and Krzysztof Szczypiorski

Institute of Telecommunications, Warsaw University of Technology, Poland

Contents

5.1	Introduction	190
5.2	Steganography Techniques for C2 Channels	191
5.2.1	Basic Concepts of Modern Steganography	191
5.2.2	Models of Steganographic C2 Channels	194
5.2.3	Cyber Threat Modelling of Steganographic C2 Channels	197
5.2.4	Countermeasures for Steganographic C2 Channels	198
5.2.5	Challenges for Steganographic C2 Channels	199
5.3	Case Studies of Steganographic C2 Channels	200
5.3.1	Introduction	200
5.3.2	Computer Botnets	201
5.3.2.1	C2 Channels Based on Digital Steganography	201
5.3.2.2	C2 Channels Based on Network Steganography	203
5.3.2.3	C2 Channels Based on Hybrid Steganography	206
5.3.3	Mobile Botnets	210
5.4	Summary	212

5.1 Introduction

The aim of steganography is to conceal secret data by utilizing various features of the different objects called *carriers*. Since the ancient times through the medieval ages until today, steganography has been widely used to hide information against observers on the way to recipients. Steganography was generally recognized in the context of hiding communication between adversaries or criminals, whereas other applications were considered as very specific or mostly theoretical without a possibility of the correct implementation. In last years, the increasing evidence of the real applications of steganography for the covert data storage and the covert data communication has given another security factor to consider by engineers and cyber security experts. To emphasize steganography as the trending topic for information security, recent reports by Kaspersky [1], McAfee [2] or Fortinet [3] warned that information hiding techniques applied by computer malicious software designers are highly emerging cyber threats. Applying steganography for computer malware operations and communication enables to:

- bypass common security mechanisms, such as antiviruses, Intrusion Detection/Intrusion Prevention systems, firewalls. All of them would allow a network traffic or multimedia files with hidden data as they would recognize them as normal, non-violating and non-suspicious network communication or data exchange.
- evade or make a detection a harder. Steganography introduces an additional level of difficulty in the forensic and malware analyses.

The modern approach tends to examine the cyberattacks as a complete *process* of doing harm by cyber adversaries in which executing the malicious code or command and control communication (C2) would be only one of the stages. In this approach, a cyberattack is modelled by a concept of advanced persistent threats (APTs) [4]. APT represents the model of multilayer intrusion campaigns, conducted in a long time frame by well-resourced and trained groups who target highly sensitive information, such as economic, proprietary, or national security intelligence. Information hiding techniques must be recognized as one of the tools that adversaries could utilize to achieve their goals. The evolution of APTs impacts the development of new defense approaches because the earlier methodologies are not sufficient anymore. One of the solutions is an intelligence-based network defense approach [5]. It leverages Cyber Kill Chain model to describe stages of intrusion, finding kill chain indicators of actions, identifying patterns that link particular intrusions and incidents into broader campaigns. Furthermore, the defenders' efforts are set in an iterative process of gathering and exchanging knowledge about adversaries and their techniques. It creates

intelligence feedback loop to enable defenders to decrease the likelihood of adversary's success with each following intrusion attempt.

Following the introduced strategy of *defense by proactive research*, this chapter would serve as the know-how for analysis, detection and breaking the C2 stage of APTs when it is secured by steganography. It focuses on applicability of information hiding techniques to the C2 stage. In Section 5.2, theoretical analysis of C2 channels established with steganographic techniques is conducted. Basic concepts of modern steganography (Section 5.2.1) and models of C2 channels based on steganography (Section 5.2.2) are introduced. Section 5.2.3 provides the model view on C2 channels by applying Cyber Kill Chain® and MITRE ATT&CK™ [6] methodologies. The discussion on the countermeasures for steganographic C2 channels (Section 5.2.4) are evaluated subsequently. Section 5.3 presents the state of knowledge of the real malicious software and botnets, where C2 channels are based on different steganographic methods. Both botnets in traditional computer systems and mobile systems are considered. It is concentrated on years 2010–2018 when several malicious campaigns and APTs with the modern steganographic capabilities were discovered. The chapter is concluded in Section 5.4.

5.2 Steganography Techniques for C2 Channels

5.2.1 Basic Concepts of Modern Steganography

On the basis of the applicability, modern information hiding techniques could be classified into two categories:

- Covert data storage methods: It means the application of storing techniques to hide data. Security is based on secret of localization of stored data and the algorithm to properly extract data from the hidden storage.
- Covert data communication methods: It means the application of network communication techniques to transmit data in a way that the observers are not aware of such communication. Security is based on the secret of localization of data inside the legitimate network data stream and the algorithm to properly extract data from it.

There are plenty of different steganographic methods that belong to one of these main categories. This chapter focuses on types of steganography methods presented in Figure 5.1. Table 5.1 compares features of these modern steganography methods.

Digital steganography utilizes digital media files such as images, audio and video files as carriers of hidden data. Johnson and Katzenbeisser [8] distinguished few types of techniques that are the foundation of modern digital steganography:

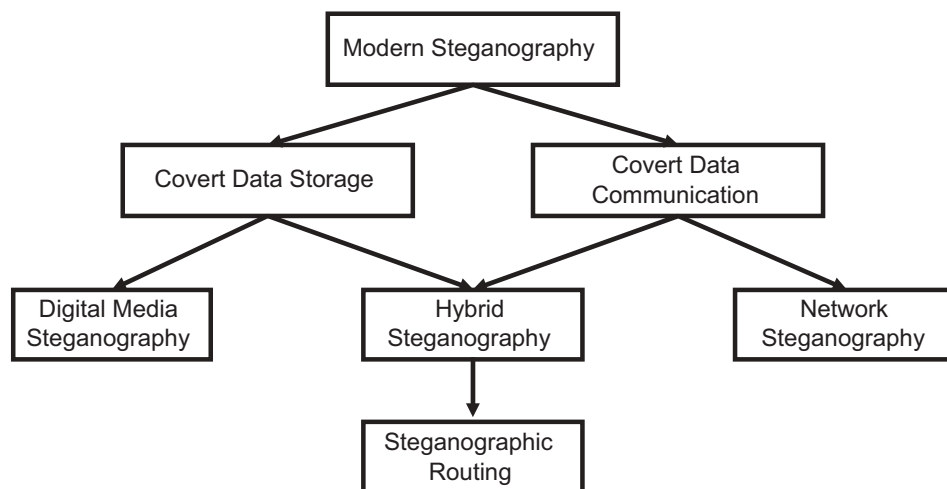


Figure 5.1 Classification of modern steganographic techniques.

Table 5.1 Comparison of modern steganography methods presented in Figure 5.1

<i>Feature</i>	<i>Digital Media Steganography</i>	<i>Network Steganography</i>	<i>Steganographic Routing</i>
Type	Covert data Storage	Cover data Communication	Hybrid
Modelling	High entropy [7], carriers [8]	Good/bad/ugly methods [9], carriers [10]	Low entropy [7], hybrid models
Detectability	Easily detectable for simple methods, need of image processing and machine learning for complex methods	Good methods hard to detect	Very hard, covered by multilayer operations
Robustness	Depending on hosting service	Vulnerable to network conditions	Vulnerable to network conditions and setup
Steganographic cost	Quality degradation	Network traffic anomalies	Some anomalies, but covered by multilayer operations
Implementation	Easy	Hard	Very hard

- Substituting redundant data in digital objects
- Embedding data in the transform space of digital signal
- Spread spectrum as a carrier
- Utilization of statistical properties of a digital file
- Secret embedding by direct malformation of digital signal
- Creating an artificial digital object as a carrier of hidden data (no alteration of the existent one)

The second most important technique of information hiding is network steganography. Whereas digital steganography is focused on digital files, network steganography utilizes networks and communication within them to transport hidden data. The aim of network steganography is to generate network packet flows with as normal characteristics as possible to establish covert communication channels among them. There are two main categories of network steganography depending on embedding procedure:

- Modifying protocols by, for example, using unused header fields or changing sequence in protocol data unit
- Manipulating time of protocol messages by, for example, introducing artificial, delays, loss or re-transmissions

More information about basic information hiding techniques in communication networks is included in [10]. Other methods that utilize different techniques into one design are called *hybrid methods*. They are recognized as the most resilient to any forensic procedure. Especially, steganographic C2 channels could profit for such methods where overlay communication patterns are encapsulated into several basic steganography techniques.

The original idea of distributed communication system with applied steganography is TrustMAS platform [11]. It developed the concept of *steganographic routing*, which is built on the design of *distributed steganographic router*. It provided the ability of creating covert channels between chosen endpoints called *StegAgents*. Links and paths between these endpoints may be established by using any of the steganographic techniques from TCP/IP protocol stack. Digital media steganography can be also used as streaming of files over application layer. It should be noted that one communication session can be realized over several paths and links where all of them are established with different steganographic technique. *Distributed steganographic router* is responsible for a reliable routing of such communication sessions. Furthermore, TrustMAS offers also a capability of covering original senders by applying random-walk algorithm for passing messages around without pointing the originator. In summary, TrustMAS is a fundamental achievement in research domain of steganographic communications systems. In upcoming years, many marks of ideas introduced by TrustMAS were found during forensic processes of malwares and botnets.

5.2.2 Models of Steganographic C2 Channels

Every steganographic C2 channel combines one or more of steganographic techniques (digital and network) with the right tactics on C2 stage of the cyberattack. C2 channels are realized with the standard communication models of *peer-to-peer* or *client-server* [12]. On the basis of these two common models of communications with application of steganography, message passing patterns for steganographic C2 channels could be identified: (1) *peer-to-peer* pattern; (2) *request-reply* pattern; (3) *observer* pattern; (4) *publish-subscribe* pattern; (5) *push-pull* pattern. Every method of covert C2 channel applies to one of these patterns directly or combines them in hybrid or chaining manner to introduce next layers of complexity.

Peer-to-peer pattern consists of direct communication over steganographic channels. Any other pattern or protocol can be encapsulated inside this, but there is no requirement of adding any mechanisms such as acknowledgments or control of losing data. Such covert channels could be realized by direct streaming of digital media files with embedded hidden data or with network steganography by transmitting hidden data through TCP/IP stack protocols. Both of them are applied directly for passing commands and to exfiltrate data from victims. Figure 5.2 shows operational scheme of this pattern.

Request-reply pattern distinguishes from *peer-to-peer* pattern with additional layer of having required replies for one or more requests. Figure 5.3 presents scheme of this pattern.

Observer pattern defines two parties of communication: observers and observable objects. Observers are required to maintain information about objects that are the subject of observation, so in the beginning there is a procedure of registering such observation. In steganographic C2 channels, it is realized in one of the two scenarios:

1. *Observing* C2 servers for operational data as normal files. Malicious bot *pulls* this data if it is available, using network steganography transmission.
2. *Observing* C2 servers for operational data as digital media files with embedded hidden data. Malicious bot *pulls* this data if it is available, using network steganography transmission or normal network traffic of TCP/IP protocol stack.

Operational schemes of *observer* pattern are showed in Figure 5.4.

Publish-subscribe pattern is extended realization of *observer* pattern with unique features. It assumes utilization of an additional bridge for bot herders and bots for two-way communication, which consists of commanding on the way from bot herder to malicious bot and exfiltrating data from victims to a bot herder. Furthermore, there is no need of registration or maintaining communication endpoints, but both sides need to know the publishing place only. Passing data from bot herder to malicious bot in this setting could consist of:

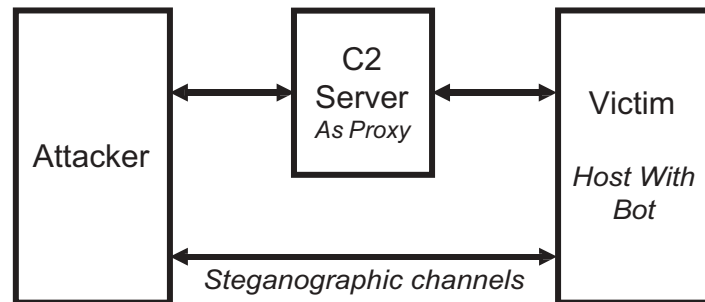


Figure 5.2 Peer-to-peer messaging pattern of steganographic C2 channel.

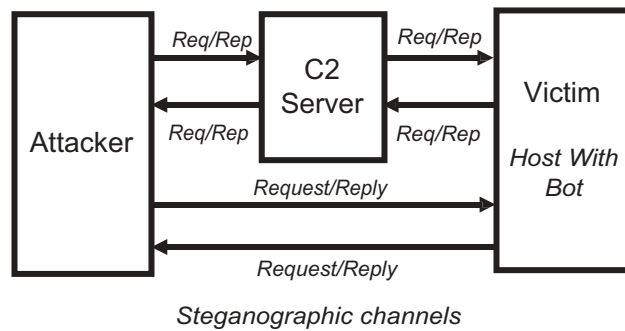


Figure 5.3 Request-reply messaging pattern of steganographic C2 channel.

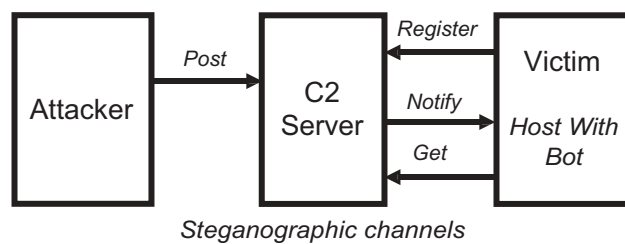


Figure 5.4 Observer messaging pattern of steganographic C2 channel in case of registering bot and pulling new data from C2 server.

1. *Publishing* commands and other C2 operational data published to C2 server as normal files. Malicious bot *pulls* this data if it is available, using network steganography transmission.
2. *Publishing* commands and other C2 operational data published to C2 server as digital media files with embedded hidden data. Malware bot *pulls* this data if it is available, using normal TCP/IP transmission, for example, HTTP.

It must be noted that a malicious bot needs to check availability of new data on C2 server regularly in this pattern. Exfiltrating data from victim to bot herder could be realized as:

1. *Publishing* stolen data embedded inside digital files into C2 servers. Bot herder *pulls* this data from C2 server if it is available.
2. *Publishing* stolen data as normal files utilizing network steganography to transmit data from victim's host to C2 server. Bot herder *pulls* this data from C2 server if it is available.

Operational schemes of *publish-subscribe* pattern are presented in Figure 5.5.

Push-pull pattern is the last recognized model of message passing. It is characterized by defined pipeline of sender ("pusher") and receiver ("puller"). It could be understood as *peer-to-peer* pattern with unique point-to-point data feeds. Scheme of *push-pull* pattern is described on Figure 5.6.

Data streams for steganographic channels could be classified as:

1. *Direct data streams*—realized by standard communication streams established for streaming digital media steganography or by using network steganography.
2. *Altered data streams*—realized by embedding hidden data into legitimate traffic and extract secret before normal flow achieves its destination. It is for utilization of *proxy* and *man-in-the-middle* concepts.

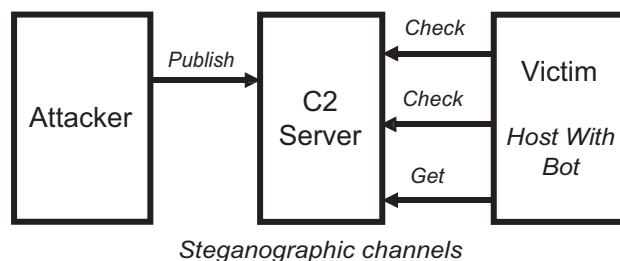


Figure 5.5 *Publish-subscribe* messaging pattern of steganographic C2 channel in case of pulling new data from C2 server.

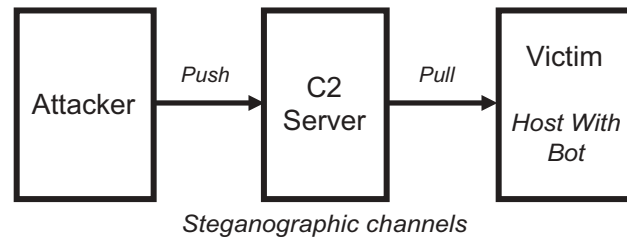


Figure 5.6 *Push-pull* messaging pattern of steganographic C2 channel in case of pulling new data from C2 server.

5.2.3 Cyber Threat Modelling of Steganographic C2 Channels

One of the common models of cyber threats used by the industry experts and scientists is the Cyber Kill Chain model [5]. This model focuses on defining stages of cyber threats representing a *kill chain* of the damage. The aim is to find those parts of chains and break the chains as early as possible. One of the defined stages of Cyber Kill Chain is C2 stage. Typically, compromised hosts must beacon outbound to an Internet controller server to establish a C2 channel. APT malware especially requires manual interaction rather than conducting activity automatically. It is concluded that steganography can be applied at this stage of cyber espionage presented by this model very clearly. Using steganography to violate the security of cyberspace was never as vital as today. The best strategy is to be proactive when preparing defensive strategies. We promote research on new methods, considering the real scenarios of executing information hiding techniques and implementing proof-of-concepts. Output of these activities serves as the basis to design methods, tools, processes and methodologies for protecting the cyberspace.

Continuing the introduced model of Cyber Kill Chain, we recognize botnets—a network of malicious software client (bots) as the type of technical component utilized in APTs or low-scale cyber malicious campaigns. From that perspective, a botnet could be logically constructed in C2 stage, where communication channels are established. The main strategy to compromise a modern botnet is to break the C2 communication channel. Sometimes it could be possible to take over the channel and mimic the bot to infer the protocol or to find the bot herders, but in most scenarios we would like to block actions of the adversaries.

To complete the model view on C2 channels, they should be referred to the model of MITRE ATT&CK [6]. MITRE ATT&CK is a globally accessible knowledge base of adversary tactics and techniques based on real-world observations. The ATT&CK knowledge base is used as a foundation for the

development of specific threat models and methodologies. It utilizes the matrix view of main categories of tactics and the techniques belonging to these tactics. There are two types of categories of tactics: defending tactics and attacking tactics. C2 channels in the context of MITRE ATT&CK model are classified as:

- Attack—Enterprise: Tactics—TA0011—Command and Control
- Attack—Mobile: Tactics—TA0037—Command and Control

5.2.4 Countermeasures for Steganographic C2 Channels

Section 5.2.3 established models of steganographic C2 channels, their features and communication patterns. In the first step, detection analysis process should be targeted at finding higher level patterns of steganographic C2 communications. This effort is hard as it needs to distinguish the malicious traffic from the background of normal network packet flows. It could be referred to “finding needle in the haystack” problem. After having suspicious network packet flows, breaking and reverse engineering procedures for steganography techniques are applied in:

- steganalysis methods for digital media files if they are transmitted over considered communication session;
- steganalysis methods for network traffic itself if suspicious traffic does not contain any digital files.

The domain of finding malicious objects between normal ones is called *anomaly detection*. It assumes that any malicious activity would feature a mark of being extraordinary in comparison to normal activity. Any *anomaly detection* algorithm would be consisted in two steps: (1) establishing a baseline model of observation and (2) examining current observations against baseline model. The modern approach for these efforts are classical mathematical statistics and recently applied as machine learning [13, 14].

After having suspicious network packet flows, fuzzing for any steganography applications is realized. Steganalysis of digital media files is realized by media file analysis targeted on file type. Image steganalysis tools utilize feature-based steganalysis and machine learning. The process consists of a noise residual computation, feature construction and binary classification. As reference, concepts and methods for digital steganalyzers resulting from [15] and [16] are considered. Very promising subset of machine learning algorithms for image steganalysis is deep machine learning [17]. It leverages deep machine learning architectures to extract different features of images. It can support uncovering the secret and

hidden data inside it. Network steganalysis can be at first realized by validating TCP/IP protocols to look, for example, for extra data in unused header fields. Main tools for network steganography detection and analysis are statistics and machine learning, for example, methods presented in [18] and [19]. There are also plenty of other approaches such as visualizations [20]. Another perspective on anomaly detection and finding sources of attacks is added by approaches such as idea of *moving observer*. It was practically realized as MoveSteg technique, proposed in [21] and implemented in [22]. It evaluates features of network flows to detect time-delay network steganography. It assumes that the observed delays result in changes of vector of observers moving around the network.

5.2.5 Challenges for Steganographic C2 Channels

Challenges for steganographic C2 channels result from the wider context of challenges for APTs [4]. Applying of information hiding techniques within APT tactics is emerging threat in recent time. The trend is proved by the fact that the most impacting cyber attackers, such as that backed by governments and armies, are implementing steganographic capabilities into their arsenal. In next years, research should focus on misusing hardware, software and networks by a combination of steganographic methods to hide multilayer operations in the logic of using them. Unfortunately, classical steganalysis methods are not sufficient anymore. There is increasing need for a new approach to deal with such activities. It is broadly recognized as *looking for a needle in a haystack*. The main questions concentrate on what to observe, how to observe and how to establish verifiable evidence of such operations. All of these aspects are related together. The very first answer for such problems is the big data approach with behavioral analysis. For example, collecting users' activity, such as

- uploading and downloading multimedia files from Internet services;
- correlations between using different services, hardware, software and networks;
- timing and
- distribution of activity

could establish a valuable source of indicators for big data analysis. The next challenge is choosing monitoring architectures and methodology. The promising algorithms to investigate such problems are:

- machine learning and big data analytics on large datasets;
- graph modelling and
- distinguishing automatic from human activity.

These algorithms could be used for data collected by observing the central network with aggregated data or in more distributed designs where the particular actors could be detected more accurately.

5.3 Case Studies of Steganographic C2 Channels

5.3.1 Introduction

This section presents the review of the real malicious software and botnets where C2 channels were based on the different steganographic methods. It is concentrated on years 2011–2017 when different malicious campaigns and APTs with modern steganographic capabilities were discovered. Some of the attacks started much earlier than the time of being caught, so dates have been picked from the official reports. Among these attacks, the three common information hiding techniques were mainly utilized for C2 channels:

- Embedding hidden data into network traffic;
- embedding hidden data into a digital file by modifying its structure or by digital steganography methods and
- combining simple steganography methods with the overlay communication protocols to establish multilevel and hybrid information hiding techniques.

Hiding data into network traffic was realized by abusing protocols from the standard TCP/IP stack, especially text protocols like DNS or HTTP. Another option was to mimic the network traffic of popular Internet applications, for example, chats and video players. Malware with digital media steganography capabilities uses different but majorly the simplest hiding algorithms. Two main scenarios of sharing these digital files were applied:

- Utilizing of different Internet services including but not limited to simple websites, dedicated file stores and social networks to contain the carriers with hidden data
- Sharing by streaming digital files with hidden data peer to peer

In this section, C2 channels based on steganography methods were evaluated in the context of the following:

- Detection: C2 channels are detected in the real attacks conducted by cyber threat groups. A new method for defending systems can be developed *a posteriori*—after occurrence of an attack and investigation of it.
- Prevention: C2 channels are designed in as academic and industrial research to broaden the knowledge of the problem. A new method for defending

systems can be developed *a priori*—before occurrence of an attack and with prediction of possible behavior of the attackers.

5.3.2 Computer Botnets

5.3.2.1 C2 Channels Based on Digital Steganography

One of the first recognized malware with utilization of covert data storage methods for C2 is Duqu [23] from 2011. The Duqu malware targeted many industrial manufacturers around the world to collect data about their industrial control systems (ICS). Duqu utilized covert storage data methods to exfiltrate secrets from the target. It appended digital images with the encrypted information stolen from the compromised environment. Next, the prepared carriers were sent to C2 servers. The generated network stream with the leaked data was recognized as benign flow of pictures. Duqu is a vital concept for cybersecurity community, as it represents a milestone in broadening knowledge about tactics of the modern cyber adversaries and APTs. It was analyzed that only high skill programmers dedicated to such tasks were able to develop that complex design. It was also connected with Stuxnet, as there were many resemblances between them.

The next intensification in finding utilization of digital file steganography for operations of botnets happened in 2014. Main example of malware with steganography capabilities is a morph of malware from Zeus family, Lurk [24]. Zeus is a well-known family of malware with long history of evolution. First discovered versions have been dated to 2007. The morph of Zeus from 2014 [25] added the utilization of steganography to complicate analysis and to bypass the intrusion detection systems. It used JPG image files to pass the C2 configuration files to bots. The encrypted C2 URLs data were hidden inside the malware's base configuration. URLs restored using decryption were like, for example, `hXXps://arrowtools.ru/xEZnzZEQuj8vJwsZ/flash-player.jpg`. As it was analyzed, such C2 URLs contained a path to JPG image file. The ZeusVM could request this JPG file over an HTTP or HTTPS GET request. The file was a legitimate JPG image that could be properly rendered. The observer could classify it as a simple and innocent image file. The JPG image consists of a sequence of segments, each beginning with a marker, each of which begins with a `0xFF` byte followed by a byte indicating what kind of marker it is. One of the markers (`0xFF`, `0xFE`) indicates a text comment. The interesting part of image file started 14 bytes after the comment marker (`0xFF`, `0xFE`). The data was encoded in base64. A DWORD value that contains the size of the encoded data in base64 was 10 bytes after the comment marker. It is 82,584 bytes in this case, but in practice the comments were always at the end of the JPG file followed by the 2-byte End of Image marker (`0xFF`, `0xD9`). As expected, the encoded data contained configuration files for C2 infrastructure. Next version of ZeusVM enhanced this steganography method with embedding the configuration using multiple comments inside JPG files. Each of these base64

comments were extracted and concatenated together in the same order as in the source JPG file. Example configuration of C2 infrastructure is presented subsequently:

```
Prologue
=====
Size: 61933 bytes
config flags: 0x00000000
# sections: 61
MD5: 73611d81
url_10ader (20002)
=====
http://icpiedimulera.it/flash.exe
url server (20003)
=====
https://arrowtools.ru/xEZNzZEQuijstZ/tree.php
AdvancedConfigs (20004)
=====
https://reybomerte.ru/xEZNzZEQuj8vasZ/flashplayer.jpg
https://suemnopshot.ru/xEZNzZEQuj8vasZ/flashplayer.jpg
http://unchangeclust.ru/xEZNzZEOujSVstZ/flashplayer.jpg
WebFilters (20005)
=====
!*microsoft.com/* (don't log)
!http://*myspace.com* (don't log)
!*googleusercontent.com* (don't log)
!*pipe.skype.com* (don't log)
!http://*odnoklassniki.ru/* (don't log)
!http://vkontakte.ru/* (don't log)
@*/login.osmp.ru/* (screenshot)
```

The most interesting part is AdvancedConfigs as it hints that more parts of the configuration were dispersed among different pictures.

In Lurk malware, steganography was applied on side of C2 server to hide URL from which a bot could download an executable. After installation, the malware could send the innocent request over HTTP on port 80 or over HTTPS on port 443 to establish a valid communication session to bypass a signature-based network detection. In response, the C2 server sent a bitmap image that contained a URL of a malware executable. The URL was encrypted and embedded in the bitmap image using steganography. The downloader's URLs utilized a steganographic technique that embeds information in the least significant bit (LSB) of every byte. The malware embedded data in the individual color pixels of a bitmap image. It was structured as follows:

- Byte 0–1: Signature of the bitmap
- Byte 2–5: Size of the file

- Byte 6–9: Reserved
- Byte 10–13: Data offset
- Byte 14–53: Information headers of the bitmap
- Byte 54+: Embedded data

A value of 0xFF is used to encode a bit “1,” and 0xFE is used to encode a bit “0.” Using this algorithm, Lurk could encode one bit of information for every eight bits (or 1 byte) of data. The malware decoded the hidden information from the first 32 bytes. The resulted value was added to a hard-coded value of 0x76 to locate the offset of the encoded malware URL. The same algorithm of decoding bits from bytes of data was applied to extract the malware URL. After the bytes are extracted from the image, the URL is decrypted, for example: `hxxp://zvld.alphaeffects.net/d/1721174125.zl`. Next, Lurk issued an HTTP GET request to the URL specified in the bitmap image to download the payload. The payload was prepared with obfuscation technique of using a four-byte XOR key. Finally, Lurk created an Internet Explorer process and injected the restored payload to execute it.

5.3.2.2 C2 Channels Based on Network Steganography

The first well-recognized example of malwares discovered with utilization of covert data communication method for C2 is Win32.Morto from 2011 [26]. This is the one of the first designs of malware with C2 communication over DNS protocol. DNS is important protocol from the perspective of attackers because:

- DNS is the critical protocol of Internet infrastructure based on TCP/IP protocol stack as it provides the base mechanism of mapping names of hosts represented by domains to IP network addresses. It means that the port of DNS service, UDP/53, will be opened and passed by firewalls, IDS and the other network security devices.
- DNS is the text protocol. It establishes the surface of abusing DNS protocol controllers. They do not have any specific procedures for validating values passed by protocol over the standard checks during resolving domains to IP addresses. The only way to find such abuses is to analyze DNS traces manually.

Win32.Morto used DNS TXT records for its C2 communication protocol. TXT records consists of alphanumeric strings stored within a DNS record. The malware infrastructure used these DNS TXT records for issuing commands. The bot once installed on a victim’s machine, attempted to request a DNS record for a number of hardcoded URLs. Instead of asking for mapping of a domain name to IP address, the malware queried for TXT records only. The returned TXT record contained commands that the malware should perform in

compromised environment. After receiving the response, the bot proceeded to validate and to decrypt the returned TXT record. The decrypted record contained a binary signature and an IP address at which another data could be download, typically a file of another malware for execution. Win32.Morto targeted Windows workstations and servers with propagation via RDP protocol. Below, there is an example of TXT record response passed to Win32.Morto malware:

```
Non-authoritative answer: malicious.url.net text = "p66662  
n1T!36666666666666666666666666666666kJ6666666666666716wTjuUj  
Ih2Njm7euX8oBU79qUDU1LDvfU8Tfx79Wa=0J66666666666666666666  
666666666666666666666666666666666666666666666666666666666  
666666666666666666666666666666666666666666666666666666666"
```

Covert data communication methods based on DNS protocol is one of the standard strategies to realize C2 stage of APTs. Throughout the years, cyber threat groups tried to apply other common TCP/IP protocols to hide their operations. Next case is Fokitor reported in 2013. The attackers utilized a stealthy Linux backdoor with the camouflage technique of hiding within the Secure Shell (SSH) protocol and other server processes. The backdoor provided remote command execution without leaving any footprint in the system, for example, opened network socket or attempts of connecting with C2 server. Instead, the malware code was planted in the SSH protocol in the man-in-the-middle setting to oversee the network stream for a specific sequence of characters consisting of colon, exclamation mark, semi-colon, period (“:!.,”). When this sequence of characters was appearing, the malware intercepted the following stream as the own payload. This interception was leaving no evidence in SSH logs. The commands in that payload was encrypted with Blowfish and encoded with base64. This technique provided stealthiness for attackers as their operations looked as legitimate connections through SSH, the other protocols and the other processes. To identify the presence of this malware, monitoring over the whole network and looking for SSH traffic with string of “:!.,” is needed. Another way to find this malware was to dump the SSHD process and search for the following strings: key=[VALUE]; dhost=[VALUE]; hbt=3600; sp=[VALUE]; sk=[VALUE] and dip=[VALUE], where [VALUE] can be an arbitrary value.

In next years, the more sophisticated techniques were in the field of cyber threat groups' interest. The malware gained the modular character, so it was no longer appropriate to simple categorize C2 channels by application of simple steganography techniques as one technique could be just a malware plugin. As the main strategy of cyber threat groups' operations is to trick defenders as much as possible, it is seen as the emerging trend of combining different covert data communication techniques into their malicious applications. The first example is

the PlugX malware (2014), which was used for example by Threat Group 3390 for their operations [27]. In one of the PlugX morphs, the ICMP protocol was utilized for the joining procedure in C2 infrastructure [28]. The case of implementing ICMP for C2 channels is similar to case of DNS protocol:

- ICMP is the crucial protocol of Internet infrastructure based on TCP/IP protocol stack as it provides the base mechanisms of network diagnosis, tracing and controlling transmissions. It means that every network appliance has this service opened. ICMP flows will be passed by firewalls, IDS and the other network security devices.
- ICMP is the text protocol. It establishes the surface of abusing ICMP protocol controllers. They do not have any specific procedures for validating values passed by protocol over the standard checks during the decision-making process. The only way to find such abuses is to analyze ICMP packets manually.

The data was transmitted as a payload of Echo reply (ICMP Type 0) packets. In the next level, the HTTP protocol was involved. Data was transmitted in a POST request matching the following pattern: POST/%p%p%p, where the %p values were random hexadecimal DWORDs. The following quadruples of headers are used in the request:

```
HHV1 / HHV2 / HHV3 / HHV4
LZ-ID / LZ-Ver / LZ-Compress / LZ-Size
IXP / IXL / IXK / IXN
FZLK1 / FZLK2 / FZLK3 / FZLK4
CC1 / CC2 / CC3 / CC4
ASH-1.0 / ASH-1.1 / ASH-1.2 / ASH-1.3 X-Session / X-Status /
X-Size / X-Sn
```

The latest version used the HTTP protocol in the new manner. The request was transmitted via GET header instead of POST. Furthermore, the data encoded in base64 was embedded in the Cookie header statement. After decoding this value from base64, a ciphered buffer was received. The encryption key was the first DWORD and the ciphered data was the remaining part. This version of PlugX was shipped with a new module allowing the malware to contact its C2 over DNS. The data was also encoded in base64 encoded and sent as a subdomain of the C2 in the DNS query.

Multigrain [29] is the next example of using DNS protocol to C2 communication. Malware utilized DNS queries with hardcoding the domain for the following:

- Initial beaconing: The malware collected the volume serial number and part of the MAC address to create a hash using DJB2 algorithm. The resulting hash was then combined with the computer name and a version number.

The resulted string was encoded with base32 and embedded into the domain name.

- Data exfiltration: Each Track 2 record founded inside the infected system was at first encrypted with a 1024-bit RSA public key, encoded with base32 and finally stored in a buffer. Every five minutes, the malware checked this buffer if it is not empty. If card data was present, the record of the buffer was embedded into DNS query within the domain name the following pattern: log.<encoded Track 2 data>.evildomain.com.

5.3.2.3 C2 Channels Based on Hybrid Steganography

In 2015, the first applications of steganography methods by emerging cyber threat groups for their operations were observed. One of the caught malware campaign is Hammertoss [30], which is probably run by a cyber-threat group from Russia, APT29. They utilized together a few techniques to establish C2 channels:

- Using steganography in image files to embed commands for bots
- Spreading handles to image files with commands through Twitter
- Storing image files with commands in Github and the other public Internet services with the storing capabilities
- Generating names of Twitter accounts in pseudo-random manner to look for tweets with image URLs. It can be compared to the idea of random generation of domains (domain generation algorithm) to overcome the problem of hard-coding the list of C2 server URLs.

The scheme of Hammertoss operations is presented in Figure 5.7. The malware at first looked for Twitter accounts whose names were generated with the included algorithm (STEP 1). If accounts existed and had the tweet (STEP 2), a bot downloaded its content (STEP 3). In the valid tweet, the attackers included URL, offset in file where hidden data is appended and the part of decryption key. Using Internet Explorer, the bot downloaded image from the URL (STEP 4). Next, the malware searched the cache of Internet Explorer for any images at least as large as the offset specified in the original post on Twitter. Hammertoss located the encrypted data at the offset specified in the tweet (STEP 5). It decrypted the data using a key composed of hard-coded data from the malware binary appended with the characters from the tweet. In this case, the image contained (STEP 6) appended and encrypted data that Hammertoss would decrypt and execute. The data might also include other commands or the login credentials to upload a victim's data to a cloud storage service (STEP 7).

Hammertoss utilized a set of techniques to hide its operations and to complicate forensic methods. It goes further than simple application of image steganography, as it adds the overlay of communication over social network (Twitter).

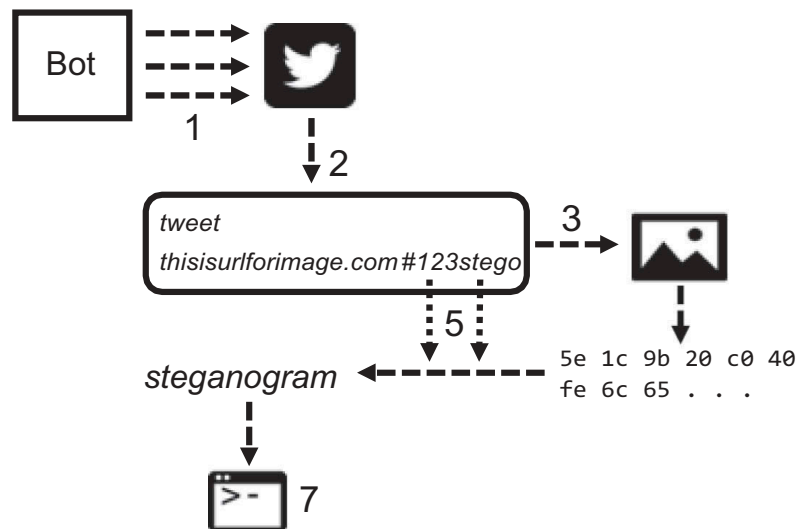


Figure 5.7 Scheme of Hammertoss steganographic C2 channel basing on [30].

In 2018, Talos reported catching of a malware called “VPNFilter,” [31] controlled by another cyber threat group APT28 (Fancy Bear). VPNFilter malware was a multistage, modular platform with capabilities of supporting both data gathering and destructive attack operations. Attackers implemented several redundant techniques for establishing C2 channels to improve reliability and robustness of their infrastructure. One of these mechanisms was to search and download images from particular Photobucket’s profiles. The example of URL is as follows: <http://photobucket.com/user/bob7301/library>. Once the malware completed initialization, it started downloading pages from the Photobucket URLs. The malware downloaded the first image from the gallery the URL is referencing, and then proceeded to extract the IP address of C2 server. The address was extracted from six integer values of GPS latitude and longitude in the EXIF information of the image. If this procedure failed, malware tried to download image with IP address of C2 server from the hard-coded backup domain. Finally, if earlier methods failed, malware started to listen for a specially crafted packet as per the following procedure:

- Looking for all IPv4/TCP SYN packets
- Validating packet for:
 - destination IP address if it matches what it obtained when the malware started this procedure. Malware could also skip this step if it failed to get an IP from api.ipify.org
 - size of 8 bytes or more
- Searching for the sequence of bytes 0x0C15222B in the validated packet

- Interpreting bytes after sequence of 0x0C15222B as the IP address. For example, 0x01020304 is treated as IPv4 address of 1.2.3.4

This algorithm can be recognized as a covert data storage method with utilization of payload of IPv4/TCP packets.

Application of hybrid steganography methods for C2 channels was recognized as the most dangerous type much earlier. It was clear that combining different techniques to obfuscate the real operation of C2 channel introduces a hard challenge for malware analysts and forensic investigators. The established way to tackle with the problem is the prevention approach. Throughout the years, the several academic and industrial teams tried to develop their own proof-of-concepts of such C2 channels for synthesizing detection algorithms. Two well-recognized results of botnets with application of hybrid hiding techniques for C2 channels are Stegobot (2011) and Instegram (2016).

Stegobot, introduced in 2011 [32], was a new-generation botnet that communicated over probabilistically unobservable communication channels. If the C2 communication is unobservable then botnet detection can be significantly more difficult than where communication is not hidden. Unlike conventional botnets to date, Stegobot traffic did not utilize a communication endpoint between bots. Instead, it applied a covert communication over a social network. It introduced an overlay for bot-to-botmaster communication that took place along the edges of this social network. Bots used digital image steganography to hide the presence of communication within image sharing behavior of user interaction. A bot executed on the infected computer can communicate with bots running on different computers, if users of these machines were connected by in the social network. The social network offers a peer-to-peer overlay in which the information is transferred from each bot to the botmaster. The steganographic C2 channel is constructed by hiding the data within images. By keeping the size of the hidden data to a limit, it was possible to make the presence of bot communication difficult to discover by examining the communication channel alone. Communication is realized in a push-pull model with restricted flooding routing. In this model, when a user uploads an image to a social network from an infected host (STEP 1), the bot does a man-in-the-middle (MITM) attack to intercept the image (STEP 2). It inserts the data into the image using an image steganography technique. Upon completion of image upload, all the neighbors of the user from the same social network are notified (STEP 3). When a neighboring user of the publisher logs into the social network from the infected machine and views the picture, the bot downloads it (STEP 4). After downloading, it extracts the steganographically embedded data carried by the image (STEP 5). The botmaster has a view of all uploaded images with hidden data by controlled bots. When the botmaster intends to put a command, it does by preparing a hidden message and uploading to its social networking account from where bots can pull it to execute.

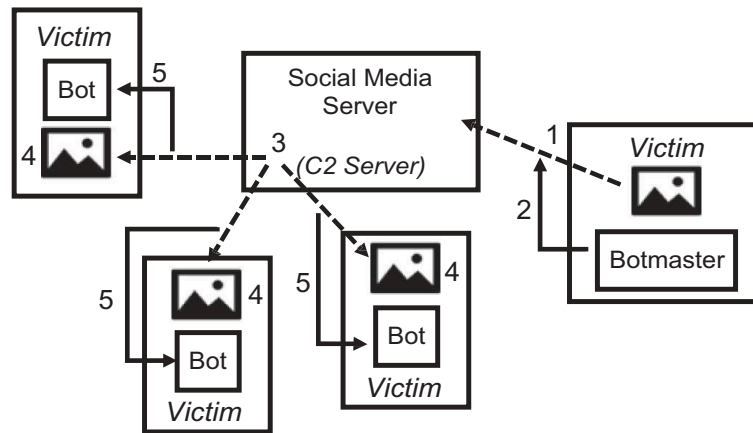


Figure 5.8 Scheme of Stegobot steganographic C2 channel based on [32].

The possibility of designing such a botnet even with a less-than-optimal routing mechanism such as restricted flooding was shown. Analysis of Stegobot's network throughput indicated its stealthiness, but also a capability of channeling enough quantities of data from its victims to the botmaster, estimated at tens of megabytes every 30 days.

Another example of research on C2 channels established by combination of steganography social networks is Instegogram [33] from 2016. Researches developed a proof-of-concept of their system. The remote access trojan was configured to communicate with the specific Instagram accounts on which images containing messages encoded with a steganographic scheme would be published. The malware included a steganographic decoder to extract a hidden payload from each downloaded image. Next, a restored data could be executed on the system. The bot continuously checked the Instagram accounts feeds for the next images with commands. If there was a new one, it was downloaded, decoded and executed. The bot also could share the results in the same way by uploading images to Instagram accounts with the embedded data. In that simple *proof-of-concept*, the limit of characters that could be reliably transmitted in the JPG images was established at 40. The capacity could be increased using other coding techniques. The main challenge taken by researches was avoiding image processing algorithms of Instagram, which can distort a hidden data inside JPG steganography images. They conducted a few methods to increasing robustness of their digital image steganography technique:

- Resizing all cover images to a size-and-aspect ratio that Instagram would accept without resizing.

- Avoiding double-compression problem, by extracting the quantization table from an existing Instagram image. It was discovered that Instagram utilized the same quantization table across all Instagram images.

5.3.3 Mobile Botnets

Another category of emerging cyber threats is abusing mobile platforms, smart-phones and other ubiquitous devices. According to [34], in most cases mobile malicious software shares the same attack surfaces as that targeting computers. Although there are several aspects of mobile platforms, which impact security of cyberspace in new directions:

- Mobile network communication: mobile platforms utilizes another protocol stack of GSM/UMTS/LTE/5G for communication. Furthermore, these networks are outside control of the mobile network providers. TCP/IP protocol stack for Internet connectivity is just one of the options to transfer data over these networks.
- Device-specific modules: Mobile platforms consist of many modules that are not present in traditional computers such as GPS, sensors and NFC.
- Networking in such devices is always on, whereas in traditional computer networks, users switch them off, for example, during night period. It means that mobile devices are constantly accessible through physical networking interfaces or by applications.
- Networks of mobile devices are dynamic. For example, IP addresses keep changing and cyber attackers could use this fact as a layer of hiding their operations. It means a harder task to monitor, detect and block them.

The first case of mobile malicious software with steganographic C2 channels is Pegasus [35]. Pegasus was professionally developed with highly advanced capabilities such as exploiting mobile systems (Android, iOS), zero-day vulnerabilities and anti-forensic mechanisms such as code obfuscation and encryption. It could evade systems and application security of voice/audio calls and apps, which included mobile apps such as Gmail, Facebook, WhatsApp, FaceTime, Viber, WeChat, Telegram, and Apple's built-in messaging and email apps. It could steal contact lists, GPS locations and router passwords stored on the device. The iOS version of this malicious software is known as "Trident." Pegasus consisted of methods to establish stealth C2 channels. It utilized SMS service to pass commands through them. An example of such a message is as follows:

```
Your Google verification code
is: 5678429\nhttp://gmail.com/?z=FEcCAA==&i=MTphYWxhY
W4udHY6NDQzLDE6bWFub3Jhb25saW51
Lm51dDo0NDM=&s=zpvzPSYS674=
```

This message actually contained a command to update a list of available C2 servers. Pegasus was capable of receiving five types of commands via SMS channel. Command ID was determined based on the last number in the verification code. The command in message presented above has an ID of 9. Further analysis of the captured binary of Pegasus showed that C2 communication in general abuses several legitimate SMS messages of two-factor authentication processes for Google, Facebook or Evernote. This functionality allowed Pegasus to be updated if Internet was not available, for example, in the case of breaking C2 infrastructure. Adversaries could provide a new list of C2 servers via SMS channel.

Another mobile malware was reported by Unit42 of Palo Alto Networks called “SpyDealer” [36]. SpyDealer was capable of communicating C2 servers via a few channels: SMS, UDP and TCP. The most interesting application in SpyDealer is C2 channel based on SMS. SpyDealer used an interception technique of registering SMS receiver with a higher priority than default messaging application in Android OS. The commands could be received through this channel to decode, parse and process. Every SMS with a command consists of an index of command and arguments for command split by a newline. SpyDealer could also update its C2 server address following one of the procedures: interpreting a command index with length larger than 4 as IP address or parsing IP address from SMS message body, which starts with L112 string. SpyDealer needed to acknowledge some of the commands. It was realized by a reply message in format of msg:repcall|<phone number>. All intercepted SMS messages with commands were aborted. It means that user would never see any of these messages in the messaging application. From the user’s perspective, C2 communication is hidden against him. Other SMS messages could be also blocked if SpyDealer was set to do so or the incoming number was included in the blocking list.

In case of establishing out-of-band channels on mobile platforms, there are primarily *proof-of-concepts*. Such comprehensive research on that topic is presented in [37]. Authors reviewed different vectors of abusing physical interfaces to establish out-of-band channels for steganographic C2 communication. They investigated the applicability of sensing-enabled covert channels in mobile phones. The main advantage from perspective of cyber attackers is that malware using such channels would be very difficult or impossible to detect. Researchers prepared *proof-of-concept* malware to verify the range of problem. They achieved a system with capability of sending C2 messages without using any wireless or cellular networks, only using popular hardware and Android-based mobile phones. They also presented the results for several steganographic carriers such as music, video, household lighting and magnetic fields.

5.4 Summary

This chapter introduced the concepts of applying steganography for C2 channels. In the theoretical part of the chapter, the problem has been analyzed from different perspectives:

- Reviewing steganography techniques: network, digital, hybrid methods toward steganographic routing
- Referring cyber threat modelling perspective to steganography by applying standard methodologies: Cyber Kill Chain and MITRE Att&ck
- Defining theoretical communication models and messaging patterns of steganographic C2 channels
- Considering countermeasures for steganographic C2 channels

Next, Section 5.3 presented the state of knowledge of the real malicious software and botnets, where C2 channels were based on the different steganographic methods. Botnets for computer and mobile systems were considered. The focus has been on years 2010–2018 when a few malicious campaigns and APTs with modern steganographic capabilities were discovered. Table 5.2 categorizes the reviewed steganographic C2 channel methods to theoretical concepts introduced in this chapter.

It can be concluded that malware designers use many different techniques, which in turn introduce hard efforts in modern forensic procedures. This chapter

Table 5.2 Reviewed steganographic C2 channels referred to models of such communications

<i>Method</i>	<i>Botnet</i>	<i>Type of steganography</i>	<i>Model</i>	<i>Messaging pattern</i>
Win32.Morto	Computer	Network	Client-server	Publish-subscribe
Fokitor	Computer	Network	Peer-to-peer	Push-pull
PlugX	Computer	Network	Peer-to-peer	Request-reply
Multigrain	Computer	Network	Peer-to-peer	Peer-to-peer
Duqu	Computer	Digital	Client-server	Publish-subscribe
ZeusVM	Computer	Digital	Client-server	Publish-subscribe
Lurk	Computer	Digital	Client-server	Request-reply
Hammertoss	Computer	Hybrid	Client-server	Publish-subscribe
VPNFilter	Computer	Hybrid	Client-server	Observer
Stegobot	Computer	Hybrid	Client-server	Push-pull
Instegram	Computer	Hybrid	Client-server	Publish-Subscribe
Pegasus	Mobile	Hybrid	Peer-to-peer	Push-pull
SpyDealer	Mobile	Hybrid	Peer-to-peer	Push-pull

not only summarizes state-of-the-art practical examples of steganographic C2 channels but also establishes standard models of such channels. This analysis could be very useful for present and future analysts working in the C2 ecosystem. It should help focusing on developing new algorithms to detect C2 channels and to break the kill chain of cyberattacks as soon as possible.

Acknowledgment

This work has been supported by the National Centre for Research and Development (Agreement No. CYBERSECIDENT/369532/I/NCBR/2017) under the CyberSecIdent Programme.

References

- [1] Kaspersky Lab. Kaspersky lab identifies worrying trend in hackers using steganography. https://usa.kaspersky.com/about/press-releases/2017_kaspersky-lab-identifies-worrying-trend-in-hackers-using-steganography, 2017. [Online; accessed 10.01.2019].
- [2] McAfee Labs. McAfee labs threats report. www.mcafee.com/enterprise/en-us/assets/reports/rp-quarterly-threats-jun-2017.pdf, 2017. [Online; accessed 10.01.2019].
- [3] Jeannette Jarvis. Steganography: Combatting threats hiding in plain sight. www.fortinet.com/blog/threat-research/steganography-combatting-threats-hiding-in-plain-sight.html, 2018. [Online; accessed 10.01.2019].
- [4] Adel Alshamrani, Sowmya Myneni, Ankur Chowdhary, and Dijiang Huang. A survey on advanced persistent threats: Techniques, solutions, challenges, and research opportunities. In *IEEE Communications Surveys Tutorials*, page 1, 2019.
- [5] Eric M. Hutchins, Michael J. Cloppert, and Rohan M. Amin. Intelligence driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Leading Issues in Information Warfare and Security Research*, 1: 1, 2011.
- [6] MITRE ATT&CK: Globally-accessible knowledge base of adversary tactics and techniques based on real-world observations. <https://attack.mitre.org>. [Online; accessed 10.01.2019].
- [7] F. Beato, E. De Cristofaro, and K. B. Rasmussen. Undetectable communication: The online social networks case. In *2014 Twelfth Annual International Conference on Privacy, Security and Trust*, pages 19–26, July 2014.
- [8] Neil F. Johnson and Stefan C. Katzenbeisser. A survey of steganographic techniques. Chapter 3 in *Information Hiding Techniques for Steganography and Digital Watermarking*, edited by Stefan Katzenbeisser and Fabien A. P. Petitcolas, Artech House Books, USA, 2000.
- [9] Krzysztof Szczypiorski, Artur Janicki, and Steffen Wendzel. “The good, the bad and the ugly”: Evaluation of wi-fi steganography. *Journal of Communications*, 10: 8, 2015.

- [10] Wojciech Mazurczyk, Steffen Wendzel, Sebastian Zander, Amir Houmansadr, and Krzysztof Szczypiorski. *Information Hiding in Communication Networks: Fundamentals, Mechanisms, Applications, and Countermeasures*. Wiley-IEEE Press, USA, 1st edition, 2016.
- [11] Wojciech Mazurczyk, Krzysztof Szczypiorski, and Igor Margasinski. Steganographic routing in multi agent system environment. *CoRR*, abs/0806.0576, 2008.
- [12] Andrew S. Tanenbaum and David J. Wetherall. *Computer Networks*. Prentice Hall Press, Upper Saddle River, NJ, USA, 5th edition, 2010.
- [13] Pedro Casas, Johan Mazel, and Philippe Owezarski. Unsupervised network intrusion detection systems: Detecting the unknown without knowledge. *Computer Communications*, 35(7): 772–783, 2012.
- [14] Monowar H. Bhuyan, Dhruva Kumar Bhattacharyya, and Jugal K Kalita. Network anomaly detection: Methods, systems and tools. *IEEE Communications Surveys & Tutorials*, 16(1): 303–336, 2014.
- [15] Jessica Fridrich and Jan Kodovsky. Rich models for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security*, 7(3): 868–882, 2012.
- [16] Weixuan Tang, Haodong Li, Weiqi Luo, and Jiwu Huang. Adaptive steganalysis against wow embedding algorithm. In *Proceedings of the 2Nd ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec '14, pages 91–96, New York, NY, USA, 2014. ACM.
- [17] Jian Ye, Jiangqun Ni, and Yang Yi. Deep learning hierarchical representations for image steganalysis. *IEEE Transactions on Information Forensics and Security*, 12(11): 2545–2557, 2017.
- [18] Yongfeng Huang, Shuhong Tang, C Bao and Yau Jim Yip. Steganalysis of compressed speech to detect covert voice over internet protocol channels. *IET Information Security*, 5(6): 26–32, 2011.
- [19] Artur Janicki, Wojciech Mazurczyk, and Krzysztof Szczypiorski. Steganalysis of transcoding steganography. *annals of Telecommunications - Annales des te'le'Communications*, 69(7): 449–460, 2014.
- [20] Wojciech Mazurczyk, Krzysztof Szczypiorski, and Bartosz Jankowski. Towards steganography detection through network traffic visualisation. In *2012 IV International Congress on Ultra Modern Telecommunications and Control Systems*, pages 947–954, Oct 2012.
- [21] Krzysztof Szczypiorski, Artur Janicki, and Steffen Wendzel. The good, the bad and the ugly: Evaluation of wi-fi steganography. *Journal of Communications*, 10(10): 749–750, 2015.
- [22] Krzysztof Szczypiorski and Tomasz Tyl. MoveSteg: A method of network steganography detection. *International Journal of Electronics and Telecommunications*, 62(4): 335–341, 2016.
- [23] Symantec Security Response. W32.Duqu. The precursor to the next Stuxnet. www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_duqu_the_precursor_to_the_next_stuxnet.pdf, 2011. [Online; accessed 10.01.2019].

- [24] Dell Secureworks. Malware analysis of the Lurk downloader. www.secureworks.com/research/malware-analysis-of-the-lurk-downloader, 2014. [Online; accessed 10.01.2019].
- [25] Adam Greenberg. New variant of Zeus banking trojan concealed in JPG images. www.scmagazine.com/home/security-news/new-variant-of-zeus-banking-trojan-concealed-in-jpg-images, 2014. [Online; accessed 10.01.2019].
- [26] Symantec. Morto worm sets a (DNS) record. www.symantec.com/connect/blogs/morto-worm-sets-dns-record, 2011. [Online; accessed 10.01.2019].
- [27] Dell Secureworks. Threat Group 3390 Cyberespionage, 2015. www.secureworks.com/research/threat-group-3390-targets-organizations-for-cyberespionage [Online; accessed 23.06.2019].
- [28] Fabien Perigaud. PlugX "v2": meet "SController", 2014. <http://blog.airbuscybersecurity.com/post/2014/01/PlugX-v2%3A-meet-SController> [Online; accessed 23.06.2019].
- [29] FireEye. MULTIGRAIN - point of sale attackers make an unhealthy addition to the pantry. www.fireeye.com/blog/threat-research/2016/04/multigrain_pointo.html, 2016. [Online; accessed 10.01.2019].
- [30] FireEye Threat Intelligence. HAMMERTOSS: Stealthy tactics define a Russian cyber threat group. www.fireeye.com/blog/threat-research/2015/07/hammertoss_stealthy.html, 2015. [Online; accessed 10.01.2019].
- [31] Cisco Talos Intelligence. New VPNFilter malware targets at least 500K networking devices worldwide. <https://blog.talosintelligence.com/2018/05/VPNFilter.html>, 2018. [Online; accessed 10.01.2019].
- [32] Shishir Nagaraja, Amir Houmansadr, Pratch Piyawongwisal, Vijit Singh, Pragya Agarwal, and Nikita Borisov. Stegobot: A covert social network botnet. In Tomáš Filler, Tomáš Pevný, Scott Craver, and Andrew Ker, Eds., *Information Hiding*, pages 299–313, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.
- [33] Endgame. Instegogram: Leveraging Instagram for C2 via image steganography. www.endgame.com/blog/technical-blog/instegogram-leveraging-instagram-c2-image-steganography, 2016. [Online; accessed 10.01.2019].
- [34] Marios Anagnostopoulos, Georgios Kambourakis, and Stefanos Gritzalis. New facets of mobile botnet: Architecture and evaluation. *International Journal of Information Security*, 15(5): 455–473, 2016.
- [35] Lookout. Technical analysis of Pegasus Spyware. <https://info.lookout.com/rs/051-ESQ-475/images/lookout-pegasus-technical-analysis.pdf>, 2016. [Online; accessed 10.01.2019].
- [36] Palo Alto Networks. SpyDealer: Android trojan spying on more than 40 apps. <https://unit42.paloaltonetworks.com/unit42-spydealer-android-trojan-spying-40-apps>, 2016. [Online; accessed 10.01.2019].
- [37] Ragib Hasan, Nitesh Saxena, Tzipora Haleviz, Shams Zawoad, and Dustin Rinehart. Sensing-enabled channels for hard-to-detect command and control of mobile devices. In *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*, ASIA CCS '13, pages 469–480, New York, NY, USA, 2013. ACM.

A.4. Artykuł [A4]

Journal of Universal Computer Science, vol. 25, no. 9 (2019), 1109-1130
submitted: 29/1/19, accepted: 9/7/19, appeared: 28/9/19 © J.UCS

Mobile Agents for Detecting Network Attacks Using Timing Covert Channels

Jędrzej Bieniasz, Monika Stępkowska
Artur Janicki, and Krzysztof Szczypiorski

(Division of Cybersecurity, Institute of Telecommunications

Warsaw University of Technology, Poland

{j.bieniasz, m.stepkowska, a.janicki, k.szczypiorski}@tele.pw.edu.pl)

Abstract This article addresses the problem of network attacks using steganographic techniques based on the manipulation of time relationships between IP packets. In the study, an efficient method to detect such attacks is presented. The proposed algorithm is based on the Change Observation Theory, and employs two types of agents: base and flying ones. The agents observe the time parameters of the network traffic, using proposed meta-histograms and trained machine learning algorithms, in the node where they were installed. The results of experiments using various machine learning algorithm are presented and discussed. The study showed that the Random Forest and MLP classifiers achieved the best detection results, yielding an area under the ROC curve (AUC) above 0.85 for the evaluation data. We showed a proof-of-concept for an attack detection method that combined the classification algorithm, the proposed anomaly metrics and the mobile agents. We claim that due to a unique feature of self-regulation, realized by destroying unnecessary agents, the proposed method can establish a new type of multi-agent intrusion detection system that can be applied to a wider group of IT systems.

Key Words: network security, traffic analysis, anomaly detection, intrusion detection, steganography, multi-agent systems

Category: C.2.0, K.6.5, I.2.0

1 Introduction

Ensuring a high level of network security is a must in these times of global penetration of cyberattacks. Cybercriminals constantly improve their methods. They search for new targets that can be easily attacked. They work on new ways of delivering, installing and controlling malware, in a way that these processes are difficult to detect by security systems. They use hidden channels to control botnets, for example, to realize a DDoS attack. Last but not least, they create hidden channels of communication to allow the leakage of sensitive data.

In many of the above described actions, steganographic techniques are used. They allow the fact of communication itself to be hidden [Lubacz et al., 2014]. Steganographic methods exploit the unused fields in various protocol headers (e.g., TCP, UDP, RTP) [Murdoch and Lewis, 2005], modify payload data or time relationships between arriving packets [Mazurczyk et al., 2012]. While there are already various methods of steganalysis which try to detect protocol stegano-

graphy or payload manipulation (e.g., [Janicki et al., 2015]), not much research has been conducted to efficiently detect timing network steganography.

This article aims to fill this gap. We would like to propose a system that will employ mobile agents. They will move within the network according to the proposed algorithm, to get closer to the attacker, in line with the Change Observation Theory and the moving observer concept [Szczypiorski and Tyl, 2016]. We will present a proof-of-concept of our method applied to detecting timing network steganography.

1.1 Network intrusion detection

As a response to the growing activity of cybercriminals, in recent years a variety of countermeasures against network intrusions have emerged. These solutions are usually referred to as intrusion detection systems (IDS). In general, they are divided into network-based IDS systems (NIDS) and host-based IDS systems (HIDS) [Kwon et al., 2017]. The former group of systems analyses the network traffic in selected nodes of the computer network, while the latter one is installed locally on a machine, usually a server. HIDS systems are dedicated to the given operating system and even to the specific services offered by the host, e.g., they can be dedicated to a web server. NIDS systems are by far more universal, which is why they gain significant attention of the scientific world, and also in this case, a focus will be devoted to them.

Another division in IDS systems is based on the detection method. While signature-based IDS systems try to identify similarities between the analyzed data and the patterns in the threat database, the anomaly-based IDS systems aim to detect anomalies in the network traffic [Bhuyan et al., 2014, Casas et al., 2012]. In addition, the current study will follow the latter approach, as it does not rely on any signature database, and, therefore, is able to detect new threats.

Many IDS systems search for anomalies when analyzing various parameters of the network data [Ahmed et al., 2016]. Some researchers use the basic netflow set of the IP network traffic parameters, such as the number of packets in the flow, duration of the flow or used port numbers. However, several studies show that IDS systems gain much in their effectiveness if the basic netflow set of parameters is extended by their various derivatives. In [Su, 2011], the authors showed that adding detailed parameters, such as ICMP checksum error count or count of the TCP packets with a sequence number equal to 0, significantly improved the detection of DoS attacks. Careful selection of the feature space is crucial to reach high effectiveness in the intrusion detection. However, to the best of our knowledge, hardly any IDS employs the measurement of time relationships between packets.

Several researchers showed the applicability of histogram analysis for intrusion and anomaly detection. In [Kind et al., 2009], the authors described which

parameters can be most useful in detecting network anomalies when observing their distributions using histograms. They showed that, e.g., histograms of the source and the destination port numbers can help in the detection of network scans or worm propagation, while histograms of the TCP flags can inform about the SYN flooding attacks or server failures, while the histograms of the flow duration values can reveal a DoS attack. The authors showed that histograms can be a synthetic tool to capture statistical information about a given parameter. In [Miller et al., 2011], the researchers found histograms of the network parameters useful in the detection of various HTTP attacks.

The problem of intrusion detection is often addressed using a machine learning (ML) approach [Buczak and Guven, 2016]. A typical procedure consists of training a ML-based classifier using data, usually labeled, acquired from life or emulated environments. Next, the classifier is tested using evaluation data. A large variety of ML algorithms have been proposed by researchers, starting from the basic as k-Nearest Neighbors (k-NN) and ending with genetic algorithms or various types of neural networks, such as multilayer perceptrons (MLP), Kohonen maps [Tsai et al., 2009] or deep learning techniques [Kwon et al., 2017].

1.2 Multi-agent cyber defense solutions

The proposed system combines the idea of network intrusion detection systems with the concept of multi-agent systems. Such approach has been investigated earlier in literature, where the biological inspirations have been involved. [Haack et al., 2011] introduced the system implementing swarming-agent-based approach to network infrastructure defense. They implemented lower-level software agents to carry out tasks which administrators would not be able to perform quickly enough to mitigate security threats. They called this method as *Cooperative Infrastructure Defense*, where the ant-based approach enables cooperation between administrators and agents to foster a collaborative problem solving. The same team of researches developed their investigations in [Fink et al., 2014], by providing more results and developing the state-of-the-art solution in ant-based cyber defense.

Cyber defense based on multi-agent systems is recognized as a modern and a very efficient approach. It gains interest of academics, industry, law enforcement agencies and even military. In [Kott, 2018], the author argues that intelligent autonomous agents will be the standard on the battlefield of the future. It means that intelligent autonomous cyber defense agents are going to become the main cyber fighters. The author introduced several novel ideas and summarized the other ones into a unified and reference architecture of such multi-agent system for cyber defense.

1.3 Attacks using steganography

Steganography conceals secret data by utilizing various features of different objects called *carriers*. It is generally recognized as the method of communication used by many kinds of adversaries or criminals. Other applications were considered as very specific or mostly theoretical. In recent years, evidence of the real applications of steganography has increased. To emphasize that steganography is a trending topic, recent reports warned about an emerging cyber threat of the application of steganography methods by malicious software designers [Kaspersky Lab, 2017, McAfee Labs, 2017, Jarvis, 2018].

Applying steganography for computer malware operations and communication enables to:

- avoid the operations of common security mechanisms, such as antivirus, IDS systems and firewalls, etc. Network traffic or multimedia files with hidden data would be recognized as benign network communication or data exchange, so they would be allowed.
- make detection more difficult or even evade forensic malware analysis.

Network steganography methods can be characterized by [Mazurczyk et al., 2016]:

- steganographic bandwidth – rate of transmitted secret data per time unit.
- undetectability – the level of inability of adversaries to compromise the method.
- robustness – the possible level for modifying the carrier of the steganographic data without distortion to that data.

Steganographic methods are subject to a trade-off between them following a *magic triangle* introduced in [Fridrich, 1999].

This paper focuses on one of the basic types of network steganography – so called *timing network steganography*. This kind of steganography utilizes varying timings of sending consecutive packets in the network streams. It means that for nearly every network protocol such a method can be applied. The parties in the communication secretly establish the algorithm of how to decode the embedded hidden data from the timing features of the network packet streams.

One of the examples of using time delays in steganography is the *Lost Audio Packets Steganography* (LACK) algorithm [Mazurczyk and Szczypiorski, 2008]. It utilizes intentionally delayed packets of VoIP streams to hide the steganograms. In the case of an extended delay, the endpoints of the VoIP communication consider such packets as lost and discard them. These packets do not take part in the reconstruction of a voice signal, and the steganograms can be retrieved from their payload.

1.4 Aim and structure of this study

In this study we propose an effective method for detecting attacks using timing covert channels, such as the ones presented in Section 1.3. Our method is based on mobile agents that can observe network traffic from various perspectives. According to [Mazurczyk et al., 2019], such a method can be classified as a combination of the passive warden concept [Anderson and Petitcolas, 1998] with capabilities of the network-aware active wardens [Lewandowski et al., 2007] in the scope of:

- sensing the topology of the network by agents and Central Unit;
- implementing interface to router logic in agents;
- defining states of detection logic and utilizing them into new decisions;

Using a testing environment, with various ML algorithms, we show experiments related to detecting timing network steganography attacks. For various testing scenarios, using a proof-of-concept, we will prove that the suggested method is able to effectively detect such a network attack. The proposed method is directly derived from the Change Observation Theory, which we presented in Section 2. Next, in Section 3 we propose our detection method. In Section 4 we describe the experimental setup used to prove out hypothesis. The results from the experiments are presented in Section 5. The article will end with conclusions described in Section 6.

2 Change Observation Theory

2.1 Introduction

Change Observation Theory combines ideas around objects, their features, changes in these features and observations of these changes. It introduces the universal framework for the evaluation of any algorithm for anomaly detection, which is one of the main efforts of academic and industrial research on cybersecurity. Anomaly detection is related to change detection events for which some of them are indicators of emerging cyber threats. The topic is well-established in the literature, but still applications to real problems such as detecting cyber threats are sought. One of the first, comprehensive introductions to this topic is presented in [Basseville and Nikiforov, 1993]. It established a unified approach for the design and the performance analysis of the algorithms for solving change detection problems defined as the problem of detecting a change in the parameters of a static or dynamic stochastic system.

The book [Poor and Hadjiliadis, 2008] is another contribution to the field of change observation theory basics. Its authors for the first time brought together

results from several sources. They defined another unified treatment of several different approaches to the change detection problem. This section tries to summarize definitions and concepts of the Change Observation Theory, especially towards developing new anomaly detection algorithms for cybersecurity problems.

2.2 Definitions

In every object, features can be observed, which may be permanent, but some of them are under a process of change. The change is related to the transition of the feature into another. The observation of features is important if, and only if, information about the features' change can be utilized practically. It means that not all observations are equally important. Objects usually have many features, say N . We assume that the object has at least one feature ($N \geq 1$). An object without features cannot be the subject of observation, thus no change could be monitored.

In many cases it is impossible to be aware of all features of objects. The more complicated situation is when some features could become hidden whereas the others could appear. This aspect refers to the dynamism of features. Before any procedure of observation, the number M of features for practical measuring should be chosen, where $M \leq N$.

Let assume that the number is K , where $K \leq M \leq N$. In many cases, some of the features are not relevant to the observation of the change. The worst aspect would be the impossibility of measuring *the significance of the features*. Therefore, the initial phase of the problem could be stated as the effort to determine M number of features of an object and select a subset of size K of such features marked as the most significant for observation. Fig. 1a shows the presented idea of the object with four significant features (C_1, C_2, C_3, C_4), among which two are under observation (C_1, C_2).

Traditionally, observation is considered to be a time-oriented process. After distinguishing a feature, for example, C_1 , a change in its properties is observed. A single unit of measurement is referred to as the state of a feature at the moment of measuring. Throughout the observation process, many samples of such states are collected and compared with one another. The essence of the described Change Observation Theory is to break the dependency of the observation process within a time domain. According to this theory, the key relationships between features are these that could be isolated from properties changing in time. The first step is to create an object model, and then determine the method for detecting change. The model of the object can introduce some inaccuracy in the mapping, which means the study of the object model does not coincide with the study of the object; however, it can provide relevant information. Features in the model

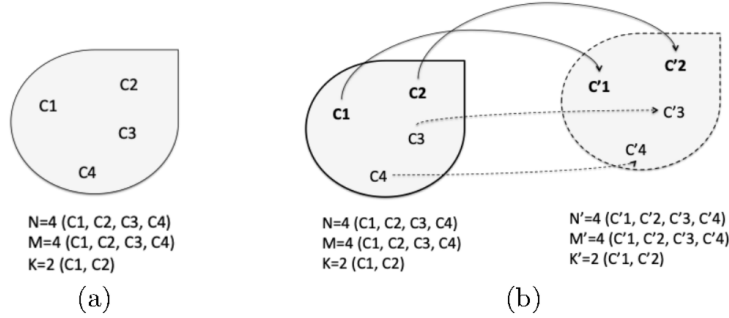


Figure 1: (a) Object with four features (C_1, C_2, C_3, C_4) among which two are under observation (C_1, C_2); (b) Model of object.

of the objects can be seen as relations. This idea is presented in Fig. 1b. For example, if C'_1 and C'_2 are integers, their relationship could be defined as:

- simple ratio of $\frac{C'_1}{C'_2}$;
- correlation between these two values.

The correlation dependence between the features C'_1 and C'_2 is characterized by the fact that the values of one feature are strictly mapped to the mean values of the second feature. The purpose of the correlation analysis is to say whether there are any dependencies between the variables and the strength of this dependency. The established models of objects with relations between features can be used as a standard model in further applications. The appearance of other relationships between the features will be considered a change.

One example of applications of the presented theory to design a real anomaly detection method is the MOVESTEG technique, as proposed in [Szczypiorski et al., 2015] and implemented in [Szczypiorski and Tyl, 2016]. It is based on the concept of a *moving observer* that can evaluate the measured features of the network flows to detect time-delay network attacks. The main result was to find the utilization of the observed facts in delay changes into the vector of the observer moving around the network. The aim of this was to discover the source of the attack. The results from the cited papers were the inspiration for the work in this paper to go further when designing a whole intrusion detection system based on Change Observation Theory.

3 Mobile agents – proposed approach

The main idea behind using mobile agents is to observe the network traffic from various points in the network, according to the Change Observation Theory, presented in Section 2. The agent will be responsible for collecting the data on

the traffic flowing through the router network interfaces and sending the preprocessed data to the central unit (CU) every T_{upload} time. The data preprocessing will include maintaining a network flow table and calculation of the statistics of the packet interarrival time values, e.g., by creating their histograms.

Two types of mobile agents are foreseen: *Base Agents* (BAs) – placed in the network nodes with the most intensive traffic, and *Flying Agents* (FAs) – sent temporarily to a network node by the CU or BA in case of a suspected anomaly, to perform a closer observation. Contrary to BAs, the FA's lifetime will be limited; when it has elapsed the FA instance will be deactivated, having obtained prior approval from the CU.

3.1 Observing time relationships between packets

To be able to capture anomalies in the packet interarrival times, a common approach is to analyze histograms of these values (see e.g., [Kind et al., 2009]), in which the frequency density for bin i , using Inverson bracket notation, is calculated as:

$$k_i = \frac{\sum_{j=0}^{N-1} [i \cdot \tau \leq t_j < (i+1) \cdot \tau]}{N} \quad (1)$$

where N denotes the number of packets within a certain time window, t denotes the time distance between the current packet j and the packet $j-1$, τ is the arbitrary set bin width.

However, our initial experiments revealed that such an approach did not allow the extraction of enough information to efficiently detect timing network steganography attacks. Therefore, we proposed that the mobile agents will calculate meta-histograms based on the original ones, using the following formula:

$$m_i = \begin{cases} k_c & i = 0 \\ k_{c-i} + k_{c+i} & i = \langle 1, M \rangle \\ k_{c-(i-M)} - k_{c+(i-M)} & i = \langle M+1, 2M \rangle \end{cases} \quad (2)$$

where c denotes the index of the gravity center of the original histogram and M denotes half the number of its bins. The proposed meta-histogram is supposed to capture in a better way the symmetry (or its lack) within the basic histogram.

A decision, whether a given meta-histogram represents a benign or a malicious network flow, can be made by a ML-based classifier. Prior to using it for detection purposes, the ML algorithm needs to be trained using labelled data.

If multiple network flows are analyzed, e.g., on a single router's interface, the ratio between positive decisions (*true positives* – TP and *false positives* – FP) of the classifier against the total number of the flows, can be treated as an anomaly metric d :

$$d = \frac{TP + FP}{all\ decisions} \quad (3)$$

This metric will be used later in this study.

3.2 Role of flying agents

When an FA is sent to a network router, it calculates the anomaly metric d , based on the algorithm proposed in Section 3.1, on each of the router's interfaces. If the d value on any of the router's interfaces is higher than in the node the FA came from (i.e., on the preceding router's interface), the FA continues to walk in the direction pointed by that interface, that is, towards the higher anomaly metrics. If the d value on all the router's interfaces is lower than the original value, the FA continues its walk in the random direction (different, however, from the agent's provenance).

Such a walk of the FA continues until the FA is **no longer** able to move, i.e., until it reaches a tree leaf (or gets to the last programmable router). It means that it is now likely to be close to the potential attacker. The FA continues to send its data to the CU. The CU, knowing that the FA is unable to walk any further, makes a decision based on the current d value. If it exceeds a certain d_{alarm} threshold, the CU detects a timing covert channel and performs the necessary actions. If the d_{alarm} threshold is not reached, the FA is deactivated.

3.3 Central unit

The central unit will be responsible for tracking the location of both types of agents (BAs and FAs), aggregating the data acquired from the mobile agents, and, last but not least, handling the main process of anomaly detection. The CU will implement the anomaly detection logic introduced in Sections 3.1 and 3.2. Its main role will consist of realization of the communication between the CU and the distributed agents. It will be in a major part hierarchical and realized in the following way:

- The CU allocates BAs to network nodes with the most intensive traffic.
- The agents observe the traffic, perform basic analyses and send the digested results to the CU.
- An FA can be sent to a network node close to a potential source of attack, following a request by the CU, or based on autonomous decision of a BA.
- The FA sends their observation results to the CU directly.
- The CU, based on the data acquired by the agents, may make a decision to trigger an alarm.

- When the T_{life} time passes, the FA asks the CU if it is allowed to deactivate. If the temporary observation is finished, the CU approves the request and removes the given FA from the registry.

3.4 Setting decision threshold

The proposed mechanism of activation and deactivation of FAs implies that the decision threshold of the classifiers needs to be properly set. In our case, higher false positives (FPs) can be tolerated, but false negatives (FNs) need to be low. Therefore we will prioritize setups yielding high recall scores.

This is a consequence of the fact that our system offers another layer of FP verification – deactivation of less effective agents. Therefore, spawning unnecessary agents is an acceptable cost of this approach. On the other hand, the number of FNs should be minimized, as missing an attack is highly undesirable.

4 Experimental setup

In this section, the setup of the tools for testing the scenarios and emulating steganographic attacks is presented. It consists of a network emulator and applications for network packet stream generation. Furthermore, the process of generating the dataset for evaluating the proposed method is described. It considers the format of the collected raw samples of the network flows.

4.1 Testing scenarios

To test the proposed method we used a network consisting of 28 nodes, serving as routers, and 16 client machines, representing both clients and servers, with the exception of one that acts as the CU node.

Figure 2 shows a sample network topology. Both servers and hosts were used to generate network traffic considered as legitimate for the purpose of the tests and attackers #1 and #2 were used to generate illegitimate traffic.

We tested the effectiveness of the method proposed in the two separate scenarios involving either Attacker #1 or Attacker #2. They both attacked the same Victim machine (Server #2), but the attacks did not happen simultaneously. Those attacks used the following traffic path:

1. Attacker #1 \rightarrow R13 \rightarrow R12 \rightarrow R8 \rightarrow R1 \rightarrow R15 \rightarrow R16 \rightarrow R18 \rightarrow Victim
2. Attacker #2 \rightarrow R28 \rightarrow R26 \rightarrow R22 \rightarrow R15 \rightarrow R16 \rightarrow R18 \rightarrow Victim

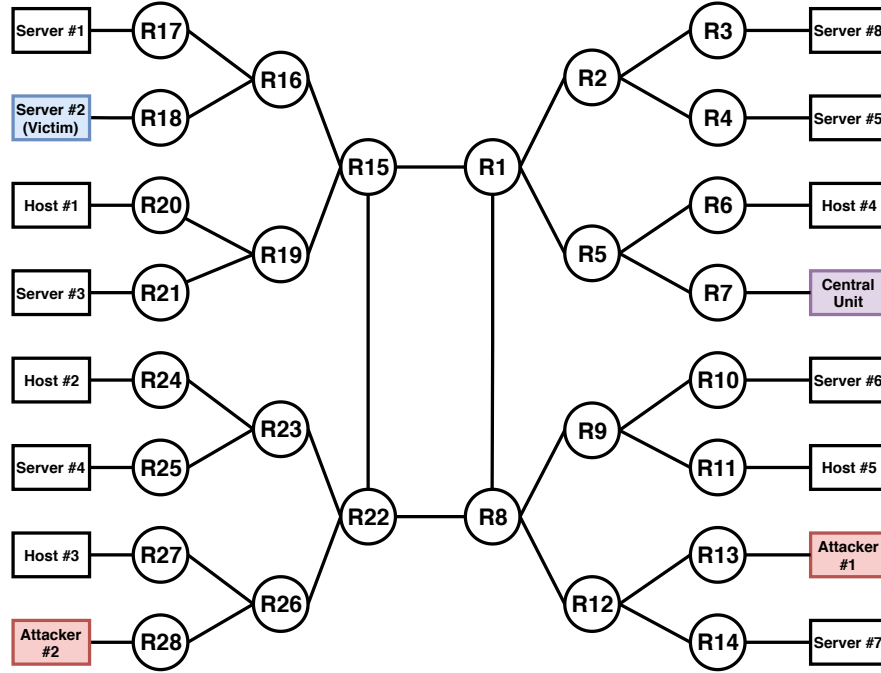


Figure 2: Sample network topology generated during emulations (blue denotes victim of attack, red – two separate attackers).

4.2 Emulating steganographic attacks

To emulate attack methods we used an open-source platform called Distributed Internet Traffic Generator (D-ITG) [Botta et al., 2012]. It was chosen due to its potential to define packet interarrival times. We emulated the steganography attacks with various percentages of traffic utilized for steganographic transmission, starting from 33% up to 100%. To execute this method we created a text file that consisted of values of interarrival times between packets. In Table 1 we present exemplary snippets of such files. They show values needed to create timing covert channels with an average interarrival time of 100ms. The value *null* means no steganographic bit assigned.

Mobile agents monitored the network traffic on the routers they were installed on and sent the data every minute to the CU. Apart from the meta-histogram data (see Section 3.1), the agents sent additional information, such as the timestamp, router ID, interface ID, source and destination addresses and ports etc.

The data, including both benign traffic and the malicious traffic were collected separately in two disjoint simulations with two attack scenarios in each of them, with 0.5% malicious traffic in the first simulation and 5.0% in the second

Table 1: Exemplary snippets of text files used by D-ITG to create steganography with different bitrate. Values in brackets denote mapping of time interval to steganographic bit.

Steganography 100%	Steganography 66%	Steganography 33%
50 [0]	50 [0]	50 [0]
150 [1]	100 [null]	100 [null]
50 [0]	150 [1]	100 [null]
150 [1]	50 [0]	100 [null]
50 [0]	100 [null]	100 [null]
150 [1]	150 [1]	150 [1]

Table 2: Datasets collected and utilized during experiments in this study.

Simulation	Attack path	Dataset tag	Utilization of datasets
Simulation 1	R13 - R18	DATASET 1	80% used for choosing hyperparameters (GridSearch with CV) and training classifiers 20% of used for testing classifiers
	R28 - R13	DATASET 2	80% used for training classifiers 20% used for testing
Simulation 2	R13 - R18	DATASET 3	100% used for: - final testing of classifiers - testing the proposed attack detection method
	R28 - R13	DATASET 4	

one. This resulted in four datasets in total (see Table 2 for summary). DATASETS 1 and 2 were used for the training and tuning of the ML-based classifiers. DATASETS 3 and 4 were used for the final testing of the trained classifiers and to verify if the proposed attack detection method was effective.

4.3 Machine learning algorithms

Several ML algorithms were considered as classifiers in the proposed method. After initial tests, the chosen MLs were SVM, RANDOM FOREST, BERNOULLI NAÏVE BAYES, GAUSSIAN NAÏVE BAYES and MLP, all from the supervised learning algorithm group. They were implemented using the Python `scikit-learn` library [Pedregosa et al., 2011].

Special attention was paid to selecting a data subset from the training data to train the ML algorithms, which is usually critical to the effectiveness of a ML-based classifier. Several options were considered, such as choosing the data that

were acquired in the direct vicinity of the attacker, remotely from the attacker, or as a mixture of the above cases.

5 Experiments, results and their discussion

5.1 Evaluation metrics

Selecting the best set of hyperparameters for each ML-based classifier, training the models and testing the classification performance were based on standard metrics:

- Receiver Operator Characteristics (ROC) curve and area under ROC curve (AUC). The ROC curve shows the relationship of the TP rate (TPR) against FP rate (FPR) for various decision thresholds. An AUC value of 1 means a perfect classification; the lower the AUC, the more misclassifications that can be expected. An AUC exceeding 80% is often considered as acceptable [Tape, Thomas G., 2017].
- Equal Error Rate (EER) yields the error value for which the classifier has the same probability of misclassifying a positive as well as a negative sample.
- accuracy – the ratio of the number of correct detections to all detections.
- attack recall (TPR for attack detection) – the ratio of correctly detected attacks referred to all attack samples.
- precision – informs how often the classification was correct.
- F_1 score – harmonic mean of the precision and recall.

5.2 Training and tuning ML-based classifiers

To tune the selected classifiers, the methodology of data science analysis was applied. As the initial experiments revealed that the detection algorithm worked best if trained on data acquired close to the attacker, the data from the nodes adjoining the attackers from DATASET 1 and DATASET 2 were selected for training purposes.

The hyperparameters of the classifiers were tuned using a GRID SEARCH algorithm with 10-fold cross validation (CV) procedure (for the SVM, the CV order was set to 3 as a higher value caused overfitting). This step ensured that all the algorithms worked in the best condition possible for the given data. GAUSSIANNB was excluded from this process as it did not have parameters that could be tuned. The best hyperparameters for each classifier were chosen according to the mean scores of the metrics in the following order of importance:

Table 3: Evaluation metric results from hyperparameters optimization process of GRID SEARCH with cross validation.

ML model	Mean					Hyperparameters results
	AUC	Accuracy	Recall	Precision	F_1	
BERNOULLINB	0.9036	0.9416	0.7550	0.4128	0.5337	alpha: 0, binarize: 0, fit_prior: True
GAUSSIANNB	0.7500	0.5409	0.9363	0.0829	0.1534	default parameters
SVM	0.9130	0.9562	0.1445	0.4108	0.1990	C: 1, decision_function_shape: ovo, gamma: 2, kernel: poly, shrinking: False
RANDOMFOREST	0.9776	0.9818	0.6307	0.9355	0.7507	bootstrap: True, criterion: entropy, max_depth: 48, max_features: log2, min_samples_leaf: 2, min_samples_split: 2, n_estimators: 36
MLP	0.9681	0.9812	0.7385	0.8196	0.7753	activation: tanh, alpha: 1e-05, hidden_layer_sizes: (100, 40), learning_rate: adaptive, max_iter: 1000, solver: adam

Table 4: Classification results for testing part of DATASET 1 for various ML algorithms trained on data from R13.

ML model	AUC	Accuracy	Recall	Precision	F_1
RANDOMFOREST	0.8652	0.9834	0.7361	0.8548	0.7910
BERNOULLINB	0.8493	0.9403	0.7500	0.3941	0.5167
GAUSSIANNB	0.7226	0.5325	0.9305	0.0785	0.1448
SVM	0.8404	0.9739	0.6944	0.6944	0.6944
MLP	0.8591	0.9716	0.7361	0.6463	0.6883

1. AUC, as most universal metrics.
2. Recall, as we wanted to minimize number of FNs.
3. F_1 and precision, which were less important in our case.

The best hyperparameters identified for the considered ML classifiers are presented in Table 3. The classification results for these classifiers are presented in Table 4 for the testing part of DATASET 1 for node R13. Figure 3 shows a

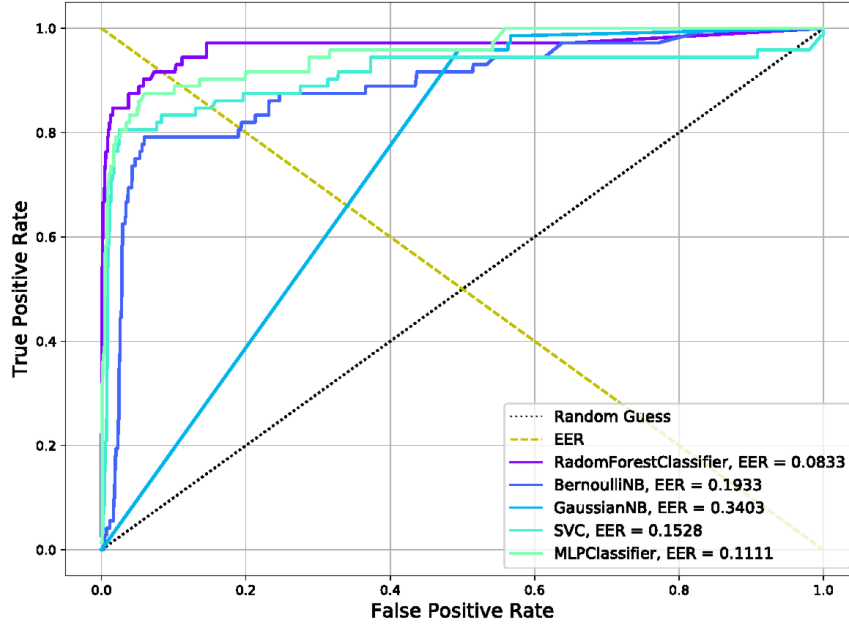


Figure 3: ROC curves calculated on testing part of DATASET 1 data for classifiers trained on R13 data from training part of DATASET 1.

sample ROC curve with EERs for the testing of classifiers for models trained on the training part of DATASET 1 for node R13.

The best detection was achieved for the MLP, Random Forrest, SVM and Bernoulli Naïve Bayes classifiers – they all yielded high values of AUC (above 80%), accuracy (above 90%) and recall (70-75%). It is noteworthy that the results for the testing data are only slightly inferior to those for the training data, still yielding high recognition scores for these classifiers. In contrast, the Gaussian Naïve Bayes algorithm performed worse, with ca. 72% AUC and accuracy of 55% only.

5.3 Results for detection of timing network steganography attacks by mobile agents

We analyzed how the trained classifiers would perform if used for detecting timing network steganography attacks by mobile agents. We compared two sets of models: trained on R13 data and trained on R28 data (i.e., using DATASET 1 and DATASET 2, respectively). Therefore we considered four cases:

1. Attack path from R13 to R18 with model trained on DATASET 1.

Table 5: Anomaly metric d along attack path and on other routers in cases 1 and 2 with ML classifiers: (A) BERNOULLINB, (B) GAUSSIANNB, (C) SVM, (D) RANDOMFOREST and (E) MLP.

CASE	Model	Metrics d along attack path							Metrics d on other routers						
		R13	R12	R8	R1	R15	R16	R18	R14	R22	R9	R2	R5	R19	R17
1	(A)	32.42	15.30	13.62	11.65	10.64	19.94	29.79	9.75	6.17	6.03	5.06	5.16	2.73	0.00
	(B)	45.94	41.36	40.33	36.85	36.40	33.75	31.39	40.72	37.31	42.50	37.05	42.53	37.96	0.00
	(C)	1.70	0.24	0.20	0.32	0.54	0.42	0.38	0.24	0.30	0.35	0.46	0.20	1.01	0.00
	(D)	6.86	1.46	0.24	0.09	0.07	0.22	0.19	0.02	0.08	0.09	0.01	0.00	0.07	0.00
	(E)	3.50	1.18	0.47	0.51	0.63	0.74	0.58	0.59	0.53	0.58	0.70	0.41	1.14	0.00
2	(A)	34.12	15.56	13.85	12.62	10.92	20.78	31.66	9.42	6.12	6.033	5.14	5.16	2.73	0.00
	(B)	45.94	41.36	40.33	36.85	36.40	33.75	31.39	40.72	37.31	42.50	37.05	42.53	37.96	0.00
	(C)	6.25	1.08	0.64	0.78	1.04	0.95	0.86	0.88	0.72	1.08	1.09	0.49	2.00	0.00
	(D)	13.50	3.84	1.80	1.06	1.09	1.74	2.07	0.38	0.58	0.60	0.56	0.21	0.42	0.00
	(E)	10.87	3.54	1.62	1.22	1.15	1.56	1.30	1.32	1.13	1.30	1.25	0.74	1.30	0.00

2. Attack path from R13 to R18 with model trained on DATASET 2.
3. Attack path from R28 to R18 with model trained on DATASET 2.
4. Attack path from R28 to R18 with model trained on DATASET 1.

Tables 5 and 6 show the anomaly metrics d for various classifiers for these cases. Figure 4a-4f display these results graphically – as an example for case 3 and the RANDOMFOREST classifier, which was among the best performing ones. The red color shows the nodes where the mobile agents yielded the highest results. The yellow color marks nodes where the FAs were spawned according to the proposed algorithm. The green nodes denote routers where the FAs would be deactivated, as the anomaly metric d would be lower than on the node from which the FA was sent. It can be seen from the last figure that red nodes were perfectly aligned with the path leading to the attacker.

5.4 Discussion

The first step – selecting and tuning the ML-based classifiers – was very important for the correct functioning of the attack detection algorithm, as the proposed anomaly metric d is based on the classifier’s positive decisions (both TPs and FPs). Therefore, they need to be as accurate as possible. RANDOMFOREST and MLP turned out to perform best in our case. NAÏVE BAYES algorithms gave the worst results and would not work properly in our solution. The other models presented some drawbacks, e.g., too low a number of positive detections (SVM) which could result in missing an attack. The GAUSSIANNB classifier should also be avoided, as its low accuracy would spawn too many unnecessary FAs. Even though our proposed system is able to regulate this by the mechanism of the

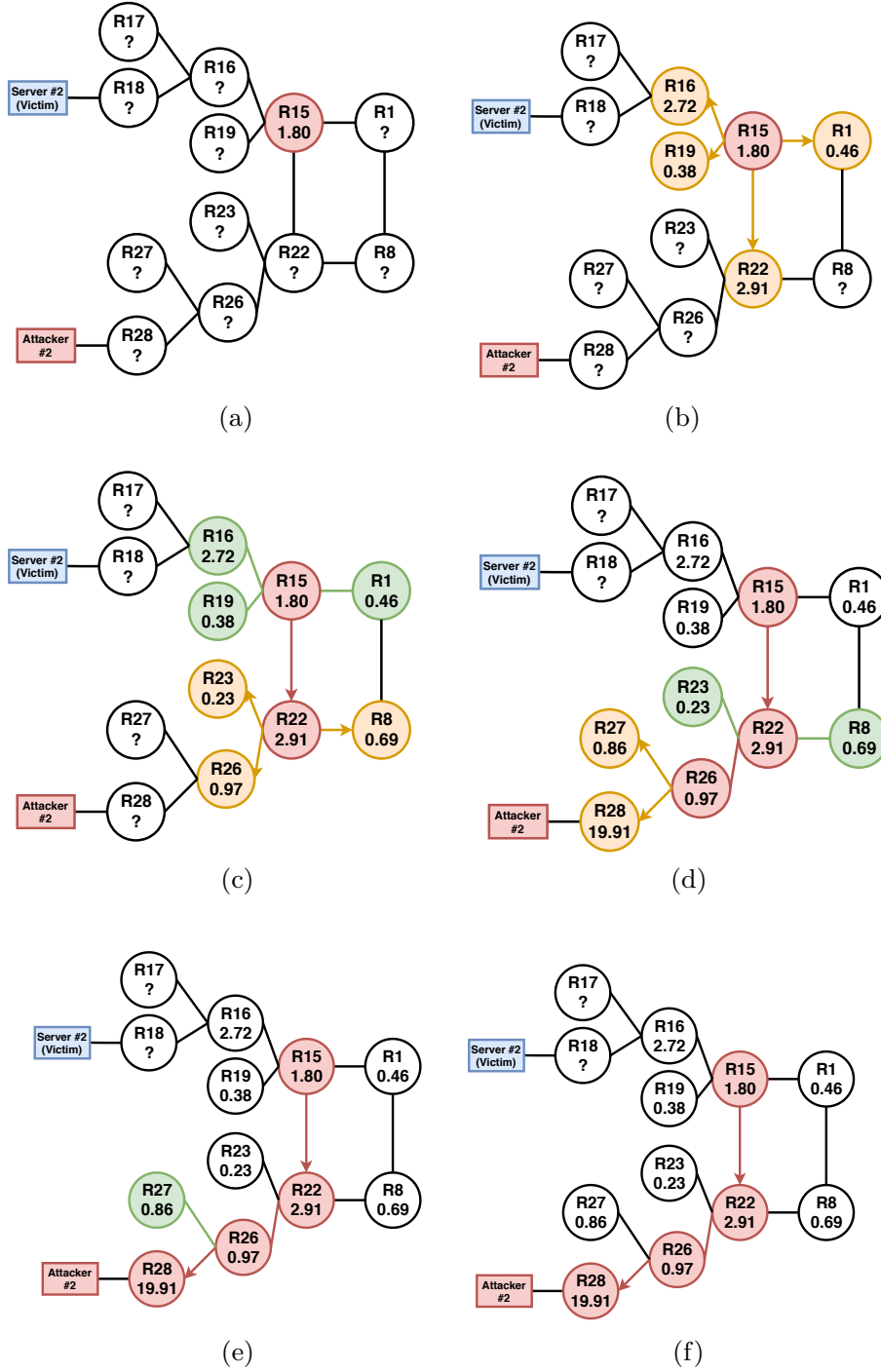


Figure 4: Anomaly metric results from the attack by R28 to R18, with RAN-
DOMFOREST model trained on DATASET 2. (a)-(f) present consecutive phases
of BAs and FAs cooperation to identify the attacker. Red nodes mean that BA
or FA yielded a high value for metric d and raised the attention. Yellow node
means spawning of FA. Green node means that a FA ends its life.

Table 6: Anomaly metric d along attack path and on other routers in cases 3 and 4 with ML classifiers: (A) BERTOUILLNB, (B) GAUSSIANNB, (C) SVM, (D) RANDOMFOREST and (E) MLP;

CASE	Model	Metrics d along attack path						Metrics d on other routers					
		R28	R26	R22	R15	R16	R18	R27	R23	R8	R1	R19	R17
3	(A)	23.60	19.92	11.80	8.79	17.14	24.96	7.40	4.95	6.87	5.29	2.70	0.00
	(B)	49.81	47.53	41.55	40.51	41.28	41.44	43.48	40.49	45.21	39.77	42.44	0.00
	(C)	12.65	3.49	0.82	1.22	1.14	1.05	1.53	0.81	0.86	0.92	2.28	0.00
	(D)	16.91	9.77	2.91	1.80	2.72	2.78	0.86	0.23	0.69	0.46	0.38	0.00
	(E)	16.90	9.19	2.70	1.88	2.29	2.16	1.52	0.77	1.28	1.03	1.46	0.00
4	(A)	23.26	19.61	11.61	8.67	16.69	24.18	7.32	4.95	6.93	5.29	2.69	0.00
	(B)	50.00	47.88	43.79	41.62	41.36	40.83	44.59	44.68	43.82	40.39	43.18	0.00
	(C)	3.86	0.46	0.30	0.63	0.50	0.52	0.95	0.34	0.34	0.43	1.21	0.00
	(D)	12.43	5.43	0.76	0.25	0.26	0.31	0.12	0.00	0.07	0.05	0.05	0.00
	(E)	9.43	4.34	0.71	0.82	0.81	0.77	1.16	0.50	0.55	0.57	1.31	0.00

programmed death of FAs, in this case it would introduce too high a load on the system. In addition, too high a number of false detections could lead to incorrect CU decisions and a FA could be sent in a wrong direction, and, as a consequence, it would not be able find the source of the attack.

Analyses of Tables 5 and 6 reveal that the anomaly metric d increases the closer the agent is to the source. Neighboring nodes to the ones along the attack path have this value significantly lower, so by using this metric the CU should easily send the FAs towards the growing anomaly, i.e., towards the attacker. It is noteworthy, that it is not required for the FAs in the nodes outside of the attack path to return $d = 0$ (this would be the case for perfect classifiers) – it is enough that the FAs are send in the direction of the growing d (which, by the way, resembles behavior of gradient methods). This shows a great advantage of the proposed method: even using non-perfect classifiers, together with the mechanism of moving agents, it can create a highly effective method of network attack detection.

Figure 5 presents the anomaly metric d as a function of distance to the source of an attack (RANDOMFOREST classifier was used). The x-axis represents the distance (in nodes) from the current node to the attacker (value 1 denotes the node closest to the attacker). It shows that in all cases d is the highest for the shortest distance to the attacker; this value decreases with the growing distance. However, it can be observed that it increases slightly again the closer to the victim. It is mainly a result of less traffic going through these nodes what decreases the denominator in the d formula. It should not, however, negatively impact neither the quality nor the effectiveness of the method. It additionally proves that the model can be more universal. It means that creating models

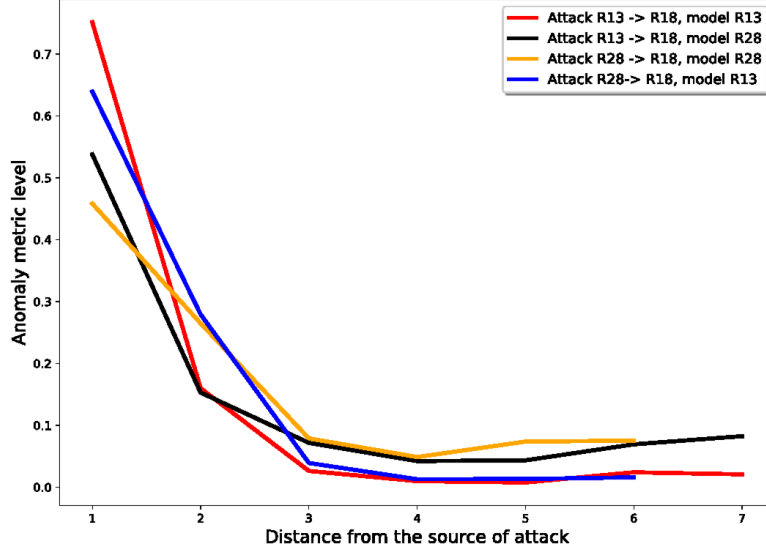


Figure 5: Distribution of anomaly metric value as function of distance from source of attack.

from data close to the possible attackers (i.e., from the nodes to which the end devices are connected) will provide the method with a proper model that should work regardless of where the attacker is placed or how the network has changed.

As probably there would be several BAs in the network, the CU should have even more information to determine if the network is under an attack, and, if it is suspected, from which node to start spawning the FAs. If the d value is high in several places in the network, the CU will need to decide if this is caused by a single attack with a single source, a single attack with multiple sources or multiple, unrelated to one another, network attacks.

6 Conclusions and future work

This paper presented results on applying the Change Observation Theory and the moving observer concept to a new kind of intrusion detection system. It is characterized by a distributed character of observation in cyberspace, cooperation between observing entities and using their mobility to identify the sources of anomalies and attacks.

We applied the proposed method to detecting attacks using timing network steganography. Therefore, as feature vectors we used meta-histograms with statistics of packet interarrival time. We trained various classifiers and proposed

an anomaly metrics based on their classification decisions. We showed, using a proof-of-concept, that this method allowed the mobile agents to move towards the source of an attack, and therefore the proposed method is effective in detecting timing network steganography. However, to apply the method to other types of attacks, the feature vector should be extended.

Our future efforts will be focused on improvements in emulating various network topologies and attacks other than those using timing network steganography. The modeling and data analysis will also be strengthened by (a) introducing new features, (b) selecting the most important ones, (b) optimizations of data size and (d) experimenting with other ML-based algorithms. In further research we will also try to take into consideration the fact that a node is visited by a flying agent several times, without a detection of an attack. We are going to investigate if this may be an indicator of a potential security breach in the given node.

The concept of using classifiers (even non-ideal ones) with the decision threshold biased towards increased TP rates, but combined with the deactivation procedure for the mobile agents, appears to be highly novel. It proved to be very promising and will continue to be researched in the future.

Acknowledgements

This work was supported by the National Centre for Research and Development (agreement No. CYBERSECIDENT/369532/I/NCBR/2017) within the CyberSecIdent Programme. The authors wish to thank Piotr Januszewski, MSc, for his contribution to setting up the experiments.

References

- [Ahmed et al., 2016] Ahmed, M., Mahmood, A. N., and Hu, J. (2016). A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, 60:19–31.
- [Anderson and Petitcolas, 1998] Anderson, R. J. and Petitcolas, F. A. P. (1998). On the limits of steganography. *IEEE Journal on Selected Areas in Communications*, 16(4):474–481.
- [Basseville and Nikiforov, 1993] Basseville, M. and Nikiforov, I. V. (1993). *Detection of Abrupt Changes - Theory and Application*. Prentice Hall, Inc. - <http://people.irisa.fr/Michele.Basseville/kniga/>.
- [Bhuyan et al., 2014] Bhuyan, M. H., Bhattacharyya, D. K., and Kalita, J. K. (2014). Network anomaly detection: methods, systems and tools. *IEEE communications surveys & tutorials*, 16(1):303–336.
- [Botta et al., 2012] Botta, A., Dainotti, A., and Pescapè, A. (2012). A tool for the generation of realistic network workload for emerging networking scenarios. *Computer Networks*, 56(15):3531 – 3547.
- [Buczak and Guven, 2016] Buczak, A. L. and Guven, E. (2016). A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials*, 18(2):1153–1176.

- [Casas et al., 2012] Casas, P., Mazel, J., and Owezarski, P. (2012). Unsupervised network intrusion detection systems: Detecting the unknown without knowledge. *Computer Communications*, 35(7):772–783.
- [Fink et al., 2014] Fink, G. A., Haack, J. N., McKinnon, A. D., and Fulp, E. W. (2014). Defense on the move: Ant-based cyber defense. *IEEE Security Privacy*, 12(2):36–43.
- [Fridrich, 1999] Fridrich, J. (1999). Applications of data hiding in digital images. In *ISSPA '99. Proceedings of the Fifth International Symposium on Signal Processing and its Applications (IEEE Cat. No.99EX359)*, volume 1, pages 9 vol.1–.
- [Haack et al., 2011] Haack, J. N., Fink, G. A., Maiden, W. M., McKinnon, A. D., Templeton, S. J., and Fulp, E. W. (2011). Ant-based cyber security. In *2011 Eighth International Conference on Information Technology: New Generations*, pages 918–926.
- [Janicki et al., 2015] Janicki, A., Mazurczyk, W., and Szczypiorski, K. (2015). On the undetectability of transcoding steganography. *Security and Communication Networks*, 8(18):3804–3814.
- [Jarvis, 2018] Jarvis, J. (2018). Steganography: Combatting Threats Hiding in Plain Sight. <https://www.fortinet.com/blog/threat-research/steganography--combatting-threats-hiding-in-plain-sight.html>. [Online; accessed 10.01.2019].
- [Kaspersky Lab, 2017] Kaspersky Lab (2017). Kaspersky Lab Identifies Worrying Trend in Hackers Using Steganography. https://usa.kaspersky.com/about/press-releases/2017_kaspersky-lab-identifies-worrying-trend-in-hackers-using-steganography. [Online; accessed 10.01.2019].
- [Kind et al., 2009] Kind, A., Stoecklin, M. P., and Dimitropoulos, X. (2009). Histogram-based traffic anomaly detection. *IEEE Transactions on Network and Service Management*, 6(2):110–121.
- [Kott, 2018] Kott, A. (2018). Intelligent autonomous agents are key to cyber defense of the future army networks. *The Cyber Defense Review*, 3(3):57–70.
- [Kwon et al., 2017] Kwon, D., Kim, H., Kim, J., Suh, S. C., Kim, I., and Kim, K. J. (2017). A survey of deep learning-based network anomaly detection. *Cluster Computing*.
- [Lewandowski et al., 2007] Lewandowski, G., Lucena, N. B., and Chapin, S. J. (2007). Analyzing network-aware active wardens in ipv6. In Camenisch, J. L., Collberg, C. S., Johnson, N. F., and Sallee, P., editors, *Information Hiding*, pages 58–77, Berlin, Heidelberg. Springer Berlin Heidelberg.
- [Lubacz et al., 2014] Lubacz, J., Mazurczyk, W., and Szczypiorski, K. (2014). Principles and overview of network steganography. *IEEE Communications Magazine*, 52(5):225–229.
- [Mazurczyk et al., 2012] Mazurczyk, W., Cabaj, K., and Szczypiorski, K. (2012). What are suspicious VoIP delays? *Multimedia Tools and Applications*, 57(1):109–126.
- [Mazurczyk and Szczypiorski, 2008] Mazurczyk, W. and Szczypiorski, K. (2008). Steganography of VoIP streams. In Meersman, R. and Tari, Z., editors, *On the Move to Meaningful Internet Systems: OTM 2008*, pages 1001–1018, Berlin, Heidelberg. Springer Berlin Heidelberg.
- [Mazurczyk et al., 2019] Mazurczyk, W., Wendzel, S., Chourib, M., and Keller, J. (2019). Countering adaptive network covert communication with dynamic wardens. *Future Generation Computer Systems*, 94:712 – 725.
- [Mazurczyk et al., 2016] Mazurczyk, W., Wendzel, S., Zander, S., Houmansadr, A., and Szczypiorski, K. (2016). *Information Hiding in Communication Networks: Fundamentals, Mechanisms, Applications, and Countermeasures*. Wiley-IEEE Press, 1st edition.
- [McAfee Labs, 2017] McAfee Labs (2017). McAfee Labs Threats Report. <https://www.mcafee.com/enterprise/en-us/assets/reports/>

- `rp-quarterly-threats-jun-2017.pdf`. [Online; accessed 10.01.2019].
- [Miller et al., 2011] Miller, Z., Deitrick, W., and Hu, W. (2011). Anomalous network packet detection using data stream mining. *Journal of Information Security*, 2(04):158.
- [Murdoch and Lewis, 2005] Murdoch, S. J. and Lewis, S. (2005). Embedding covert channels into tcp/ip. In *International Workshop on Information Hiding*, pages 247–261. Springer.
- [Pedregosa et al., 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.
- [Poor and Hadjiliadis, 2008] Poor, H. V. and Hadjiliadis, O. (2008). *Quickest Detection*. Cambridge University Press.
- [Su, 2011] Su, M.-Y. (2011). Real-time anomaly detection systems for denial-of-service attacks by weighted k-nearest-neighbor classifiers. *Expert Systems with Applications*, 38(4):3492–3498.
- [Szczypiorski et al., 2015] Szczypiorski, K., Janicki, A., and Wendzel, S. (2015). The Good, The Bad And The Ugly: Evaluation of Wi-Fi Steganography. *Journal of Communications*, 10(10):749–750.
- [Szczypiorski and Tyl, 2016] Szczypiorski, K. and Tyl, T. (2016). MoveSteg: A Method of Network Steganography Detection. *International Journal of Electronics and Telecommunications*, 62(4):335–341.
- [Tape, Thomas G., 2017] Tape, Thomas G. (2017). The Area Under an ROC Curve. <http://gim.unmc.edu/dxtests/roc3.htm>. [Online; accessed 10.01.2019].
- [Tsai et al., 2009] Tsai, C.-F., Hsu, Y.-F., Lin, C.-Y., and Lin, W.-Y. (2009). Intrusion detection by machine learning: A review. *Expert Systems with Applications*, 36(10):11994–12000.

Article

Dataset Generation for Development of Multi-Node Cyber Threat Detection Systems

Jędrzej Bieniasz  and Krzysztof Szczypiorski 

Institute of Telecommunications, Faculty of Electronics and Information Technology, Warsaw University of Technology, 00-661 Warsaw, Poland; k.szczypiorski@tele.pw.edu.pl

* Correspondence: J.Bieniasz@tele.pw.edu.pl

Abstract: This paper presents a new approach to generate datasets for cyber threat research in a multi-node system. For this purpose, the proof-of-concept of such a system is implemented. The system will be used to collect unique datasets with examples of information hiding techniques. These techniques are not present in publicly available cyber threat detection datasets, while the cyber threats that use them represent an emerging cyber defense challenge worldwide. The network data were collected thanks to the development of a dedicated application that automatically generates random network configurations and runs scenarios of information hiding techniques. The generated datasets were used in the data-driven research workflow for cyber threat detection, including the generation of data representations (network flows), feature selection based on correlations, data augmentation of training datasets, and preparation of machine learning classifiers based on Random Forest and Multilayer Perceptron architectures. The presented results show the usefulness and correctness of the design process to detect information hiding techniques. The challenges and research directions to detect cyber deception methods are discussed in general in the paper.

Keywords: cybersecurity; data science; machine learning; datasets; cyber threats modeling; multi-agent systems; cyber deception



Citation: Bieniasz, J.; Szczypiorski, K. Dataset Generation for Development of Multi-Node Cyber Threat Detection Systems. *Electronics* **2021**, *10*, 2711. <https://doi.org/10.3390/electronics10212711>

Academic Editor: Qusay H. Mahmoud

Received: 28 September 2021

Accepted: 3 November 2021

Published: 7 November 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In recent years, threats in cyberspace have evolved into well-organized, long-term, and resource-intensive intrusion campaigns known as Advanced Persistent Threats. As a result, there is a need to increase research into and the implementation of new cyber defense solutions, methods, operations, and procedures. Cybersecurity research activity is very broad, but it could be summarized by offering new developments and solutions for new use cases for each function of the NIST Cybersecurity Framework (CSF) [1]. Examples of a tailored solution that has been developed based on a new use case for cybersecurity would be physical unclonable functions [2]. The need for a new secure identification and authentication method was driven by the restricted requirements of cyber-physical systems. The resulting concept offers low computational cost and resource requirements, whereas Identify and Protect functions are easily provided for such systems. The same strategy for research in cyber threat detection is followed in this paper. One of the most important scientific and technological areas that are increasingly being used for cyber defense is data science and data-driven methods. The following list summarizes the five areas of work around data science in cybersecurity:

1. Modeling cyber threats to leverage across the data pipeline, from observation to flaw detection to actionable cyber threat data—for example modeling techniques: NIST Incident Response [3], Cyber Kill Chain [4], and MITRE ATT&CK [5].
2. Applying new models of cyber threats and setting up platforms to simulate them in near-production environments.
3. Elaboration of detection algorithms that are technically feasible in modern networks and systems.

4. Sharing the collected information on cyber threats and applying this information in technical solutions.
5. Accelerating the decision-making process within cybersecurity teams and departments together with the company's decision-makers.

For a more detailed overview of the challenges of data-driven cyber threats and intrusion detection, see [6]. This paper presents efforts to create an end-to-end process that combines aspects 1, 2, and 3. This is possible by extending the established approach for cyber threat detection systems [7], mainly realized by Network and Host Intrusion Detection Systems (NIDS, HIDS), to Multi-Node Cyber Threat Detection (MNCTD) systems. The cyber defense action matrix [4] shows that classical NIDS or HIDS can be used individually in four out of seven phases of the Cyber Kill Chain—weaponization, exploitation, installation, and C2 (Command and Control). MNCTD goes further and proposes the combination of the detection capabilities of all steps into a cyber threat detection system that focuses on network communications.

Any research project on such models, algorithms, and systems suffers from the availability of the right data. The recognized problem of availability of specific datasets for the particular research hypothesis and preparation of appropriate datasets for cyber threat detection is a critical challenge [8]. In the first part, this paper summarizes the current state of datasets for cyber security research available in academia and industry. It then proposes an approach to create specific datasets for hybrid cyber threat detection systems research, as such datasets are scarcely available in the public domain. Most of these available datasets focus on network attacks, such as Distributed Denial of Service (DDoS), SSH Brute Force, or botnet communication over open text protocols such as HTTP or IRC. Modern cyber threat modeling shifts thinking to identify threat phases (Cyber Kill Chain) or tactics realized through various techniques (MITRE ATT&CK) to block a threat as early as possible. Developing new solutions for cyber defense is about defining the aspects of the threat using the chosen modeling method and creating certain observable indicators that can be analyzed by detection algorithms. Such an approach could provide the desired ability to block and counter cyber threat campaigns as soon as indicators of threat are detected. Another novelty of this paper is the emphasis on the increasing importance of detecting information concealment techniques used in cyber attacks, especially in APT campaigns. One of the most important reports on the rise of steganomware was the June 2017 McAfee report [9]. In it, steganography was identified as the emerging element used in new malware campaigns. Information hiding techniques can be used at any stage of a cyber threat campaign, but the focus is on methods that work with communication activities over networks:

- Delivery and C2 Phase designated by Cyber Kill Chain methodology.
- Defense Evasion, Exfiltration, and C2 Tactics classified by MITRE ATT&CK methodology.

This paper presents the possibility of preparing datasets with information hiding techniques to develop the concept of a Multi-Node Cyber Threat Detection platform. The created multi-agent system for collecting network packet traces was applied in the automatically generated environment of network nodes and with the random setup of malicious pairs of hosts (sender-receiver) per experimental run. Then, the collected sample datasets were used in the data science workflow for cyber threat detection. The classical pipeline of a data science experiment includes data cleaning, feature selection, under- or over-selection of rare class examples, and development of the default solution for classification problems.

The structure of the paper is as follows:

- Section 2 briefly presents the available datasets and the systematic approach to evaluating self-generated datasets. It builds the context for the need for the research in this paper:

- Generating datasets for cyber threat detection research in the domain of information hiding techniques applied by modern malware and malicious cyber operations like APTs.
- Establish the possibility to generate these datasets in different and randomized networking environments with a varying set of sources and destinations for the simulated cyber attacks.
- Section 3 presents the methodology used to establish the framework for end-to-end dataset generation for cyber threat detection research. It follows the context of the research established in Section 2. This section covers the concept of the system for capturing network traffic in a multi-node setup, simulation of benign and malicious network flows and scenarios for generating the final datasets, and a simple methodology for generating datasets.
- Section 4 shows examples of data science experiments enabled by the generated data. This part presents the empirical evaluation of the datasets generated by the methods introduced in Section 3.
- Section 5 concludes the paper with a summary of the results and further research directions that could be based on this paper.

Contributions of the Paper

The main contributions of the paper are:

- An approach to collect datasets for cyber threat detection research in a multi-node setup using the developed agent system. This contribution goes far beyond the state-of-the-art presented in Section 2.3. The majority of the available datasets are focused on providing indicators for simulated cyber attacks from single endpoints like central collectors, whereas this research tackles multi-node cyber data collection to follow the cyber attack path of execution.
- Application of the information hiding techniques in communication networks [10] to research cyber threats as an emerging problem in cyberspace. The paper shows how to generate network data streams using information hiding techniques. This is a key effect, as most of the state-of-the-art datasets presented in Section 2.3 include the classic types of cyber attacks only with no covert communication samples. The introduction of this paper and Section 2.4 show the increase in malware applying information hiding techniques for Command and Control channels, to exfiltrate data or to persistently maintain the presence in the compromised environments. It means that any research into cyber threat detection methods in the area of steganography used in malicious operations has never been as important.
- Development of an automated and randomized tool for setting up network configurations (nodes and links) when performing simulations of network communication scenarios. According to the state-of-the-art cyber data collection environments of the datasets presented in Section 2.3 they were mostly configured once with the chosen sources and destinations of cyber attacks. The contribution of this paper offers a solution to mitigate the biases in datasets related to the shape and topology of the environment in which they were collected.
- The execution of reference cyber threat detection experiments on the collected datasets. Most of the state-of-the-art research papers related to datasets included in Section 2.3 present the datasets and collection process. This paper contributes to the approach applied by the authors where the collected datasets were evaluated to be feasible in data-driven cyber threat detection workflows.

2. Related Work

2.1. Multi-Node Cyber Defense Solutions

The systems that could be built upon the results of this paper combine the idea of network intrusion detection systems with the concept of multi-agent systems into a multi-node cyber threat detection system. In the last 30 years, it has been investigated in

different aspects related to architectures, computational aspects (for example, involving AI), effective collaboration within multi-agent platforms, and applications. One of the milestones is the paper [11], where the idea of intrusion detection using autonomous agents was proposed. Publications such as [12–14] combined are drawing a comprehensive review of the state-of-the-art in multi-agent cyber defense solutions.

Nowadays, a cyber defense based on multi-agent systems is recognized as a modern and very efficient approach with continuous emerging. Interest in such systems has been extensively revisited recently within academia, industry, law enforcement agencies, and even the military. In [15], the author developed the idea that intelligent autonomous agents will be the standard on the battlefield of the future. It means that intelligent autonomous cyber defense agents are going to become the main element of any entity involved with the battlefield, where cyberspace will become the crucial area of conflict. The paper introduced several novel ideas with summarization of the other ones into the reference architecture of any multi-agent system for cyber defense.

A current industrial application of such systems could be any Internet of Things networks or, in general, cyber-physical systems and networks. The justification behind this is that these systems are by default distributed and multi-node. Furthermore, the requirements on the lightness of the computation on the nodes implicates that only multi-agent cyber threat detection solutions would fit such environments. For example, the state-of-the-art in this field from two papers [16,17] introduce the intrusion detection system in connected vehicles (Vehicle-to-Vehicle, V2V). The system presented in [16] consists of the part that is analyzing the node of the environment—a vehicle—with the option of centralized data analytics in the cloud. The main contribution of the authors was to consider each single element of the vehicle as the valuable source of data to detect cyber threats. Next, it was proposed to combine the real time data from such different and distributed elements together for the classification algorithm based on Bayesian networks. The paper [17] investigates such multi-agent cyber threat detection within a single vehicle more deeply in terms of how to combine data from different sensors to detect intrusions. Such an approach complies with the general idea of multi-agent intrusion detection systems and it is an important example of how to apply it to solve the modern problems of security in cyberspace. As the use case of a connected vehicle will be rapidly adopted, cyber defense solutions involving multi-agent concepts crucially need to be developed.

2.2. Generation of Datasets for Cyber Threat Detection Research

The general prerequisite for any discovery problem to be addressed by data science methods is to have the right data. There are three main approaches to obtaining data for cyber threat detection:

- Collecting data from actual production networks and cyber intrusions,
- Building models of production networks and simulating network communications (malicious and benign),
- The use of mathematical, statistical, machine learning, and other algorithms to generate the data.

The first approach is highly desirable, as working on actual data should guarantee low-fault detection algorithms that are ready for actual cyber attacks. The main problem with the reliability of this approach is that few organizations could use such data for cyber threat detection research. Cyber attacks are very rare if we consider the total observation time. This means that it would take a very long time to collect enough examples to train a detection algorithm on these indicators. Another challenge with such data is the privacy concern. It is impossible to share such information, so the cybersecurity community cannot benefit from it for cyber threat detection.

The simulation approach is usually used in modern research into intrusion detection systems in industry and academia. One of the most well-known research institutes in this field is the Canadian Institute of Cybersecurity (CIC) at the University of Brunswick. The institute has published nearly 30 datasets over the past decade, while researchers have

developed reference methods for generating such data. A 2018 paper [18] summarizes the current approach to generating the simulated networks and data for cyber threat detection research over them. The main outcome was the development of a parametric configuration of the network communication patterns to be simulated, called profiles. This improved the quality of the resulting datasets. Another systematic approach was presented in [19]. This paper adds the new idea of simulated datasets for cyber threat detection systems based on a novel architecture:

- Collecting sensors distributed on network nodes,
- Allowing for continuous communication and coordination between sensors,
- The use of a central processing unit to improve detection decisions,
- The automation of the network scenarios in which the data was collected,
- The use of data science methods to oversample the least representative samples of malicious data.

The details are presented in Section 3. The latter approach exploits the mathematical foundations of modeling and data analysis, in particular, to apply machine learning methods for data generation. It could help to increase the similarity of generated data with actual production or to address shortcomings of simulations (complementary between approaches). An example of applying machine learning to improve detection rates and complement the small number of malicious samples is presented in [20]. It uses adversarial machine learning methods for cyber threat detection research. Generative Adversarial Networks (GAN) are implemented to generate synthetic samples. Then, the module IDS was trained on them along with the original samples. It also fixes the problems of unbalanced or missing data on input. This approach greatly improves the performance of the IDS detection algorithm. The major challenge in applying machine learning for cyber threat detection is the explainability and transparency of such algorithms.

2.3. Availability of Datasets for Cyber Threat Detection Research

Historically, the first milestones in the public availability of datasets for cyber threat detection research were in 1998–1999, when the DARPA'98 and KDD'99 datasets were released. Since then, many other and different datasets have been created, but there are still not enough publicly available datasets for cybersecurity research. This section presents some examples of publicly available datasets that are generally recognized as comprehensive, well-prepared, and appropriate for cybersecurity research on cyber threat detection systems.

Canadian Institute of Cybersecurity datasets: ISCX 2012 Dataset [21] was the first participation of the Canadian Institute of Cybersecurity that provided a systematic approach for creating datasets for cyber threat detection systems research. They introduced the concept of profiles, which contain detailed descriptions of intrusions and abstract distribution models for lower-level applications, protocols, or network entities. Previously, they analyzed real-world traces to create these profiles. The dataset created included benign and malicious network traffic traces of HTTP, SMTP, SSH, IMAP, POP3, and FTP. The NSL-KDD ISCX Dataset [22] was created as a solution to the inherent problems of the original KDD'99 dataset. It still suffers from some of the problems and may not perfectly represent real-world networks. Nevertheless, it can be used as a useful benchmark dataset to help researchers compare different cyber threat detection methods. The CIC 2017 dataset [23] contains benign and the most recent widespread attacks stored as network traffic traces from actual real-world executions. Implemented attacks include Brute Force FTP, Brute Force SSH, DoS, Heartbleed, web attack, infiltration, botnet, and DDoS. Due to the nature of the prepared profiles, they can be directly applied to a variety of network protocols with different topologies to create a dataset for specific requirements. The CSE-CIC 2018 dataset [24] follows the pattern in the scaled infrastructure of 500 devices. The dataset provides the network traffic traces and system logs from each of these devices.

UNSW-NB15 Dataset [25]: The raw network packets of the UNSW-NB 15 dataset were created in the Cyber Range Lab of the Australian Center for Cyber Security (ACCS). It contains a mixture of actual normal activities and synthetic current attack behaviors

from fuzzers, backdoors, DoS, exploits, generic cyber attacks, reconnaissance, shellcode, and worms. Argus, Zeek (formerly *BroIDS*), and the authors' tools were used for data collection. Class tagging was also provided. The number of records in the training set was 175,341, and the testing set was 82,332 from different types of network traffic (benign and malicious).

UGR'16 Dataset [26]: The dataset was created with real traffic and actual attacks. The network traffic was recorded by Netflow v9 collectors strategically placed in the network of one of the Internet Service Providers from Spain. It consists of two datasets split into weeks:

- CALIBRATION set was collected from March to June 2016 (four months) with accurate background traffic data.
- itemize TEST was collected from July to August 2016 with factual background and synthetically generated traffic data of various known attack types.

The main advantage of this dataset is its usefulness for evaluating cyber threat detection algorithms with a long-term perspective. The models can also take into account differentiation by day/night or working days/off days.

CAIDA Datasets: The Center for Applied Internet Data Analysis (CAIDA) collects various types of data from geographically and topologically diverse locations and makes these data available to the research community. Information was collected from active and passive measurement infrastructures that provide insights into global Internet behavior. CAIDA collects, curates, archives, and shares the datasets resulting from these measurements. It also processes and shares several derived datasets. Datasets through April 2016 are available at [27]. One of the most well-known CAIDA datasets is the DDoS 2007 dataset, which contains network traffic traces from large-scale distributed denial-of-service attacks. More recent datasets are made available on the Impact Cyber Trust Project [28] system. The Information Marketplace for Policy and Analysis of Cyber-risk Furthermore, Trust (IMPACT) project was created by the U.S. Department of Homeland Security to support the global cyber-risk research community through the coordination and development of real-world data and information sharing capabilities. The IMPACT project enables the sharing of empirical data and information among the global cybersecurity research and development (R&D) community in academia, industry, and government to accelerate solutions to cyber risk and infrastructure security. Datasets are available exclusively to researchers from the U.S. and collaborating countries.

2.4. Malware with Information Hiding Techniques Applied

Network steganography, as a branch of information hiding techniques, is rapidly evolving and has attracted tremendous interest from cybersecurity researchers since the paper [29]. Any network steganography technique must meet three conditions [10]:

- Modified properties of the protocol;
- Modified properties of the protocol may refer to mechanisms related to inadequacies of the communication channel, the nature of the messages exchanged, or their form;
- Communication parties trying to prevent the observer from detecting the transmission of data using information hiding techniques.

The *Morto* worm [30], a malware with network steganography capabilities, used records stored on Domain Name System (DNS) servers to communicate with C2 servers. This was the first actual implementation of network steganography in malware ever discovered. Over the years, DNS has proven to be one of the most popular network protocols abused for information concealment techniques. Any system from IT that has access to the Internet must use it, so port 53 is wide open and allowed by firewalls and cyber threat detection systems. The DNS protocol is characterized by open text messages that provide many opportunities to hide data in them using text steganography methods. Another protocol that has been used for network steganography in malware in recent years is the Secure Shell (SSH) protocol. It was discovered in 2013 in the *Fokitor* Trojan [31].

The motivation to use SSH for such operations is the same as DNS: widely used in IT systems, port 22 open and allowed. In this method, the SSH protocol connections merely carried the hidden information as a payload. The Regin malware [32], discovered in 2014, was equipped with three mechanisms to prevent network communication:

- Stealth data tunneling in ICMP protocol traffic (ping).
- Insertion of steering commands in cookies in the HTTP protocol header.
- Insertion of steering commands into specially prepared TCP protocol segments or UDP datagrams.

This is the ongoing trend of implementing different steganographic C2 channels and using them depending on the deployment conditions. Steganography, a cyber deception method, provides the ability to bypass the detection and measures of standard network security applications, such as blocking by firewalls or triggering alerts by cyber threat detection systems.

Another trend is the combination of different methods to hide information, e.g., combining multimedia steganography with hidden communication via TCP/IP protocols. The typical approach for combining multimedia steganography and network communication to form hybrid steganography is as follows:

- Use of multimedia steganography to hide the data.
- Use of standard protocols of the TCP/IP stack, especially application network traffic, to smuggle multimedia files between victims and attackers either directly or via C2 servers.

The first practical application of such an approach was a 2011 malware campaign. Duqu [33] used multimedia steganography to hide data in JPEG images and then sent them to the C2 server. This communication looks like an ordinary image file transfer, but in reality it is used to establish a covert C2 channel. A similar technique was used for the 2014 Zeus Trojan morph, Lurk [34], where images were the carriers of the hidden control commands. In the following years, the C2 channels used in modern cyber threat campaigns were considered for information hiding techniques. More recently, the techniques have evolved, spreading multimedia steganography over open social networks (OSN) and adding methods of text steganography. This introduced a new level of complexity to any forensic analysis, making it a problem similar to finding a needle in a haystack. An example of a practical application is Hammertoss APT, applied by the group APT29 [35]. They used Twitter to exchange URLs to image files that contained hidden data. Each Twitter message also contained a specially prepared hashtag needed to decode the hidden part of the image. The project attracted interest from cybersecurity researchers who were looking for models to define detection techniques, as the classical signature approach was insufficient. Interesting proofs-of-concept of steganography systems include:

- Stegobot [36]—one of the pioneering systems using OSNs as an overlay network for the technical operations.
- Instegogram [37]—a technique that uses the image feed of a given Instagram account to decode C2 messages from images. The main achievement here was using a popular internet service to smuggle malware communications.
- StegHash with SocialStegDisc [38]—The StegHash technique was used to distribute multimedia files with hidden data portions across many Internet services and accounts. The mechanism of hashtags creates an invisible chain through which the original message can be recovered. SocialStegDisc implemented the StegHash technique to address the scheme in a novel steganographic file system.

Therefore, the use of hybrid and network steganography to breach the security of computer systems, in particular, is an important area of research to identify vulnerabilities and methods to combat them. This is the critical goal of this work, to improve the security of cyberspace.

3. Generating Datasets for Cyber Threat Detection Research

3.1. Application of Multi-Node Cyber Threat Detection System

A multi-node cyber threat detection system operates in the environment of distributed network devices running open operating systems (e.g., Linux), mainly programmable routers. Each router contains the execution environment of mobile agents that are interconnected to form a platform that controls the Central Unit. For the purpose of this study, the monitoring mode of such a system is considered.

On the execution platform, it is possible to run agents with different purpose settings:

- Agents collect network traffic logs in a specific format and send this data to the Central Unit for analysis.
- Agents equipped with motion logic that follows the developed algorithm for computing anomaly metrics and cooperates in selecting additional areas of the observation network. The goal is to discover the sources of the attack.

To build a multi-agent peer-to-peer communication, the concept of the actor system [39] has been used. The main purpose of the actor system is to develop a high-level and non-blocking parallel execution model for computation. The atomic execution units, called actors, execute their assigned tasks and then share the results via the message box communication abstraction. The actor system is responsible for creating and managing the lives of the actors (agents) in various distributed environments. The scheme of the prepared platform is shown in Figure 1. It shows the main nodes of the architecture:

- The Central Unit node, which manages the actor system of the whole platform and coordinates the life cycle of the distributed agents and of itself. More details are presented in Table 1.
- A router with the actor system instance in which the single node managing agent is instanced and connected with the whole platform managed by the Central Unit. Furthermore, this agent could spawn other node agents to operate different functions. Figure 1 shows such an agent called the Interface Sniffer Agent.

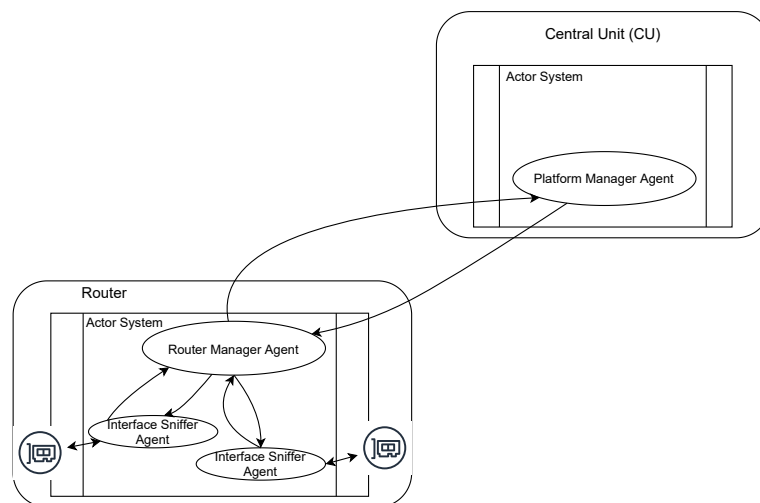


Figure 1. Scheme of monitoring platform based on actor system approach.

As the prototype of the platform is utilized in the monitor mode (sniffing and collecting network data), the main functionalities to be included within the main nodes of the system are:

- Sniffing network traffic on all interfaces of a router;
- Storing PCAP files at the nodes;

- Collecting PCAP files across nodes in the Central Unit.

The interface Sniffer Agent would provide the first and second functionality. The third is implemented by the communication scheme between the Platform Manager Agent, Router Manager Agents, and Interface Sniffer Agents. The whole platform (node agents and Central Unit) provides the other functions, such as life cycle management, PCAP file management, or controlling the operation mode of the system. Table 1 summarizes the operational aspects for each main component of the platform: Central Unit, a router, Router Controlling Agent, and Internal Sniffing Agent. It includes the functional role (*Objectives* column) realized by each of them and the communication patterns with the other components that are utilized to fulfill the role (*Communication patterns* column).

Table 1. Summary of operational aspects of multi-node Network Traffic Monitoring Platform.

Module	Objective	Communication Patterns
Central Unit	1. Hosting main control agent of entire platform	1. Initializing the connection between Central Unit and any router hosting platform 2. Requesting creation of new controlling agent on remote router platform 3. Submitting configuration settings to newly created controlling agent on remote router platform 4. Receiving notification about availability of new PCAP file 5. Retrieving PCAP file over HTTP protocol
	2. Managing entire platform	
	3. Managing joining process of routers hosting platforms	
	4. Managing joining process of router control agents	
	5. Maintaining status of remote router hosting platforms and control agents on each of them	
	6. Receiving message about new PCAP file available on router	
	7. Downloading PCAP file from selected remote router	
	8. Managing collection of PCAP files downloaded from remote routers	
A router	Computing and networking platform that hosts remote portion of entire agent system	Receiving requests to create new control agent
Router Controlling Agent	1. Router management	1. Receiving configuration settings from Central Unit 2. Requesting creation of internal agents to sniff network interfaces 3. Controlling start and stop of network sniffing per selected interface 4. Notifying Central Unit about availability of new PCAP file
	2. Joining platform agent system	
	3. Managing network interfaces and setting up sniffing on them	
	4. Providing HTTP server through which PCAP files can be downloaded from Central Unit	
	5. Managing status of availability of new PCAP files	
Internal Sniffing Agent	1. Creating sniffing process on bound network interface	1. Receiving request to start sniffing 2. Receiving request to stop sniffing 3. Collecting information when new PCAP file is available 4. Notifying router control agent of availability of new PCAP file
	2. Managing sniffing strategy	
	3. Managing end of sniffing action by passing callback to router control agent	

The concept of architecture realized by the presented proof-of-concept is easily expandable by embedding processing and detection algorithms together with any distributed computing strategies to be imposed within the system. Any complexity in terms of logical distribution of the processing and decision-making could be considered. However, the constraints and limitations of such an expansion for any logical workflow of the cyber threat detections and mitigations are driven by:

- The performance related to the type of the networks and its protocols.
- The data flow rates and processing performance.

- The physical bandwidth of interfaces within network nodes (routers and the other network appliances).
- The computational resources within a single network node where the cyber threat detection agent would operate.
- The multi-node cyber threat detection system management links to the performance.

Network packets need to be processed in the time imposed by the bandwidth of the interfaces within a network node. If there is an objective to detect and react to cyber threats inline, then the detection and computation architecture is required to be able to draw decisions within the time frame of the network packet processing. For 10 Gb/s networks, one packet of 300 bytes (average size in the Internet) needs to be processed in 240 nanoseconds. Otherwise, the system would process the copy of the data, so the main constraints will be limited to copy operation, transferring data to the other agents, and the size of the generated data in time (directly based on the network flow data rates).

In fact, the proof-of-concept was implemented in Python as the most efficient for fast prototyping. It was used in the monitor mode only to collect PCAPs as datasets. However, the real production multi-node cyber threat detection system should be implemented in more suitable hardware and software for technological stacking. Software programming languages for data processing within constrained environments in terms of computational resources are C, C++, or Rust. If the software processing cannot fulfill the processing requirements, then hardware solutions to accelerate the computations needs to be considered, such as ASICs or FPGAs. The Central Unit node or any other node considered in general as the “computational center” could be built upon Big Data technological stacks characterized by high scalability, efficiency, and possibility to parallelize computations. The main limitation would be related to the available hardware resources and if it is possible to implement several computational servers as the component of a production multi-node cyber threat detection system.

3.2. Network Traffic Streams Simulations

3.2.1. Malicious Network Data Streams

The network data streams within the scope of this paper must contain steganographic techniques of the various types. For this work, the implementation could be simple so that the required data can be generated for further research. The choices of such methods are:

- Method based on intentionally lost packets that can carry hidden payloads. It could be implemented in various network protocols such as SIP or RTP, with one important characteristic rule—a packet is detected as lost even if it eventually reaches the destination, it is simply discarded. No verification is performed. This fact can be directly applied to network steganography in the following way:
 - Some packets must be intentionally delayed to be detected as lost.
 - The payload of such packets could be overwritten to carry steganograms.
 - When such a packet finally arrives at its destination, it is simply discarded. If a steganographic receiver is installed, it could intercept these packets to extract the hidden payload.
- Method based on modulating the transmission time between packets to encode bits ‘0’ and ‘1’. Delay-based network steganography is a type of time-based steganography. It uses modulation of the transmission times of successive packets in network traffic to encode ‘1’ and ‘0’ bits of hidden data. Probably any network protocol can be used for such a method. The secret between sender and receiver is to encode and decode the hidden data in the temporal relationships between the packets. The sender side must be parameterized with the type of distribution used to generate the network stream. The receiver side must also be parameterized with this distribution and with decision thresholds in the decoding module.

The dedicated applications were prepared as the element of the whole end-to-end framework for cyber threat detection research. The signalization over lost packets in

the multimedia stream of packets utilizes the RTP protocol. The hidden communication over packets with the modulated time of sending uses the ICMP protocol. The prepared applications could also be executed in benign mode to generate the expected network flows of the selected protocols (RTP or ICMP).

3.2.2. Benign Network Data Streams

The approach to generating benign network traffic was developed by analyzing the typical patterns of network communications in consumer and enterprise networks LAN/WAN. Several specific applications and protocols were identified:

- Surfing the Internet and using the HTTP protocol.
- VoIP communication using SIP, RTP, UDP, HTTP, and TCP protocols.
- Video streaming using RTP and HTTP protocols.
- Data transfer using HTTP, FTP, SSH/SFTP, TCP, UDP, or email protocols.
- Using network-related protocols such as ICMP.

For this study, some publicly available applications and scripts were used to simulate such traffic. The complementary method uses publicly available network traces to replay them within a network. The applications mentioned in Section 2.4 are also used in benign mode to generate legitimate traffic without using information hiding techniques.

3.2.3. Engine of Generation of Network Topologies for Experimentation

A network emulation engine should be used for functions such as:

- Enabling rapid prototyping of new use cases.
- Enabling automatic generation of new network topologies, i.e., setting up a new dataset.

When generating a new topology, the basic features must be specified, such as:

- The number of routers and hosts,
- IP address ranges and routing,
- Whether to provide access to the Internet,
- Whether a firewall should be included,
- Placement of the Central Unit of the entire Cyber Threat Detection Agent system.

Based on these parameters, a random graph should be generated and then fed into a network emulation engine via an API or configuration file. Preparing such an automation promises to minimize any bias in the network topologies on the measured effectiveness of the newly developed cyber threat detection algorithms. This means that well-generalized cyber threat detection models should be created that can work equally efficiently in any network topology.

For this research, we developed such a tool for the automatic generation of test application scenarios, which consist of the following elements:

- The backend network engine and simulation tool—GNS3 [40];
- The text file to hold the network configuration—nodes and their types;
- the main script in Python, which
 - Interprets and validates the entered network configuration,
 - Randomly generated connections between nodes,
 - Automatically sets up the network using the GNS3 API;
- The set of scripts in Python to configure the senders and receivers of the network traffic depending on the purpose—to run benign, malicious, or mixed scenarios using the agent system presented in Section 3.1.

It should be noted that the crucial aspect of the prepared solution is the automated and randomized mechanism for:

- Generation of links between the configured set of nodes.
- Selection of senders and receivers for each network data stream profile (benign or malicious).

Such an approach allows the generation of datasets from many network scenarios and data generation configurations. It could also provide the ability to mitigate any factors associated with configuration bias across the broad spectrum of research in data-driven cyber threat detection. Data sets were collected in specific network scenarios with a small degree of variation in the sender-receiver network data configuration (benign or malicious).

3.2.4. Generation of Example Datasets

As presented in Section 3.2.3, the application to automatically generate the network configurations and network communication scenarios was implemented in this article. Figure 2 shows an example output topology of the fully working network of nodes (routers, PC hosts, firewall, Internet connection) prepared by this tool for the purpose of this article.

Among the setup presented in Figure 2, the seven different network scenarios of hidden communication between transmitters and receivers were run. The configuration for each scenario is shown in Table 2. A given scenario (Table 2, first column) consists of the setup of the sender and receiver for a hidden communication over lost packets (second column in Table 2) and the setup of the sender and receiver for a hidden communication over lost packets (third column in Table 2). The order of operations to collect the datasets is as follows:

1. Select nodes (computer hosts) that represent the pairs of a sender and a receiver of a hidden communication for both techniques in this study.
2. Run benign network communication patterns along with hidden network communication.
3. Collect PCAPs for each network node used.
4. Drag all PCAPs to the Central Unit of the platform.
5. Finish generating and collecting data.

Table 2. Setup of pairs of sender-receiver for generation of hidden communication simulations.

Scenario	Hidden Communication over Lost Packets	Hidden Communication over Time Modulation
Scenario 1	Sender: Host H1 Receiver: Host H2	Sender: Host H6 Receiver: Host H7
Scenario 2	Sender: Host H2 Receiver: Host H3	Sender: Host H1 Receiver: Host H6
Scenario 3	Sender: Host H3 Receiver: Host H4	Sender: Host H1 Receiver: Host H2
Scenario 4	Sender: Host H4 Receiver: Host H5	Sender: Host H2 Receiver: Host H3
Scenario 5	Sender: Host H1 Receiver: Host H5	Sender: Host H3 Receiver: Host H7
Scenario 6	Sender: Host H6 Receiver: Host H7	Sender: Host H4 Receiver: Host H5
Scenario 7	Sender: Host H4 Receiver: Host H7	Sender: Host H5 Receiver: Host H6

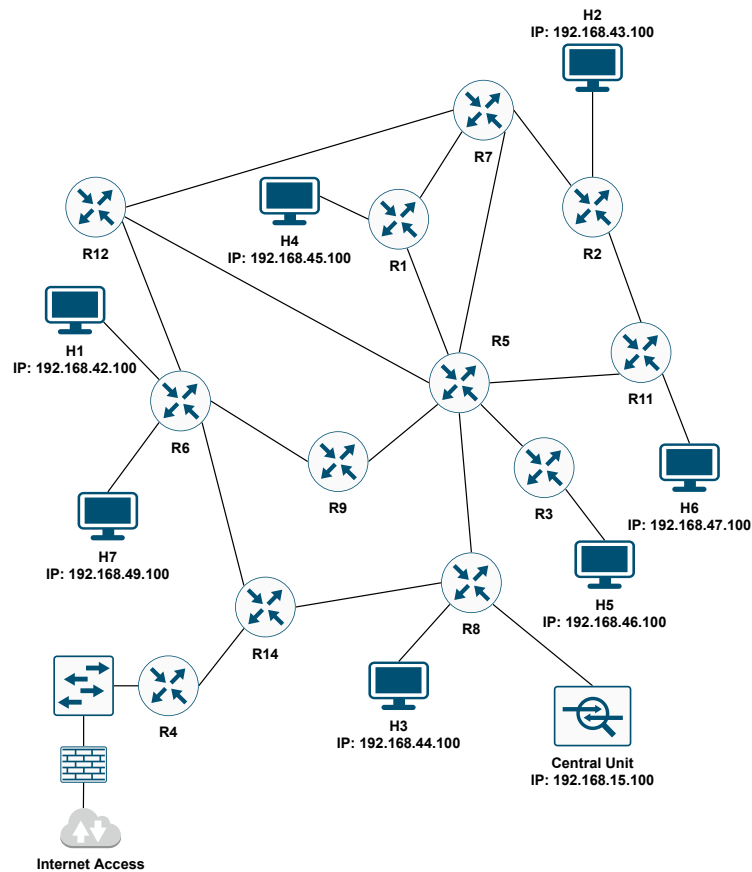


Figure 2. Network setup for simulations of hidden communication techniques.

4. Cyber Threat Detection Research Enabled

4.1. Cyber Threat Detection as Standard ML Classification Problem

In terms of machine learning, cyber threat detection is defined by the principles of the classification problem adapted to the chosen goal of detection. The two classical objectives for cyber threat detection algorithms are:

- Anomaly detection to anomalously detect observations defined as a deviation from the specified base model. The detected anomaly is examined in more detail to determine if it is a cyber threat.
- Detection of cyber threats by finding patterns of the known nature of a cyber attack. Tagged records are required.

Since the datasets generated for this work contain the detailed labels of the cyber threats, the set of experiments follows the second option to be presented. Table 3 shows the general workflow used in this work for cyber threat detection experiments on the hidden communication techniques. It presents the objectives to be achieved for each step of the workflow.

Table 3. Generic procedure to research cyber threat detection methods.

Workflow Step	Objective
Step 1	Benign and malicious network communication simulation scenarios
Step 2	Collect raw source data
Step 3	Generate network data representation from raw source data
Step 4	Data labeling
Step 5	Feature selection
Step 6	Prepare train and test datasets
Step 7	Train data augmentation to balance samples per each label
Step 8	Train, test, and evaluate a machine learning classifier

4.1.1. Network Flows Generation

The records were collected by the system presented in Section 3.1 as network data traces in PCAP format. These PCAPs were then processed into network datasets using CICFlowMeter [41].

Each network flow dataset was bidirectional and consisted of 84 metrics. Network flows were identified by the classic 5-tuple key (source IP, destination IP, source port, destination port, Layer 4 protocol code). When a flow exceeded the configured flow timeout (in seconds) or the flow was inactive (activity timeout), the status was saved and exported. In the exported flow table, the timestamp adds the 5-tuple key to distinguish the flows. In the case of this experiment, the parameters were set to:

- flow timeout—120 s
- activity timeout—30 s

The output of the application is a CSV file. Another step was the labeling. It was done manually through the script based on the coding scheme shown in Table 2. The last step in the data preparation was to combine all CSV files into a final dataset with all collected observations from the simulated scenarios. This dataset was then preprocessed in the data experimentation phase according to the state of the art in data science.

4.1.2. Training Classifiers for Cyber Threat Detection

This part shows how to use the prepared datasets to find the classifier to be used as a cyber threat detector. The first step was to analyze the metrics in the dataset. The aim was to check which metrics were more important for predicting the target class (feature selection). The process involved pairwise correlation between the metrics and the explained variable (class or label). Figure 3 shows the filtered matrix of the metrics for which the absolute value of the correlation coefficient of any metric with a label greater than 0.05. The most positively or negatively correlated metrics were related to time (inter-arrival time (IAT), time of activity) and volume of the network data (number of packets, number of bytes). The practical aspect of this step was to reduce the dimension of the problem. The number of metric outputs was 28 since 52 metrics were filtered and three metrics were skipped since they were not relevant to the problem (flow ID, timestamp, IP addresses).

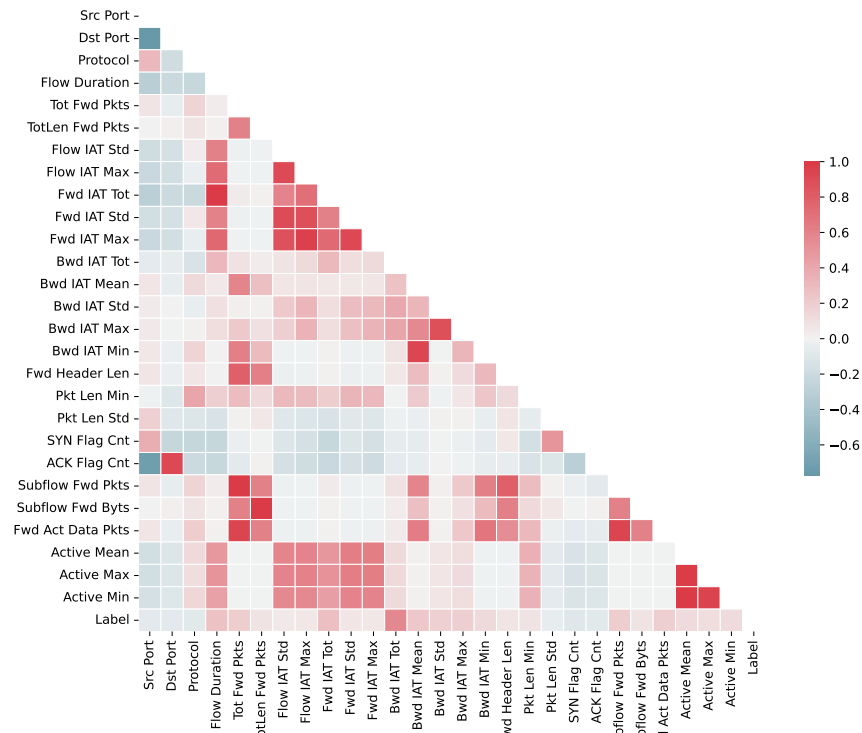


Figure 3. Pairwise correlation matrix of prepared dataset. Filtered according to level of correlation between label column and other parameters for feature selection purposes.

Subsequently, the filtered data were divided into training (80% of the datasets) and testing (20% of the datasets) datasets. The number of collected data streams distributed among the training and testing datasets after the split is shown in Table 4.

Table 4. Distribution of flows among training and test datasets split from collected dataset as described in Section 3.2.4.

Types of Flows	Flows in Training Dataset	Flows in Testing Dataset
Benign traffic (Label: 0)	92,737	23,163
Hidden communication over lost packets (Label: 1)	742	195
Hidden communication over time modulation (Label: 2)	457	127

The first problem that arises is the imbalance of the class examples in the training dataset. The imbalanced classifications poses a challenge to any predictive algorithm. In the case of imbalance in the training examples, the trained models might have poor predictive performance, especially for the minor classes. On the other hand, the minor classes are important because the context of the experiment is cyber threat detection research, where cyber attacks are sporadic compared to harmless attacks. The Synthetic Minority Oversampling Technique (SMOTE) [42] and Edited nearest neighbor (ENN) [43] were applied as data extensions to overcome the problem. SMOTE is used to increase the number of samples in the minority class by linear interpolation, and ENN is used to remove the noise from the majority samples. Table 5 contains the number of data streams before and after applying SMOTE-ENN techniques to the training dataset.

Table 5. Data augmentation results—number of flows in training dataset before and after SMOTE-ENN.

Types of Flows	Flows in Training Dataset before SMOTE-ENN	Flows in Training Dataset after SMOTE-ENN
Benign traffic (Label: 0)	92,737	92,737
Hidden communication over lost packets (Label: 1)	742	92,706
Hidden communication over time modulation (Label: 2)	457	92,643

The experimental phase was conducted to measure the possibility of predicting cyber threats by applying selected ML classifiers. Two classifiers were chosen to test the preparation of the example solution for this paper:

- Random Forest (RF) [44] was implemented based on RandomForestClassifier from the Scikit-learn [45]. The initial setup was based on the default parameters as described in [46].
- Multilayer Perceptron (MLP) classifier was implemented as the custom architecture in TensorFlow [47] using the Keras subpackage. The setup of this classifier in terms of architecture, optimizer, loss function, and evaluation metrics is presented in Table 6.

Table 6. Custom MLP classifier architecture setup per each layer (6) with selected optimizer, loss function, evaluation metric, and training procedure.

Parameter	The Custom MLP Classifier
Layer 1 (Input)	Input, size: 28×20 , activation: relu
Layer 2	Dense, size: 20×20 , activation: relu
Layer 3	BatchNormalization, size: 20×20 , activation: relu
Layer 4	Dense, size: 20×150 , activation: relu
Layer 5	Dense, size: 150×20 , activation: relu
Layer 6 (Predictions)	Dense, size: 20×20 , activation: softmax
Optimizer	Adam with the learning rate: 0.001
Loss	Categorical Cross Entropy
Evaluation metrics	accuracy
Training setup	batch size: 128, epochs: 20, validation split: 0.15

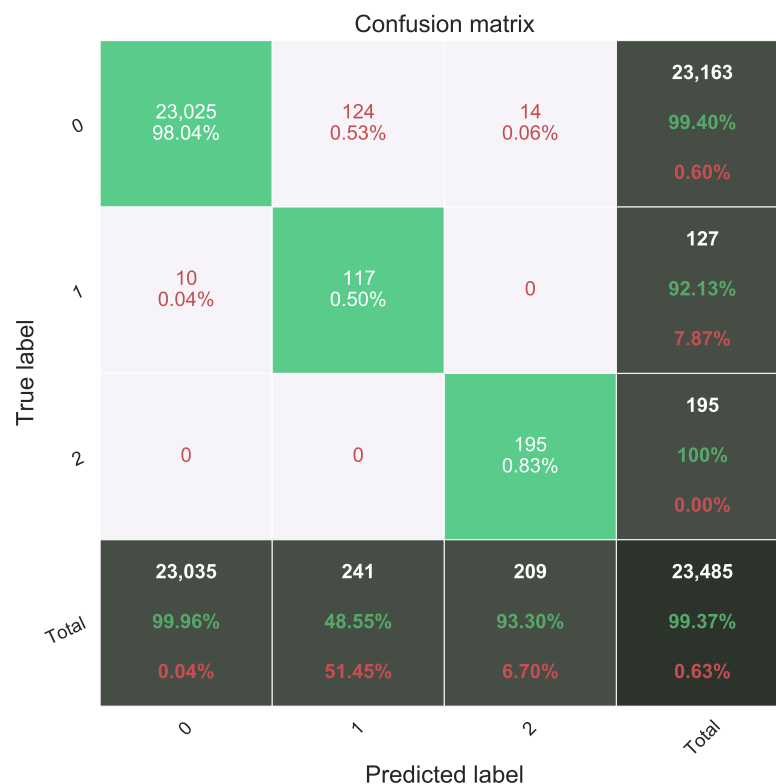
Both classifiers were trained with the SMOTE-ENN training datasets and evaluated with non-SMOTE-ENN test datasets. The metrics of accuracy, precision, recognition, and F1 score were used to evaluate the performance. The result metrics for the RF and MLP classifier are shown in Tables 7 and 8, respectively. The last rows of both tables show the overall accuracy of the respective classifier. Figures 4 and 5 show the confusion matrices of the two selected classifier models. The classification was conducted within three classes (0, 1, or 2), so the size of each confusion matrix was 3×3 . Each cell presented the number of instances of a given true class (rows) classified into a given predicted class (columns). The diagonal of each matrix included true positives. The other cells could be classically interpreted as false positives, false negatives, and true negatives in relation to a selected class.

Table 7. Performance of RF classifier in terms of precision, recall, F1 score, and overall accuracy metrics.

Types of Flows	Precision	Recall	F1 Score
Benign traffic (Label: 0)	1.00	0.99	1.00
Lost packets attack (Label: 1)	0.49	0.92	0.64
Packet timing attack (Label: 2)	0.93	1.00	0.97
Overall accuracy	0.99		

Table 8. Performance of custom MLP classifier in terms of precision, recall, F1 score, and overall accuracy metrics.

Types of Flows	Precision	Recall	F1 Score
Benign traffic (Label: 0)	1.00	0.99	1.00
Lost packets attack (Label: 1)	0.42	0.99	0.59
Packet timing attack (Label: 2)	0.84	1.00	0.91
Overall accuracy	0.99		

**Figure 4.** Confusion matrix of RF classifier.

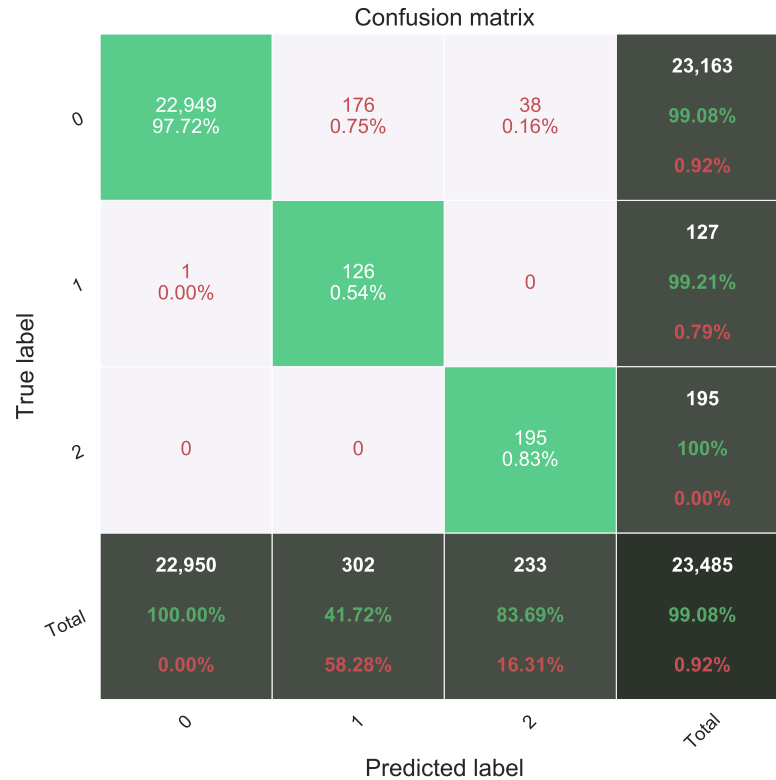


Figure 5. Confusion matrix of custom MLP classifier.

4.2. Conclusions and Future Directions

The prepared models show almost perfect results for the Label 2—Time Modulation Information Hiding Attack. The results for Label 1—Flows with Lost Packets Information Hiding Technique—were noticeably worse. The presented results should be considered as an example for the research work with the generated datasets. Table 9 supplements Table 3 with a summary of the actions performed for each step in this work.

Each step of the workflow shown in Table 9 could be considered to be different research directions, such as:

- Generating more data using the prepared application for simulations, especially for more examples of hidden communication techniques.
- Selection of a different predictive model or design of a more complex architecture combining some classifiers.
- To find a data representation that is better suited to the context of detecting information hiding techniques. The output data representation of CICFlowMeter contains several different metrics related to temporal aspects of network communication. Thus, this is the most likely answer as to why the detection of temporally modulated hidden communication had better performance.
- With other feature selection or data augmentation methods.

Table 9. Summary of realized actions per each step of generic procedure to research cyber threat detection methods.

Workflow Step	Objective	Realization in This Paper
Step 1	Benign and malicious network communication simulations	Implementation of the dedicated tool to set up networks and simulations; proof-of-concept implementation of Multi-node Cyber Threat Detection System to use in monitor and collect mode
Step 2	Collecting raw source data	Network traffic traces collected as PCAP files
Step 3	Generating network data representation from raw source data	Generation of network flows including 80 metrics from [41]
Step 4	Data labeling	Labeling based on Table 2 by adding the column Label with the respected coding to the corresponding network flows
Step 5	Feature selection	Selecting features using correlation coefficient between Label and the other features
Step 6	Preparing train and test datasets	Train/test split method from [45]
Step 7	Train data augmentation to balance samples per each label	Augmentation of the training dataset with SMOTE-ENN method
Step 8	Train, test, and evaluate a machine learning classifier	Preparing Random Forest and the custom MLP classifiers.

5. Summary

This paper presents an approach for the availability of datasets for cyber threat detection research and the application to Data Science. As a first step, the multi-agent platform for collecting network flows was implemented for this purpose. Such a platform enabled the collection of network flow data in a multi-node setup. One of the achievements was the implementation of an automated solution for generating network configurations and running application scenarios for cyber threat activities. This is a promising aspect to scale research experiments for cyber threat detection. Another future aspect to be explored is the ease of practical implementation of such solutions. The input problem of any practical cyber threat detection solution is the working environment in which the method is implemented. The standard approach is the learning phase, which assumes that the cyber detection engine must be adapted to the network environment through monitoring and learning. After reaching *readiness*, the cyber threat detection engine is partially trained with new examples of malicious activity or feedback data from human operators. The ability to scale the data generated in different network configurations would improve cyber threat detection engines for the sake of greater generality. Other practical implications could include easier implementation in working environments and reduced relearning and manual tuning efforts.

The unique value of this research was to emphasize the recognition of information concealment techniques, which are generally considered concepts within the broad domain of cyber deception. The most important prerequisite for working on cyber threat detection is the availability of the right data. All known data sets that are available focus on the publicly known types of cyber attacks. The examples of the use of cyber deception techniques are very rare or non-existent. One of the main contributions of this work was the development of a network environment integrated with the tools to collect such examples. This was achieved by proposing the implementation of a multi-agent system as a Multi-Node Cyber

Threat Detection platform utilizing the monitor mode. Based on the collected data, the reference data science workflow was evaluated by applying methods for data representation and classification of malicious network flows. The final result confirms the usefulness of the presented end-to-end approach for researching the discovery of information hiding techniques. The authors will apply it in further research and development projects. Cyber attackers are increasingly using cyber deception techniques. Moreover, advanced cyber attacks, for example, APT (Advanced Persistent Threat) campaigns, could combine more than one deception technique in two dimensions:

- Within different abstraction layers within the ISO-OSI 7-layer model, with the application layer in particular considered a significant threat.
- Per each step of a cyber attack modeled as Cyber Kill Chain. [4]

This poses a massive threat to the security of cyberspace, so more efforts need to be made in the coming years to improve cyber defense capabilities in the area of cyber deception.

Author Contributions: Conceptualization, K.S.; Investigation, J.B.; Methodology, J.B.; Software, J.B.; Supervision, K.S.; Validation, J.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by The Polish National Centre for Research and Development under project No. CYBERSECIDENT/369532/1/NCBR/2017.

Data Availability Statement: The data cannot be shared due to project restrictions.

Acknowledgments: The authors wish to thank Monika Stepkowska, for her contribution to setting up the experiments.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Barrett, M. NIST Cybersecurity Framework (CSF): Framework for Improving Critical Infrastructure Cybersecurity. Version 1.1. 2018. Available online: <https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04162018.pdf> (accessed on 22 October 2021).
2. Fragkos, G.; Minwalla, C.; Plusquellic, J.; Tsiropoulou, E.E. Artificially Intelligent Electronic Money. *IEEE Consum. Electron. Mag.* **2021**, *10*, 81–89. [CrossRef]
3. Cichonski, P.; Millar, T.; Grance, T.; Scarfone, K. NIST SP 800-61: Computer Security Incident Handling Guide. 2012. Available online: <https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-61r2.pdf> (accessed on 22 October 2021).
4. Hutchins, E.; Cloppert, M.J.; Amin, R.M. Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains. *Lead. Issues Inf. Warf. Secur. Res.* **2011**, *1*, 80. Available online: <https://www.lockheedmartin.com/content/dam/lockheed-martin/rms/documents/cyber/LM-White-Paper-Intel-Driven-Defense.pdf> (accessed on 22 October 2021).
5. MITRE ATT&CK. Available online: <https://attack.mitre.org/> (accessed on 5 September 2021).
6. Chou, D.; Jiang, M. Data-Driven Network Intrusion Detection: A Taxonomy of Challenges and Methods. *arXiv* **2020**, arXiv:2009.07352.
7. Ptacek, T.; Newsham, T. *Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection*; Secure Networks, Inc.: Mandaluyong, Philippines, 1998. Available online: <http://www.icir.org/vern/Ptacek-Newsham-Evasion-98.ps> (accessed on 22 October 2021).
8. Nehinbe, J.O. A Simple Method for Improving Intrusion Detections in Corporate Networks. In *Information Security and Digital Forensics*; Weerasinghe, D., Ed.; Springer: Berlin/Heidelberg, Germany, 2010; pp. 111–122.
9. McAfee Labs Threats Report—June 2017. Available online: <https://www.mcafee.com/enterprise/en-us/assets/reports/rp-quarterly-threats-jun-2017.pdf> (accessed on 5 September 2021).
10. Mazurczyk, W.; Wendzel, S.; Zander, S.; Houmansadr, A.; Szczypiorski, K. Background Concepts, Definitions, and Classification. In *Information Hiding in Communication Networks: Fundamentals, Mechanisms, Applications, and Countermeasures*; IEEE-Wiley Press: New York, NY, USA, 2016; Chapter 2, pp. 39–58.
11. Balasubramanian, J.; Garcia-Fernandez, J.; Isacoff, D.; Spafford, E.; Zamboni, D. An architecture for intrusion detection using autonomous agents. In Proceedings of the 14th Annual Computer Security Applications Conference (Cat. No. 98EX217), Phoenix, AZ, USA, 7–11 December 1998; pp. 13–24.
12. Herrero, A.; Corchado, E. Multiagent Systems for Network Intrusion Detection: A Review. *Comput. Intell. Secur. Inf. Syst.* **2009**, *63*, 143–154.

13. Docking, M.; Uzunov, A.V.; Fiddymment, C.; Brain, R.; Hewett, S.; Blucher, L. UNISON: Towards a Middleware Architecture for Autonomous Cyber Defence. In Proceedings of the 2015 24th Australasian Software Engineering Conference, Adelaide, SA, Australia, 28 September–1 October 2015; pp. 203–212.
14. Saeed, I.A.; Selamat, A.; Rohani, M.F.; Krejcar, O.; Chaudhry, J.A. A Systematic State-of-the-Art Analysis of Multi-Agent Intrusion Detection. *IEEE Access* **2020**, *8*, 180184–180209. [CrossRef]
15. Kott, A. Intelligent Autonomous Agents are Key to Cyber Defense of the Future Army Networks. *Cyber Def. Rev.* **2018**, *3*, 57–70.
16. Pascale, F.; Adinolfi, E.A.; Coppola, S.; Santonicola, E. Cybersecurity in Automotive: An Intrusion Detection System in Connected Vehicles. *Electronics* **2021**, *10*, 1765. [CrossRef]
17. Lombardi, M.; Pascale, F.; Santaniello, D. EIDS: Embedded Intrusion Detection System using Machine Learning to Detect Attack Over the CAN-BUS. In Proceedings of the 30th European Safety and Reliability Conference and 15th Probabilistic Safety Assessment and Management Conference, Venice, Italy, 1–5 November 2020; pp. 2028–2035. Available online: <https://www.rpsonline.com.sg/proceedings/esrel2020/pdf/5090.pdf> (accessed on 22 October 2021).
18. Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. *ICISSP* **2018**, *1*, 108–116.
19. Ring, M.; Wunderlich, S.; Grödl, D.; Landes, D.; Hotho, A. Creation of Flow-Based Data Sets for Intrusion Detection. *J. Inf. Warf.* **2017**, *16*, 41–54.
20. Shahriar, M.H.; Haque, N.I.; Rahman, M.A.; Alonso, M. G-IDS: Generative Adversarial Networks Assisted Intrusion Detection System. In Proceedings of the 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC), Madrid, Spain, 13–17 July 2020; pp. 376–385.
21. Canadian Institute for Cybersecurity. Intrusion Detection Evaluation Dataset (ISCXIDS2012). Available online: <https://www.unb.ca/cic/datasets/ids.html> (accessed on 5 September 2021).
22. Canadian Institute for Cybersecurity. NSL-KDD Dataset. Available online: <https://www.unb.ca/cic/datasets/nsl.html> (accessed on 5 September 2021).
23. Canadian Institute for Cybersecurity. Intrusion Detection Evaluation Dataset (CICIDS2017). Available online: <https://www.unb.ca/cic/datasets/ids-2017.html> (accessed on 5 September 2021).
24. Canadian Institute for Cybersecurity. A Realistic Cyber Defense Dataset (CSE-CIC-IDS2018). Available online: <https://www.unb.ca/cic/datasets/ids-2018.html> (accessed on 5 September 2021).
25. Moustafa, N.; Slay, J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS), Canberra, ACT, Australia, 10–12 November 2015; pp. 1–6.
26. Maciá-Fernández, G.; Camacho, J.; Magán-Carrión, R.; García-Teodoro, P.; Therón, R. UGR'16: A new dataset for the evaluation of cyclostationarity-based network IDSs. *Comput. Secur.* **2018**, *73*, 411–424. [CrossRef]
27. Center for Applied Internet Data Analysis. CAIDA Datasets. Available online: <https://www.caida.org/catalog/datasets/overview/> (accessed on 5 September 2021).
28. Information Marketplace For Policy and Analysis of Cyber Risk & Trust. Available online: <https://www.impactcybertrust.org> (accessed on 5 September 2021).
29. Szczypiorski, K. *Steganography in TCP/IP Networks—State of the Art and a Proposal of a New System—HICCUPS*; Institute of Telecommunications' Seminar, Warsaw University of Technology: Warsaw, Poland, 2003.
30. Mullaney, C. Morto Worm Sets a (DNS) Record. 2011. Available online: <http://www.symantec.com/connect/blogs/morto-worm-sets-dns-record> (accessed on 22 October 2021).
31. Attackers Hide Communication within Linux Backdoor. Available online: <https://www.securityweek.com/attackers-hide-communication-linux-backdoor> (accessed on 5 September 2021).
32. Regin: Top-Tier Espionage Tool Enables Stealthy Surveillance. 2015. Available online: <https://docs.broadcom.com/doc/regin-top-tier-espionage-tool-15-en> (accessed on 5 September 2021).
33. Bencsáth, B.; Pék, G.; Buttyán, L.; Félegyházi, M. Duqu: A Stuxnet-like malware found in the wild. *CrySyS Lab Tech. Rep.* **2011**, *14*, 60–141. Available online: <https://www.crysys.hu/publications/files/bencsathPBF11duqu.pdf> (accessed on 22 October 2021).
34. Dell Secureworks. Malware Analysis of the Lurk Downloader. Available online: <https://www.secureworks.com/research/malware-analysis-of-the-lurk-downloader> (accessed on 5 September 2021).
35. FireEye Threat Intelligence. HAMMERTOSS: Stealthy Tactics Define a Russian Cyber Threat Group. Available online: https://www.fireeye.com/blog/threat-research/2015/07/hammertoss_stealthy.html (accessed on 5 September 2021).
36. Nagaraja, S.; Houmansadr, A.; Piyawongwisal, P.; Singh, V.; Agarwal, P.; Borisov, N. Stegobot: A Covert Social Network Botnet. In *Information Hiding*; Filler, T., Pevný, T., Craver, S., Ker, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2011; pp. 299–313.
37. Deutsch, J.; Garrie, D. Instegogram: A New Threat and Its Limits for Liability. *J. Law Cyber Warf.* **2017**, *6*, 1–7.
38. Bieniasz, J.; Szczypiorski, K. Methods for Information Hiding in Open Social Networks. *JUCS-J. Univers. Comput. Sci.* **2019**, *25*, 74–97.
39. Hewitt, C.; Bishop, P.; Steiger, R. A Universal Modular Actor Formalism for Artificial Intelligence. In Proceedings of the 3rd International Joint Conference on Artificial Intelligence (IJCAI'73), Stanford, CA, USA, 20–23 August 1973; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 1973; pp. 235–245.
40. GNS3 Network Simulation Tool. 2021. Available online: <https://www.gns3.com> (accessed on 5 September 2021).

41. Canadian Institute for Cybersecurity. CICFlowmeter—Network Traffic Bi-Flow Generator and Analyzer for Anomaly Detection. Available online: <https://github.com/ahlashkari/CICFlowMeter> (accessed on 5 September 2021).
42. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic Minority over-Sampling Technique. *J. Artif. Int. Res.* **2002**, *16*, 321–357. [[CrossRef](#)]
43. Beckmann, M.; Ebecken, N.F.; de Lima, B.S.P. A KNN undersampling approach for data balancing. *J. Intell. Learn. Syst. Appl.* **2015**, *7*, 104. [[CrossRef](#)]
44. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [[CrossRef](#)]
45. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
46. Random Forrest Classifier from Scikit-Learn Framework. 2018. Available online: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html> (accessed on 20 September 2021).
47. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. 2015. Available online: tensorflow.org (accessed on 5 September 2021).

Towards Empowering Cyber Attack Resiliency Using Steganography

Jędrzej Bieniasz, Krzysztof Szczypiorski

Institute of Telecommunications
Warsaw University of Technology
Warsaw, Poland
e-mail: J.Bieniasz@tele.pw.edu.pl

Abstract—The fog computing has emerged as the extension of cloud computing to the network edge. The idea could be considered as a promising mechanism for cybersecurity by assuming that higher uncertainty of information from perspective of adversaries would improve the security of networks and data. This approach is recognized as *cyberfog security* in which data, split into fragments, is dispersed across multiple end-user devices. Even if some of them would be compromised, the adversary could not decode information and the availability of data would not be affected. This paper considers applying two steganographic proposals (StegHash and SocialStegDisc) for a new distributed communication system by fulfilling assumptions of *cyberfog security* approach. The initial design of such system is proposed. Features and limitations were analyzed to prepare recommendations for further development and research.

Keywords—fog computing; information hiding; steganography; distributed filesystems; StegHash; SocialStegDisc; cyber-physical systems

I. INTRODUCTION

Cloud computing (centralized model) is extensively developed and widely applied in last years. Parallely, Internet of Things with deployment of smart and interconnected devices, for example mobile phones, wearables, sensors, power grids etc. could overgrow 50 billion units by 2020 [1]. It triggers defining a new paradigm of distributed computing called *fog computing* [2] that complement the centralized cloud computing. National Institute of Standards and Technology (NIST) went further with unification of the description of computing systems. They defined a whole new framework called *cyber-physical systems (CPS)* [3] and IoT could be considered as the implementation of CPS. In [4] authors consider the fog computing as a network and data security mechanism. *Cyberfog security approach* means that higher dispersion of data supports greater attack resiliency. An adversary, which compromises the system, could take only a part of the system and it would be useless.

The authors noticed a possible connection of their current research with this topic. Recently, they revisited the idea of text steganography [5] in combination with social networks. The original idea of StegHash [6] method is to use multimedia objects as carriers of hidden information and to disperse them across open social networks (OSNs). A logical connection between them is established by means of a

mechanism of hashtags as the text markers in the form of #<tag>, commonly applied in OSNs. In addition, the author of StegHash proposes an option of applying the StegHash technique to create a steganographic filesystem analogous to the existing classic filesystems such as FAT (File Allocation Table) or NTFS (New Technology File System). Research on that option was furtherly conducted in [7] to follow the pattern of researching steganography by development of applications preceded by the appearance of new steganographic technique. SocialStegDisc proposed the application of basic concepts of classic filesystems, such as create-read-update-delete operations or defragmentation process to the original idea of StegHash. Furthermore, time-memory trade-offs were proposed by the design.

This paper proposes to apply methods [6] [7] directly to design a new kind of distributed communication system for *cyberfog security* approach. The design utilizes the indexing scheme introduced by StegHash [6] and provides the basic operations like SocialStegDisc [7]. Furthermore, the concept expands the ideas beyond the original ones by using end-user devices' memory as a carrier for objects with hidden data. The methods of routing and finding next parts physically in such peer-to-peer network is addressed. The security is majorly ensured by the character of logical connection between the distributed parts of data. The system could only be properly compromised when the adversary takes over the secret generation function. Without it, the captured parts, e.g. by compromising one or part of devices would be unusable what fulfill the basic *cyberfog cybersecurity* approach requirement.

II. REVIEW OF LITERATURE

A. Cyberfog Security

Cyberfog security approach was introduced by Kott et al. in [4]. They proposed to use the fog computing [2] as a method for mitigating a cyber adversary. It assumes that presenting adversaries with uncertainty could provide greater attack resiliency. A direct realization of this approach is splitting data into numerous fragments and dispersing them across multiple end-user devices. Even if the system could be partly compromised, captured information would be useless for adversary, while still being useful to us. Authors recognized numerous benefits, but also formidable challenges with respect to data and network management complexity, bandwidth, storage, battery-power demands,

data-reassembly latency and intermittent connectivity. They see that the network might need to manage a complex tradeoff between availability and confidentiality in real time depending on users' tasks and circumstances.

B. Steganography and Open Social Networks

Two models of communication: high-entropy and low-entropy was presented by Beato et. al in [8]. In high entropy model the steganogram is transported by a single multimedia object such as pictures, video or music. This is considered as a classic method of steganographic communication. This model is characterized by high steganographic throughput, but the channel is easily detectable. Low-entropy model utilizes text data (e.g., status update, group text message) to carry secret information. To determine the steganogram location a pre-shared secret is used to decode the actual message. This could be used mainly for signaling due to the low steganographic throughput.

It should be noticed another low-entropy steganographic method proposed by Castiglione et al. in [9]. They took an advantage of the feature of inserting tags in images. The proposed stealth communication channel requires the uploading of multiple images and to tag multiple users.

III. CONCEPT OF THE SYSTEM

A. Reminder of StegHash and SocialStegDisc

The proposed method of StegHash [6] is based on the use of hashtags on various open social networks to connect multimedia files, like images, movies or music, with embedded hidden data. For every set of hashtags containing n elements there is the factorial of n permutations, which are individual indexes of each message. With a secret value (password) and a secret transition generator (function), the link between these indexes could be established and then explored as a chain from one message to another, with each containing hidden content. The example of the chain is presented in the Figure 1. It establishes a new paradigm of unlimited data space, but limited address space. SocialStegDisc [7] is an application of StegHash which introduced filesystem construction on the top of StegHash chaining mechanism. Furthermore, time-memory trade-offs were proposed. Increasing the number n of hashtags is followed by the increasing volume of the dictionary proportionately to $n!$ what for higher n would be unacceptable. Proposed mechanism of the linked list were taken directly from filesystem theory to introduce basic operations - read, write, update and delete - without comprising the level of security. To sum up main assumptions:

- An unique permutation of the set of hashtags is an index on data fragment;
- The set of n hashtags means $n!$ of unique indexes to generate;
- There could be more than one service, where data is uploaded, so the index system for services need to be designed. For StegHash and SocialStegDisc was to

use each of n hashtags on the last position as a key for one of n services.

- The used pseudorandom generation function should produce the same chain of indexes with the same seed;

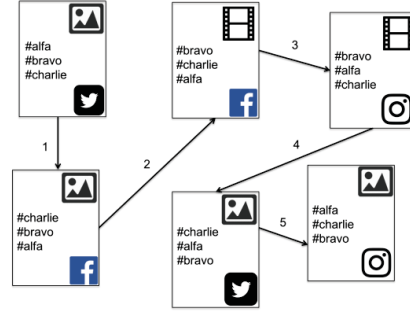


Figure 1. Example of StegHash method [6]

B. Transforming StegHash/SocialStegDisc into Cyberfog Security Domain

We need to identify how well-defined and examined concepts of StegHash and SocialStegDisc fit into a domain of fog of the devices. Firstly, there are no public and open internet services like open social networks. Instead, everything will operate between a defined set of end users' devices. The next main difference is a size of a problem. Earlier, the n was from a few to a dozen or so. Now, this number increases many times, so indexing the services for uploading data by one of n hashtags in an index is impossible. So, we propose to apply this mechanism in another setting:

- n devices would create a *memory sector* of size n . For such memory sector the mechanism of addressing the carrier of data (end-user device) by one of n hashtags is still applicable inside this memory sector.
- System needs a new layer for the communication on the level of the memory sectors;

Basing on this, there would be many memory sectors available in the system. Every device is locally associated with a particular memory sector. It also means that every device have its own instance of StegHash/SocialStegDisc as a memory controller to serve all the basic filesystem operations inside the memory sector. It should be noticed that the design could be scaled to support multiple membership of devices in memory sectors.

From the user perspective, sending data between devices or generally saving the data in the system is realized by using associated StegHash/SocialStegDisc controller to load data to the local memory sector by dispersing fragments between n other devices. Every device needs to have a synchronized copy of an allocation table of the sector. Figure 2. presents the conceptual model of applying StegHash/SocialStegDisc for the fog architecture.

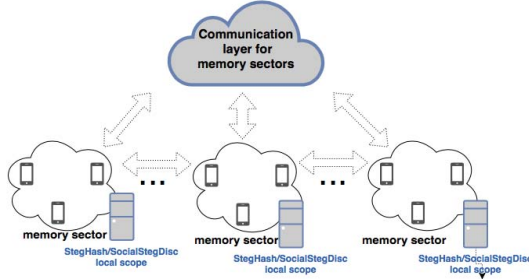


Figure 2. Applying StegHash/SocialStegDisc into the fog architecture

C. The Communication on the Level of the Memory Sectors

In the former paragraph the concept of the memory sector under control of StegHash/SocialStegDisc was introduced. To complete the design of the system we needed to design a communication scheme between the memory sectors. We evaluated two ideas which are presented in this paper.

1) Device as the memory sector's gateway

In this scheme, when a device X saves the data inside its memory sector, it would inform other devices that starting from the index I , there are B bytes to read and the device X is a gateway to read. This kind of scheme is combining the classic methods of routing (networking) and the file allocating methods (filesystem theory). Other devices would have a view of allocated data and they would know where to send read requests to download data. This approach requires a synchronization of allocated addresses across the memory sectors, but the state of generation function is needed to be synchronized only across the memory sector.

2) Direct reading from the memory sector

This idea is about not using a one of memory sector's devices as a gateway. Instead, the routing information for uploaded data requires including all n devices keys and the coding scheme of how to choose the next device for reading data. With this information, the device could read a requested file directly from the memory sector. There are two options of sharing the routing records:

- *Proactive*: sharing the routing records with every single upload to the system and caching it by every device;
- *Reactive*: sharing only metadata of the file and gateway with every single upload to the system and caching it by every device. When the data is wanted to be retrieved, at first the request of memory scheme is issued to the gateway. The gateway responds back with all n devices keys and coding scheme. Next, the device could read a requested file directly from the memory sector.

D. The Communication on the Device-to-Device Level

The layer of network communication and reachability between devices is needed to be considered by design. This consist of two functions:

- *Streaming data end-to-end*. Providing communication by TCP/IP transport layer protocol – TCP or UDP – is an obvious choice. Other protocols from upper levels could be also used.
- *Reachability between devices*. Every device needs to know the network addresses of other devices in network. It could be achieved through static routing configuration or through dynamic routing protocols.

To serve the described communication layer we propose to use the TrustMAS platform [10]. This system would provide all required features, but in a steganographic manner. Every device would have use a StegAgent instance introduced by TrustMAS design to support covert streaming of system data. It would serve as a network communication endpoint for StegHash/SocialStegDisc memory controller. Figure 3 presents the scheme of the layer architecture of the platform.

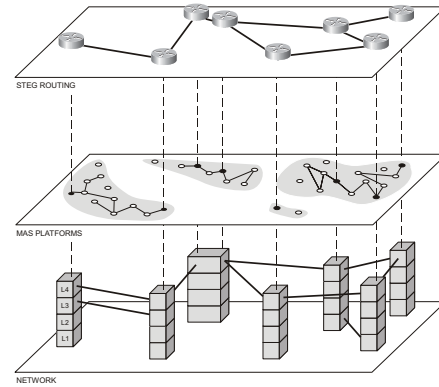


Figure 3. Three-layer architecture of TrustMAS [10]

E. Overview of the Design

Until this paragraph we introduced a new area of application for StegHash [6] and SocialStegDisc [7] which is a distributed communication system between end-user devices. We defined the memory sectors and how to manage them by StegHash/SocialStegDisc controller, the communication layer between them and we chose a platform for device-to-device communication – TrustMAS [10]. Figure 4. provides the conceptual scheme of components inside a particular end-user device and interaction with user, physical memory and another device. On Figure 5. a view of the system by a stack of operational layers is presented. Every layer provides the set of operations logically connected:

- *Application layer* – an application which uses the system from the perspective of user;
- *Filesystem/messaging service* – provides the interface for application layer with read-write-delete operations inside device and with managing point-to-point connection sessions; it passes commands to lower layers;

- *StegHash/SocialStegDisc controller* – the implementation of StegHash [6] indexing method and SocialStegDisc [7] memory management;
- *Memory sector and its controller* – defining set of operations on the memory sector as a group of n devices. It provides the view of the memory sector for StegHash/SocialStegDisc controller and manages the state of the memory sector. Furthermore, they cooperatively manage the scheme of allocation of the memory over devices, memory sectors and the whole storage system.
- *Physical memory* – a non-volatile memory of the device, where data is stored;
- *StegAgent* – treated as a network service introduced by TrustMAS [10]. It offers a network communication and reachability of devices.

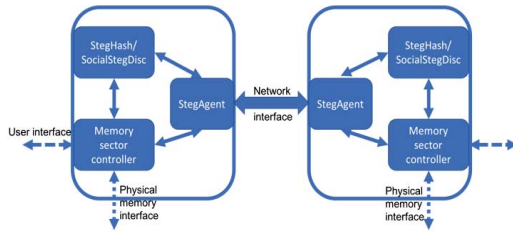


Figure 4. Components of the system and their interaction scheme

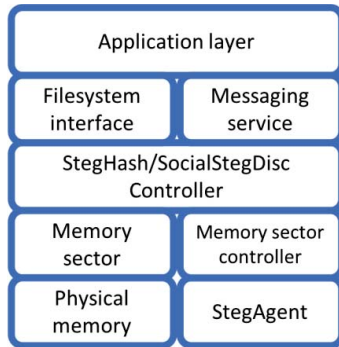


Figure 5. The layer view of the system

IV. DISCUSSION

A. Expanding the Original Idea of StegHash

The concept expands the idea beyond the original ones [6] [7] by:

- Using end-user devices' memory as a carrier for objects with hidden data;
- Using any type of object as a carrier of hidden data;
- Data could be also written directly into end-user device memory with embedding hiddenly into a carrier;

The design introduced the concept of the memory sector created from n devices. For such memory sector the mechanism of addressing the carrier of data (end-user device) by one of n hashtags is still applicable inside this memory sector. This idea was needed to solve the incompatibility between number of a few services in StegHash/SocialStegDisc [6] [7] and hundreds of devices in the fog architecture. The trade-off is a need of a new layer for the communication on the level of the memory sectors and for managing the allocation procedures over them. To support this operations, a new type of a distributed filesystem would be explored, but this is planned for the future extension on that topic by authors.

B. Level of Security Provided by the Design

The security of the design is provided from two perspectives:

- The fog of devices and dispersion of data between them supported by a type of a logical chain created by StegHash [6] indexing concept.
- Security of a data transmissions provided by the StegAgent of TrustMAS [10] platform;

For the fog architecture, security is majorly ensured by the distribution of data and by the character of logical connection between the parts. The system could only be compromised when the adversary takes over the generation function. The adversary could sniff the network or compromise one or a part of devices, but the captured parts of data are unusable without knowledge about logical connection between them. It could look like a chaotic set of bytes. The authorized operators of the system could still retrieve data properly as they have a knowledge of the correct connection chain among parts of data. Only compromising the generation function module or taking over a unit of it is the strongest threat, so any mechanism for triggering the automative destruction would be taken under consideration.

Another level of security is provided by application of parts of TrustMAS [10] platform. For the design proposed in this paper, we would apply a trust management system for communicating agents and agents communicating through covert channels. Main trust model proposed for TrustMAS is based on following the behavior of agents. If they realize an expected scenario and the protocol of communication is correctly executed that means that agents are trusted. TrustMAS utilizes cross-layer steganography what means a capability of using many different types of steganography like network steganography in every TCP/IP layer or application layer steganography through images, videos or text hiding. Furthermore, between two StegAgents a communication path could be created from links with other methods of steganography used by every of them. It has an advantage of adapting the exchanging hidden data what is harder to uncover.

There is one more feature of TrustMAS [10] which is an anonymous technique based on the random-walk algorithm. The message is forwarded by an agent to next in a probabilistic manner, so any other agent cannot conclude about originator of the message. We see the possibility of

including such mechanism in the design of the system, but we omit it now to address it in further research papers. We would consider other anonymity algorithms.

C. Technical Challenges for Implementations

It should be mentioned following Kott et al. in [4] that in a cyberfog security approach, the network might need to manage a complex tradeoff between availability and confidentiality in real time depending on users' tasks and circumstances. Going through the proposed design we see that the every next mechanism of security means adding a greater level of complexity. In Table I. we summed up if the mentioned mechanisms and algorithms affects one of the aspects: memory, time and reliability. We marked the positive effect on th aspect by "+" and the negative effect by "-". "+/-" means that there is no evidence or cause to mark them positive nor negative.

TABLE I. IMPACT OF THE DESIGN ON THE MEMORY, TIME AND THE RELIABILITY OF THE SYSTEM

Design components	Aspect			
	Memory	Time	Reliability	Security
StegHash indexing [6]	+/-	-	+/-	+
SocialStegDisc operations and improvements [7]	+	-	-	+
Memory sector	-	-	+/-	+/-
StegAgent [10] covert communication	-	-	-	+
StegAgent [10] steganographic routing	+/-	-	-	+
Storage system	+/-	-	+/-	++
Messaging service	+/-	-	+/-	++

It is noticeable that all components impose a time penalty on the system. It is caused majorly by the fact that many mechanisms need to be executed in the order and time of computation is obviously higher.

In this paper we focused on the basic aspects such as memory, time and reliability. In the future work, we will evaluate executing the design in the limited environment of Internet of Things' devices, where the consumed energy, available memory and computing power. They are decisive features impacting the successfulness of the implementation of IoT solutions.

V. SUMMARY

In this work a new concept of the communicating system realizing the idea of cyberfog security [4] was presented. The

design combines and adapts a few components such as StegHash [6] for indexing data, SocialStegDisc [7] for filesystem operations and TrustMAS [10] for device-to-device data transmission. As they are adapted for the environment of the fog of devices, they establish a new kind of a secure communication platform. Security is characterized by the fact that the partly compromising of the system does not interfere the operations, whereas captured samples are useless for the adversary.

In the future work we would deepen the design of the system by considering more mechanisms for operations of the platform and for achieving higher level of security. A *proof-of-concept* implementation would be prepared to deliver results from the real working example of the system.

REFERENCES

- [1] D. Evans, "The Internet of Things. How the Next Evolution of the Internet Is Changing Everything". Cisco Internet Business Solutions Group (IBSG). 1. 1-11.
- [2] M. Chiang and T. Zhang, "Fog and IoT: An Overview of Research Opportunities," in IEEE Internet of Things Journal, vol. 3, no. 6, pp. 854-864, Dec. 2016.
- [3] NIST SP 1500-201, Edward R. Griffor, Christopher Greer, David A. Wollman, Martin J. Burns (June 2017), Framework for Cyber-Physical Systems: Volume 1, Overview, <https://dx.doi.org/10.6028/NIST.SP.1500-201>
- [4] A. Kott, A. Swami and B. J. West, "The Fog of War in Cyberspace," in Computer, vol. 49, no. 11, pp. 84-87, Nov. 2016.
- [5] M. Chapman, G. Davida, and M. Rennhard, "A Practical and Effective Approach to Large-Scale Automated Linguistic Steganography", Proceedings of the Information Security Conference, October 2001, pp. 156-165.
- [6] K. Szczypiorski. 2016. "StegHash: New Method for Information Hiding in Open Social Networks". IJET International Journal of Electronics and Telecommunication, 62 (4) : 347-352.
- [7] J. Bieniasz, K. Szczypiorski: "SocialStegDisc: Application of steganography in social networks to create a file system", In Proc. of 3rd International Conference on Frontiers of Signal Processing (ICFSP 2017), Paris, France, 6-8 September 2017
- [8] F. Beato, E. De Cristofaro and K. B. Rasmussen, "Undetectable communication: The Online Social Networks case," Privacy, Security and Trust (PST), 2014 Twelfth Annual International Conference on, Toronto, ON, 2014, pp. 19-26.
- [9] A. Castiglione, B. D'Alessio and A. De Santis, "Steganography and Secure Communication on Online Social Networks and Online Photo Sharing," Broadband and Wireless Computing, Communication and Applications (BWCCA), 2011 International Conference on, Barcelona, 2011, pp. 363-368.
- [10] K. Szczypiorski, I. Margasiński, W. Mazurczyk, K. Cabaj, and P. Radziszewski, "TrustMAS: Trusted Communication Platform for Multi-Agent Systems", In Proceedings of the OTM 2008 Confederated International Conferences, CoopIS, DOA, GADA, IS, and ODBASE 2008. Part II on On the Move to Meaningful Internet Systems (OTM '08), Robert Meersman and Zahir Tari (Eds.), Springer-Verlag, Berlin, Heidelberg, pp. 1019-1035.



StegFog: Distributed Steganography Applied to Cyber Resiliency in Multi Node Environments

JĘDRZEJ BIENIASZ^{ID}, PATRYK BAK, AND KRZYSZTOF SZCZYPIORSKI^{ID}, (Senior Member, IEEE)

Institute of Telecommunications, Warsaw University of Technology, 00-665 Warszawa, Poland

Corresponding author: Jędrzej Bieniasz (Jedrzej.Bieniasz@pw.edu.pl)

This work was supported by the Polish National Centre for Research and Development under Project CYBERSECIDENT/369234/I/NCBR/2017.

ABSTRACT This paper presents a new study about the communication system named StegFog. It is based on applying distributed steganography linked with the concept of cyberfog security. The idea of cyberfog is to secure data by splitting it into numerous fragments and spreading them to end-user devices. The design of StegFog combines steganographic methods previously investigated by the Authors – StegHash and SocialSteg Disc. The reference design of the system is proposed and analyzed in terms of security, reliability, performance and timing. To prove the feasibility of the proposed solution, the prototype of StegFog was implemented, consisting of steganographic nodes operating over BitTorrent P2P networks with a hidden communication channel established by the multimedia files shared within it. Experiments were conducted for the implemented prototype to understand the basic features of the system, such as operational aspects of the address generator constructed using the modified StegHash method, timing of the basic methods within the system – reading, writing and notifying, steganographic rate of the system and overall security provided by the system. The presented results show how system such as StegFog aligned with the cyberfog security concept could be considered as the protection measure for secure network communications. The practicality, challenges and research directions to extend the concept are also discussed.

INDEX TERMS Cyberfog, distributed steganography, fog computing, information hiding, network security, StegHash, SocialStegDisc, peer-to-peer networks.

I. INTRODUCTION

Recently, the main computing paradigms have been rapidly evolving. Centralized computing is generally developing as Cloud Computing and Data Centers with involving some distribution and redundancy. Distributed computing is mostly realized with Edge Computing and Internet of Things (IoT) applications. The examples of such smart and interconnected devices are mobile phones, wearables, sensors, power grids, etc. As both of these computing solutions are solving many industrial and life use-cases, there is a highly need for the methods complementing them, especially in case of cybersecurity. One of such emerging ideas is fog computing, which is recognized by [1], [2] as the area of developing a new solutions for network security, data security or privacy. Cyberfog as one of the concepts of combining fog

computing and cybersecurity was recently introduced by [3]. Such approach is characterized with higher dispersion and distribution of data that establishes cyber attack resiliency. In the case of a data breach, an adversary could steal only part of the data from the system. Furthermore, such data would be structurally dispersed and without knowing the secret, the attack would not have achieved its goals. Such idea assumes that security of networks and data would increase by *maximizing the “foginess” of information* [3]. Whereas these benefits are understandable, there are numerous challenges of cyberfog such as management complexity, network bandwidth, memory consumption, power demand, latency and connectivity.

The presented characteristics of cyberfog is corresponding with Authors’ previous research on distributed steganography. One of the milestones was the concept of TrustMAS introduced by [4]. TrustMAS is described at Related Work (Section II). Recently, the idea of text steganography [5] has

The associate editor coordinating the review of this manuscript and approving it for publication was Byung-Seo Kim ^{ID}.

been combined with social networks into new distributed information hiding methods. StegHash [6] is the method to use multimedia objects as carriers of hidden information and to disperse them across open social networks (OSNs). A logical connection between them is established by means of a mechanism of hashtags as the text markers in the form of #<tag>, are commonly applied in OSNs. In addition, the Author of StegHash proposes an option for applying the StegHash technique to create a steganographic filesystem analogous to the existing classic filesystems, such as FAT (File Allocation Table) or NTFS (New Technology File System). SocialStegDisc [7] follows up the StegHash research by proposing the application of basic concepts of classic filesystems, such as create-read-update-delete operations or defragmentation processes to the original idea of StegHash. Furthermore, time-memory trade-offs were proposed by the design. The summarized research results were presented in [8].

The motivation of this work is to propose a new distributed steganographic system for protecting networks conformed with cyberfog and tackling its challenges. The system needs to be practically implementable with the end-user devices and to offer acceptable tradeoff between availability and confidentiality of data. A combination of cyberfog with distributed steganography was firstly investigated by the Authors in [9]. This paper introduces a new method called StegFog following the theoretical ideas from [9].

StegFog utilizes the indexing scheme introduced by StegHash [6] and provides basic operations like in SocialStegDisc [7]. The design of the system expands them by using the end-users' devices' memory as a carrier for objects with hidden data. The methods of data fragments routing and finding the next parts among such peer-to-peer networks are also proposed. The working prototype of a such system is implemented for testing purposes. Then, it is evaluated in accordance to the challenges defined by [3], especially data fragments distribution, network management complexity, data transfer speed, endpoint disk capacity and computing power consumption. The security of StegFog is mainly ensured by the character of logical connection between the distributed parts of the data. The system could only be properly compromised when the adversary takes over the secret generation function. Without it, the captured parts, e.g., by compromising one or part of a device, would be unusable, thereby fulfilling the basic cyberfog requirement.

The main contributions of this paper are:

- 1) Further development of the initial concept presented in the paper [9] into a new distributed steganography method called StegFog with defining the functional and operational architecture, the underlying communication protocol with all required supporting functions and providing the security evaluation of the design.
- 2) Implementation of StegFog prototype to experiment on the aspects such as timing of operations, fragments management and security of the addressing scheme, reliability and network bandwidth utilization.

The structure of the paper is as follows:

- Section II briefly presents the development of the distributed steganography concept towards practical realizations and implications in the area of using it with cyber attackers in the wild. Then, it discusses the fog computing and cyberfog security concept and how it could be connected with distributed steganography.
- Section III defines the StegFog reference concept and architecture as the extension of the initial idea presented in [9]. The expanded security analysis is also conducted.
- Section IV shows how the prototype of StegFog was developed and implemented based on the reference design introduced in Section III.
- Section V evaluates the prototype of StegFog implemented for the purpose of this research through a few experiments to understand the basic features of StegFog in practical application.
- Section VI concludes the paper with a summary of the results and further research directions that could be developed from this paper.

II. RELATED WORK

A. DISTRIBUTED STEGANOGRAPHY

The term steganographic communication usually covers the exchange of messages via a steganographic channel between one sender and one receiver. For the first time, the term distributed steganography was used in [10], in 2011. Distributed steganography is characterized by communication through a hidden channel between several independent senders and one receiver. In [10] three examples of the use of distributed steganography are given:

- Military use – used by spy networks to communicate over untrusted networks in foreign countries.
- Business use – some investing projects could involve competing companies. Companies should not know about the financial contributions from other companies and the fact they are participating in a fundraiser.
- Knowledge Management use – when individual units should not be aware of the existence of secret information from other units, but it is needed to pull such information together.

In each of the referred examples, the advantage of steganography over cryptographic solutions is hiding the communication between the recipient and the senders. In these cases, any potential observer has no knowledge of the number of parties sharing information, nor are they able to establish their identity.

An example opposite to that mentioned in [10] is when one sender shares information over a steganographic channel with multiple recipients. This approach was named Broadcast Steganography and presented in [11]. In [11] the authors additionally used encryption to make the hidden message readable only to the authenticated users. To illustrate the use of steganographic broadcasting, the authors used the following examples:

- Military application. To detect internal threats, a secret investigative group would use a hidden channel for communication. Thus, the observer is not aware of an ongoing investigation, nor is he able to identify the investigating group.
- Human Rights activism. An activist can bypass censorship or, by skillful authentication of user groups, detect members infiltrating the association.

In [12] the Authors attempted to define a distributed steganographic system. According to the paper, a distributed steganographic system can be described by the following equation:

$$Pr_r[DSD([C]_s, [h]_s, DSE(r, [C]_t, [h]_t, m, e) < C_s >) = m] \geq 1 - \mu(k) \quad (1)$$

where:

- DSE, DSD - pair of probabilistic algorithms
- r - seed
- $[C]_t$ - target set of steganographic channels
- $[h]_t$ - historical log of steganographic channels
- m - message to be hidden
- e - minimal number of channels to preserve the message
- $[C]_s$ - the overt environment
- $[h]_s$ - set of historical channels

According to the Equation 1, a steganographic distributed system should, with a probability greater than or equal to $1 - \mu(k)$, allows the message to be restored using no less than e steganographic channels.

The idea of steganographic communication in a distributed system arose before the definition of a distributed steganographic system was proposed. In 2007, a distributed steganographic router operating in a multi-agent environment was introduced in [4]. The authors focused on ensuring the anonymity of network agents and ensuring hidden communication between them. Steganographic agents are designed to conduct covert communication in more than one layer of the TCP / IP model. In the TrustMAS platform, apart from ordinary agents using the anonymity function, steganographic agents are distinguished as StegAgents (SA). Behavior consistent with the definition of distributed steganography is the discovery process followed by SAs, where once a message is sent, it can reach a target node from more than one intermediate node.

Another area of development in the domain of distributed steganography is combination the concept with a popular Internet services. One of the target environments for such applications are Peer-to-Peer (P2P) network protocols. Such idea is summarized in [13], especially concentrating on application of BitTorrent protocol [14]. The described method of network steganography based on BitTorrent uses the modification of the order of the network packets in the data exchange protocol between nodes. It can be recognized as the timing method of network steganography applied on BitTorrent protocol message order.

The new distributed steganography methods are also developed around cloud computing. The paper [15] is proposing a distributed steganographic filesystem based on computation clusters. The main idea behind the method presented there consists in using multiple files to encode hidden information by means of their relative positions in clusters. Recently, the idea of using cloud computing resources was revisited by papers [16] for multi-cloud setup and its developed version [17] for single-cloud configuration. Distributed steganography was examined as an approach to hidden data in several files featuring leaving less traces than the classical methods. The method proposed in [16] offered the approach to steganography in multi-cloud storage environment with preservation of the integrity of files used as the hidden data carriers. However, such approach requires the management of several cloud storage environments and a more complex scheme to preserve secrecy of the steganographic protocol. [17] improves it by using a single cloud storage environment and reducing the complexity of secrecy management.

Another research results with using cloud storage capabilities for distributed steganography are presented in [18]. The main difference from the previous cited works is that it focuses on distributed steganography method at the level of hidden data carriers (e.g. image files) and storing it within a given system. It proposes an adaptive payload distribution for multiple images steganography stored in a cloud environment. For that purpose, Authors designed two payload distribution strategies based on image texture complexity and distortion distribution. They also provided a theoretical security analysis from the steganalyst's point of view.

B. FOG COMPUTING AND SECURITY TOWARDS CYBERFOG CONCEPT

The increasing adaption of the Internet of Things (IoT) solutions and applications outcomes as that a cyberspace is ever more connected, especially with the heterogeneous and mobile devices. The cyberspace of cyber-physical systems (CPS) [19] on one hand is producing a lot of data via sensors, actuators or RFID/NFC tags. On the other hand many end-user devices, such as smartphones, tablets, and PCs are consuming such data to make decisions and actions. Many components, standards and protocols must cooperate together to efficiently gather, process, and share such data. The fog computing [1], distributed by default, represents a promising solution which offers interoperability, scalability, security, and privacy. It operates as a middle layer between data consumers/producers and traditional cloud computing environments. Authors of [2] are analyzing the evolution in modeling the new methodologies related to fog computing and IoT. Paper shows how moving security and privacy tasks toward the edge of the network provide both advantages and new challenges to be faced in this research field.

One of the proposed solutions of how to apply fog computing for security is the cyberfog security approach was introduced by Kott *et al.* in [3]. They proposed to use fog computing as a method for mitigating a cyber adversary with

uncertainty could provide greater attack resiliency. One of the functions to ensure security in the cyberfog approach is the division of valuable information into parts and their dispersion among different devices. Prof. Kott referred to the division of the secret according to Shamir's pattern [20]. This behavior is intended to increase the resilience of the system in case parts of the network are compromised. It assumes that the adversary should not be able to recreate the secret with fewer parts of the secret than the assumed threshold allowing for its complete reconstruction. The dispersed nature of the system also means that the attacker of such a system will have a difficult task, because to disable it, it will be necessary to take over or destroy a large number of devices that can be physically located in different places. To ensure data availability in a changing network, Prof. Kott additionally proposed the use of data fragment replication, known from database products, such as Apache Cassandra. Thanks to replication, it will be possible to recreate the required number of secret fragments despite the compromised part of the network.

The challenges with such an approach are mentioned in [3] and include a few aspects. The first one is how to distribute fragments of a secret. Fragments should be divided and distributed in the system in a way that prevents a potential adversary from knowing how they are dispersed to predict the addressees of the following parts. Another challenge is a situational awareness. Not all data fragments are equally important in the context, so that the data should be divided in such a way that the potential intercepted data fragments are of little significance to the adversary. The authors of the cyberfog concept also suggest obfuscating the information contained in the scattered fragments so that, if intercepted, there would be several equally probable interpretations of them. Going a step further, the data can be additionally secured so that, without knowing how to read them, they could lead the adversary to misinterpreting them. These mechanisms can be implemented not only at the information level, but also at the level of network traffic or network topology. More general challenges also include the level of network management complexity, data transfer speed, limiting endpoint disk capacity and computing power consumption.

C. StegFog IN CONTEXT OF THE RELATED WORK

As it was mentioned in Introduction (Section I), a combination of fog computing and cyberfog security with distributed steganography was firstly investigated by the Authors of this paper in [9]. They provided the idea of how to combine their previous works into a new way of secure communications aligned with the cyberfog concept. Based on the related work walk-through, there is a need of designing, implementing and validating a new security mechanisms in fog computing approach. This paper complements the current state-of-the-art in distributed steganography research by the comprehensive design of a new method characterized as the hybrid one and applied for the rising paradigm of fog computing. It further combines three types of classic steganography

approaches into the one distributed steganography method StegFog:

- using any file steganography method, e.g. image steganography on at *data-at-rest* level
- using text-based steganography method to hiddenly manage pointers to hidden data fragments (*data-at-rest* and *data-in-transit* level)
- using network protocol steganography on *data-in-transit* level

The paper is also presenting the implementation of a working prototype, which could be considered as one of the first working systems aligned with cyberfog approach defined by [3]. Then, it made possible to empirically measure several features of the system, such as as data fragments distribution security, network management complexity, data transfer speed or required endpoint disk capacity.

III. REFERENCE ARCHITECTURE OF StegFog SYSTEM

The idea behind this paper is to extend examination if the StegHash [6] and SocialStegDisc [7] mechanisms can be apply for communication systems for the fog in the devices. The paper [9] introduced the idea which is reminded by Section III-A. Section III-B generalizes the layered architecture of StegFog beyond the one presented in [9]. Section III-C evaluates security aspects and requirements of StegFog system. It includes the conclusions formulated in [9] and develop it further with [21] security assessment approach.

A. StegHash/SocialStegDisc APPLIED FOR StegFog

In StegFog there is no public and open internet services, like social networks in the original idea for both StegHash and SocialStegDisc. Instead, the system would operate between a defined set of end users' devices connected into Wide Area/Local Area networking setup. The next new element is defining the size of the problem. Previously, indexing the services that carry the uploaded steganograms could be indicated by one of the n hashtags in an index. In StegFog it is impossible, as the number of carrying devices increases many times. It could be approached by assuming that n devices would create a *memory sector* of size n . For each memory sector the mechanism of addressing the carrier of the data (end-user device) by one of the n hashtags would still be applicable within the memory sector. The tradeoff in such a concept is that the system would need a new layer for the communication on the level of the memory sectors. Every device would be locally associated with one memory sector from one of the many available in the system. It means that every device has its own instance of a memory controller to serve all the basic filesystem operations inside the memory sector working with algorithms introduced by StegHash/SocialStegDisc. During sending or storing data in the system, such a controller disperses fragments between n devices. Every device needs to have a copy of an allocation table for any memory sector and to support multiple membership devices in the memory sectors. As part of the design of

the system for data storage and access schemes between the memory sectors are considered:

- Access through memory sector gateways. When a device saves the data to a memory sector it would notify the other devices that starting from the selected index there is data of B bytes stored. The storing device is selected as a gateway to access the stored data. This approach combines methods of routing and the file allocating methods. This approach introduces a synchronization of allocated addresses across the memory sectors and a state of generation function within the memory sector. Finally, the other devices have a global view of the allocated data.
- Direct access to read from any memory sector. In comparison to the first proposition, the routing information for uploaded data requires extensions to have all n device identifiers with an algorithm to select the following devices for retrieving data. The routing records are shared with every single upload to the system and cached by devices (proactive way).
- Hybrid approach of 1) and 2). It assumes sharing only metadata of the file and a selected gateway with every single upload to the system. When the data is retrieved, at first the request for the memory scheme is issued to the gateway. The gateway responds back with all n devices' identifiers and an identification scheme. Next, the requested part of the data could be read directly from the memory sector.

B. StegFog LAYERED ARCHITECTURE

Generally, the system needs to provide two network services for communication:

- Sending data end-to-end between network nodes. It could be provided by TCP/IP transport layer protocols – TCP or UDP. The other protocols from upper layers could be also considered.
- Reachability between network nodes. The devices within the system need to establish a network, so routing is necessary. Network addresses could be distributed with a static routing configuration or through a dynamic routing protocol.

The described network communication should be built upon a platform that offers hidden distributed communication while preserving privacy. Any device in StegFog would work as a *StegPeer* utilizing steganographic components to provide covert channels to realize communication. The concept of StegFog expands beyond the original ideas [6], [7] by:

- Using the end-users' devices' memory as a carrier for objects with hidden data.
- Using any type of object as a carrier of hidden data.
- Data could also be written directly into the end-users' device's memory by embedding it into a carrier.

The proposed concept of StegFog could be described using layers abstraction as presented in Figure 1. The layers included in the design which are utilized by each peer:

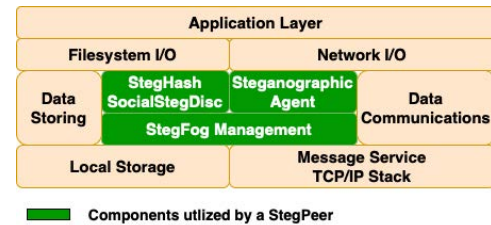


FIGURE 1. The layered architecture of StegFog.

- Application layer – an application that uses the system from the perspective of the user.
- Filesystem I/O and Network I/O – a standard system API for a filesystem and network operations
- Data communications layer – provides the interface for preparing data to be sent through networks to the other peers.
- Message service layer – it utilizes the standard TCP/IP protocol stack for any communications to be occurred between endpoints.
- Data storing layer – provides the interface for the application layer with read-write-delete operations inside the device and with managing point-to-point connection sessions; it passes commands to lower the layers.
- Local storage – represented as a non-volatile memory in the device, where data is stored.

The other layers marked as green in Figure 1 extend any peer into StegPeer with:

- StegHash/SocialStegDisc layer – the implementation of the StegHash [6] indexing method, SocialStegDisc [7] memory management and any of steganographic methods on files to hide data within the local storage [13].
- StegFog management layer – defining a set of operations on the memory sector as a group of n devices. It provides the view of the memory sector for the StegHash/SocialStegDisc layer and it manages the state of the memory sector. Furthermore, they cooperatively manage the scheme of allocation of the memory over devices, memory sectors and the whole storage system.
- Steganographic Agent – treated as a service using steganographic techniques for communications. It combines the different types of steganography such as network steganography over TCP/IP protocol stack and multimedia steganography as it is needed in StegFog.

C. SECURITY EVALUATION OF StegFog SYSTEM

The security of StegFog could be evaluated from two perspectives:

- Security of the data transmissions provided by the hidden communication layer such as TrustMAS [4] described in Section II-A. Many methods for steganography could also be applied. A summary of them is presented in [13].
- Security based on dispersion of data between devices with a logical chain created by StegHash [6].

TABLE 1. Evaluation of security requirements for StegFog.

Security criterion	If required?
Availability	YES
Usability	YES
Integrity	MIXED
Authenticity	YES
Confidentiality	YES
Ownership	MIXED

It is complemented by cyberfog security evaluation based on a well-recognized framework [21].

In StegFog, security is provided by the distribution of data and its logical connections. The resilience of the system relies on the secret of a generation function that creates the usable chain from dispersed data. When an adversary attacks the network to compromise one or a part of the devices, it would be unusable without knowledge of how to retrieve the data logically. The remaining data still could be properly read even if some parts are missing. Compromising the generation function module or capturing a device, they are very high threats, so a mechanism for triggering the automatic destruction would be taken under consideration.

This paper introduces the security analysis of StegFog by means of the cyberfog approach. It is based on the template from [21] with an extension of the classic model of the information security CIA triad (Confidentiality Integrity Availability), with three new aspects: Usability, Authenticity and Ownership. Below is a short description of the criteria of such an analysis that will be performed for the StegFog concept:

- Availability - information should be available at the request of the owner.
- Usability - information should be available in a form that allows it to be read effectively.
- Integrity - all information holders should have a consistent view of the information.
- Authenticity - the owner of the information should have access to the author of the given information.
- Confidentiality - an entity that is not the owner of the information should not have access to it.
- Ownership - the owner of the information should have access to it.

Table 1 presents the evaluation results of the analysis by assigning YES, NO or MIXED answers regarding the fulfillment of the security requirements. To summarize:

- Availability is achieved by dispersing pieces of information between end devices. Any potential attack needs to target independently operating systems. There is one known operational single point of failure (SPOF), which would lead to the unavailability of the entire system.

- Usability is realized by the introduction of an information sharing mechanism similar to the sharing of Shamir's [20] secret. Shamir's secret breakdown assumes that the information is useful - recoverable, given the k of the n message part.
- Integrity of information is not considered by cyberfog, but it could be provided by the dedicated methods in the practical implementation, such as checksums or digital signatures.
- Authenticity is provided by default in the concept. To mimic the node of the StegFog, an attacker would have to take over or modify all pieces of information that, according to the scheme of action, are distributed among the end devices. Such a scenario is highly unlikely to be realized.
- Confidentiality is realized by default in the concept and could be hardened by implementation. Nodes that store pieces of data do not have access to all information. Fragments can be encrypted before the dispersing procedure, making them impossible to be read by a potentially hostile node. Notifications about a new piece of data in the system can also be encrypted. StegFog by design assumes that nodes in the network that are not instructed to store data would not be even aware of its existence among other devices.
- Ownership in cyberfog is based on the assumption that the owner of the information can request access to a given fragment at any time. In the case of compromising the node storing some information, it can be secured against interception of a given fragment using physical methods - device destruction. This is the situation that could compromise the original ownership of data in cyberfog.

In the cyberfog security approach [3], the network needs to manage a complex tradeoff between availability and confidentiality in real time depending on the users' tasks and circumstances. Evaluating the proposed design it is noticed that each mechanism of security impacts the level of complexity significantly. Table 2 compares the impact of the design on security together with operational features: memory, time and reliability. The improving effect imposed by a component on a feature is marked by \nearrow . The negative effect is represented by \searrow , i.e. higher memory consumption, slower time of operations or worse reliability. *N/D* means that there is no evidence or cause to mark them as improving nor negative effect. It is noticeable that all components impose a time penalty on the system. It is caused mainly by the fact that many mechanisms need to be executed in the order and time of computation is obviously higher. The general tradeoff of the system is to improve security in cost of memory, time and reliability.

IV. StegFog PROOF-OF-CONCEPT IMPLEMENTATION

A. INTRODUCTION TO PRACTICAL REALIZATION OF StegFog SYSTEM

In Section III the reference design of StegFog is introduced. This section presents the practical realization of the concept

TABLE 2. Evaluation of StegFog components in regard to how they are impacting system's features. The improving effect - ↗, the negative effect - ↘, N/D - nor improving or negative (no evidence to mark ↗ or ↘).

System component	Security	Memory	Time	Reliability
StegHash indexing scheme	↗	N/D	↘	N/D
SocialStegDisc operations scheme	↗	↗	↘	↘
Steganographic protocol agent	↗	↘	↘	↘
Memory sector	N/D	↘	↘	N/D
Storage system	N/D	N/D	↘	N/D
Messaging service	N/D	N/D	↘	↗

as a proof-of-concept implementation of elements of the system for empirical verification of its features. StegFog at its conceptual level is aligned with the peer-to-peer (P2P) networking approach, therefore the methods presented in the following sections could be applied by each node participating in the system. For this implementation, the protocol assumes that nodes know the identifiers of other nodes in the network. The mechanism of discovering and adding new nodes to the system are beyond this paper. Furthermore, the implementation skips the mechanism of the group secret determination, which is a random seed for the implemented generator addressing message fragments. This problem could be solved by using known protocols, such as Tree-based Group Diffie-Helman (TGDH) [22].

The key element of the implemented prototype is the addressing algorithm, which was created based on the generation of addresses proposed in the StegHash method [6]. This implementation assumes the existence of one or more spaces visible to all nodes participating in the protocol allowing the reading of the materials added to a given space. In addition, the spaces should not restrict the protocol methods and allow for the addition of new materials. Finally, the protocol does not require permission to delete or modify elements in the common space.

B. ADDRESSING MECHANISM FOR StegFog

In StegFog, the hashtags that are elements of the address in the StegHash method are replaced with alphanumeric strings. It provides that the nodes implementing the protocol can unambiguously cast the last element of the address to the node ID in the network. The common feature for the addressing in StegFog and StegHash method is the presence of a common secret, which is the random seed for the address generator. It is crucial that this secret is known only to the participants of the protocol, as this information is critical to the security of the protocol. Another common feature is the location of the address element responsible for indicating the place to look for a given piece of information. This element is at the end of each permutation. The address generation uses the

mechanism of permutation counters proposed in [7], which allows for memory optimization and no need to store address dictionaries. The address generator designed for the needs of the protocol was additionally extended with the possibility of building addresses from elements that are not mapped in the node identifiers. The set of strings given to the generator input can therefore be divided into two subsets, containing:

- Significant elements - elements related to the network node identifiers.
- Insignificant elements - additional elements, not related to the protocol.

A critical element, necessary for the correct operation of the address recovery algorithm is the use of a random seed with the same value as for the generation of addresses. The seed value may change for subsequent address generation and recovery operations. The protocol does not mandate the method of data fragmentation. The requirement for correct addressing of information fragments is assigning them in the order of the generated addresses: address 1 → fragment 1, address 2 → fragment 2 ..., thus they will be correctly recreated in the reading procedure. The protocol assumes the use of steganographic channels to distribute pieces of data. The protocol in StegFog is independent of the steganographic method applied, but it requires a few elements to be stored by each steganogram:

- Data fragment.
- Address of given fragment.
- Starting address (starting permutation) needed to recover the following addresses.
- The iteration count needed to recover the next address.
- Flag coding the message type: '0' - data fragment; '1' - request a fragment of the message; '2' - notification about the availability of a new message; '3' - response to fragment request.

C. COMMUNICATION METHODS SUPPORTED BY StegFog PROOF-OF-CONCEPT

Three basic methods of the protocol are supported by the StegFog prototype: sending the message in StegFog, notifying the recipient that the message is available and reading the message from the system. Sending the message, as shown in Figure 2, can be divided into the following steps:

- 1) Message to be sent at a node is passed to the StegPeer module where the message is split into fragments (Frag).
- 2) The message fragments are placed into carriers using the selected steganographic method to create steganograms (Steg).
- 3) Steganograms are then addressed using strings obtained from the addressing algorithm. The iteration count and flag of the message type are also added to the steganogram.
- 4) Steganograms are then published in a shared space.
- 5) If the steganogram is addressed to the node reading it and the flag is set to '0', the node saves the data in the

local memory in the form: <fragment address, <permutation count, message fragment > >. If the steganogram is not addressed to the node it does not take any further steps.

To read the stored data, the recipient of the message should be informed about the possibility of downloading its fragments. The notification message consists of: identifier of the node to which the message is directed, a flag code of the notification message type and the address of the first fragment in the message. The notification is made available through a steganographic channel, which makes it invisible to peers using the shared space, but not participating in the StegFog. It is shown in Figure 3.

The trust between the parties is assumed by default, however, it is possible to extend it with additional encryption of the content. It could provide protection against attacks within the network and unauthorized access attempts. Reading data from StegFog is more complex than sending and notifying procedures and follows the order:

- 1) Request data fragment with the address received in the notification or generated by the algorithm. The content of the request consists of: data fragment address, identifier of StegPeer sending the request and the flag informing that it is a request for a fragment. The request is hidden in a steganogram and then put into the shared space where it can be retrieved by other StegPeers.
- 2) StegPeers fetching new data from a shared space. If a requested fragment with a given address is in the local memory of a node it makes this fragment available in the shared space. This is realized by preparing and sharing steganogram with a data fragment, a permutation counter, the identifier of the node that requested the fragment and a flag code for a response type.
- 3) StegPeer requesting data monitors the space and downloads any multimedia materials added there. The steganogram including the flag that indicates the response to the request and the correct identifier of the requesting node is saved in its memory. It is repeated until the value of the counter that indicates the end of the whole message by '0'. The node assembles the data fragments in the correct order.

The Reading scheme is presented in Figure 4.

D. PROTOTYPING StegFog SYSTEM

For the purpose of this research, as the building block for the platform, the BitTorrent protocol [14] was utilized, for which the operations are realized by similar nodes that share files with each other. It enables the establishment of communication in a decentralized environment of nodes easily using off-the-shelf open source software. The choice of the BitTorrent protocol is also justified due to the need to implement a steganographic channel. The basis of steganographic communication is the universality of the carriers used to create steganograms. Recently, there is an increase in the network traffic using the BitTorrent protocol [23]. The percentage

of traffic is not as high as it was in 2011, where it was 52.01% for North America, but according to [23] it is was 32% for the Middle Eastern, European and African regions in 2018. It suggests that the BitTorrent protocol is recognized as a viable system to embed steganographic communication due to its prevalence. The following elements were used to implement the prototype of the steganographic node of the StegFog system and the runtime environment:

- A library with development API of BitTorrent protocol [24].
- The Java programming language, by which the StegPeer node is implemented. The compatibility with the BitTorrent library needs to be assured.
- B-Encode library [25] used to implement a custom metadata generator with.torrent extension compliant with BitTorrent protocol specification [14].
- Tracker [26] server used in the BitTorrent protocol to track torrents that exist in the network and peers (BitTorrent protocol clients).
- Containerization based on Docker to create a network for the test environment, prepare images for the tracker and peers and to make testing runtimes repeatable.
- Logging enabled for research on the protocol.
- Automating the environment using Bash scripts.

Figure 5 shows the logical view on the network where StegFog is prototyped. StegPeer can participate in the BitTorrent protocol just like other network nodes. The BitTorrent protocol relies on two networking protocols: TCP used for data communication between Peers and HTTP for communication between any Peer and the Tracker server. The Tracker server tracks Peers and torrents available on the network. On request it provides a list of Peers that have a given fragment of the message, so that Peer knows where to get the interesting fragment. The Shared Space shown previously in Figures 2, 3 and 4 is represented in BitTorrent by:

- Utilizing files with the description of the torrent with the extension.torrent, which are often hosted by other websites.
- Utilizing a memory space in devices that have downloaded a given fragment of the torrent by making them available to the other nodes.

The BitTorrent uses the underneath of the Kademila protocol [27] for its operation. Existing research, such as [27], suggests how to harden communication based on the Kademila protocol by optimizing the protocol parameters. The resilience of the network could be increased by the proper setup of a bucket size k-bucket, where routing information to other network nodes is stored. The tracker server could be recognized as a single-point-of-failure, however it is possible to run the system without a tracking server by including the BitTorrent DHT protocol controller [28]. Operating without a tracking server is aligned with the cyberfog concept (Section II-B). Researching this aspect is beyond this paper and it could be investigated in the future. Still, using the

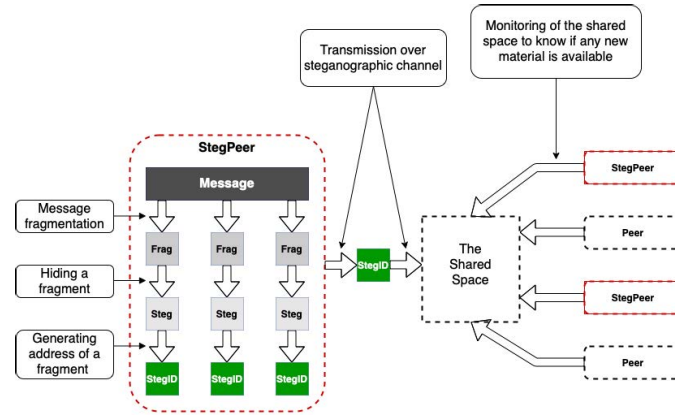


FIGURE 2. StegFog communication methods: Sending.

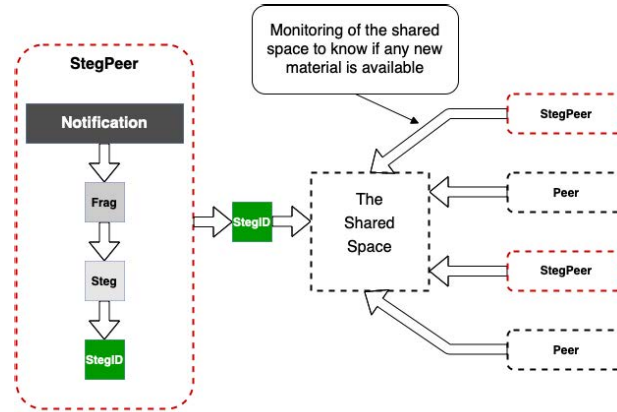


FIGURE 3. StegFog communication methods: Notification.

Tracker server is closer to real networking environments where the BitTorrent protocol is used.

V. EXPERIMENTS

A. SIMULATION ENVIRONMENT

To conduct research on the implemented prototype of StegFog, three simulation networks were deployed consisting of four, six or eight nodes together with the tracker node (Tracker). The tests were performed in the cloud environment as a virtual machine with Ubuntu 16.04 installed. The instance had 60 GB of RAM and a 16-core 6th generation Intel Xeon CPU. The software modules were managed by Docker and its relevant configuration files. The first step to prepare the simulation environment was to create subnets for future containers. In the next step, the tracker docker image was created based on the public system image Ubuntu 16.04 and the BitTorrent tracker library [26]. The container, which was created on the basis of the tracker image, was configured to enable outside communications and to monitor

the current statistics, such as the number of connected peers, their type and the version of the BitTorrent protocol client. The image of the communication node was created with Ubuntu 16.04 and the correct Java runtime environment. TheStegPeer image exposes ports 6891 and 49001 for the BitTorrent protocol library [24]. It also creates the required folder structure to copy the carriers (multimedia files) to be used by StegFog to the runtime of a container. Alternatively, the StegPeer container provides logs and torrent files, which can be accessed from the hosting system through a web portal or command line interface.

B. EVALUATION OF StegFog PROOF-OF-CONCEPT IMPLEMENTATION

1) ADDRESSING SCHEME GENERATOR ANALYSIS

The address generator described in Section IV-B differs from a simple hash function for a given input string by combining significant elements to identify a steganographic node with insignificant elements of pseudorandom characters.

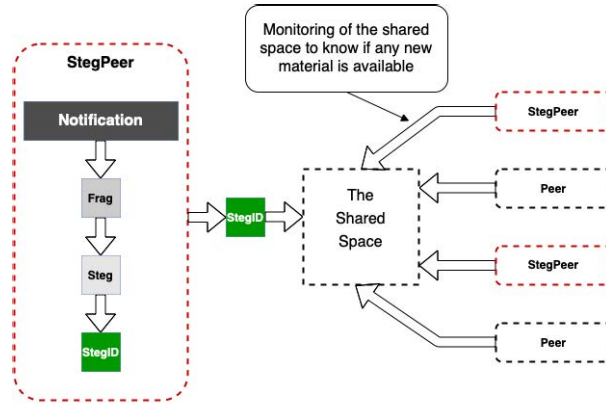


FIGURE 4. StegFog communication methods: Reading.

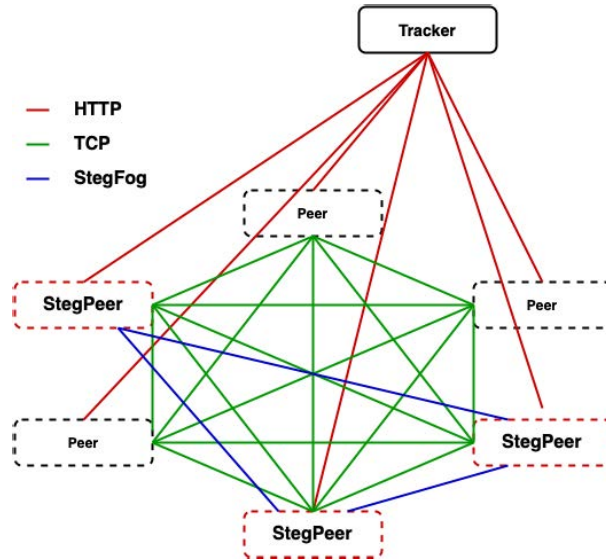


FIGURE 5. Logical view on the network with StegFog communication applied.

Referring to the concept presented in Section III to correctly address a given fragment of a message, the identification substring should be placed at the end of the set given at the input of the generator. This procedure limits the space of possible combinations of the input set. The maximum number of usable addresses can be calculated with the following equation:

$$P = k * (n - 1)! \quad (2)$$

where:

- P - the maximum possible number of permutations to generate addresses.
- n - size of the input set to be permuted (sum of significant and insignificant elements).
- k - the number of significant elements in the set.

To investigate the impact of such a procedure used in the address generator the probability distributions of characters appearing in the addresses were analyzed. The hash function was used to construct the reference distribution, which generates permutations of the input set in a pseudo-random manner, without repetitions. This function takes as a parameter the number of addresses needed to be generated and a set of elements. During the experiment, the probabilistic algorithm implemented in the generator and the hash function used the same random seed value. The result of the address generator and hash function is a list of permutations of the elements for the input file in the following format:

iB, Hu, im, Xt, QB, lI, xQ, Hz

Relative entropy, also known as the Kullback-Leibler divergence [29], was chosen as the measure determining the

discrepancy of the probability distributions:

$$d_{KL}(P, Q) = \sum_i p(i) \log_2 \frac{p(i)}{q(i)} \quad (3)$$

where:

- d - relative entropy of the distributions of discrete random variables P and Q .
- p - probability of the sample value of a random variable P .
- q - probability of the sample value of a random variable Q .

This measure is a classic element for information theory with practical considerations in cybersecurity. An example of the use of this measure in cybersecurity is the detection of automatically generated domains by the so-called algorithms DGA (Domain Generation Algorithms) [30], [31], and [32]. These algorithms are implemented in the malware to create C2 channels between infected machines and the control server. Examples of this type of software are Skybot and Styx [33]. To find anomalies in strings, e.g., domain names, file names or processes, entropy is also used. In [34], this approach was used to detect anomalies in the domain names. Furthermore, it was shown by [34] that the relative entropy allows the detection of the differences between the tested samples more clearly when compared to the classical entropy.

The set consisting of the addresses obtained from the generator and the hash function was modeled using the n -gram model [35]. This model is used in cybersecurity, for example, to analyze executable files for the presence of malware [36]. N -grams are also used in other fields such as biology [37] and computational linguistics [38]. In this study, in addition to understanding the impact of the disturbance in the probability distribution introduced by the generator based on the StegHash [6] method, four other questions are raised:

- Question 1: How does the size of n in the n -gram model affect the presence of the disturbance?
- Question 2: What is the influence of the size of the examined address space on the presence of the disturbance?
- Question 3: How does a change in n in the n -gram model and address space size affect the entropy of distributions?
- Question 4: Does changing the ratio of identifying elements to insignificant elements affect the relative entropy of the address distributions as defined by Eq. 3?

Converting the set of addresses to the form n -grams consists of concatenating all addresses in the set into one string and creating the appropriate length n -grams. In the examined case, unigrams (1-gram), bi-grams (2-gram) and tri-grams (3-gram) were generated. The n -gram distribution is discrete, therefore the probabilities of n -gram occurrences in the set are calculated based on the number of occurrences of the given n -gram. Before calculating the relative entropy, the previously calculated n -grams probability vectors from the different distributions must be standardized. The standardization aligns the lengths of the probability vectors. This is realized by

adding the appropriate number of zero probabilities n -grams, which previously did not appear in a given distribution. However, these distributions still need to be smoothed out to eliminate the zero-frequency problem [38]. For n -grams derived from the second distribution, a very small, non-zero, fixed-value probability is assigned, and all other probability values from the original distribution are reduced by that fixed amount so that the sum of the vector probabilities is still 1. The results of the measurements: entropy of addresses from the generator, entropy of addresses from the hash function (reference permutation) and relative entropy of these distributions, are presented in the Figures 6, 7 and 8.

The samples are from the address space range 10 through 20110, in increments of 200. During the measurements, the system was configured with the following parameter values: the number of significant elements is four and the size of the set to be permuted is eight. According to Eq. 2, the maximum number of addresses that can be generated in this configuration is 20,160.

2) HOW DOES THE SIZE OF N IN THE N -GRAM MODEL AFFECT THE PRESENCE OF THE DISTURBANCE?

Comparing the results of relative entropy measurements from Figure 6, it can be concluded that the selection of the n -gram model has an impact on the visibility of the disturbance introduced by the address generator. In the case of the uni-gram model, the relative entropy of the distributions is zero, while for the bi-gram and tri-gram it is positive and answers Question 1.

3) WHAT IS THE INFLUENCE OF THE SIZE OF THE EXAMINED ADDRESS SPACE ON THE PRESENCE OF THE DISTURBANCE?

Based on the results presented by Figures 6, 7 and 8, it is seen that the disturbance introduced by the generator is most visible for the small size of the compared address space, especially below 100 addresses. This behavior is presented in Figure 6 by higher values of the relative entropy and in Figures 7 and 8 by larger differences in the entropy values from two different distributions, compared to larger portions of the address space. This observation answers Question 2.

4) HOW DOES A CHANGE IN N IN THE N -GRAM MODEL AND ADDRESS SPACE SIZE AFFECT THE ENTROPY OF DISTRIBUTIONS?

Figures 7 and 8 show that the size of the investigated address space has no influence on the entropy values. Based on the entropy of the distribution from the generator (Figure 7) it can be seen that the values from this distribution do not show an upward or downward trend. Referring to Question 3, the entropy value increases depending on the selection of the n -gram model, which results from the structure of n -grams. Along with the increase in the parameter n , the number of characters in n -grams is increasing, which, if the characters are different, allows you to store more information in them.

Another observation seen for the bi-grams in Figure 7 and tri-grams in Figure 8 is the fact that in most measurements

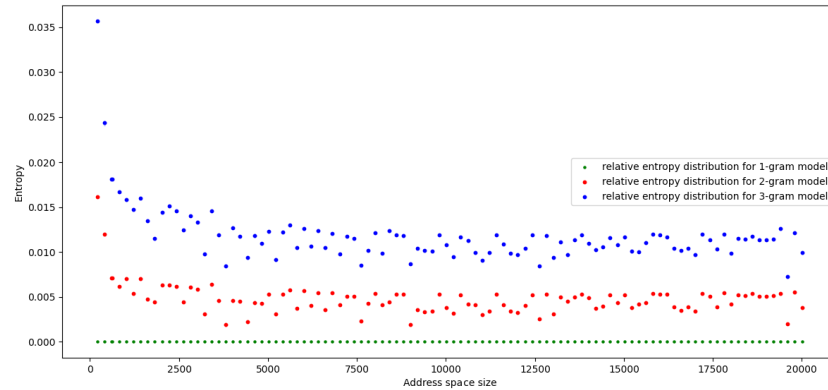


FIGURE 6. Comparison of relative entropies based on 1-gram, 2-gram and 3-gram models of strings from the generator and from the hashing function.

the entropy values from the address generator distribution are smaller than those from the hash function. In about one hundred samples visible in the discussed graphs, only in two cases the value of entropy coming from the generator is higher than that of the hash function - in the bi-gram model and only once in the tri-gram model. For the unigram model, n -grams reach the same values, therefore the influence of the generator is not visible.

5) DOES CHANGING THE RATIO OF IDENTIFYING ELEMENTS TO INSIGNIFICANT ELEMENTS AFFECT THE RELATIVE ENTROPY OF THE ADDRESS DISTRIBUTIONS?

To answer Question 4 and examine the influence of the ratio of significant elements to non-significant elements, the following system parameters were setup: length of permutation is 24 and size of set to be permuted is 16. These parameters cause the address space to increase to 159,667,200. The number of significant elements remained the same as four, and the number of insignificant elements was doubled to eight. The bi-gram model will be used to investigate the difference in relative entropy, and the samples will be in the range from 10 to 20110, with a step of 200. Based on Figure 9 in the analyzed address space it can be observed that in the vast majority of this space, the relative entropy with the hash function reaches lower values for the system settings with the ratio of significant to insignificant elements equal to 1:2 in comparison to the 1:1 setting. Thus, by increasing the number of insignificant elements, we reduce the difference in the distribution of addresses from the generator, with the distribution coming from the hash function. The mean value of the relative entropy for a 1:2 ratio of significant to insignificant is 1.6 times smaller than for a 1:1 ratio, which answers Question 4.

6) OPERATIONAL MEASUREMENTS WITHIN StegFog PROOF-OF-CONCEPT SYSTEM

During research on the proposed protocol, a series of measurements was performed to determine its properties

depending on the number of nodes in the network and the size of the transferred data. Additionally, an attempt was made to determine the steganographic bit rate of the implemented system prototype. Two test scenarios were designed to test the timing of the protocol. During the measurements, three phases in the protocol were distinguished: sending, notification, reading.

The first scenario consists of sending a message 102 characters long distributed among five steganograms. This test was performed for each of the simulation configurations with 4, 6 and 8 nodes (Section V-A). The testing message was sent between two selected nodes. It was repeated three times for each configuration. The time results for each of the phases are presented in Table 3 in the form of an arithmetic mean with the variance.

Based on the results presented in Table 3, it can be concluded that for the same message size, the duration of the loading and reading phases may differ. The time of the notification phase is constant, regardless of the size of the network. For each steganogram containing a message fragment, a unique address in the network is generated. Table 4 shows the addressees generated per each steganogram for three configurations.

The operation of the address generating function depends on the list of addresses available in the network. The node address is encoded in two characters, so the last two characters of the fragment address determines the node address that is to store the fragment. Comparing the addressing for three different network sizes in Table 4, it can be seen that the number of fragments sent between selected nodes differs depending on the configuration. The same situation occurs when the message fragments are downloaded by the second node during the Reading phase. It comes from one of two situations:

- A node requests a fragment that is stored locally on that node.
- A node wants to store a fragment and the protocol selects its own memory space to do so.

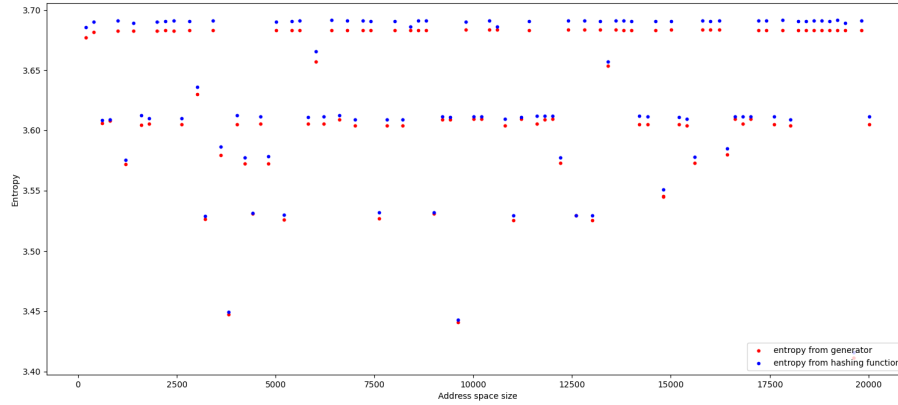


FIGURE 7. Comparison of entropies between 2-gram models of strings from the generator and from the hashing function.

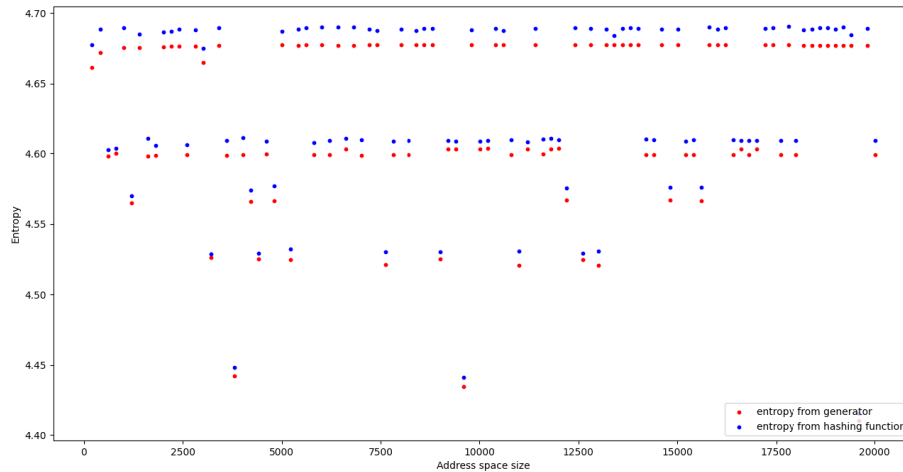


FIGURE 8. Comparison of entropies between 3-gram models of strings from the generator and from the hashing function.

TABLE 3. Time measurement results in Scenario 1 of StegFog operations. (T - time, Var - variance).

Protocol phase	4 nodes		6 nodes		8 nodes	
	T [s]	Var [s]	T [s]	Var [s]	T [s]	Var [s]
Sending	187	2	307	0	307	1
Notification	16	1	17	0	16	2
Reading	383	6	303	27	370	54

These time differences were not visible during the Notification phase, because in this case the information is always hidden in one steganogram. Taking into account these observations, Table 5 was prepared showing the time dependencies between the shared and downloaded steganograms and the network size. The unit of values in Table 5 is S [s / 1 steg], which means the number of seconds needed to transfer one steganogram in the tested configuration of nodes.

The difference in the values between the Sending and Reading phases is that:

- During sending the node makes the message fragments available for download via *seeding* mechanism of BitTorrent [14].
- Reading of each fragment consists of sending the request to provide a fragment and the fetching of the fragment from the node containing it in the *leeching* mode [14].

As can be seen in Table 4, along with the increase in the number of nodes in the network, the number of characters addressing the fragments has increased. Based on results showing the average times for sharing and downloading the steganogram (Table 5), it can be concluded that the transfer of the steganogram in the network of nodes implementing the StegFog concept is independent of the number of nodes in the network, as well as the impact of the addressing size being negligible.

In the second scenario, the important parameter is that the message length embed into a steganogram is set to 25.

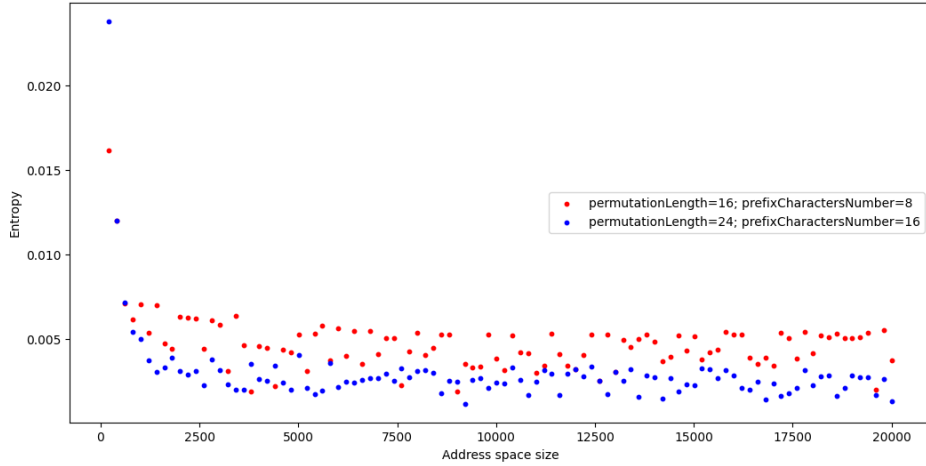


FIGURE 9. Comparison of relative entropies based on 2-gram models of strings from the generator and from the hashing function with the extended set of insignificant elements in the address string (configured by `prefixCharactersNumber` parameter).

TABLE 4. Addresses of steganograms generated in StegFog performance testing scenarios.

Steganogram No	4 nodes
1	D4B2dUA1p9wsGVC3
2	D4dUC3B2GVp9wsA1
3	A1wsdUGVp9B2D4C3
4	p9B2wsA1dUC3GVD4
5	C3dUB2D4p9wsGVA1
	6 nodes
1	qYF63sD4ypA1jWB2E5C3
2	B2A1qYD4E5C3sjWypF6
3	jWA1F6qYC3B23sypB4E5
4	qY3sA1F6D4jWypE5B2C3
5	3sE5D4A1jWqYypF6C3B2
	8 nodes
1	G7H8uTE50qpXafB2A1D4C3F6
2	pXA1E5D4H8F6B20qafuTG7C3
3	E5H8D4A10qafC3pXF6uTB2G7
4	C3F6A1uTE5G70qH8afB2pXD4
5	pXH8D4E5C3uTG70qafA1B2F6

Three messages with lengths of 50, 100 and 200 characters were prepared for sending between two selected nodes. Such communication means fragmentation into 2, 4 and 8 parts, respectively. Observing the results presented by Table 6 with the corresponding plot in Figure 10, a linear increase in the time for sending and reading messages depending on the number of steganograms sent is noticed. Additionally, it overlaps the measurements of the average sending and reading times from Table 6, to check the computational overhead in comparison with the five message fragments. It means that the time differences introduced by the computation are negligible.

7) ESTIMATING STEGANOGRAPHIC RATE OF StegFog SYSTEM

It is not possible to accurately determine the steganographic rate of StegFog as it does not need to be implemented using

TABLE 5. Measurement results in Scenario 1 of StegFog operations, S as $[s / 1 \text{ steg}]$.

Protocol phase	4 nodes $S [s / 1 \text{ steg}]$	6 nodes $S [s / 1 \text{ steg}]$	8 nodes $S [s / 1 \text{ steg}]$
Sending	62	62	61
Notification	16	17	16
Reading	38	38	37

any specific steganographic methods for local data storage on StegPeers and communications between them. The general formula for the steganographic flow in a given implementation can be expressed by the formula:

$$S = \frac{c}{v} \quad (4)$$

where S is the steganographic bit rate expressed in $[B / s]$, c is the carrier capacity expressed in $[B / 1 \text{ steg}]$ - the number of bytes that can be hidden in the carrier and v is the number of seconds needed to transmit one steganogram expressed in $[s / 1 \text{ steg}]$.

The well-known and easy to implement Least Significant Bit (LSB) steganography method was applied to hide data in PNG images with some modifications to extend the number of parameters transmitted with the data. For the purpose of this research, LSB image steganography method is considered as the reference application of a steganographic method within the StegHash/SocialStegDisc layer of StegFog system, especially to streamline the proof-of-concept implementation, experiments and performance estimations. Any other methods such as text and multimedia steganography [13], or more advanced algorithms e.g. [39] or [18] could be used on that layer shown in Figure 1. The concept as [18] could contribute to improve the security of StegFog by adding another distributed approach focusing on multimedia steganography. It could add more security on the lower layer

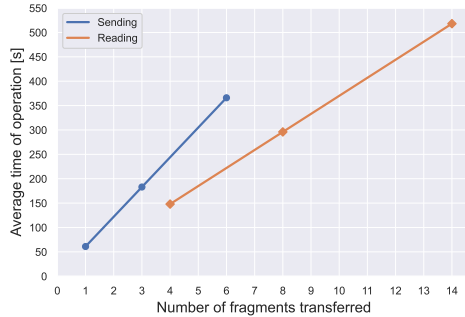


FIGURE 10. Plot of the linear dependence between number of fragments transferred during (Sending; Reading) operations and time to complete each one. Based on Table 6.

of the system as complement to the higher layer mostly developed by this paper.

Two types of carriers were used during the measurements: images in PNG format with a resolution of 1024 x 1024 and 2028 x 1521 pixels. When generating the steganogram, the carriers were collected in a pseudo-random manner, so it was assumed that the same amount was used during the research. The metadata in the steganogram has a size of 39 B. The implementation of the LSB method encodes one bit per pixel. By averaging the number of pixels available in carriers and the size of the metadata, the average capacity of the carriers used was about 26KB.

The data from Section V-B6 was used to calculate the average time in seconds needed for the transmission of one steganogram. The sum of the average times needed to perform the measurements presented in Table 3, and the total number of transferred steganograms in all protocol phases were taken into account. The total time was 1906 seconds, and 44 steganograms were transferred during it, which gives 43 [s / 1 steg]. Putting the obtained results for c and v into the Eq. 4, the steganographic bit rate is 605 [B / s].

C. DISCUSSION AND EVALUATION

This paper analyzes the information security of the cyberfog concept. For verification, an extended template was used, taken from the literature [21], which, in addition to the basic security aspects, such as confidentiality, integrity and availability, also takes into account: usefulness, authenticity and ownership of information. The obtained results show that this method does not fully protect against attacks on data integrity and ownership. Potential extensions of the method as proposed in Section IV-D may allow all security aspects to be met in the final implementation.

For the needs of the research, a distributed steganographic system was implemented, allowing the usability of the protocol prototype to be checked and its basic functions to be examined. However, this is not the final solution. As mentioned in Section IV-D, the next version of the concept should be expanded by including a node discovery mechanism, sharing secrets for the address generator seed, as well

TABLE 6. Summary of the measurements of StegFog operations in Scenario 2.

Aspect	2 fragments	4 fragments	8 fragments
Sending	63 s	186 s	373 s
Sending (avg.)	61 s	183 s	366 s
Notification	17 s	14 s	15 s
Reading	149 s	306 s	528 s
Reading (avg.)	148 s	296 s	518 s
Fragments sent	1	3	6
Fragments read	4	8	14

as developing a link reservation method or enhancing the library [24] to ensure full asynchronous steganographic communication. The protocol features related to the address space size look promising, the maximum value of which can be calculated using the Eq. 2. For the size of the element set input to the address generator n , the size of the space increases $(n-1)!$ times. It is worth noting that the number of signifiers reflecting the protocol node identifiers does not have a significant influence on the size of the space. To obtain a larger space size, the ratio of insignificant to significant elements should be increased.

The results of the relative entropy studies of the probability distributions from the generator and the pseudorandom hash function presented in Section V-B1 do not show any significant disturbance from the generator. For the 1-gram model, they are not recognized at all, and for the 2-gram and 3-gram models, respectively higher, but not exceeding 0.08. For 98% of the samples, the relative entropy of the distributions is below 0.025. From the graphs presented in Section V-B1, it can be seen that the entropy from the generator is in most cases slightly lower than that obtained from the samples obtained with the hash function. When looking at these cases, it is worth remembering that in a real scenario, there is a small probability that the observer would have a perfectly separated set of addresses coming only from steganographic communication. In further research, the degree of undetectability will be assessed in the context of addressing depending on its number among other strings of characters indexing multimedia materials in a given system, in which the protocol was embedded. Section V-B6 presents the timing results of the StegFog operations. Due to the implementation of the prototype, the reading method takes the longest time. For each of the pieces of information not stored on the requesting device, it is necessary to provide a steganogram containing the request for a given fragment, and then retrieve it from the node that owns it. This behavior is determined by the BitTorrent protocol specification. It was empirically verified that the network size for the tested configurations did not affect the average time of sending and reading the steganograms. The results of research related to the time of sending and reading information from the system show a linear increase depending on the number of steganograms involved in the communication. The values shown on Figure 10, are consistent with the average information transfer times for each protocol method obtained in this study. The difference in the average time of sending and reading is related to the protocol implementation. In the reading phase, the sending of the fragment from the node with it to

the requesting node consists of two steganograms. The first one uses the seeding mechanism of the BitTorrent protocol, and the second one downloads the material in the leaching mode. The two modes differ in their average durations, with the seeding mode being longer. The sending phase uses only the longer mode, while the reading phase uses both, which lowers the average steganogram transmission time.

For the implementation of the protocol using BitTorrent, the steganographic bit rate was calculated at 605 [B / s]. However, this value should not be treated as the maximum bit rate achievable by the protocol as it is strongly dependent on the capacity of the carriers and the chosen method of multimedia steganography, which is independent of the protocol concept defined in Section III. The above results and the prototype of the protocol can form the basis for further work bringing it closer to solving real problems, such as storing sensitive data and ensuring appropriate access to it. This example shows that steganography is used not only in malware communication, but can also protect data and privacy of citizens against abuse by the state and other institutions.

VI. SUMMARY

As part of the work, a prototype of a distributed steganographic system implementing a protocol dedicated for a new concept in the field of cybersecurity, and Cyberfog was realized. The conducted research and analysis of the protocol can constitute the basis for determining the manner and scope of the method's use in view of the problems related to the storage and sharing of sensitive information mentioned in the introduction. The results and observations from the conducted information security analysis in Section III-C can be helpful for the further development of the cyberfog method, thus supplementing it with security aspects, such as integrity or ownership. The prototype of the proposed protocol uses the BitTorrent network as a P2P platform offering multimedia sharing in a decentralized environment. The nature of this type of network may make it difficult to detect C2 channels between the managing server and devices in the botnet. The paper presented the simulation environment and the description of the research carried out on the protocol. The address generator has been analyzed. The formula for the size of the address space has been given depending on the parameters given at the generator input. Using the relative entropy and the n-gram model, the disturbance introduced by the generator was examined in comparison to the result obtained from the hash function. How the system parameters related to the generator affect the size of the disturbance as assessed. Additionally, it was examined how the selection of the language model - n-gram, influences the detection of the disturbance. The paper includes research on protocol timing. As part of this bit of the work, the following were verified:

- Influence of network size on execution times of protocol methods with constant information size.
- Relationship between the duration of sending and reading information, and the size of the information.

- Steganographic bandwidth achieved in the simulation environment.

The implemented example of the system can serve as an inspiration for entities dealing with detection and counteracting malicious software and revealing vulnerabilities in websites using the BitTorrent protocol to place steganographic channels in them. In future research, some aspects could be considered:

- In the case of using the protocol in networks with a variable number of nodes, it is worth considering extending it with the method of discovery or registration of new steganographic nodes.
- When developing a system based on the protocol defined in Section III, whose security relies on the secrecy of the address generator, it is worth considering a mechanism for finding a secret or its cyclic change.
- A further stage for research on the address generator based on the adapted StegHash method may be to check the distribution difference between the addresses created by the generator and the selected set of torrent names presented on a given website. The result of this test would be to check the discoverability of the addresses that are torrent names among other torrent names, and not from the address generator.

REFERENCES

- [1] M. Chiang and T. Zhang, "Fog and IoT: An overview of research opportunities," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 854–864, Dec. 2016.
- [2] S. Sicari, A. Rizzardi, and A. Coen-Porisini, "Insights into security and privacy towards fog computing evolution," *Comput. Secur.*, vol. 120, Sep. 2022, Art. no. 102822.
- [3] A. Kott, A. Swami, and B. West, "The fog of war in cyberspace," *Computer*, vol. 49, no. 11, pp. 84–87, 2016. [Online]. Available: <https://www.computer.org/csdl/magazine/co/2016/11/mco2016110084/13rUEgs2PJ>
- [4] K. Szczypiorski, I. Margasiński, W. Mazurek, K. Cabaj, and P. Radziszewski, "TrustMAS: Trusted communication platform for multi-agent systems," in *On the Move to Meaningful Internet Systems: OTM 2008*. Berlin, Germany: Springer, 2008, pp. 1019–1035.
- [5] M. Chapman, G. I. Davida, and M. Rennhard, "A practical and effective approach to large-scale automated linguistic steganography," in *Information Security*, G. I. Davida and Y. Frankel, Eds. Berlin, Germany: Springer, 2001, pp. 156–165.
- [6] K. Szczypiorski, "StegHash: New method for information hiding in open social networks," *Int. J. Electron. Telecommun.*, vol. 62, no. 4, pp. 347–352, Dec. 2016. [Online]. Available: <http://journals.pan.pl/Content/101655/PDF/48.pdf>
- [7] J. Bieniasz and K. Szczypiorski, "SocialStegDisc: Application of steganography in social networks to create a file system," in *Proc. 3rd Int. Conf. Frontiers Signal Process. (ICFSP)*, Sep. 2017, pp. 76–80.
- [8] J. Bieniasz and K. Szczypiorski, "Methods for information hiding in open social networks," *J. Univ. Comput. Sci.*, vol. 25, no. 2, pp. 74–97, 2019.
- [9] J. Bieniasz and K. Szczypiorski, "Towards empowering cyber attack resiliency using steganography," in *Proc. 4th Int. Conf. Frontiers Signal Process. (ICFSP)*, Sep. 2018, pp. 24–28.
- [10] X. Liao, Q.-Y. Wen, and S. Shi, "Distributed steganography," in *Proc. 7th Int. Conf. Intell. Inf. Hiding Multimedia Signal Process.*, Oct. 2011, pp. 153–156.
- [11] N. Fazio, A. Nicolosi, and I. Perera, "Broadcast steganography," in *Topics in Cryptology—CT-RSA 2014*. Cham, Switzerland: Springer, 2014, pp. 64–84. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-04852-9_4
- [12] A. Alston, "A steganographic design paradigm for general steganographic objectives," Columbia Univ., Steganography Project Rep., Spring 2017, pp. 1–30. Accessed: May 30, 2022.

- [13] W. Mazurczyk, S. Wendzel, S. Zander, A. Houmansadr, and K. Szczypiorski, *Information Hiding in Communication Networks: Fundamentals, Mechanisms, Applications, and Countermeasures*, 1st ed. Hoboken, NJ, USA: Wiley, 2016.
- [14] BitTorrent. (2008). *BEP 3: The BitTorrent Protocol Specification*. Accessed: May 30, 2022. [Online]. Available: https://www.bittorrent.org/beps/bep_0003.html
- [15] A. Venčkauskas, N. Morkevičius, G. Petraitis, and J. Čeponis, "Covert channel for cluster-based file systems using multiple cover files," *Inf. Technol. Control*, vol. 42, no. 3, pp. 260–267, Sep. 2013.
- [16] L. Moyou Metcheke and R. Ndongam, "Distributed data hiding in multi-cloud storage environment," *J. Cloud Comput.*, vol. 9, no. 1, pp. 1–15, Dec. 2020.
- [17] S. W. M. Tcheunteu, L. M. Metcheke, and R. Ndongam, "Distributed data hiding in a single cloud storage environment," *J. Cloud Comput.*, vol. 10, no. 1, pp. 1–15, Aug. 2021.
- [18] X. Liao, J. Yin, M. Chen, and Z. Qin, "Adaptive payload distribution in multiple images steganography based on image texture features," *IEEE Trans. Depend. Sec. Comput.*, vol. 19, no. 2, pp. 897–911, Apr. 2022.
- [19] E. Griffor, C. Greer, D. Wollman, and M. Burns, "Framework for cyber-physical systems: Overview," NIST-Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep. NIST SP 1500-201, Jun. 2017, vol. 1. [Online]. Available: <https://www.nist.gov/publications/framework-cyber-physical-systems-volume-1-overview>
- [20] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979, doi: [10.1145/359168.359176](https://doi.org/10.1145/359168.359176).
- [21] D. B. Parker, *Toward a New Framework for Information Security?* Hoboken, NJ, USA: Wiley, 2012, ch. 3, pp. 3.1–3.23. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118851678.ch3>
- [22] Y. Kim, A. Perrig, and G. Tsudik, "Tree-based group key agreement," *ACM Trans. Inf. Syst. Secur.*, vol. 7, no. 1, pp. 60–96, Feb. 2004.
- [23] A. Sachdeva. (2018). *BitTorrent Popularity and Online Piracy is Increasing Again: Here's Why*. Accessed: May 30, 2022. [Online]. Available: <https://fossbytes.com/bittorrent-popularity-online-piracy-is-increasing-again-netflix/>
- [24] BT: A Full-Featured BitTorrent Implementation in Java 8. Accessed: May 30, 2022. [Online]. Available: <https://github.com/atomashpoltskiy/bt>
- [25] B-Encode: Library to Encode and Decode B-Encoded Documents. Accessed: May 30, 2022. [Online]. Available: <https://github.com/adaxi/Bencode>
- [26] BitTorrent-Tracker: Simple, Robust, Bittorrent Tracker (Client & Server) Implementation. Accessed: May 30, 2022. [Online]. Available: <https://github.com/webtorrent/bittorrent-tracker>
- [27] H. Heck, O. Kieselmann, and A. Wacker, "Evaluating connection resilience for the overlay network Kademia," in *Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2017, pp. 2581–2584.
- [28] BitTorrent. (2020). *BEP 5: DHT Protocol*. Accessed: May 30, 2022. [Online]. Available: https://www.bittorrent.org/beps/bep_0005.html
- [29] T. Cover and J. Thomas, *Elements of Information Theory*. Hoboken, NJ, USA: Wiley, 2012.
- [30] S. Yadav, A. K. K. Reddy, A. L. N. Reddy, and S. Ranjan, "Detecting algorithmically generated malicious domain names," in *Proc. 10th Annu. Conf. Internet Meas. (IMC)*. New York, NY, USA: Association for Computing Machinery, 2010, pp. 48–61, doi: [10.1145/1879141.1879148](https://doi.org/10.1145/1879141.1879148).
- [31] T. Barabosch, A. Wichmann, F. Leder, and E. Gerhards-Padilla, "Automatic extraction of domain name generation algorithms from current malware," in *Proc. NATO Symp. IST-111 Inf. Assurance Cyber Defense*, Koblenz, Germany, 2012, pp. 2–12–14.
- [32] MITRE ATT&CK. (2020). *T1568.002: Dynamic Resolution: Domain Generation Algorithms*. Accessed: May 30, 2022. [Online]. Available: <https://attack.mitre.org/techniques/T1568/002/>
- [33] SANS Internet Storm Center. (2015). *Detecting Random—Finding Algorithmically Chosen DNS Names (DGA)*. Accessed: May 30, 2022. [Online]. Available: <https://isc.sans.edu/forums/diary/Detecting+Random+Finding+Algorithmically+chosen+DNS+names+DGA/19893/>
- [34] Red Canary. (2018). *Using Entropy in Threat Hunting: A Mathematical Search for the Unknown*. Accessed: May 30, 2022. [Online]. Available: <https://redcanary.com/blog/threat-hunting-entropy/>
- [35] W. B. Cavnar and J. M. Trenkle, "N-gram-based text categorization," in *Proc. 3rd Annu. Symp. Document Anal. Inf. Retr. (SDAIR)*, 1994, pp. 161–175.
- [36] R. Zak, E. Raff, and C. Nicholas, "What can N-grams learn for malware detection?" in *Proc. 12th Int. Conf. Malicious Unwanted Softw. (MALWARE)*, Oct. 2017, pp. 109–118.
- [37] Z. Volkovich, V. Kirzhner, A. Bolshoy, E. Nevo, and A. Korol, "The method of N-grams in large-scale clustering of DNA texts," *Pattern Recognit.*, vol. 38, no. 11, pp. 1902–1912, Nov. 2005.
- [38] D. Jurafsky and J. H. Martin, *Speech and Language Processing*, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2009.
- [39] X. Liao, Y. Yu, B. Li, Z. Li, and Z. Qin, "A new payload partition strategy in color image steganography," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 3, pp. 685–696, Mar. 2020.



JĘDRZEJ BIENIASZ was born in Warsaw, Poland. He received the M.Sc. degree (Hons.) in telecommunications from the Faculty of Electronics and Information Technology (FEIT), Warsaw University of Technology (WUT), in 2017. He was involved in establishing B.Sc. studies under cybersecurity program at WUT, in 2019, where he is teaching various topics in cybersecurity (introduction courses, network security, application security, digital forensic, and incident response). He is currently an Assistant Professor with the Cybersecurity Division, FEIT, Institute of Telecommunications (IT), WUT, where he is also the Head and the Co-Founder of the Cybersecurity Center. He is the author or the coauthor of over 20 papers. He has various experience from projects with academia and industry around cybersecurity, IT, and networking. His research interests include cybersecurity data science and analytics, cyber threat modeling and hunting, secure systems, applications, and architectures.



PATRYK BĄK was born in Pionki, Poland. He received the B.Sc. and M.Sc. degrees in telecommunications from the Faculty of Electronics and Information Technology (FEIT), Warsaw University of Technology (WUT), Poland, in 2018 and 2020, respectively. He started professional career in IT industry, in 2017. In 2019, he was promoted to the Technical Project Leader and he lead the development of multiple projects for banking industry. He is currently building DevOps platforms in on-premises data centers and public clouds. He is a Certified Google Cloud Engineer. He is the author or coauthor of two papers.



KRZYSZTOF SZCZYPIORSKI (Senior Member, IEEE) was born in Warsaw, Poland. He received the M.Sc. (Hons.), Ph.D. (Hons.), and D.Sc. (Habilitation) degrees in telecommunications from the Faculty of Electronics and Information Technology (FEIT), Warsaw University of Technology (WUT), Warsaw, in 1997, 2007, and 2012, respectively. He also completed his postgraduate studies in psychology of motivation at the University of Social Sciences and Humanities, Warsaw, in 2013, and the Master of Business Administration (M.B.A.) degree from the Warsaw University of Technology Business School, in 2022. He graduated in advanced networking from Budapest Tech (now Óbuda University), Hungary, in 2003, and the Hass School of Business, University of California at Berkeley, in 2013. Since 2015, he has been the Co-Founder and the Research and Development Director of Cryptomage S.A., a cybersecurity company. He has been the Head and the Co-Founder of the Research Centre for Cybersecurity and Data Science, WUT, since 2020, and the B.Sc., M.Sc., and M.B.A. programs in cybersecurity at WUT, since 2019. He is currently a Professor of telecommunications with the Institute of Telecommunications (IT), FEIT, WUT, where he is also the Head and the Co-Founder of the Cybersecurity Division. He is the author or the coauthor of over 220 papers, three patent applications (one is granted), and over 80 invited talks.

...

B. Oświadczenia współautorów


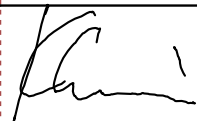
Poniższy rozdział zawiera oświadczenia współautorów dotyczących publikacji [A1]–[A7] będących w zakresie niniejszej rozprawy doktorskiej:

- Oświadczenie współautorów dotyczące artykułu [A1]: **SocialStegDisc: Application of steganography in social networks to create a file system** ► str. 175
- Oświadczenie współautorów dotyczące artykułu [A2]: **Methods for Information Hiding in Open Social Networks** ► str. 176
- Oświadczenie współautorów dotyczące artykułu [A3]: **Steganography Techniques for Command and Control (C2) Channels** ► str. 177
- Oświadczenie współautorów dotyczące artykułu [A4]: **Mobile Agents for Detecting Network Attacks Using Timing Covert Channels** ► str. 178
- Oświadczenie współautorów dotyczące artykułu [A5]: **Dataset Generation for Development of Multi-Node Cyber Threat Detection Systems** ► str. 179
- Oświadczenie współautorów dotyczące artykułu [A6]: **Towards Empowering Cyber Attack Resiliency Using Steganography** ► str. 180
- Oświadczenie współautorów dotyczące artykułu [A7]: **StegFog: Distributed Steganography Applied To Cyber Resiliency In Multi Node Environments** ► str. 181

Warszawa, 5 września 2022 r.

Oświadczenie współautorów



Niniejszym deklaruje się wkład poszczególnych współautorów w opracowanie artykułu [A1]: Jędrzej Bieniasz, Krzysztof Szczypiorski. (2017). **Social-StegDisc: Application of steganography in social networks to create a file system**. 2017 3rd International Conference on Frontiers of Signal Processing (ICFSP), 2017, pp. 76-80. DOI: 10.1109/ICFSP.2017.8097145.

Współautor	Wkład %	Zakres zadań	Podpis
Jędrzej Bieniasz	75%	Stan sztuki Projekt rozwiązania Metodologia Implementacja Badania Walidacja Opracowanie artykułu	
Krzysztof Szczypiorski	25%	Konceptualizacja Nadzór Opracowanie artykułu	

Warszawa, 5 września 2022 r.

Oświadczenie współautorów



Niniejszym deklaruje się wkład poszczególnych współautorów w opracowanie artykułu [A2]: Jędrzej Bieniasz, Krzysztof Szczypiorski. (2019). **Methods for Information Hiding in Open Social Networks**. Journal of Universal Computer Science, 25(2), 74-97. DOI: 10.3217/jucs-025-02-0074.

Współautor	Wkład %	Zakres zadań	Podpis
Jędrzej Bieniasz	65%	Stan sztuki Projekt rozwiązania Metodologia Implementacja Badania Walidacja Opracowanie artykułu	
Krzysztof Szczypiorski	35%	Stan sztuki Projekt rozwiązania Konceptualizacja Nadzór Opracowanie artykułu	

Warszawa, 5 września 2022 r.

Oświadczenie współautorów



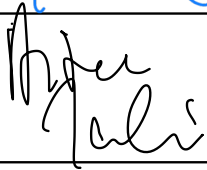

Niniejszym deklaruje się wkład poszczególnych współautorów w opracowanie artykułu [A3]: Jędrzej Bieniasz, Krzysztof Szczypiorski. (2019). **Steganography Techniques for Command and Control (C2) Channels**. In Botnets. Architectures, Countermeasures, and Challenges. (pp. 189-216). CRC Press. DOI: 10.1201/9780429329913-5.

Współautor	Wkład %	Zakres zadań	Podpis
Jędrzej Bieniasz	75%	Stan sztuki Metodologia Analiza teoretyczna Modelowanie Opracowanie artykułu	
Krzysztof Szczypiorski	25%	Konceptualizacja Nadzór Opracowanie artykułu	

Warszawa, 5 września 2022 r.

Oświadczenie współautorów


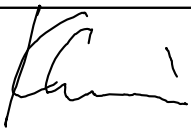
Niniejszym deklaruje się wkład poszczególnych współautorów w opracowanie artykułu [A4]: Jędrzej Bieniasz, Monika Stępkowska, Artur Janicki, Krzysztof Szczypiorski. (2019). **Mobile Agents for Detecting Network Attacks Using Timing Covert Channels**. Journal of Universal Computer Science, 25(9), 1109-1130. DOI: 10.3217/jucs-025-09-1109.

Współautor	Wkład %	Zakres zadań	Podpis
Jędrzej Bieniasz	65%	Stan sztuki Projekt rozwiązania Metodologia Implementacja Zbieranie danych Badania Walidacja Opracowanie artykułu	
Monika Stępkowska	5%	Implementacja Zbieranie danych Opracowanie artykułu	
Artur Janicki	5%	Stan sztuki Projekt rozwiązania Nadzór Opracowanie artykułu	
Krzysztof Szczypiorski	25%	Stan sztuki Konceptualizacja Projekt rozwiązania Nadzór Opracowanie artykułu	

Warszawa, 5 września 2022 r.

Oświadczenie współautorów


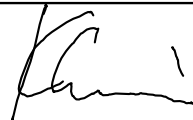
Niniejszym deklaruje się wkład poszczególnych współautorów w opracowanie artykułu [A5]: Jędrzej Bieniasz, Krzysztof Szczypiorski. (2021). **Dataset Generation for Development of Multi-Node Cyber Threat Detection Systems**. Electronics. 2021; 10(21):2711. DOI: 10.3390/electronics10212711.

Współautor	Wkład %	Zakres zadań	Podpis
Jędrzej Bieniasz	75%	Stan sztuki Projekt rozwiązania Metodologia Implementacja Zbieranie danych Badania Walidacja Opracowanie artykułu	
Krzysztof Szczypiorski	25%	Konceptualizacja Nadzór Opracowanie artykułu	

Warszawa, 5 września 2022 r.

Oświadczenie współautorów



Niniejszym deklaruje się wkład poszczególnych współautorów w opracowanie artykułu [A6]: Jędrzej Bieniasz, Krzysztof Szczypiorski. (2018). **Towards Empowering Cyber Attack Resiliency Using Steganography**. 2018 4th International Conference on Frontiers of Signal Processing (ICFSP), 2018, pp. 24-28. DOI: 10.1109/ICFSP.2018.8552068.

Współautor	Wkład %	Zakres zadań	Podpis
Jędrzej Bieniasz	75%	Stan sztuki Projekt rozwiązania Metodologia Walidacja Opracowanie artykułu	
Krzysztof Szczypiorski	25%	Konceptualizacja Nadzór Opracowanie artykułu	

Warszawa, 5 września 2022 r.

Oświadczenie współautorów

Niniejszym deklaruje się wkład poszczególnych współautorów w opracowanie artykułu [A7]: Jędrzej Bieniasz, Patryk Bąk, Krzysztof Szczypiorski. (2022). **StegFog: Distributed Steganography Applied To Cyber Resiliency In Multi Node Environments**. IEEE Access, 2022. DOI: 10.1109/ACCESS.2022.3199749.

Współautor	Wkład %	Zakres zadań	Podpis
Jędrzej Bieniasz	60%	Stan sztuki Projekt rozwiązania Metodologia Ocena bezpieczeństwa Wsparcie implementacji Badania Walidacja Opracowanie artykułu	
Patryk Bąk	15%	Projekt rozwiązania Implementacja Badania	
Krzysztof Szczypiorski	25%	Konceptualizacja Projekt rozwiązania Nadzór Opracowanie artykułu	