



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 stegomalware 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.

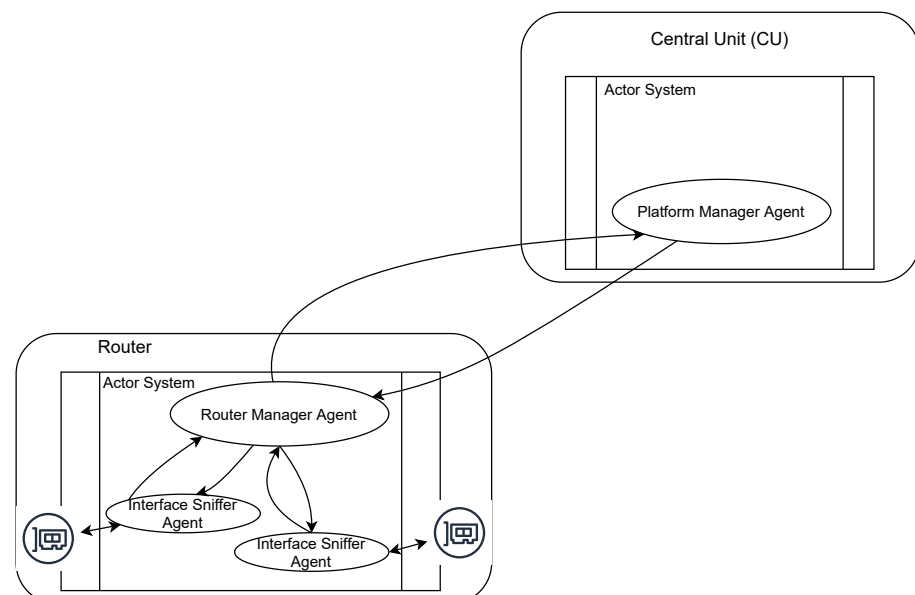


Figure 1. Scheme of monitoring platform based on actor system approach.

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	<ol style="list-style-type: none"> 1. Hosting main control agent of entire platform 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 	<ol style="list-style-type: none"> 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
A router	Computing and networking platform that hosts remote portion of entire agent system	Receiving requests to create new control agent
Router Controlling Agent	<ol style="list-style-type: none"> 1. Router management 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 	<ol style="list-style-type: none"> 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
Internal Sniffing Agent	<ol style="list-style-type: none"> 1. Creating sniffing process on bound network interface 2. Managing sniffing strategy 3. Managing end of sniffing action by passing callback to router control agent 	<ol style="list-style-type: none"> 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

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

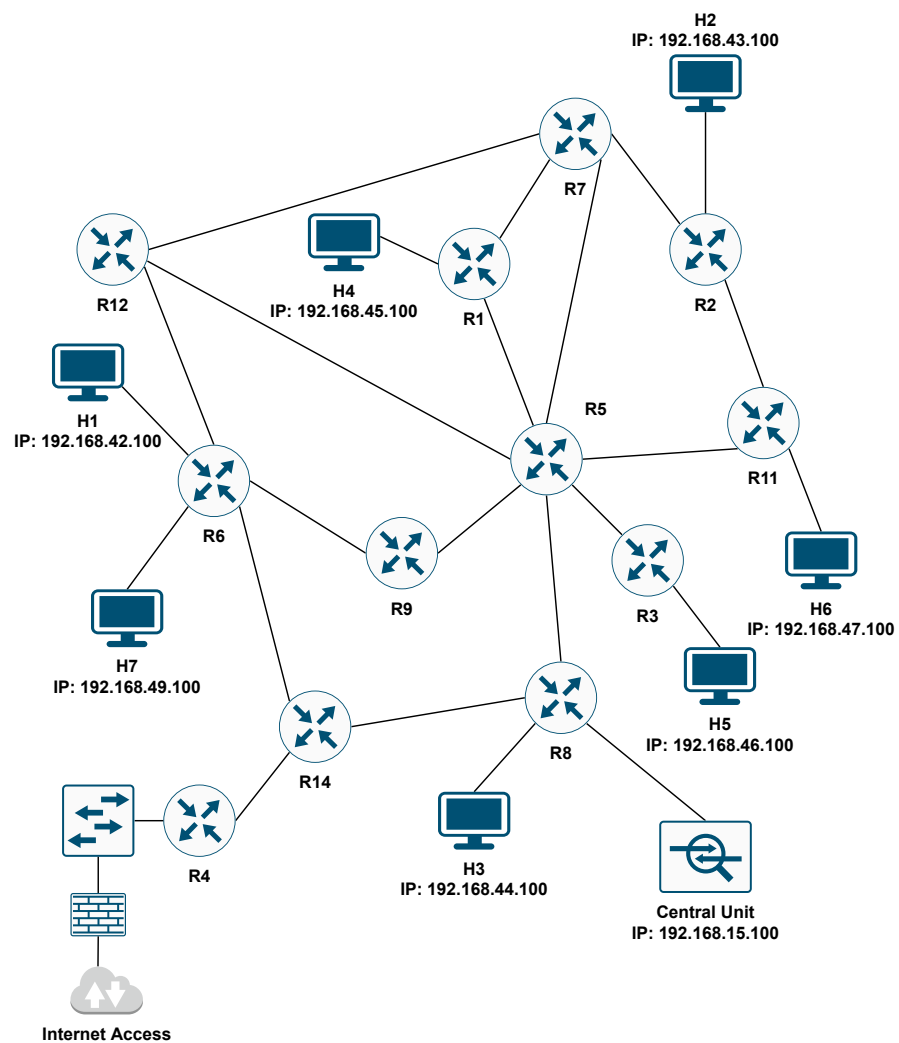


Figure 2. Network setup for simulations of hidden communication techniques.

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).

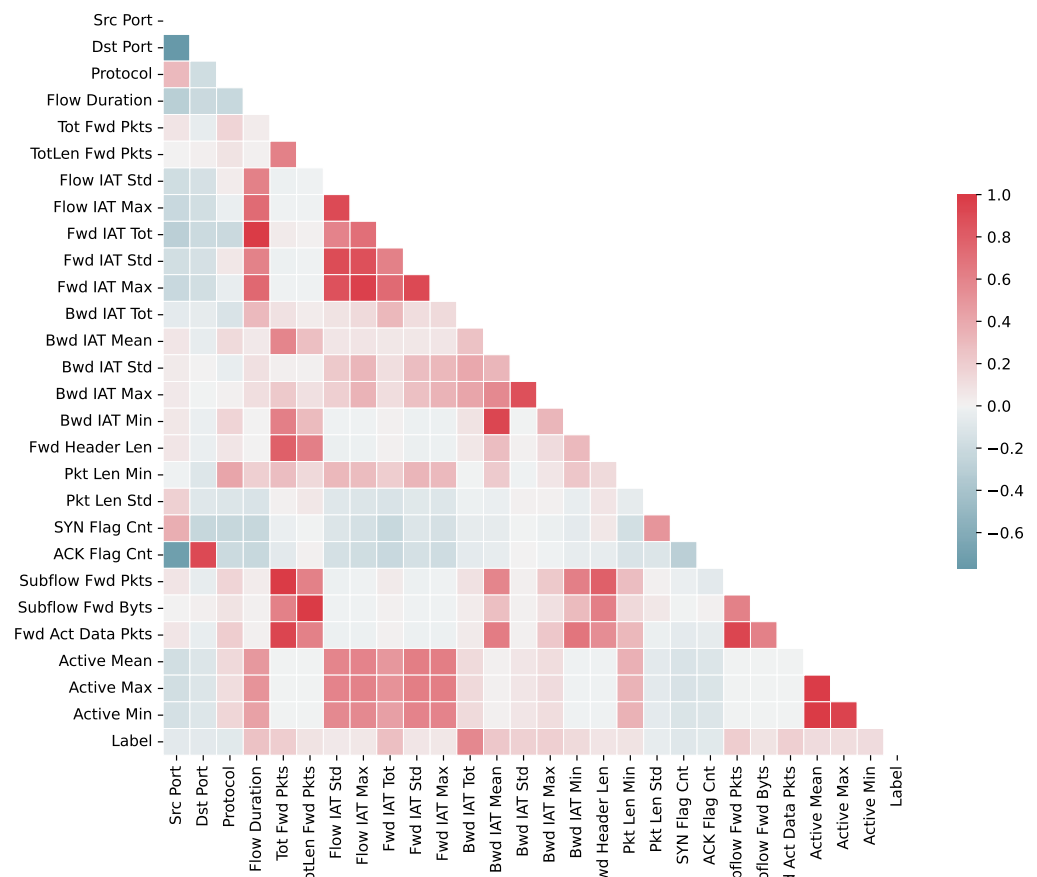


Figure 3. Pairwise correlation matrix of prepared dataset. Filtered according to level of correlation between label column and other parameters for feature selection purposes.

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		

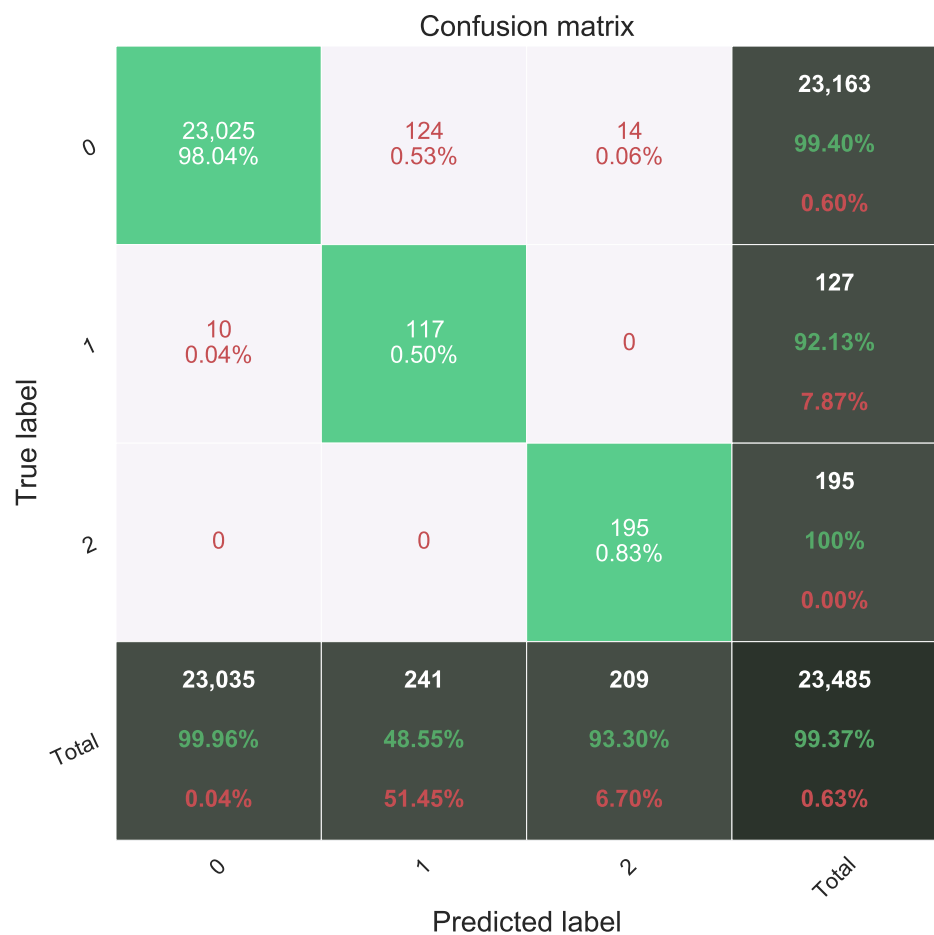


Figure 4. Confusion matrix of RF classifier.

Confusion matrix

True label	0	22,949 97.72%	176 0.75%	38 0.16%	23,163 99.08% 0.92%
	1	1 0.00%	126 0.54%	0	127 99.21% 0.79%
	2	0	0	195 0.83%	195 100% 0.00%
	Total	22,950 100.00% 0.00%	302 41.72% 58.28%	233 83.69% 16.31%	23,485 99.08% 0.92%
		0	1	2	Total
		Predicted label			

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/I/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. Balasubramaniyan, 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.; Fiddymont, 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.; Grüdl, 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 (ISXIDS2012). 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).