Czech Technical University in Prague
Faculty of Information Technology
Department of Digital Design

Stream-wise Parallel Anomaly Detection in Computer Networks

by

Tomáš Čejka

Doctoral Degree Study Program: Informatics

Dissertation thesis statement for obtaining
the academic title of "Doctor" abbreviated to "Ph.D."

Prague, May 2018

The dissertation thesis was written during full-time doctoral study at the Department of Digital Design, Faculty of Information Technology of the Czech Technical University in Prague.

Ph.D. Candidate:	Tomáš Čejka Department of Digital Design Faculty of Information Technology Czech Technical University in Prague Thákurova 9, 160 00 Prague 6, Czech Republic cejkato2@fit.cvut.cz
Supervisor:	Hana Kubátová Department of Digital Design Faculty of Information Technology Czech Technical University in Prague Thákurova 9, 160 00 Prague 6, Czech Republic kubatova@fit.cvut.cz
Reviewers:	______________________________ ______________________________ ______________________________

The dissertation thesis statement was distributed on ....

The defence of the dissertation thesis will be held before the Committee for the presentation and defence of the dissertation thesis in the doctoral degree study program Informatics on .... at .... in the meeting room No. ....

In accordance with Para. 8 of Art. 30 of the Study and Examination Regulations for Students of the CTU, those who are interested may look into the dissertation thesis and make notes, copies, or duplicates from it at their own expense. A copy of the dissertation thesis is available in the Science and Research Office of the Faculty of Information Technology, room No. 308.

....

Chairman of the Committee for the presentation and defence of the dissertation thesis
Faculty of Information Technology
Czech Technical University in Prague
Thákurova 9, 160 00 Prague 6, Czech Republic

Contents

1 Introduction and Motivation

1.1 Introduction

Network security plays an essential role in the modern world of digital communication. It is a research field of many researchers and security companies around the world. Many topics in the network security area that have been elaborated by various researchers. However, there are still many open research issues. This dissertation thesis focuses on several challenges related to network security and parallel processing of high data volume of network traffic. The dissertation thesis is a collection of peer-reviewed research papers on the topic of anomaly detection in large computer networks.

Network technology continuously evolves, network stack is implemented in many hardware chips, and software implementation of communication protocols can be easily integrated into any system or device. Therefore, lots of devices, even small sensors, are being equipped with a network interface, and according to the current trend, the number of network-enabled devices will grow. Devices can communicate via network interfaces not only with humans, as it used to be at the beginning of the computer networking, but also with other "things" or machines connected to the network. Due to automation, machines are becoming both the primary consumers and producers of data. This trend aims to help people to make their life/work easier and more efficient. We are living in the era of the Internet of Things.

It is necessary to perform comprehensive analysis of network traffic to discover the infected devices and other sources of attacks or malicious traffic. The aim is to monitor the activity and status of the whole infrastructure to preserve its operability and keep so-called situational awareness. Monitoring systems observe network traffic and gather data about communication between connected devices. Monitoring systems produce such a high data volume that it is challenging to do the analysis and detection manually. Additionally, it is a challenging task to store all the data due to slow storage or to keep a long history of data from large networks. A logical used approach is to aggregate or sample data to decrease the volume of data.

1.2 Motivation

Network monitoring, as well as anomaly detection, are essential parts of every well-running network infrastructure. Since the network infrastructures grow and the bandwidth increases, it is necessary to adapt monitoring and analysis systems to handle high data volumes. Additionally, many security threats are difficulty detectable using traditional approaches based on simple flow data (NetFlow, or more preferred IPFIX). Therefore, our work deals with several challenges to overcome the current state and prepare the systems for the future.

At first, we want to avoid limits of the fixed time windows. The aim is to process flow records immediately when they arrive without waiting for the end of time interval. Such processing is closer to real-time, and so the delay can be shortened.

The second objective is related to the detection results and ability to detect security threats. Many security threats are detectable using basic flow data. However, advanced threats at application layer are usually represented by similar flows, i.e., they are not distinguishable from legitimate traffic. The challenge is to improve ability and reliability of detection algorithms to detect application layer threats using flow data (that can be extended by additional information).

Finally, the last scope of interest is parallel processing of the flow data to handle growing data volume. The aim is to design a parallel infrastructure for network traffic analysis that can process much more data than the original single stream processing tools. Additionally, the challenging requirement is to use existing detection algorithms without any necessary change.

1.3 Problem Statement

Let us have a *set of monitoring probes* and a *central flow collector* that receives data from the probes. Let us have a computing node that can receive a continuous stream of data via a flow collector. The computing node runs *various detection algorithms* which process a *stream of input data* and detect malicious traffic. The node can be easily cloned and started multiple times on separate machines to get more processing power. The challenging part is a *distribution of a single stream of data among the multiple nodes*, whereas each node should process just a subset of the original stream.

This approach does not require a complete redesign of the deployed detection or analysis algorithms. However, a suitable mechanism of data distribution must be designed to preserve results of detection. This work proposes an approach to scalable processing flow data using independent computing nodes.

Besides parallel processing, another challenge is related to the reliability of detection algorithms when they are used to detect application layer threats. Use of traditional flow records with just basic information is not sufficient for detection of advanced threats that are very similar to legitimate traffic.

Finally, there is a lack of existing (open source) tools to develop prototypes of detection and analysis algorithms. Such tools are essential for experiments and evaluation of detection algorithms.

1.4 Contributions of the Thesis

Contributions are related to the security analysis of the flow data. They can be divided into the following topics:

1. **stream-wise approach** to network traffic analysis and anomaly detection using flow data without long-term data storage,
2. design and development of **stream-wise and application-aware detection methods**, that were implemented as parts of the developed open source system **NEMEA**,
3. scalable architecture for **parallel processing** that is based on the proposed methodology about **semantic relations** in the flow data, i.e., *witnesses* explained in this dissertation thesis,
4. **practical experiments** and observed statistics about real network traffic that were presented in the published reviewed papers,
5. the dissertation thesis also contains a set of **formal symbolic descriptions** of the developed detection algorithms, that were used to evaluate the proposed features and parallel infrastructure.

1.5 Structure of the Statement

The statement is organized as follows: Section 1 describes the context of this statement. Section 2 describes the current state-of-the-art and lists the related existing works. Section 3 briefly presents the main points of this work, and they are described in more detail in Section 4. Section 5 concludes this document.

2 Background and State-of-the-Art

There are many sources of information utilized in network monitoring. According to the source of information, we can recognize two main types of monitoring: *host-based*, which uses information from each device (e.g., system log files), and *network-based*, which uses information exported from network devices such as routers, switches, or specific monitoring probes. This work focuses on the network-based monitoring which is a useful and feasible approach in large network infrastructures where operators usually do not have full control over connected devices.

Current state-of-the-art systems use IP Flows, which are defined in [11] as follows:

> *An IP Flow, also called a Flow, is defined as a set of IP packets passing an Observation Point in the network during a certain time interval. All packets that belong to a particular Flow have a set of common properties derived from the data contained in the packet and from the packet treatment at the Observation Point.*

For the sake of simplicity and our purposes, information of IP Flows is represented by data structures *flow records*, whereas each one is usually a unidirectional communication between two network addresses. Such flow records are identified by an n-tuple consisting of: source IP address (*srcip*), destination IP address (*dstip*), source port of transport protocol (*srcport*), destination port of transport protocol (*dstport*), transport protocol (*proto*), timestamp of the first packet in the flow (*timefirst*), timestamp of the last packet of the flow (*timelast*). Besides these fields for identification, flow records contain some additional fields — counters that represent volume of the transferred traffic: number of transferred bytes (*bytes*), number of transferred packets (*packets*). The described list of fields of a flow record is referred as a *basic flow record* in the later text of this thesis. Flow data (sequence of flow records) are sent from monitoring probes, where the flow records are exported, to a flow collector, where they are processed and stored.

2.1 Detection Methods

There are many surveys and taxonomies of detection methods or tools [1, 2, 17]. Naturally, each type of a network attack or a traffic anomaly is elaborated in many other papers, e.g., a DDoS topic presented in papers such as [3, 24, 34]. Types of detection methods are highly related to the specific types of attacks or the types of traffic anomaly that ought to be detected. In addition, the detection mechanisms are usually presented in combination with defense techniques to mitigate the traffic or at least to allow post-mortem investigation.

An anomaly in network traffic usually appears at unpredictable points in time. The start of an anomaly can be observed as a change of statistical distribution of the random variable that represents some characteristic of network traffic [28]. The choice of a set of the observed characteristics depends on a detection method and on an anomaly type that should be detected. Characteristics are usually monitored and analyzed as a time series that is defined as a sequence of measured values in time. The first insight to an anomaly detection using time series can be based on simple thresholds and limits where an alert is raised when the current value reaches some predefined value.

There are many types of network attacks or other anomalies that appear in computer networks. The anomalies differ in many ways such as a volume of traffic, or a duration of anomaly. For successful anomaly detection, a model of the network traffic is needed. It is a complicated issue to create a working and precise model of a computer network traffic as it is described in [31]. The absence of such model can lead to false alerts signaled from detection methods. Based on long-term observation, some similarities and trends can be seen in network traffic. More sophisticated detection methods are based on this knowledge and use various kinds of smoothing such as moving averages or prediction models to normalize measured characteristics.

Some types of a network anomaly are significant enough to be found by manual analysis of the network traffic. The most significant anomaly that can be observed in the network traffic is a DoS attack or its distributed version (DDoS) attack. They are observable as a huge increase in the total traffic. However, from the ISP network point of view, even a high increase of traffic at a link of a user can be below the resolution of the global view and capacity of ISP's backbone links. In addition, a manual analysis is not always possible due to a huge volume of data and the efficiency of such analysis.

SYN flood attacks[1] have been studied for more than ten years, and various detection approaches were published in many papers. However, this issue is currently still relevant because of the increase of network traffic volumes, the growth of networks and the higher level of sophisticated attacks. The detection based on NP-CUSUM is used in [30], where the authors present their observation about SYN-FIN pairs in network traffic under the normal condition:

> *(1) there is a strong positive correlation between the SYN and RST packets;*

[1]a well-known type of flood attacks that is based on sending of several TCP packets with SYN flag, these packets are used for establishment of a TCP session.

> *(2) the difference between the number of SYN and FIN packets is close to the number of RST packets.*

The authors bring an experimental evaluation of flood detection using NP-CUSUM and counts of TCP flags observation; however, they mention a possible disadvantage of aggregated counting of packets that can be spoofed by an emission of mixed packet types by an attacker.

The authors of [27] compare a straightforward *adaptive threshold algorithm*, which can bring satisfactory performance for attacks with high intensity, and an *algorithm based on CUSUM*. The adaptive threshold algorithm uses a deviation from a moving average computed by exponentially weighted moving average (EWMA) algorithm. An alert is signaled when a measured value is higher than moving average in the last k consecutive intervals. The authors propose to use some prediction method such as Holt-Winters to remove non-stationary behavior before applying the CUSUM algorithm. However, because of time-consuming calculations with minor gains compared to more straightforward approaches, the authors used an approach based on CUSUM and the difference between measured value and result of EWMA algorithm.

The idea to use EWMA for smoothing, Holt-Winters for prediction of seasonal trends of traffic and CUSUM for change point detection was published in [16]. The authors designed a prototype of a monitoring probe extended by anomaly detection as a software plug-in for an existing monitoring probe. The paper proposed two algorithms with fixed parameters evaluated on a dataset from backbone network. In [32] the authors also use the EWMA algorithm for a detection. They propose to estimate parameters from training data that is free of intrusions or attacks. Important parameters for detection are upper and lower control limits — thresholds.

The Holt-Winters method was used for detection also in [5]. The authors described the formula of the method as well as the way of its usage for a detection. The detection is based on a prediction of a confidence band and a comparison of current measured value with the confidence band.

The detection methods listed in this section have usually been applied on time series of some aggregated counters such as the number of packets with TCP SYN flag. The aggregation of characteristics does not allow to trace back the suspicion flows. Therefore, it is complicated to identify the real reason of an anomaly. The authors of [26] propose an architecture that uses a multi-stage Bloom filter or a sketch structure. The multi-layer reversible sketch (MLRS) and the count-min sketch (CMS) that were used in [26] are briefly described in [A.17].

The authors of [26] use multi-chart CUSUM (MNP-CUSUM) proposed in [29] with the input taken from the sketch structure. The sequential MNP-CUSUM over a sketch for anomaly detection allows for detection of changes with a small delay and a low false alarm rate. An alert is raised when at least one of the test statistics reaches the threshold h_i.

The sketches are used in various detection systems, sometimes, in combination with different sophisticated mechanisms. Callegari et al. have proposed an architecture of a detection system that is based on reversible sketches and neural networks in [8]. The paper is concentrated on short-term anomalies detection (without seasonal effect). However, the authors publish the comparison of neural networks and training algorithms that can be used for anomaly detection in computer networks. Another example of usability of the sketches is in [7] where the detection is based on Principal Component Analysis (PCA).

The study of detection methods of DDoS attacks was published in [3]. The authors discuss open issues and challenges. The example of the challenges is a development of new detection methods that can detect even sophisticated attacks generated by tools that are being rapidly improved. The paper uses the taxonomy of DDoS attacks and DDoS defense mechanisms published by [24]. According to the taxonomy, DDoS detection methods can be classified as a) *statistical* b) *knowledge-based* c) *soft computing* d) *data mining* and *machine learning*.

The group of *statistical* detection methods contains various methods based on Change-Point Detection (CPD) algorithms such as CUSUM or NP-CUSUM [4, 28] (described in [A.17]), Change aggregation trees (CATs) [10] or D-WARD [23] based on continuous monitoring of bidirectional traffic flows between the network and the rest of the Internet with analysis of deviation from the typical flow patterns.

Knowledge-based methods are based on predefined rules and patterns of attacks. Detection methods exploit various heuristics and data structures. *Soft computing* methods are based on learning paradigms. Methods from this group use neural networks, radial basis functions, and genetic algorithms. The *data mining and machine learning* algorithms are also exploited for network attacks detection. However, the training and the deployment of these methods are challenging because of lack of suitable datasets.

The authors of [3] propose some requirements on detection methods to be ideally usable in real networks. Detection should be as fast as possible with acceptable resource consumption. Detection should be accurate with low false alarms. Detection methods should be prepared for real network traffic that means they should be scalable for high-speed networks and the performance should be extensively evaluated. Detection methods should be prepared for up-to-date sophisticated network attacks. Detection methods should accurately segregate high-rate attack traffic from legitimate traffic such as flash crowds with minimum resource consumption and low false alarm rate. Detection methods should be able to handle an increased volume of data for processing and persist operable. Detection methods should be able to handle spoofed IP addresses. Detection methods should be dependent on a minimum number of input parameters if not independent of parameters and should also be based on a minimum number of traffic parameters or characteristics.

2.2 Big Data Processing

Big Data (described, e.g., in [25]) is a term describing a particular character of complex data that are challenging for storage, processing, and visualization. Big Data are significant for its three characteristics: velocity, variety, and volume. According to the definition, network traffic can be considered as the Big Data problem.

There are many papers about the parallel processing of network data. We focus on flow-based processing that is the current state-of-the-art. There are several software solutions for Big Data processing; one of the most popular is Hadoop. Technical report [33] shows that existing Big data frameworks are not efficient enough for some applications. However, Hadoop tools that support MapReduce algorithm have been used by many researchers.

It is necessary to split data for parallel processing, but this problem is usually handled just by general mechanisms of the frameworks. For instance, the authors of [21, 22] use a distributed database (like Hive or HBase) or a distributed filesystem (HDFS) to store data files. Authors of [18] analyzed IP, TCP and HTTP traffic stored in offline data files. Paper [6] presents experiments with several types of MapReduce jobs to analyze campus network traffic (e.g., computing volume of traffic per subnet).

Authors of [20] use Spark with Netmap to extract traffic features for detection of different types of DDoS attacks in real-time. The detection uses machine learning methods and relies on a distributed storage and an abstraction of objects called Resilient Distributed Dataset (RDD). The paper notes that the use of sampled data produces many false-positives.

Detection of DDoS and SYN flood is presented in [35]. The processing is based on HDFS and HBASE that uses a Bloom filter with several hash functions and a KEY that consists of the fixed n-tuple (*src* and *dst* IP addresses and ports, protocol and TCP flags). The authors do not explicitly explain the reason they chose the KEY, but they mention a condition of detection of an attack with 100 % accuracy.

In 2016, Cermak et al. in [9] presented a benchmark of stream processing systems (such as S4, Spark, Samza, Storm, and Flink) for parallel processing. The chosen operations (Identity, Filter, Count, Aggregation, TOP N, SYN DoS) were proposed for performance comparison.

Jirsik et al. in [19] presented the performance of the proposed system for stream processing based on Apache Big Data analytics tools as well. The benchmark was focused on a set of typical analysis queries, and according to the results, the systems were able to process up to 2 million flows/s on a cluster with 32 CPU cores in total. Contrary, the solution proposed in this dissertation thesis is obviously less resource consuming since we can handle approximately the same flows/s with only half number of CPU cores (16) (described in [A.3]).

The dissertation thesis by Garofalo [14] deals with anomaly detection using a traditional Big Data approach. It describes existing Big Data analytics tools such as Apache Hadoop, Apache Storm, Apache Kafka, and Apache Spark. The author proposes anomaly detection using these tools. However, the datasets are currently out-of-the-date, and the work focuses only on a few types of anomaly (DoS, DDoS, horizontal scan, vertical scan).

Papers [9, 14, 19] do not explicitly mention evaluating the system with extended flow records with some L7 information (they only state that the flow records might be extended). Therefore, we can assume that they used only traditional flow data with information up to L4.

2.3 Semantic Relations in Data

Semantic relations in data and adverse effects of data splitting were mentioned in [13]. The authors show experiments with Hashdoop, an improved Hadoop, that splits data using CRC hashes of *srcip* and *dstip*. The authors chose a packet counting and ASTUTE algorithm for parallel processing. If more different algorithms were used, this hashing would not be efficient enough. The splitting based on IP addresses is just a particular case of our methodology.

The authors of [22] face the topic of data with semantic relations which require being processed together. Accurately, the paper describes an analysis of TCP connection using stitching flow parts (into one bidirectional connection).

It is done by putting the whole traffic of the same IP addresses together.

The authors and their works that were mentioned in Section 2.2 use the existing frameworks for Big Data processing and distributed storage. Hence, they rely on the internal mechanisms of data distribution by the frameworks. The algorithms must handle this state, and it is usually solved by additional communication between the computing nodes to exchange data or results. That is the reason why the authors need not care about the semantic relation, even though the processing is not optimally efficient.

3 Overview of Our Approach

Our approach is based on the following main parts that are explained in the following text:

1. stream-wise analysis of flow data,
2. application-aware detection algorithms,
3. methodology about usage of *witnesses*,
4. combining altogether into parallel infrastructure for flow data analysis and detection of malicious traffic.

4 Main Results

There were many goals of this thesis, and they were commonly focused on processing the flow data in large backbone networks. The aim was to design mechanisms to process **huge volume** of flow data as soon as possible especially for network security purposes. There was a need to design new detection and processing algorithms that could work in a **stream-wise** way, i.e., to process data immediately without long-term storage. This approach allows for processing in **real-time**. In addition, this research showed that some application layer information might help to identify advanced security threats. Therefore, the next goal was to extend the monitoring and analysis infrastructure to support additional information, i.e., to make the monitoring "aware of application" (**application aware**). Because of the growing volume of the flow data, the processing on just one machine would not be scalable enough. Therefore, the next goal of the thesis was **parallel processing** based on splitting a stream of flow data into independent subsets that can be processed on separate machines. There was a related requirement about the **balanced load** of each machine, which is the reason why the sizes of the split subsets should be as similar as possible. Splitting the stream of flow data affects the amount of information that is processed by a single machine. The goal was to design an algorithm for splitting that ensures the machine receives enough data to identify security threats, i.e., **to preserve detection results**.

Contributions of this thesis can be generalized and divided into the five main topics that are covered in the following sections.

4.1 Network Measurements Analysis Framework (NEMEA)

Before this research has started, the existing tools worked mostly in a traditional way of batch processing, where the flow data are stored into files on a file system before an analysis can start. Each file contains flow data within a fixed time window. Since each file must be closed before the analysis, there is a necessary delay before the processing starts. Also, one file may contain flow records that belong to a different time window due to delayed exporting. This variability complicates the development of detection algorithms. However, it is usually not an issue in practice. The need for data storage and delayed processing are the significant limits that we avoided using a *stream-wise approach* (described in Sec. 4.2).

To support this research and to allow experiments, there was a need for a stream-wise framework for the development of prototypes of detection algorithms. The aim was to test and evaluate new stream-wise detection algorithms with real data. Because of lack of such tools in 2013, it was decided to develop a new universal framework that can fulfill the needs. Even though the batch approach of processing is still commonly used in practice, it is not efficient enough for processing a continuous stream of data from large networks anymore.

As a result of years of research and development, Network Measurements Analysis (NEMEA) Framework was released. NEMEA was deployed and analyzes flow data from the perimeter of the CESNET2 network, the Czech national academic network. It is an open source project that is publicly available at GitHub[2] for a world-wide community of network operators and researchers. Since the existence of NEMEA, several research activities were done, and there are published papers that show some of the developed detection methods and

[2]`https://github.com/CESNET/NEMEA`

their feasibility in large-scale networks. Besides that, NEMEA itself was successfully presented in Canada in 2016.

The main advantages of NEMEA are:

1. high modularity — each NEMEA module is an independent system process that receives and transmits flow data and runs in an operating system[3],
2. stream-wise approach of data processing (described in Sec. 4.2) that aims at low memory consumption and decreases a delay of detection,
3. application-awareness (described in Sec. 4.3), which is an ability to analyze any information field even from application layer headers and any NEMEA module may define new fields without affecting the rest of the system.

In 2017, NEMEA was used for educational purposes in the Network security course at the Faculty of Information Technology, Czech Technical University in Prague for the first time. Students can quickly do hands-on experiments with the stream-wise flow data analysis. Besides just usage of the existing NEMEA modules, it is possible to improve them or invent own modules with novel detection algorithms.

In the time of writing of this dissertation thesis, NEMEA contains 11 detection modules and about 22 general purpose modules form manipulation with UniRec messages. The detection modules can detect malicious traffic of the following categories:

- DNS tunnels,
- amplification attacks,
- anomalous volume of traffic per observed host,
- brute-force attacks (guessing passwords, user accounts, open SIP URI prefixes),
- communication with blacklisted servers,
- horizontal scanning,
- traffic from miners of cryptocurrency,

[3]GNU/Linux CentOS 7 is primarily supported

- vertical scanning (port scan),
- volumetric DDoS attacks characterized by a significant increase of traffic volume and the number of its sources.

Some of the detection algorithms are described or referenced from this thesis.

4.2 Stream-wise Approach to Flow-Based Analysis (SW)

Monitoring systems of large networks produce high data volumes of flow data. Operators ought to store the flow data by law; however, only basic flow records with information up to transport layer (L4) are being stored. Additionally, the traditional approach to an analysis of the basic data is based on processing the fixed length time windows. The fixed time windows necessarily cause higher detection delays. Therefore, we have started our research on stream-wise processing. The stream-wise algorithms process data immediately without any need of waiting for the end of a time window. Besides, the stream-wise algorithms do not store any data permanently by design. They are designed to process infinite stream of flow data on-the-fly.

Traditional approaches of flow-based processing worked in batches, i.e., datasets are split into time intervals and processed when the time interval elapses. On the other hand, the stream-wise approach, which is preferred in this work, is based on an on-the-fly analysis of the flow data without any long-term storage. The input data come from monitoring probes continuously, and it is processed immediately. For the analysis, it is not intended to store data permanently but all detection methods store only necessary state information temporarily.

Modern flow exporters send flow records continuously because the active timeout is checked on every update of a flow record in the flow cache and the inactive timeout is checked regularly. Therefore, the approach of processing (batch or stream-wise) depends mainly on a flow collector, where the records are being processed. This work focuses on stream-wise processing (security analysis) in a flow collector.

Stream-wise approach is inspired by traditional pipeline processing, where the input data is also processed continuously, i.e., as it is received without long-term storage. Therefore, it is possible to process a stream of Big Data using computing nodes with insufficient resources. To imagine the situation, let t_0 be the time of starting the detection system that processes data observed at monitoring probes. For practical reasons, the detection system should run without any interruption, so ideally, processing runs to infinity (t_∞).

There are flow records F_i received by the detection system at t_i. Naturally, for $i \in (-\infty, 0)$ the flow records were not processed because they were exported before the start of the detection system.

The detection system usually consists of various detection algorithms that are implemented as independent modules. Stream-wise approach of each module can be represented by the following equation:

$$S_t = A(F_t, S_{t-1}), \quad S_{\leq 0} = s_0, \tag{1}$$

where A is the algorithm that expects the current flow record F_t in time t and keeps its previous state S_{t-1} internally as an input. The algorithm returns a current state S_t that is related to the observed traffic. The algorithm aims to identify malicious traffic. When the detection module that runs the detection algorithm is started, the state S_0 is set to some initial constant s_0.

As a result, detection modules should depend only on a currently received flow record and the internal state of the algorithm. However, the internal state can be a complex structure of information. For instance, there are detection methods that analyze changes of behavior of network entities in time and look for an anomalous behavior or changes. Such methods need some information about history that can be contained in the previous state.

Additionally, the information about entities might be updated by appending new data into the structure. This way, the required memory grows. Therefore, it is the most important task when designing a stream-wise module to use some aging and cleanup of the current state. Some approximation should be considered to sustain low resource consumption. For instance, there can be some maximal limit of stored data per each analyzed entity (e.g., *srcip* or *dstip*).

4.3 Application-Aware Detection Methods (AA)

Our research showed that some advanced security threats are invisible for the traditional NetFlow tools. Such security events look like legitimate flow records. The application-aware detection methods were studied and designed to overcome the lack of visibility. The application-aware approach is based on extended flow records containing information from higher protocol layers (up to application layer — L7), i.e., extension fields in IPFIX are used. Application headers from unencrypted network traffic allow for reliable detection of some application threats such as brute-force dictionary attacks. Naturally, this approach requires capable tools that can process extended flow records. This was the reason for developing the NEMEA framework.

Various security threats can be detected using aggregated flow data with a low level of detail. Many methods can recognize attacks like network scanning, volumetric attacks or even brute-force password guessing, as it was described in [15]. However, many types of malicious traffic have very similar characteristics as the legitimate traffic, so it is hard to distinguish malicious and legitimate traffic reliably. There are two issues: either malicious traffic is not recognized, or some legitimate traffic is detected by mistake (false positive alerts). For example, multiple unsuccessful attempts of registration to the SIP device looks like a normal TCP or UDP communication (many short IP flows or, sometimes, only two longer IP flows) between two IP addresses. Meanwhile, it should be identified as a brute-force attack.

Several detection algorithms described in this dissertation thesis use extended flow records with L7 information. The paper [A.15] is the most straightforward usage of domain names and URLs in a filter that checks information from the flow records. For this purpose, flow records were enriched by `DNS_NAME` — a domain name extracted from a DNS request or response, `HTTP_HOST` — a hostname extracted from a `Host:` header of an HTTP request, `HTTP_URL` — URL from an HTTP request. These information fields are checked in publicly available blacklists, which are related to the identifiable suspicious activity of known malware samples. For instance, a connection to a host that is listed as a C&C server might indicate some malware infection.

It is known that users can encode and transfer user data via any communication protocol. Various scientific papers describe methods of communication tunnels also known as covert channels. The paper [A.8] presents another usage of information from DNS messages to detect a covert channel over DNS. This type of the covert channel can be established using, e.g., iodine[4]. The developed detection module that can detect iodine's tunnels uses information extracted from `DNS_RDATA` — an information field containing RDATA part of DNS answers.

The scope of some Voice over IP (VoIP) security events was elaborated in the papers [A.2] and [A.7]. The papers show detection algorithms that analyze `SIP_CALLED_PARTY` — called SIP URI that is contained in the SIP *To:* header. Analysis of the values of this information field disclosed very suspicious traffic generated by penetration testing tools (according to `SIP_USER_AGENT` field) that tries to exploit a victim's SIP device. In addition, `SIP_CSEQ` field (*CSeq* header of SIP) is needed to match SIP requests and responses correctly. It was discovered

[4]http://code.kryo.se/iodine/

that brute-force guessing the SIP account names or their password is observable as a high number of unsuccessful SIP requests, whereas the `SIP_STATUS_CODE` field represents the return value.

There is also an unpublished work described in the diploma thesis [A.29] on detection of spoofing the victim's system time using NTP (Network Time Protocol). The developed detection module analyzes time series of `NTP_ORIG` (origin timestamp) and `NTP_RECV` (received timestamp) and looks for suspicious changes.

Even though the described approach based on extended flow records increases the precision and reliability of detection, it must be used very carefully. This improvement of security (reliable detection of advanced application layer attacks) might conflict with the privacy of users. It is worth noting the described detection algorithms work with unencrypted traffic only meanwhile all private data should be transferred encrypted according to the best practices. Regardless, many types of unencrypted traffic might disclose information about users. It is essential to design monitoring and detection systems so they can reliably protect users against malicious traffic instead of spying them or disrupt their activities.

Various documents deal with privacy issues and data protection. IETF community created a document RFC7258 [12] that claims a pervasive monitoring as a privacy issue. In addition, there is a General Data Protection Regulation (GDPR) of the European Union about protection of user data and restricting service providers in frivolous storing and processing user data. Naturally, many exceptions allow providers to process user data (e.g., because of security purposes). However, the scope of information and purpose of processing must always be kept in mind to avoid breaking users' privacy.

Sometimes, it is possible to avoid the danger of privacy issues by using transformed "anonymized" or "pseudonymized" data[5]. Processing such data is harmless since it is not possible to identify users. The application-aware approach described in this thesis is based on the extraction of particular headers from packets meanwhile the user content (payload) is completely skipped. This set of information fields is selected according to the requirements of the detection algorithm. The fields can be trivially transformed so that it does not affect the

[5]Both anonymized or pseudonymized data alone do not allow identification of users. However, pseudonymized data are usually retrieved by some encryption mechanism so they can be transformed back to the original data using a secret key. It is assumed that the detection system does not have access to the secret key and decryption must be done during the alert reporting or incident handling.

algorithm and the privacy of users is still preserved.

Concerning stream-wise approach, the detection methods do not store any piece of information permanently. In addition, information about IP addresses is deleted after a given time interval to keep the memory consumption low.

4.4 Semantic Relations in Flow Data (WITNESS)

This section defines some terms to explain the principle of parallel processing that respects semantic relations in flow data.

Definition 4.1. Network traffic: A sequence of all packets[6] that were sent or received via a computer network in a time window.

Definition 4.2. Malicious traffic: A subset of network traffic that is unwanted, either it aims to spend capacity of links, or it threatens a target device or service.

Generally, malicious traffic might be generated by any device or software that can send packets into the network. The source of malicious traffic can be, e.g., attackers, malware, botnets.

Examples of malicious traffic can be a flood (TCP SYN, UDP), general DoS or DDoS, communication of an infected device with command and control servers, unsolicited e-mail messages (SPAM, phishing, . . .), communication with a server hosting malware/ransomware. In some network infrastructures especially in a business environment, an information exfiltration can be a severe issue. Therefore, covert channels can also be classified as malicious traffic.

Definition 4.3. Detector: An algorithm that identifies malicious traffic in the network traffic and generates alerts with information about the detected malicious traffic.

Definition 4.4. Witness type: A characteristic or feature of network traffic that is analyzed by a detector to identify particular malicious traffic. The witness type is dependent on the detector, its parameters and a type of malicious traffic that ought to be detected.

Examples of witness types are *SYN packet rate* or for more precise detection *ratio of SYN, FIN, RST packets in time* for TCP SYN flood detection, number of unsuccessful authentication requests, number of different prefixes of SIP URI or domain names.

[6]A *packet* is a basic transferred data unit, a sequence of bytes that represents protocol headers and payload.

Definition 4.5. Witness: A particular instance of a network traffic subset that was identified (detected) as the malicious traffic by a detector and that resulted in an alert created by the detector.

Since the malicious traffic is detectable using one or more witness types, many different witnesses can exist for one alert. Some examples of witness types and witnesses are listed below:

- Let us have communication of a piece of malware with a C&C server. If the content of the communication is known (e.g., using reverse engineering), the witness type can be a *sequence of bytes* of a packet, and the witness is any *reassembled message* containing the sequence of bytes discoverable by some pattern matching algorithm.

- Having a malware sample with the fixed IP address as a C&C server, the witness type can be an *IP address* and the witness is any *packet* with this IP address as source or destination.

- Similarly, a C&C server can be specified as a domain name. The witness type can be the *domain name* in such case, and the witness is any *DNS query* from a client or any *HTTP request* containing this domain name as a hostname.

- When a C&C server uses specific communication patterns (e.g., deterministic timing such as once per hour of requests and responses with the known characteristics like number of packets, number of bytes, port numbers), the witness can be any *sequence of packets* that represents the communication and shows the sought characteristic. The same information might be equivalently contained in a *sequence of flow records* aggregating the information about the packets.

It should be noted that a witness need not be a complete communication between two addresses. Statistic based detectors with thresholds may report an alert at any time after processing a minimal subset of network traffic that contains enough data (bytes, packets, flow records, . . .) to reach the threshold. In such case, an alert is reported for every randomly selected subset of the network traffic that contains a witness.

4.5 Parallel Analysis of Flow Data (PARINFRA)

This dissertation thesis presents a parallel approach to flow data processing at (near) real-time. The infrastructure for parallel processing that was built for our experiments is shown in the high-level view in Fig. 1. The Flow Scatter box splits a single stream of flow records (incoming flow data) into subsets and distributes the subsets among the multiple equivalent computing nodes. Every node has the same configuration of the detection modules. Each node processes independent set of flow records without additional communication with other nodes. The results are alerts containing detected security events.

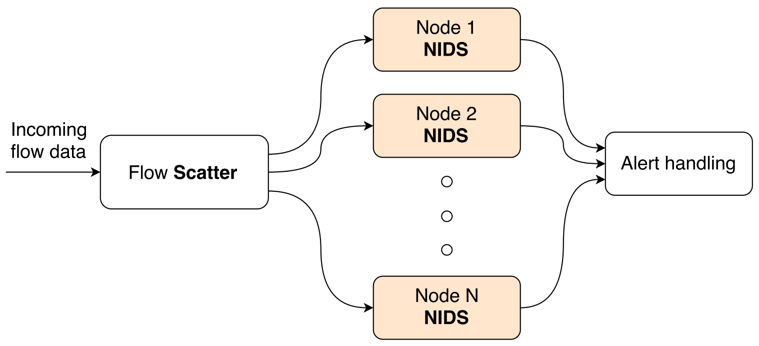

Figure 1: A high-level view of the infrastructure for parallel stream-wise processing using a Flow Scatter.

The heart of the infrastructure is the Flow Scatter that must decide which computing node processes which flow record. The decision must be made with minimal delay; otherwise, the Flow Scatter is a bottleneck of the whole infrastructure. In addition, the decision about node number must be made concerning semantic relations in data, whereas the importance of such relations was explained in in the published paper [A.1]. The experiments about splitting the stream of flow records were described in the published paper [A.3]. The experiments showed that disrupting the semantic relations affects the performance of the detection algorithms, i.e., a security event might be undetected if there is not enough data together on one computing node.

Construction of the Flow Scatter that respects semantic relations depends on a set of detectors that are used. The practical experiments with the flow data captured from a real backbone network and the set of detection modules (from the NEMEA system) showed that a Flow Scatter can distribute flow records uniformly among computing nodes to the algorithms of the same group, and there are different groups of algorithms (scattering groups). Each scattering group is identified by common characteristics that define “what data should stay together.” Therefore, the first challenge is to find the scattering groups, whereas each group is then implemented as a hashing method in the scatter.

The dissertation thesis of the author shows a straightforward algorithm how to find such groups of algorithms. We have a set of the scattering groups S, which is empty at the beginning, and the list A of the detection algorithms, which are formally described. The set S contains descriptions of subsets of flow records that contain complete non-broken witnesses. Every scattering group S_i contains algorithms with the similar witness types, and therefore it defines a scattering function of the scatter that chooses the computing node number for processing a particular flow record by the group of algorithms.

The experiments from [A.3] identified three different scattering groups. Generally, some algorithms analyze flow records of the same *srcip*, *dstip*, and the last group contained algorithms that expected bi-directional flow records, i.e., traffic between two IP addresses in both directions. Therefore, the Flow Scatter had three scattering functions inside. Since all scattering functions are computed for each incoming flow record, it is possible that the Flow Scatter duplicates a flow record and resends it to at most three computing nodes. However, the flow record is processed only once by all detection algorithms (even though it might be on different nodes).

Scalable Infrastructure with Flow Scatters

To overcome a possible throughput limit of a single Flow Scatter that would affect the performance of the whole infrastructure, we designed an extended scheme with multiple Flow Scatters. In addition, there is a set of Flow Scatters that receive the flow data. In the figure, there is also a Round-Robin Scrubber, which simply distributes the single stream of flow data among the Flow Scatters, however, it is also possible that each Flow Scatter receives a separate flow data stream (e.g., from flow exporters). The point is that all Flow Scatters have completely same configuration and so the resulting node number is always the same no matter which Flow Scatter computed it.

4.6 Discussion

Section 4.1 presented the NEMEA system that we designed and developed. Development of NEMEA was essential for this research in the area of online analysis of extended flow records. It is based on the stream-wise approach to processing (described in Section 4.2), and the proposed "application-aware" detection algorithms can analyze L7 information contained in the flow records (described in Section 4.3). Section 4.4 defines terminology to describe semantic relations in the flow data (*witnesses*). This terminology can be used to design a *Flow Scatter* module that splits a single stream of flow data on-the-fly to create independent subsets. These subsets are processed in parallel. The parallel scalable infrastructure was described in Section 4.5.

The valuable achievement of this many-years work is the deployment of NEMEA in the Czech national research and education network (NREN) — CESNET2, in a significant commercial data center & ISP network in the Czech Republic, and in the NREN of Switzerland. The author of this dissertation thesis has successfully started to build a community of researchers and developers that contribute to this project. NEMEA was also successfully used for educational purposes, and it is planned to continue with this effort because NEMEA is a suitable tool for students and researchers to study network traffic and detection algorithms.

5 Conclusions & Future Work

5.1 Summary

Network monitoring, as well as anomaly detection, are essential parts of every well-running network infrastructure. Since the network infrastructures grow and the bandwidth increases, it is necessary to adapt monitoring and analysis systems to handle high data volume. Additionally, many security threats are difficult to detect using traditional approaches based on *basic flow data*. Therefore, this dissertation thesis deals with several challenges to overcome the current state and prepare the systems for the future.

This dissertation thesis is a collection of published papers on anomaly detection in computer networks. Each paper represents a particular contribution in the network security area, particularly, in the detection of suspicious traffic using flow data from monitoring systems. This work focused on large high-speed networks, especially the backbone ones. All works have been tested and deployed in the Czech national academic network and therefore the papers usually contain the results and statistics from our experiments with the real network traffic that was observed.

5.2 Contributions of the Thesis

The contributions of this thesis, as they were described in the previous chapters, might be divided into several areas. From the practical point of view, we have designed and developed a NEMEA framework for the analysis of IP flows. The framework allowed us to perform practical experiments with real network traffic monitoring as well as to start this research in multiple scopes: *stream-wise processing*, *application-aware detection algorithms*, *semantic relations in the flow data*, *parallel processing the flow data*.

Monitoring systems of large networks produce high data volume of flow data. Operators ought to store the flow data by law; however, only basic flow records with information up to transport layer (L4) are being stored. Additionally, the traditional approach to an analysis of the basic data is based on processing the fixed length time windows. The fixed time windows necessarily cause higher detection delays. Therefore, this research focused on stream-wise processing. The stream-wise algorithms process data immediately without any need of waiting for the end of a time window, i.e. they work *online*. Besides, the stream-wise algorithms do not store data permanently by design. They are designed to process infinite stream of flow data on-the-fly.

This research showed that extended flow data allow for detection of some advanced security threats. Such security events look like legitimate flow records. The application-aware detection methods were studied and designed to overcome the lack of visibility. The application-aware approach is based on extended flow records containing information from higher protocol layers (up to application layer — L7). Application headers from unencrypted network traffic allow for reliable detection of some application threats such as brute-force dictionary attacks. Naturally, this approach requires capable tools that can process extended flow records. This was the reason for developing the NEMEA framework.

Years of studying and development of the detection algorithms brought experiences about the semantic relations in the flow data. Practical experiments with the real network traffic from the Czech academic network showed that breaking the relations in the flow data has a significant impact on the detection results. However, it is useful to split a stream of flow data into independent substreams. It was hence necessary to find a methodology that would preserve those semantic relations during the splitting. The published paper describes the semantic relations as witnesses in the flow data that must remain complete to keep the security events detectable.

The methodology about witnesses allows for parallel processing that is essential for analysis of the increasing volume of flow data. This dissertation thesis describes a Flow Scatter module that is constructed according to the witness types. The Flow Scatter splits the stream of flow records among multiple computing nodes. As a result, the monitoring and detection system can scale up.

Finally, the valuable achievement of this many-years work is the deployment of NEMEA in the Czech national research and education network (NREN) — CESNET2, in a significant commercial data center & ISP network in the Czech Republic, and in NREN of Switzerland. NEMEA was also successfully used for educational purposes, and it is planned to continue with this effort because NEMEA is a suitable tool for students and researchers to study network traffic and detection algorithms.

5.3 Future Work

As a future research, there are several unsolved challenges related to this dissertation thesis:

- automatic adaptation of the Flow Scatter based on, e.g., Machine learning algorithms,
- automatic mitigation triggered by the detected security events,
- granularity of the Flow Scatter — find minimal subsets and sample data among the computing nodes without affecting the detection results.

5.4 Shrnutí

Tato práce je koncipována jako soubor publikovaných recenzovaných článků z oblasti síťové bezpečnosti a analýzy síťového provozu v podobě síťových toků (flow). V době, kdy tento výzkum začínal, se běžně používal tradiční způsob analýzy síťových toků založený na zpracování dat uložených podle pevně daných časových intervalů. Tento časový interval byl obvykle nastaven na 5 minut. Dalším charakteristickým znakem analýzy toků bylo použití informací z prvních čtyř vrstev ISO/OSI modelu. To znamená, že flow data obsahovala pouze informace z IP hlaviček a TCP/UDP/ICMP hlaviček.

Naše práce se zabývá novým proudovým způsobem zpracování síťových toků („stream-wise processing"), který má sloužit jako náhrada tradiční práce s flow daty. Tento způsob zpracování se zakládá na online analýze příchozích záznamů o síťových tocích. Analytické a deteční moduly v tomto případě nečekají na dokončení časového intervalu a zpracovávají data ihned. Kromě okamžitého zpracování se očekává, že stream-wise moduly nebudou data dlouhodobě ukládat. Je nutné, aby se veškerá data, která moduly uloží ať už v operační paměti či na pevném disku, byla pouze krátkodobá a modul se musí postarat o automatické promazávání těchto dat.

Existuje mnoho bezpečnostních hrozeb, které není možné spolehlivě detekovat pomocí tradičních základních síťových dat s informacemi omezenými na první čtyři vrstvy ISO/OSI modelu. Z toho důvodu se náš výzkum začal zabývat analýzou „rozšířených síťových toků", které obsahují navíc informace z aplikačních vrstev (L7). Tyto informace je možné do flow dat přidávat díky hardwarové akceleraci v exportérech.

Během posledních pěti let výzkumu vznikla řada detekčních modulů, které využívají proudový způsob zpracování a zároveň jsou schopny využít aplikační informace pro účely detekce. Řadu vytvořených detekčních modulů se podařilo publikovat v podobě recenzovaných příspěvků na mezinárodních konferencích, kde byly dosažené úspěchy prezentovány špičkovým specialistům z oboru. Vybrané publikované příspěvky tvoří součást této dizertační práce.

Další výzvou vedle zlepšení detekčních schopností pomocí L7 informací bylo zpracování velkého množství dat za jednotku času. Vzhledem k povaze dat, která je třeba zpracovávat, zejména pak kvůli rychlosti, variabilitě a objemu dat, je možné data o síťovém provozu považovat za Big Data. Podle aktuálních trendů bude docházet ke zvyšování objemu dat i v budoucnu. Z toho důvodu se tato dizertační práce zabývá paralelizací detekčního systému a paralelním zpracováním flow dat.

V oblasti paralelního zpracování pokrývá tato dizertační práce dvě hlavní oblasti. První oblastí je teoretická část, která popisuje sémantické vazby ve flow datech. Text práce obsahuje definice základních potřebných pojmů a vysvětluje tzv. svědka („witness") jako podmnožinu flow dat reprezentující škodlivý provoz, která musí zůstat pohromadě, aby byl tento provoz detekovatelný detekčním algoritmem. Charakteristika svědků je nazývána jako typ svědka („witness type").

Druhou oblastí je návrh a realizace paralelní infrastruktury systému pro analýzu flow dat. Navržená infrastruktura vychází z teoretické části, která vysvětluje důležitost sémantických vazeb pro detekční algoritmy. V paralelní infrastruktuře proto zaujímá důležitou úlohu komponenta pro rozdělování proudu flow dat do menších podskupin, které je následně možné zpracovávat paralelně. Tato komponenta je nazývána Flow scatter.

Dizertační práce obsahuje algoritmus pro nalezení správného chování Flow Scatteru tak, aby zachovával sémantické vazby důležité pro použité detekční algoritmy. Vytvoření Flow Scatteru je postaveno na typech svědků a jejich seskupení podle toho, jestli mezi sebou typy svědků souvisí.

Bibliography

[1] M. Ahmed, A. N. Mahmood, and J. Hu. A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, 60:19 – 31, 2016.

[2] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita. Network anomaly detection: Methods, systems and tools. *IEEE Communications Surveys Tutorials*, 16(1):303–336, First 2014.

[3] M. H. Bhuyan, H. J. Kashyap, D. K. Bhattacharyya, and J. K. Kalita. Detecting Distributed Denial of Service Attacks: Methods, Tools and Future Directions. *The Computer Journal*, page bxt031, 2013.

[4] R. B. Blazek, H. Kim, B. Rozovskii, and A. Tartakovsky. A novel approach to detection of denial-of-service attacks via adaptive sequential and batch-sequential change-point detection methods. In *Proceedings of the IEEE Systems, Man, and Cybernetics Information Assurance Workshop, West Point, NY, USA*, 2001.

[5] J. D. Brutlag. Aberrant Behavior Detection in Time Series for Network Monitoring. In *LISA*, pages 139–146, 2000.

[6] V. K. Bumgardner and V. W. Marek. Scalable hybrid stream and hadoop network analysis system. In *Proceedings of the 5th ACM/SPEC International Conference on Performance Engineering (ICPE)*, 2014.

[7] C. Callegari, L. Gazzarrini, S. Giordano, M. Pagano, and T. Pepe. A novel PCA-based network anomaly detection. In *Communications (ICC), 2011 IEEE International Conference on*, pages 1–5. IEEE, 2011.

[8] C. Callegari, S. Giordano, and M. Pagano. Neural Network based Anomaly Detection. In *2014 IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD) (CAMAD 2014)*, Athens, Greece, Dec. 2014.

[9] M. Čermák, D. Tovarňák, M. Laštovička, and P. Čeleda. A performance benchmark for netflow data analysis on distributed stream processing systems. In *IEEE Network Operations and Management Symposium (NOMS)*, 2016.

[10] Y. Chen, K. Hwang, and W.-S. Ku. Distributed change-point detection of DDoS attacks over multiple network domains. In *Proc. of Int'nl Symp. on Collaborative Technologies and Systems*, pages 543–550, 2006.

[11] B. Claise. Cisco Systems NetFlow Services Export Version 9. http://www.ietf.org/rfc/rfc3954.txt, Oct. 2004.

[12] S. Farrell and H. Tschofenig. Pervasive Monitoring Is an Attack. http://www.ietf.org/rfc/rfc7258.txt, May 2014.

[13] R. Fontugne, J. Mazel, and K. Fukuda. Hashdoop: A MapReduce framework for network anomaly detection. In *IEEE Conference on Computer Communications Workshops (INFOCOM)*, 2014.

[14] M. Garofalo. *Big Data Analytics for Flow-based Anomaly Detection in High-Speed Networks*. PhD thesis, 2017.

[15] L. Hellemons, L. Hendriks, R. Hofstede, A. Sperotto, R. Sadre, and A. Pras. *SSHCure: A Flow-Based SSH Intrusion Detection System*, pages 86–97. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[16] R. Hofstede, V. Bartos, A. Sperotto, and A. Pras. Towards real-time intrusion detection for NetFlow and IPFIX. 2013.

[17] N. Hoque, M. H. Bhuyan, R. C. Baishya, D. K. Bhattacharyya, and J. K. Kalita. Network attacks: Taxonomy, tools and systems. *Journal of Network and Computer Applications*, 40:307–324, 2014.

[18] L. T. Ibrahim, R. Hassan, K. Ahmad, and A. N. Asat. A study on improvement of internet traffic measurement and analysis using Hadoop system. In *International Conference on Electrical Engineering and Informatics (ICEEI)*. IEEE, 2015.

[19] T. Jirsik, M. Cermak, D. Tovarnak, and P. Celeda. Toward stream-based ip flow analysis. *IEEE Communications Magazine*, 55(7):70–76, 2017.

[20] A. M. Karimi, Q. Niyaz, W. Sun, A. Y. Javaid, and V. K. Devabhaktuni. Distributed network traffic feature extraction for a real-time IDS. In *IEEE International Conference on Electro Information Technology (EIT)*, 2016.

[21] Y. Lee, W. Kang, and H. Son. An internet traffic analysis method with mapreduce. In *IEEE/IFIP Network Operations and Management Symposium Workshops (NOMS)*, 2010.

[22] Y. Lee and Y. Lee. Toward scalable internet traffic measurement and analysis with hadoop. *ACM SIGCOMM Computer Communication Review*, 2013.

[23] J. Mirkovic, G. Prier, and P. Reiher. Attacking DDoS at the source. In *Network Protocols, 2002. Proceedings. 10th IEEE International Conference on*, pages 312–321. IEEE, 2002.

[24] J. Mirkovic and P. Reiher. A taxonomy of DDoS attack and DDoS defense mechanisms. *ACM SIGCOMM Computer Communication Review*, 34(2):39–53, 2004.

[25] S. Sagiroglu and D. Sinanc. Big data: A review. In *2013 International Conference on Collaboration Technologies and Systems (CTS)*, pages 42–47, May 2013.

[26] O. Salem, S. Vaton, and A. Gravey. A scalable, efficient and informative approach for anomaly-based intrusion detection systems: theory and practice. *International Journal of Network Management*, 20(5):271–293, 2010.

[27] V. A. Siris and F. Papagalou. Application of anomaly detection algorithms for detecting SYN flooding attacks. *Computer communications*, 29(9):1433–1442, 2006.

[28] A. G. Tartakovsky, B. L. Rozovskii, R. Blažek, and H. Kim. A Novel Approach to Detection of Intrusions in Computer Networks via Adaptive Sequential and Batch-Sequential Change-Point Detection Methods. *IEEE Transactions on Signal Processing*, 54(9):3372–3382, 2006.

[29] A. G. Tartakovsky, B. L. Rozovskii, R. B. Blažek, and H. Kim. Detection of intrusions in information systems by sequential change-point methods. *Statistical Methodology*, 3(3):252–293, 2006.

[30] H. Wang, D. Zhang, and K. Shin. Detecting SYN flooding attacks. In *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1530–1539, 2002.

[31] W. Willinger and V. Paxson. Where mathematics meets the Internet. *Notices of the AMS*, 45(8):961–970, 1998.

[32] N. Ye, C. Borror, and Y. Zhang. EWMA techniques for computer intrusion detection through anomalous changes in event intensity. *Quality and Reliability Engineering International*, 18(6):443–451, 2002.

[33] M. Zadnik, P. Krobot, and J. Wrona. Experience with big data frameworks for IP flow collector. In *Technical report*, 2016.

[34] S. T. Zargar, J. Joshi, and D. Tipper. A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks. *IEEE Communications Surveys & Tutorials*, 15(4):2046–2069, 2013.

[35] J. Zhang, Y. Zhang, P. Liu, and J. He. A Spark-Based DDoS Attack Detection Model in Cloud Services. In *Proceedings of 12th International Conference on Information Security Practice and Experience (ISPEC)*, 2016.

Publications of the Author

Reviewed Relevant Publications of the Author

[A.1] T. Cejka, M. Zadnik *Preserving relations in parallel flow data processing* 11th IFIP WG 6.6 International Conference on Autonomous Infrastructure, Management, and Security, AIMS 2017, Zurich, Switzerland, 2017.

[A.2] T. Jánský, T. Čejka, V. Bartoš *Hunting SIP Authentication Attacks Efficiently* 11th IFIP WG 6.6 International Conference on Autonomous Infrastructure, Management, and Security, AIMS 2017, Zurich, Switzerland, 2017.

[A.3] M. Švepeš, T. Cejka *Making flow data analysis parallel* 11th IFIP WG 6.6 International Conference on Autonomous Infrastructure, Management, and Security, AIMS 2017, Zurich, Switzerland, 2017.

[A.4] T. Cejka, V. Bartoš, M. Svepes, Z. Rosa, H. Kubatova *NEMEA: A Framework for Network Traffic Analysis* 12th International Conference on Network and Service Management (CNSM 2016), Montreal, Canada 2016.

[A.5] Z. Rosa, T. Cejka, M. Zadnik, V. Puš *Building a Feedback Loop to Capture Evidence of Network Incidents* 12th International Conference on Network and Service Management (CNSM 2016), Montreal, Canada 2016.

[A.6] T. Cejka, M. Svepes *Analysis of Vertical Scans Discovered by Naive Detection* Management and Security in the Age of Hyperconnectivity: 10th IFIP WG 6.6 International Conference on Autonomous Infrastructure, Management, and Security, AIMS 2016, Munich, Germany, 2016.

[A.7] T. Cejka, V. Bartos, L. Truxa, and H. Kubatova *Using Application-Aware Flow Monitoring for SIP Fraud Detection* Intelligent Mechanisms for Network Configuration and Security: 9th IFIP WG 6.6 International Conference on Autonomous Infrastructure, Management, and Security, AIMS 2015, S. Latré, M. Charalambides, J. François, C. Schmitt, and B. Stiller, Eds. Ghent, Belgium: Springer International Publishing 2015.

[A.8] T. Cejka, Z. Rosa, H. Kubatova *Stream-wise Detection of Surreptitious Traffic over DNS* 19th IEEE International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (IEEE CAMAD 2014), Athens, Greece 2014.

The paper has been cited in:

- Nuojua, V., David, G., Hämäläinen, T.: *DNS tunneling detection techniques – Classification, and theoretical comparison in case of a real APT campaign.* In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 10531 LNCS, pp. 280-291, 2017. DOI: 10.1007/978-3-319-67380-6_26

[A.9] P. Benacek, R. B. Blazek, T. Cejka, H. Kubatova *Change-Point Detection Method on 100 Gb/s Ethernet Interface* Proceedings of the Tenth ACM/IEEE Symposium on Architectures for Networking and Communications Systems, New York, USA 2014.

The paper has been cited in:

- Nakamura, K., Hayashi, A., Matsutani, H.: *An FPGA-based low-latency network processing for spark streaming.* In Proceedings of the IEEE International Conference on Big Data, Big Data 2016, art. no. 7840876, pp. 2410-2415, 2016. DOI: 10.1109/BigData.2016.7840876

[A.10] T. Cejka, L. Kekely, P. Benacek, R. B. Blazek, H. Kubatova *FPGA Accelerated Change-Point Detection Method for 100Gb/s Networks* MEMICS proceedings, 9th Doctoral Workshop on Mathematical and Engineering Methods in Computer Science (MEMICS 2014), Telč, Czech Republic 2014.

[A.11] P. Benáček, V. Puš, H. Kubátová, T. Čejka *P4-To-VHDL: Automatic generation of high-speed input and output network blocks*, Microprocessors and Microsystems, Vol. 56, Elsevier, pp. 22–33, 2018.

The paper has been cited in:

- Garcia, Luis Fernando Uria, et al.: *Introdução à Linguagem P4-Teoria e Prática.* Minicursos do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (Minicursos_SBRC) 36 (2018).

Remaining Relevant Publications of the Author

[A.12] T. Čejka and R. Krejčí *Configuration of open vSwitch using OF-CONFIG* IEEE/IFIP Network Operations and Management Symposium, Istanbul, Turkey, 2016.

The paper has been cited in:

- P. Bellavista, et al. *Multi-domain SDN controller federation in hybrid FiWi-MANET networks*, EURASIP Journal on Wireless Communications and Networking 2018.1 (2018).
- P. Jha, and B. Tech. *End-to-End Quality-of-Service in Software Defined Networking*, 2017.

[A.13] T. Čejka and A. Robledo *Detecting Spoofed Time in NTP Traffic* The 4th Prague Embedded Systems Workshop, Roztoky u Prahy, Czech Republic, 2016.

[A.14] M. Svepes and T. Cejka *Overload-resistant Network Traffic Analysis* The 4th Prague Embedded Systems Workshop, Roztoky u Prahy, Czech Republic 2016.

[A.15] T. Čejka, R. Bodó, and H. Kubátová *Nemea: Searching for Botnet Footprints* The 3th Prague Embedded Systems Workshop, Roztoky u Prahy, Czech Republic 2015.

[A.16] V. Bartoš, M. Žádník, T. Čejka *Nemea: Framework for stream-wise analysis of network traffic* CESNET Technical Report 2013.

The paper has been cited in:

- Y.K. Lai, T. Wellem, H.P. You. *Hardware-assisted estimation of entropy norm for high-speed network traffic*, Electronics Letters, 2014.

[A.17] T. Čejka *Hardware Accelerated Anomaly Detection in Computer Networks*, A Doctoral Study Report submitted to the Faculty of Information Technology, Czech Technical University in Prague Prague, December 2014.

[A.18] T. Cejka, R. B. Blazek *Real-time Detection of Anomalies in High-speed Computer Networks* The 1st Embedded Systems Workshop - ESW 2013, Temešvár, Czech Republic 2013.

[A.19] T. Cejka *Systém pro detekci anomálií v počítačových sítích* Počítačové architektury a diagnostika - PAD 2013, Teplá, Česká republika 2013.

[A.20] T. Cejka *Hardwarově akcelerovaná detekce anomálií v počítačových sítích s využitím FPGA* Počítačové architektury a diagnostika - PAD 2012, Milovy, Česká republika 2012.

Supervised Relevant Theses

[A.21] L. Stejskalová *System for grouping suspicious network addresses*, (in Czech), Diploma thesis, FIT, CTU in Prague, 2018.

[A.22] M. Tomáš *Extension of reputation database with information from Passive DNS*, (in Czech), Bachelor thesis, FIT, CTU in Prague, 2018.

[A.23] J. Jančička *Automatic classification of network entities*, (in Czech), Bachelor thesis, FIT, CTU in Prague, 2017. *Honored as an excellent thesis*

[A.24] A. Plánský: Automatic Analysis of Security Incident Alerts, (in Czech) Diploma thesis, FIT, CTU in Prague, 2017.

[A.25] O. Hollmann *Detection of network attacks of Denial of Service type*, (in Czech), Bachelor thesis, FIT, CTU in Prague, 2017.

[A.26] Jiří Havránek *Network flows exporter supporting application information*, (in Czech), Bachelor thesis, FIT, CTU in Prague, 2017. *Honored as an excellent thesis*

[A.27] Z. Rosa *Automatic data capture for detected events*, (in Czech) Diploma Thesis, FIT, CTU in Prague, 2017.

[A.28] M. Švepeš *Extension of the NEMEA system for deployment in a distributed environment*, (in Czech), Diploma Thesis, FIT, CTU in Prague, 2017. *Honored as an excellent thesis*

[A.29] A. Robledo *Network Time Protocol attacks detection*, Diploma thesis, FIT, CTU in Prague, 2016.

[A.30] M. Kalina *Traffic monitoring in 100Gb/s network infrastructures*, (in Czech), Diploma thesis, czech, FIT, CTU in Prague, 2016.

[A.31] Z. Kasner *Flow-Based Classification of Devices in Computer Networks*, Bachelor thesis, FIT, CTU in Prague, 2016.

[A.32] T. Jánský *Application for analysis of VoIP/SIP network flows*, (in Czech), Bachelor thesis, FIT, CTU in Prague, 2016. *Honored as an excellent thesis*

[A.33] L. Truxa *Detection of VoIP Exchanges Fraud*, (in Czech), Diploma thesis, czech, FIT, CTU in Prague, 2015.

[A.34] N. Jíša *Detection of Network Attacks to Voice over IP Infrastructure*, (in Czech), Diploma thesis, czech, FIT, CTU in Prague, 2015.

[A.35] Z. Rosa *Detection of tunneling in computer networks*, (in Czech), Bachelor Thesis, FIT, CTU in Prague, 2014, *Honored as an excellent thesis*

[A.36] M. Švepeš *Configuration and monitoring system for distributed system NEMEA*, (in Czech), Bachelor Thesis, FIT, CTU in Prague, 2014, *Honored as an excellent thesis*

[A.37] J. Neužil *P2P Botnet Detection in Computer Networks*, Bachelor Thesis, FIT, CTU in Prague, 2014.